



Maxwell Scripting Guide



ANSYS, Inc.
Southpointe
2600 Ansys Drive
Canonsburg, PA 15317
ansysinfo@ansys.com
<https://www.ansys.com>
(T) 724-746-3304
(F) 724-514-9494

Release 2024 R2
July 2024

ANSYS, Inc. and
ANSYS Europe,
Ltd. are UL
registered ISO
9001:2015 com-
panies.

Copyright and Trademark Information

© 1986-2024 ANSYS, Inc. Unauthorized use, distribution or duplication is prohibited.

ANSYS, Ansys Workbench, AUTODYN, CFX, FLUENT and any and all ANSYS, Inc. brand, product, service and feature names, logos and slogans are registered trademarks or trademarks of ANSYS, Inc. or its subsidiaries located in the United States or other countries. Icem CFD is a trademark used by ANSYS, Inc. under license. All other brand, product, service and feature names or trademarks are the property of their respective owners. FLEXlm and FLEXnet are trademarks of Flexera Software LLC.

Disclaimer Notice

THIS ANSYS SOFTWARE PRODUCT AND PROGRAM DOCUMENTATION INCLUDE TRADE SECRETS AND ARE CONFIDENTIAL AND PROPRIETARY PRODUCTS OF ANSYS, INC., ITS SUBSIDIARIES, OR LICENSORS. The software products and documentation are furnished by ANSYS, Inc., its subsidiaries, or affiliates under a software license agreement that contains provisions concerning non-disclosure, copying, length and nature of use, compliance with exporting laws, warranties, disclaimers, limitations of liability, and remedies, and other provisions. The software products and documentation may be used, disclosed, transferred, or copied only in accordance with the terms and conditions of that software license agreement.

ANSYS, Inc. and ANSYS Europe, Ltd. are UL registered ISO 9001: 2015 companies.

U.S. Government Rights

For U.S. Government users, except as specifically granted by the ANSYS, Inc. software license agreement, the use, duplication, or disclosure by the United States Government is subject to restrictions stated in the ANSYS, Inc. software license agreement and FAR 12.212 (for non-DOD licenses).

Third-Party Software

See the legal information in the product help files for the complete Legal Notice for Ansys proprietary software and third-party software. If you are unable to access the Legal Notice, please contact ANSYS, Inc.

Table of Contents

Table of Contents	Contents-1
1 - Introduction to Scripting	1-1
Scripting Help Conventions	1-1
Variable Types	1-2
Introduction to VBScript	1-2
Simple and Composite Names	1-3
VBScript Variables	1-3
Declaring Variables	1-3
Declaring Variables in Python	1-4
Variable Naming Conventions	1-4
Scope and Lifetime of Variables	1-4
Array Variables	1-4
VBScript Operators	1-5
Operator Precedence	1-5
Arithmetic Operators	1-6
Comparison Operators	1-7
Logical Operators	1-7
Controlling Program Execution	1-8
Using If...Then...Else	1-8
Using Select Case	1-8
Looping Through Code	1-8
Using a For...Next Loop	1-9
Using a Do Loop	1-9
Repeating Statements While a Condition is True	1-9
Repeating a Statement Until a Condition Becomes True	1-9
VBScript Procedures	1-9
Function Procedures	1-9
Sub Procedures	1-10

Converting Between Data Types	1-10
Including Scripts	1-10
Aborting Scripts	1-11
Interacting with a Script	1-11
Recommended VBScript References	1-11
Introduction to IronPython	1-12
Scope	1-12
Python compatibility	1-12
Advantages of IronPython	1-12
Introduction to IronPython	1-13
Scope	1-13
Python compatibility	1-14
Advantages of IronPython	1-14
IronPython Mini Cookbook	1-15
Comments	1-15
Assigning/Creating Variables	1-16
Create Lists/Arrays	1-16
Create Dictionaries/Maps	1-17
Boolean Values	1-17
Converting Numbers to Strings and Vice Versa	1-17
String Formatting/Concatenation	1-18
Looping over Lists	1-18
Looping over a Range	1-18
Indentation in IronPython	1-19
Indenting Functions	1-19
Indenting If Conditions	1-19
Methods in IronPython	1-20
Finding Methods	1-20
Help	1-20
Translating Script Commands from VBScript to IronPython	1-20

Script Method Argument	1-20
Primitive Types	1-21
Named Arrays	1-21
Named Functions	1-21
VBScript Method Call Types	1-21
Converting VBScript Function calls to IronPython Syntax	1-22
Return Values	1-22
Primitive Method Arguments	1-22
Named Array Arguments	1-23
Named Array Values with All Key Value Pairs	1-23
Named Arrays with Nested Named Arrays	1-23
Function Blocks	1-24
Scripting Using Iron Python	1-25
Translating a script in VBScript to IronPython	1-25
Writing an IronPython script from scratch	1-25
IronPython Script Execution Environment	1-26
Script Argument for IronPython	1-26
Scripting using Embedded VBScript or JavaScript	1-27
Scripting with IronPython	1-30
Standalone IronPython	1-31
Running Standalone IronPython	1-31
Using a Recorded Script	1-31
Creating an External Script	1-31
Example Script	1-32
IronPython Samples	1-33
Change property	1-34
Create a Cone using IronPython	1-35
Creating User Defined Primitives and User Defined Models in Python Scripts	1-39
Advantages Compared to C++	1-39
Changes compared to C	1-39

Structures	1-39
Return Values for UDM and UDP Functions	1-39
Constants	1-39
Methods	1-40
Output Parameters	1-40
Comparison with C function:	1-41
'List Size' Parameters	1-41
Comparison with C function:	1-42
Added Parameters	1-42
Developing a UDM/UDP	1-43
Creation	1-43
Location	1-43
Organize	1-44
Edit/Reload	1-44
UDPEExtension	1-44
Import	1-44
Main class: UDPEExtension	1-44
IUDPEExtension methods	1-45
Mandatory methods.	1-45
GetLengthParameterUnits()	1-45
GetPrimitiveTypeInfo()	1-45
GetPrimitiveParametersDefinition2()	1-45
AreParameterValuesValid2(errorMessage, udpParams)	1-45
CreatePrimitive2(funcLib, udpParams)	1-45
Optional methods	1-45
GetPrimitiveParameters()	1-45
GetRegisteredFaceNames()	1-46
GetRegisteredEdgeNames()	1-46
GetRegisteredVertexNames()	1-46
MapParametersDefinitionVersions2(oldVersion, oldUDPPParams)	1-46

GetOldPrimitiveParametersDefinition2(version)	1-46
Example UDP	1-46
UDMExtension	1-47
Import	1-47
Main class: UDMExtension	1-47
IUDMExtension methods	1-47
Mandatory methods.	1-47
GetInfo()	1-47
IsAttachedToExternalEditor()	1-47
CreateInstance(funcLib)	1-48
GetUnits(instanceld)	1-48
ReleaseInstance(instanceld)	1-48
GetAttribNameForEntityId()	1-48
GetAttribNameForPartId()	1-49
Optional methods	1-49
GetInstanceSourceInfo(instanceld)	1-49
ShouldAttachDefinitionFilesToProject()	1-50
Example UDM	1-50
UDMFunctionLibrary	1-50
Functions list:	1-51
UDM/UDP Functions	1-52
Return Values for Each UDM and UDP Function	1-52
UDP/UDM Structures and Constants	1-54
UDP/UDM Structures	1-55
List of structures	1-56
UDP/UDM Constants	1-62
Enum constants:	1-62
UDP Python Example	1-64
Introduction to CPython	1-69
Creating an External Script	1-69

Start ansysedt as GRPC server	1-70
Connect Functions:	1-70
Example:	1-71
Launching Electronics Desktop	1-71
Connecting with a Running Instance of Electronics Desktop	1-71
Closing Electronics Desktop/Ending the Script	1-72
-grpcsrv Flag	1-72
Anslys Electronics Desktop Scripting	1-72
Overview of Electronics Desktop Scripting Objects	1-73
oAnsoftApp	1-73
oDesktop	1-73
oProject	1-74
oDesign	1-74
oEditor	1-74
oModule	1-75
Example Script Opening	1-75
GetActiveProject and GetActiveDesign for Wider Use	1-76
Running a Script	1-76
Within Electronics Desktop	1-76
From the Command Line	1-77
Direct Launch	1-78
Recording a Script	1-78
Recording a Script to File	1-78
Recording a Script to a Project	1-79
Working with Project Scripts	1-80
Executing a Script from Within a Script	1-81
Electronics Desktop Scripting Conventions	1-82
Named Arguments	1-82
VBscript Example	1-83
IronPython Example	1-83

Setting Numerical Values	1-84
Event Callback Scripting	1-85
PyAEDT (Beta)	1-86
2 - Object-Oriented Property Scripting	2-1
Object-Oriented Scripting	2-1
Material Properties and Examples	2-9
Body Properties and Modification	2-11
Retrieving Variables	2-11
Retrieve Datasets and Values	2-12
GetSolutionData API	2-14
Summary	2-14
Additional Details Specific to AEDT Solvers	2-18
Additional details on Boundaries/Excitations	2-19
3D component encapsulation	2-21
Additional details on Solve setup	2-21
Materials Scripting Support	2-22
Object Oriented Scripting for Materials	2-26
Examples showing change to material property type:	2-27
Examples showing change to a vector component value	2-27
Change choice property value	2-28
Change choice property value	2-29
Project Object Script Commands	2-30
AnalyzeAll [project]	2-33
ClearMessages	2-34
Close	2-35
CopyDesign	2-36
CutDesign	2-36
DeleteDesign	2-37
DeleteToolObject	2-38
GetActiveDesign	2-39

GetConfigurableData (Project)	2-39
GetDefinitionManager	2-40
GetDependentFiles	2-41
GetDesign	2-41
GetEDBHandle	2-42
GetLegacyName	2-42
GetName [Project]	2-43
GetPath	2-44
GetTopDesignList	2-45
InsertDesign	2-45
InsertDesignWithWorkflow	2-46
InsertToolObject	2-48
Paste (Project Object)	2-48
Redo [Project Level]	2-49
Rename	2-49
RestoreProjectArchive	2-50
Save	2-51
SaveAs	2-52
SaveAsStandAloneProject	2-54
SaveProjectArchive	2-55
SetActiveDefinitionEditor	2-56
SetActiveDesign	2-57
SimulateAll	2-57
Undo [Project]	2-58
UpdateDefinitions	2-59
3 - Application Object Script Commands	3-1
GetAppDesktop	3-2
GetApp	3-2
4 - Desktop Object Script Commands	4-1
AddMessage	4-5

ClearMessages	4-6
CloseAllWindows	4-10
CloseProject	4-11
CloseProjectNoForce	4-12
Count	4-12
DeleteProject	4-14
EnableAutoSave	4-15
ExportOptionsFiles	4-15
GetActiveProject	4-16
GetAutoSaveEnabled	4-17
GetBuildDateTimeString	4-18
GetCustomMenuSet	4-18
GetDefaultUnit	4-19
GetDesigns	4-21
GetDesktopConfiguration	4-22
GetDistributedAnalysisMachines	4-23
GetDistributedAnalysisMachinesForDesignType	4-23
GetExeDir	4-24
GetGDIObjectCount	4-25
GetLibraryDirectory	4-25
GetLocalizationHelper	4-27
GetMessages	4-28
GetName [Desktop]	4-29
GetPersonalLibDirectory	4-30
GetProcessID	4-31
GetProjects	4-32
GetProjectDirectory	4-32
GetProjectList	4-33
GetSchematicEnvironment	4-34
GetScriptingToolsHelper	4-35

GetSysLibDirectory	4-35
GetTempDirectory	4-36
GetUserLibDirectory	4-37
GetVersion	4-37
LaunchJobMonitor	4-38
NewProject	4-39
OpenAndConvertProject	4-40
OpenMultipleProjects	4-41
OpenProject	4-41
OpenProjectWithConversion	4-42
Paste (Project Object)	4-43
Paste (Project Object)	4-43
PauseRecording	4-44
PauseScript	4-45
Print	4-45
QuitApplication	4-46
RefreshJobMonitor	4-47
ResetLogging	4-48
RestoreProjectArchive	4-48
RestoreWindow	4-49
ResumeRecording	4-50
RunACTWizardScript	4-51
RunProgram	4-51
RunScript	4-52
RunScriptWithArguments	4-54
SelectScheduler	4-55
SetActiveProject	4-56
SetActiveProjectByPath	4-57
SetCustomMenuSet	4-58
SetDesktopConfiguration	4-59

SetLibraryDirectory	4-60
SetProjectDirectory	4-61
SetSchematicEnvironment	4-61
SetTempDirectory	4-62
ShowDockingWindow	4-63
Sleep	4-64
SubmitJob	4-65
TileWindows	4-66
Desktop Commands For Registry Values	4-67
DoesRegistryValueExist	4-68
GetRegistryInt	4-68
GetRegistryString	4-69
SetRegistryFromFile	4-70
SetRegistryInt	4-71
SetRegistryString	4-72
ImportExport Tool Commands	4-72
ImportANF	4-73
ImportANFv2	4-75
ImportAutoCAD	4-76
ImportAWRMicrowaveOffice	4-77
ImportEDB	4-79
ImportExtracta	4-80
ImportGDSII	4-81
ImportGerber	4-82
ImportIDF	4-83
ImportIDFandMerge	4-86
ImportIDX	4-87
ImportIPC	4-90
ImportODB	4-91
ImportXFL	4-92

5 - Running Instances Manager Script Commands	5-1
GetAllRunningInstances	5-1
GetRunningInstanceByProcessID	5-2
GetRunningInstancesMgr	5-2
6 - Project Object Script Commands	6-1
AddDataset	6-4
AnalyzeAll [project]	6-7
ChangeProperty	6-8
ClearMessages	6-15
Close	6-16
CopyDesign	6-16
CutDesign	6-17
DeleteDataset	6-18
DeleteDesign	6-19
DeleteToolObject	6-20
EditDataset	6-20
ExportDataset	6-23
GetActiveDesign	6-24
GetArrayVariables	6-24
GetChildNames [Project]	6-25
GetChildObject [Project]	6-26
GetChildTypes [Project]	6-27
GetConfigurableData (Project)	6-28
GetDefinitionManager	6-28
GetDependentFiles	6-29
GetDesign	6-29
GetDesigns	6-30
GetEDBHandle	6-31
GetLegacyName	6-32
GetName [Project]	6-32

GetObjPath [Project]	6-33
GetPath	6-34
GetPropEvaluatedValue	6-34
GetPropNames [Project]	6-35
GetPropSIValue	6-36
GetPropValue [Project]	6-37
GetProperties	6-38
GetPropertyValue	6-39
GetTopDesignList	6-41
GetVariableValue	6-41
GetVariables	6-42
ImportDataset	6-43
Note About File Types:	6-44
InsertDesign	6-45
InsertDesignWithWorkflow	6-47
InsertToolObject	6-48
Paste (Project Object)	6-48
Redo [Project Level]	6-49
RemoveAllUnusedDefinitions	6-49
RemoveMaterial	6-50
RemoveUnusedDefinitions	6-51
Rename	6-52
RunToolkit	6-53
Save	6-55
SaveAs	6-55
SaveAsStandAloneProject	6-58
SaveProjectArchive	6-58
SetActiveDefinitionEditor	6-59
SetActiveDesign	6-60
SetPropValue [Project]	6-61

SetPropertyValue	6-62
SetVariableValue	6-63
SimulateAll	6-64
Undo [Project]	6-65
UpdateDefinitions	6-66
7 - Definition Manager Script Commands	7-1
AddMaterial (Maxwell)	7-2
Additional Properties for Nonlinear Property Type	7-7
Additional Properties for Vector Property Type	7-8
Additional Python Examples	7-8
Additional VB Examples	7-13
CloneMaterial	7-16
ComputeCoreLossCoefficients	7-17
DoesMaterialExist	7-19
EditMaterial (Maxwell)	7-20
Additional Python Examples	7-26
Python Example: Core Loss Y Parameter for Electrical Steel (Anisotropic Property)	7-35
Additional VB Examples	7-37
Example:Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")	7-37
Example:Litz Wire	7-39
Example: Thermal Modifier	7-40
Example: B-P Curve Core Loss Model	7-41
GetProjextMaterialNames	7-43
ExportMaterial	7-44
RemoveMaterial	7-45
RemoveUnusedDefinitions	7-46
Component Manager Script Commands	7-47
Add [component manager]	7-48
AddDynamicNPortData [component manager]	7-67
AddNPortData [component manager]	7-70

AddSolverOnDemandModel	7-76
ClearSolutionCache [component manager]	7-76
Edit [component manager]	7-77
EditSolverOnDemandModel	7-104
EditWithComps [component manager]	7-104
Export [component manager]	7-116
GetData [component manager]	7-117
GetNames [component manager]	7-118
GetNPortData [component manager]	7-119
GetSolverOnDemandData	7-122
GetSolverOnDemandModelList	7-122
IsUsed [component manager]	7-122
Remove [component manager]	7-123
RemoveSolverOnDemandModel	7-124
RemoveUnused [component manager]	7-124
Update Dynamic Link [component manager]	7-125
Material Manager Script Commands	7-126
GetData [material manager]	7-126
GetNames [material manager]	7-127
GetProperties [material manager]	7-129
IsUsed [material manager]	7-129
RemoveUnused [material manager]	7-130
Model Manager Script Commands	7-131
Add [model manager]	7-131
ConvertToDynamic	7-142
ConvertToParametric	7-142
Edit [deprecated]	7-143
EditWithComps [model manager]	7-143
Export [model manager]	7-155
GetData [model manager]	7-156

GetNames [model manager]	7-157
IsUsed [model manager]	7-157
Remove [model manager]	7-158
RemoveUnused [model manager]	7-159
Network Data Explorer Manager Script Commands	7-160
ExportFullWaveSpice	7-161
ExportNetworkData	7-162
ExportNMFData	7-165
Symbol Manager Script Commands	7-165
Add [symbol manager]	7-165
BringToFront [symbol manager]	7-176
Edit [deprecated]	7-176
EditWithComps [symbol manager]	7-177
Export [symbol manager]	7-188
GetData [symbol manager]	7-189
GetNames [symbol manager]	7-190
IsUsed [symbol manager]	7-191
Remove [symbol manager]	7-191
RemoveUnused [symbol manager]	7-192
Add [footprint manager]	7-193
Edit [footprint manager]	7-216
EditWithComps [footprint manager]	7-216
Export [footprint manager]	7-234
GetData [footprint manager]	7-235
GetNames [footprint manager]	7-235
IsUsed [footprint manager]	7-236
Remove [footprint manager]	7-236
RemoveUnused [footprint manager]	7-237
Add [padstack manager]	7-238
Edit [padstack manager]	7-245

EditWithComps [padstack manager]	7-252
Export [padstack manager]	7-259
GetData [padstack manager]	7-260
GetNames [padstack manager]	7-261
IsUsed [padstack manager]	7-261
Remove [padstack manager]	7-262
RemoveUnused [padstack manager]	7-263
Script and Library Scripts	7-264
AddScript	7-264
EditScript	7-265
ExportScript	7-266
RemoveScript	7-267
ModifyLibraries	7-268
8 - Property Script Commands	8-1
Object Script Property Function Summary	8-3
Object Path	8-3
Property Object	8-3
Project Object	8-5
Design Object	8-6
3D Modeler Object	8-6
Variable Object	8-7
Optimetrics Module Object:	8-8
Optimetrics Setup Object	8-9
ReportSetup(Results) Module Object:	8-9
ReportSetup(Results) Module Child Objects:	8-10
Radiation Module Object:	8-11
Radiation Module Child Objects:	8-11
Conventions Used in this Chapter	8-12
GetArrayVariables	8-24
GetProperties	8-25

GetPropertyValue	8-26
GetVariables	8-28
GetVariableValue	8-29
SetPropertyValue	8-30
SetVariableValue	8-31
9 - Design Object Script Commands	9-1
ApplyMeshOps	9-5
Analyze	9-6
AnalyzeAll (Maxwell menu)	9-7
AnalyzeDistributed	9-7
ClearLinkedData (Design)	9-8
ConfigureFluentConductivityCoupling	9-9
ConstructVariationString	9-10
CreateParametricCircuit	9-11
Create3DDesign [Maxwell]	9-11
DeleteFieldVariation	9-12
DeleteFullVariation	9-13
DeleteLinkedDataVariation	9-14
DeleteVariation	9-15
EditNotes	9-15
EnableHarmonicForceCalculation	9-15
ExportConvergence	9-25
ExportElementBasedHarmonicForce	9-26
ExportHarmonicTransientForce	9-27
ExportMeshStats	9-29
ExportProfile	9-30
GetConfigurableData	9-30
GetData	9-31
GetDesignValidationInfo	9-32
GetGeometryMode	9-33

GetManagedFilePath	9-33
GetModule	9-34
GetName	9-35
GetNominalVariation	9-36
GetNoteText	9-36
GetPostProcessingVariables	9-37
GetSelections [Design]	9-38
GetSolutionType	9-38
GetVariationVariableValue	9-39
Is2D	9-39
Is3D	9-40
Redo [Design]	9-40
RenameDesignInstance	9-41
ResetToTimeZero	9-42
SetActiveEditor	9-42
SetConductivityThreshold	9-43
SetDesignSettings [Maxwell]	9-43
SetFastTransformationForLayoutComponent	9-57
SetObjectAttributesForLayoutComponent	9-58
SetObjectDeformation	9-61
SetObjectTemperature	9-63
SetShowAllLayoutComponents	9-65
SetShowLayoutForLayoutComponent	9-65
SetSolutionType (Maxwell)	9-66
Solve	9-67
StopSimLink	9-68
RunToolkit	9-69
Undo [Design]	9-70
ValidateDesign	9-70
ValidateLink	9-71

10 - 3D Modeler Editor Script Commands	10-1
Conventions Used in this Chapter:	10-1
<AttributesArray>	10-1
<SelectionsArray>	10-2
Draw Menu Commands	10-4
Create3DComponent	10-6
CreateBondwire	10-10
CreateBox	10-13
CreateCircle	10-16
CreateCone	10-19
CreateCutplane	10-21
CreateCylinder	10-23
CreateEllipse	10-26
CreateEquationCurve	10-29
CreateEquationSurface	10-32
CreateHelix	10-35
CreatePoint	10-37
CreateUserDefinedPart	10-39
CreatePolyline	10-47
CreateRectangle	10-52
CreateRegion	10-55
CreateRegularPolyhedron	10-58
CreateRegularPolygon	10-61
CreateSphere	10-64
CreateSpiral	10-67
CreateTorus	10-69
CreateUserDefinedModel	10-71
CreateUserDefinedPart	10-79
Edit3DComponent	10-86
[Beta] EditNativeComponentDefinition [Maxwell]	10-89

EditPolyline	10-92
Get3DComponentParameters	10-96
Get3DComponentDefinitionNames	10-97
Get3DComponentInstanceNames	10-97
Get3DComponentMaterialNames	10-98
Get3DComponentMaterialProperties	10-99
Insert3DComponent	10-99
InsertNativeComponent [Layout Component to Maxwell with CS]	10-101
InsertPolylineSegment	10-104
SweepAlongPath	10-107
SweepAlongVector	10-109
SweepAroundAxis	10-111
SweepFacesAlongNormal	10-113
SweepFacesAlongNormalWithAttributes	10-115
UpdateComponentDefinition	10-118
Edit Menu Commands	10-119
Copy	10-120
DeletePolylinePoint	10-120
DuplicateAlongLine	10-122
DuplicateAroundAxis	10-124
DuplicateMirror	10-127
Mirror	10-130
Move	10-132
OffsetFaces	10-134
Paste (Model Editor)	10-135
Rotate	10-136
Scale	10-137
Modeler Menu Commands	10-139
AssignMaterial	10-142
Chamfer	10-144

Connect	10-146
CoverLines	10-147
CoverSurfaces	10-148
CreateEntityList	10-149
CreateFaceCS	10-151
CreateGroup	10-156
CreateObjectCS	10-157
CreateObjectFromEdges	10-164
CreateObjectFromFaces	10-165
CreateRelativeCS	10-167
DeleteEmptyGroups	10-169
DeleteLastOperation	10-170
DetachFaces	10-171
EditEntityList	10-172
EditFaceCS	10-174
EditObjectCS	10-180
EditRelativeCS	10-187
Export	10-189
ExportModelImageToFile	10-191
Fillet	10-194
FlattenGroup	10-196
GenerateHistory	10-197
HealObject	10-198
GetActiveCoordinateSystem	10-203
GetCoordinateSystems	10-204
Import	10-205
ImportDXF [Modeler]	10-208
ImportGDSII [Modeler]	10-212
Intersect	10-216
MoveCStoEnd	10-217

MoveEntityToGroup	10-218
MoveFaces	10-219
ProjectSheet	10-222
PurgeHistory	10-223
Section	10-224
SeparateBody	10-225
SetModelUnits	10-226
SetWCS	10-228
ShowWindow	10-229
Split	10-229
Subtract	10-232
SweepFacesAlongNormal	10-234
ThickenSheet	10-235
UncoverFaces	10-237
Unite	10-239
Ungroup	10-240
WrapSheet	10-241
Other oEditor Commands	10-242
AddDefinitionFromBlock	10-245
AddDefinitionFromLibFile	10-250
AddViewOrientation	10-254
BreakUDMConnection	10-261
ChangeProperty	10-262
Delete	10-269
GetBodyNamesByPosition	10-270
GetEdgeByPosition	10-271
GetEdgeIdsFromFace	10-273
GetEdgeIdsFromObject	10-274
GetFaceArea	10-274
GetFaceCenter	10-275

GetFaceByPosition	10-276
GetFaceIDs	10-277
GetGeometryModelerMode	10-278
GetModelBoundingBox	10-279
GetObjectIDByName	10-279
GetObjectName	10-280
GetObjectNameByFaceID	10-281
GetObjectsByMaterial	10-281
GetObjectsInGroup	10-282
GetMatchedObjectName	10-283
GetModelUnits	10-284
GetNumObjects	10-284
GetSelections [Model Editor]	10-285
GetPosition	10-285
GetVertexIDsFromEdge	10-286
GetVertexIDsFromFace	10-287
GetVertexIDsFromObject	10-287
GetVertexPosition	10-288
OpenExternalEditor	10-289
PageSetup	10-290
RenamePart	10-292
11 - Output Variable Script Commands	11-1
CreateOutputVariable (Maxwell)	11-1
DeleteOutputVariable	11-3
DoesOutputVariableExist	11-3
EditOutputVariable	11-4
GetOutputVariableValue	11-6
12 - Reporter Editor Script Commands	12-1
AddCartesianLimitLine	12-4
AddCartesianXMarker	12-6

AddCartesianYMarker	12-7
AddDeltaMarker	12-8
AddMarker	12-9
AddNote	12-10
AddTraceCharacteristics	12-12
AddTraces	12-13
ChangeProperty[ReportSetup]	12-16
ClearAllMarkers	12-20
ClearAllTraceCharacteristics	12-20
CopyTracesData	12-21
CopyReportsData	12-22
CopyReportDefinitions	12-23
CopyTraceDefinitions	12-23
CreateReportFromFile	12-24
CreateReport (Maxwell)	12-25
CreateReportFromTemplate	12-35
CreateReportOfAllQuantities	12-36
DeleteMarker	12-37
DeleteAllReports	12-38
DeleteReports	12-39
DeleteTraces	12-39
ExportImageToFile [Reporter]	12-41
ExportPlotImageToFile	12-42
ExportReport	12-43
ExportToFile	12-45
ExportToFile [Reporter]	12-46
ExportMarkerTable	12-47
FFTONReport	12-48
GetAllReportNames	12-49
GetAllCategories	12-49

GetAllQuantities	12-50
GetAvailableDisplayTypes	12-51
GetAvailableReportTypes	12-52
GetAvailableSolutions	12-53
GetDataExpressions	12-53
GetDataUnits	12-54
GetDesignVariableNames	12-55
GetDesignVariableUnits	12-56
GetDesignVariableValue	12-57
GetDesignVariationKey	12-58
GetDisplayType (Maxwell)	12-59
GetImagDataValues	12-60
GetPerQuantityPrimarySweepValues	12-61
GetPropertyValue	12-62
GetRealDataValues	12-64
GetReportTraceNames	12-65
GetSolutionContexts	12-65
GetSolutionDataPerVariation	12-66
GetSweepNames	12-69
GetSweepUnits	12-70
GetSweepValues	12-71
GroupPlotCurvesByGroupingStrategy	12-72
ImportIntoReport	12-72
IsDataComplex	12-73
IsPerQuantityPrimarySweep	12-74
MovePlotCurvesToGroup	12-75
MovePlotCurvesToNewGroup	12-76
PasteReports	12-77
PasteTraces	12-78
Release Data	12-78

RenameReport	12-79
RenameTrace	12-80
ResetPlotSettings	12-81
SavePlotSettingsAsDefault	12-82
UpdateAllReports	12-82
UpdateReports	12-83
UpdateTraces	12-84
UpdateTracesContextAndSweeps	12-86
UnGroupPlotCurvesInGroup	12-89
13 - Boundary and Excitation Module Script Commands in Maxwell	13-1
General Commands Recognized by the Boundary/Excitations Module	13-2
AddAssignmentToBoundary	13-3
DeleteAllBoundaries	13-4
DeleteAllExcitations	13-5
DeleteBoundaries	13-5
GetBoundaryAssignment	13-6
GetBoundaries	13-7
GetBoundariesOfType	13-7
GetCoreLossEffect	13-8
GetCoreLossEffectOnField	13-9
GetDisplacementCurrent	13-9
GetEddyEffect	13-9
GetExcitations	13-10
GetExcitationsOfType	13-11
GetNumBoundaries	13-11
GetNumBoundariesOfType	13-12
GetNumExcitations	13-13
GetNumExcitationsOfType	13-13
ReassignBoundary	13-14
RemoveAssignmentFromBoundary	13-15

RenameBoundary	13-16
ReprioritizeBoundaries	13-17
Script Commands for Creating and Modifying Boundaries in Maxwell	13-18
AssignCylindricalHField	13-19
AssignDependent (2D)	13-20
AssignDependent (3D)	13-21
AssignFluxTangential	13-23
AssignImpedance (Maxwell)	13-23
AssignIndependent (Maxwell)	13-24
AssignInsulating	13-26
AssignRadiation (Maxwell)	13-26
AssignResistiveSheet	13-27
For 3D Magnetostatic, Eddy Current, and Transient designs	13-28
AssignSymmetry (Maxwell)	13-29
AssignTangentialHField	13-29
AssignThinLayer	13-30
AssignTouching	13-33
AssignVectorPotential	13-34
AssignZeroTangentialHField	13-36
EditCylindricalHField	13-36
EditDependent (Maxwell)	13-36
EditFluxTangential	13-37
EditImpedance (Maxwell)	13-37
EditIndependent (Maxwell)	13-39
EditInsulating	13-39
EditRadiation (Maxwell)	13-39
EditResistiveSheet	13-40
EditSymmetry	13-42
EditTangentialHField	13-43
EditThinLayer	13-43

EditTouching	13-45
EditVectorPotential	13-46
EditZeroTangentialHField	13-47
Script Commands for Creating and Modifying Excitations in Maxwell	13-48
AddTerminalsToWinding	13-49
AssignCharge	13-50
AssignCoilGroup	13-50
AssignCoilTerminal	13-52
AssignCoilTerminalGroup	13-52
AssignCurrent	13-54
AssignCurrentDensity	13-55
AssignCurrentDensityGroup	13-56
AssignCurrentDensityTerminal	13-56
AssignCurrentDensityTerminalGroup	13-57
AssignCurrentGroup	13-57
AssignFloating	13-57
AssignSink (Maxwell)	13-58
AssignVoltage (Maxwell)	13-58
AssignVoltageAPhi	13-59
AssignVoltageDrop	13-60
AssignVoltageDropGroup	13-61
AssignVoltageGroup	13-61
AssignVolumeChargeDensity	13-61
AssignWindingGroup	13-62
EditCharge	13-63
EditCoilTerminal	13-63
EditCurrent	13-63
EditCurrentDensity	13-65
EditCurrentDensityTerminal	13-65
EditExternalCircuit	13-65

EditFloating	13-66
EditSink	13-66
EditVoltage	13-66
EditVoltageAPhi	13-68
EditVoltageDrop	13-69
EditVolumeChargeDensity	13-69
EditWindingGroup	13-69
SetCoreLoss	13-70
SetEddyEffect	13-70
SetMagnetizationCompute	13-71
SetMinimumTimeStep	13-73
ShowWindow	13-73
SetPermeabilityComputation	13-75
14 - Mesh Operations Module Script Commands	14-1
General Commands Recognized by the Mesh Operations Module	14-2
DeleteOp	14-2
GetOperationNames	14-3
RenameOp	14-3
Script Commands for Creating and Modifying Mesh Operations	14-4
AssignCylindricalGapOp	14-5
AssignDensityControlOp	14-8
AssignEdgeCutLayerOp	14-11
AssignLengthOp	14-12
AssignModelResolutionOp	14-15
AssignSkinDepthLayerSetting	14-17
AssignSkinDepthOp	14-18
AssignTrueSurfOp	14-19
EditCylindricalGapOp	14-22
EditDensityControlOp	14-24
EditEdgeCutLayerOp	14-26

EditLengthOp	14-27
EditModelResolutionOp	14-30
EditSkinDepthOp	14-31
EditTrueSurfOp	14-33
InitialMeshSettings	14-35
15 - Analysis Setup Module Script Commands	15-1
ClearLinkedData (Module)	15-2
CopySetup	15-3
DeleteSetups	15-3
EditSetup [Maxwell]	15-4
VB Example: 3D ElectricTransient EditSetup:	15-8
VB Example: 3D Transient EditSetup:	15-8
VB Example: 3D ACConduction EditSetup:	15-10
VB Example: 2D DCConduction EditSetup:	15-10
VB Example: 2D Transient EditSetup:	15-11
Python Example: Transient EditSetup:	15-12
ExportCircuit	15-13
ExportIcepak	15-17
ExportSolnData	15-18
GetSetupCount	15-19
GetSetups	15-20
GetSweepCount	15-21
GetSweeps	15-21
InsertSetup [Maxwell]	15-22
Example:2D Magnetostatic solve setup	15-29
Python Example: 3D Eddy Current solve setup:	15-30
Python Example: 2D Eddy Current solve setup with two-way thermal coupling	15-32
VB Example:3D Eddy Current solve setup:	15-34
Example: 3D Transient solve setup:	15-35
Example: 2D Transient solve setup	15-37

Example: 3D Electrostatic solve setup	15-38
Example: 2D Electrostatic solve setup:	15-38
Example:3D DC Conduction solve setup	15-38
Example: 2D DC Conduction solve setup:	15-39
Example: 3D ElectroDC Conduction solve setup	15-39
Example: 3D ElectricTransient solve setup	15-40
VB Example: 3D AC Conduction solve setup	15-40
Python Example:3D AC Conduction solve setup	15-41
Example: 2D AC Conduction solve setup:	15-42
PasteSetup	15-42
PasteSweep	15-43
RenameSetup	15-43
ResetAllToTimeZero	15-44
ResetSetupToTimeZero	15-44
RevertAllToInitial	15-45
RevertAllToInitialTemperature	15-45
RevertSetupToInitial	15-46
RevertSetupToInitialTemperature	15-47
16 - Optimetrics Module Script Commands	16-1
General Commands Recognized by the Optimetrics Module	16-6
CopySetup	16-7
DeleteSetups [Optimetrics]	16-8
DistributedAnalyzeSetup	16-8
ExportDXConfigFile	16-9
ExportOptimetricsProfile	16-10
ExportOptimetricsResult	16-11
ExportParametricResults	16-12
GetSetupNames [Optimetrics]	16-13
GetSetupNamesByType [Optimetrics]	16-14
ImportSetup	16-14

PasteSetup [Optimetrics]	16-16
RenameSetup [Optimetrics]	16-16
SolveSetup [Optimetrics]	16-17
SolveAllSetup	16-18
Parametric Script Commands	16-19
EditSetup [Parametric]	16-19
ExportParametricSetupTable	16-20
GenerateVariationData [Parametric]	16-21
InsertSetup [Parametric]	16-21
Optimization Script Commands	16-29
EditSetup [Optimization]	16-29
InsertSetup [Optimization]	16-35
InsertSetup for System Coupling	16-43
InitializeSystemCoupling	16-47
Sensitivity Script Commands	16-47
EditSetup [Sensitivity]	16-47
InsertSetup [Sensitivity]	16-55
Statistical Script Commands	16-59
EditSetup [Statistical]	16-59
InsertSetup [Statistical]	16-62
17 - Solutions Module Script Commands	17-1
ExportLayoutLayerGridField	17-2
ExportLayoutViaCurrent	17-5
ExportLayoutViaCurrentDensity	17-6
DeleteSolutionVariation	17-7
ExportNetworkData	17-9
ExportTransientData	17-11
FFTONReport	17-12
GetAdaptiveSettings	17-13
GetAllSourceMagnitudes	17-14

GetAllSourceModes	17-14
GetAllSourcePhases	17-15
GetAllSources	17-16
GetAntennaParameters	17-16
GetFieldType	17-17
GetIncludePortPostProcessing	17-18
GetLayerAndNetOfMeshBody	17-18
GetMeshBodiesOnLayerAndNet	17-21
GetMultipactionBreakdown	17-23
GetNetworkDataSolution	17-24
GetNetworkDataSolutionDefinition	17-25
GetNetworkPostprocSetup	17-25
GetTerminalExcitationType	17-26
GetTransientSolveTimes	17-27
GetValidSolutionList	17-27
TDROnReport	17-28
18 - Field Overlays Module Script Commands	18-1
AddMarkerToPlot	18-2
AddMarker[Fields Reporter]	18-3
AddNamedExpression	18-4
CalcOp	18-6
CalcStack	18-6
CalculatorRead	18-7
CalculatorWrite	18-8
ChangeGeomSettings	18-12
ClcEval	18-13
ClcMaterial	18-17
ClcMaterialValue	18-18
ClearAllMarkers[Fields Reporter]	18-18
ClearAllNamedExpr	18-19

CopyNamedExprToStack	18-20
CreateFieldPlot	18-21
Maxwell Field Line Trace Plot Examples	18-38
DeleteFieldPlot	18-46
DeleteMarker[Fields Reporter]	18-47
DeleteNamedExpr	18-47
DeleteUneditablePlot	18-48
DoesNamedExpressionExists	18-49
EditSurfaceMeshSummaryData	18-49
EnterComplex	18-53
EnterComplexVector	18-54
EnterCoord	18-55
EnterEdge	18-56
EnterLine	18-57
EnterOutputVar	18-57
EnterPoint	18-58
EnterQty	18-59
EnterScalar	18-59
EnterScalarFunc	18-60
EnterSurf	18-61
EnterVector	18-61
EnterVectorFunc	18-62
EnterVol	18-63
ExportFieldPlot	18-63
ExportMarkerTable	18-64
ExportOnGrid [Field Overlays]	18-65
ExportPlotImageToFile	18-68
ExportPlotImageWithViewToFile [Reporter]	18-69
ExportSurfaceMeshSummary	18-70
ExportToFile	18-72

GetFieldFolderNames	18-73
GetFieldPlotNames	18-74
GetFieldPlotQuantityName	18-74
GetMeshPlotNames	18-75
ModifyFieldPlot	18-76
GetTopEntryValue	18-78
HidePolarPlot	18-79
HideRadiatedPlotOverlay	18-80
LoadNamedExpressions	18-81
ModifyFieldPlot	18-82
ModifyInceptionParameters	18-84
ReassignFieldPlot	18-86
RenameFieldPlot	18-89
RenamePlotFolder	18-90
SaveFieldsPlots	18-90
SetFieldPlotSettings	18-91
SaveNamedExpressions	18-94
SetFieldPlotSettings	18-95
SetPlotFolderSettings	18-98
SetPlotsViewSolutionContext	18-103
UpdateAllFieldsPlots	18-104
UpdateQuantityFieldsPlots	18-105
19 - Fields Calculator Script Commands	19-1
AddNamedExpression	19-2
Syntax for Algebraic Option	19-4
CalcOp	19-5
CalcRead(deprecated)	19-6
CalculatorRead	19-7
CalcStack	19-8
CalculatorWrite	19-8

ChangeGeomSettings	19-13
ClcEval	19-13
ClcMaterial	19-17
ClearAllNamedExpr	19-18
CopyNamedExprToStack	19-18
DeleteNamedExpr	19-19
EnterComplex	19-20
EnterComplexVector	19-20
EnterLine	19-21
EnterPoint	19-22
EnterQty	19-23
EnterScalar	19-24
EnterScalarFunc	19-24
EnterSurf	19-25
EnterVector	19-26
EnterVectorFunc	19-26
EnterVol	19-27
ExportOnGrid [Fields Calculator]	19-28
ExportToFile [Fields Calculator]	19-31
GetTopEntryValue	19-33
LoadNamedExpressions	19-34
SaveNamedExpressions	19-35
20 - Motion Setup Script Commands	A
DeleteMotionSetup	A
ReassignMoving	B
Commands to Create and Edit the Band	B
AssignBand	B
EditMotionSetup	E
GetMotionSetupNames	G
Other Commands Recognized By the ModelSetup Module	G

GetSymmetryMultiplier	H
SetSymmetryMultiplier	H
21 - Parameter Setup Script Commands	21-1
General Parameter Setup Script Commands	21-1
Commands to Create and Edit Parameters	21-2
AddReduceOp	21-2
AssignForce	21-4
[Beta] AssignLayoutForce	21-5
AssignMatrix	21-7
AssignTorque	21-11
DeleteAllParameters	21-13
DeleteParameters	21-13
DeleteReduceMatrix	21-14
DeleteReduceOp	21-15
EditForce	21-16
[Beta] EditLayoutForce	21-16
EditMatrix	21-18
EditReduceOp	21-18
EditTorque	21-19
ReassignParameter	21-20
RenameParameter	21-20
RenameReduceMatrix	21-21
RenameReduceOp	21-21
22 - User Defined Document Script Commands	22-1
AddDocument	22-1
DeleteAllDocuments	22-4
DeleteDocument	22-5
EditDocument	22-5
GetDocumentDefinitionNames	22-7
GetDocumentNames	22-8

RenameDocument	22-8
SaveHtmlDocumentAs	22-9
SavePdfDocumentAs	22-10
UpdateAllDocuments	22-10
UpdateDocument	22-11
ViewHtmlDocument	22-12
ViewPdfDocument	22-13
Explication of a Sample UDD Script	22-13
Example Python Script: Defining a Document	22-15
23 - User Defined Solutions Commands	23-1
CreateUserDefinedSolution	23-1
DeleteUserDefinedSolutions	23-3
EditUserDefinedSolution	23-4
24 - Network Data Explorer Script Commands	24-1
AddDiffPair	24-4
Cascade (SPISim)	24-5
ClearDiffPairs	24-6
Clone	24-7
Close	24-8
Combine (SPISim)	24-9
Deembed (SPISim)	24-10
DeembedBack (SPISim)	24-11
DeembedFront (SPISim)	24-13
DisableDiffPairs	24-14
EnableDiffPairs	24-15
ExportCitiFile	24-16
ExportMatlab	24-18
ExportSpreadsheet	24-20
ExportTouchstone	24-22
ExportTouchstone2	24-24

Extract (SPISim)	24-27
GetFrequencies	24-28
GetFrequencyCount	24-29
GetName	24-30
GetPortCount	24-30
GetPortNumber	24-31
GetPostProcSettings	24-32
GetVariation	24-33
HasSameData	24-34
LoadSolution	24-36
Open	24-37
Rename (SPISim)	24-38
Renormalize (SPISim)	24-39
Reorder	24-41
Reorder (SPISim)	24-42
Reset	24-43
SetAllPortImpedances	24-44
SetPortDeembedDistance	24-45
SetPortImpedance	24-46
SetPostProcSettings	24-47
Smooth	24-48
Stretch (SPISim)	24-49
Terminate	24-51
25 - ComInstance Script Commands	25-1
Callback Scripting Using ComInstance Object	25-1
ComInstance Functions	25-2
GetComponentName	25-2
GetInstanceID [Component Instance]	25-3
GetInstanceName [Component Instance]	25-4
GetParentDesign	25-4

GetPropHost	25-5
GetPropServerName	25-5
26 - Schematic Scripting	26-1
Method Format	26-1
Editor Scripting IDs	26-4
Format of IDs for different schematic objects:	26-4
Format for Components	26-4
Create Method List	26-5
CreateArc (Schematic Editor)	26-7
CreateCircle (Schematic Editor)	26-10
CreateComponent (Schematic Editor)	26-13
CreateGlobalPort (Schematic Editor)	26-15
CreateGround (Schematic Editor)	26-19
CreateLine (Schematic Editor)	26-22
CreatePagePort (Schematic Editor)	26-25
CreateIPort (Schematic Editor)	26-28
CreatePolygon (Schematic Editor)	26-31
CreateRectangle (Schematic Editor)	26-34
CreateText (Schematic Editor)	26-38
CreateWire (Schematic Editor)	26-42
General Method List	26-45
Activate (Schematic Editor)	26-47
AddPinGrounds (Schematic Editor)	26-48
AddPinIPorts (Schematic Editor)	26-51
AddPinPageConnectors (Schematic Editor)	26-52
AlignHorizontal (Schematic Editor)	26-53
AlignVertical (Schematic Editor)	26-54
BringToFront (Schematic Editor)	26-55
CloseEditor (Schematic Editor)	26-57
Copy (Schematic Editor)	26-57

CopyData [Schematic Editor]	26-58
CopySubdesign [Schematic Editor]	26-59
CreatePage (Schematic Editor)	26-60
Cut (Schematic Editor)	26-61
DeactivateOpen (Schematic Editor)	26-62
DeactivateShort (Schematic Editor)	26-63
Delete (Schematic Editor)	26-64
DeletePage (Schematic Editor)	26-65
ElectricRuleCheck (Schematic Editor)	26-66
ExportImage (Schematic Editor)	26-67
ExportNetlist	26-68
FindElements (Schematic Editor)	26-69
FitToBorder [Schematic Editor]	26-71
GridSetup (Schematic Editor)	26-72
FlipHorizontal (Schematic Editor)	26-75
FlipVertical (Schematic Editor)	26-76
Move (Schematic Editor)	26-78
NameNets (Schematic Editor)	26-79
PageBorders (Schematic Editor)	26-81
Pan (Schematic Editor)	26-81
Paste (Schematic Editor)	26-83
PasteData [Schematic Editor]	26-84
PasteDesign (Schematic Editor)	26-85
PushExcitations	26-87
Rotate (Schematic Editor)	26-90
SelectAll (Schematic Editor)	26-92
SelectPage (Schematic Editor)	26-93
SetPageData [Schematic Editor]	26-94
SendToBack	26-97
ShowVariableBlock (Schematic Editor)	26-98

SortComponents (Schematic Editor)	26-98
Wire (Schematic Editor)	26-99
ZoomArea (Schematic Editor)	26-100
ZoomIn (Schematic Editor)	26-101
ZoomOut (Schematic Editor)	26-102
ZoomPrevious (Schematic Editor)	26-103
ZoomToFit (Schematic Editor)	26-104
Property Method List	26-105
ChangeProperty (Schematic Editor)	26-105
GetEvaluatedPropertyValue (Schematic Editor)	26-106
GetProperties (Schematic Editor)	26-107
GetPropertyAttribute [Schematic Editor]	26-109
GetPropertyValue (Schematic Editor)	26-110
SetPropertyValue (Schematic Editor)	26-111
Information Method List	26-112
GetAllPorts	26-113
GetCompInstanceFromRefDes (Schematic Editor)	26-113
GetComponentInfo (Schematic Editor)	26-114
GetComponentPins (Schematic Editor)	26-116
GetComponentPinInfo (Schematic Editor)	26-117
GetComponentPinLocation [Schematic Editor]	26-118
GetEditorName (Schematic Editor)	26-119
GetNetConnections (Schematic Editor)	26-120
GetNumPages [Schematic Editor]	26-121
GetPortInfo (Schematic Editor)	26-122
GetSelections (Schematic Editor)	26-123
GetSignals (Schematic Editor)	26-124
GetWireConnections (Schematic Editor)	26-126
GetWireInfo (Schematic Editor)	26-128
GetWireSegments (Schematic Editor)	26-130

27 - Definition Editor Script Commands	27-1
Symbol Editor Scripts	27-1
AddLevel (Symbol Editor)	27-3
AlignHorizontal (Symbol Editor)	27-4
AlignVertical (Symbol Editor)	27-5
BringToFront (Symbol Editor)	27-6
ChangeProperty (Symbol Editor)	27-6
CloseEditor (Symbol Editor)	27-19
Copy (Symbol Editor)	27-20
CreateArc (Symbol Editor)	27-20
CreateCircle (Symbol Editor)	27-21
CreateImage (Symbol Editor)	27-22
CreateLine (Symbol Editor)	27-23
CreatePin (Symbol Editor)	27-24
CreatePolygon (Symbol Editor)	27-25
CreateRectangle (Symbol Editor)	27-26
CreateText (Symbol Editor)	27-27
Cut (Symbol Editor)	27-27
Delete (Symbol Editor)	27-28
DisplayPinNames (Symbol Editor)	27-29
ExportFile (Symbol Editor)	27-29
FlipHorizontal (Symbol Editor)	27-30
FlipVertical (Symbol Editor)	27-31
GetProperties (Symbol Editor)	27-31
GetPropertyValue (Symbol Editor)	27-32
GetVisibleLevels (Symbol Editor)	27-32
GridSetup (Symbol Editor)	27-33
Move (Symbol Editor)	27-34
Pan (Symbol Editor)	27-35
Paste (Symbol Editor)	27-35

Redo (Symbol Editor)	27-36
RemoveLevels (Symbol Editor)	27-37
RemovePort (Symbol Editor)	27-37
Rotate (Symbol Editor)	27-38
Save (Symbol Editor)	27-39
SelectAll (Symbol Editor)	27-39
SendToBack (Symbol Editor)	27-40
SetActiveLevel (Symbol Editor)	27-40
SetInitialLevels (Symbol Editor)	27-41
SetPropertyValue (Symbol Editor)	27-42
SetVisibleLevels (Symbol Editor)	27-42
ToggleLevel (Symbol Editor)	27-43
Undo (Symbol Editor)	27-44
ZoomArea (Symbol Editor)	27-45
ZoomIn (Symbol Editor)	27-46
ZoomOut (Symbol Editor)	27-46
ZoomPrevious (Symbol Editor)	27-47
ZoomToFit (Symbol Editor)	27-47
28 - Core Global Script Context Commands	28-1
AddErrorMessage	28-1
AddFatalMessage	28-2
AddInfoMessage	28-2
AddWarningMessage	28-3
LogDebug	28-3
LogError	28-4
29 - Example Scripts	29-1
VBScript Example Scripts	29-1
Variable Helix Script	29-1
Data Export Script	29-5
IronPython Example Scripts	29-8

BH Coordinates Python Script 29-8

 Script Contents 29-9

 Posco_BH_Curve.tab Contents 29-11

Equation Based Curve Python Script 29-13

Index Index-1

1 - Introduction to Scripting

Using scripts is a fast, effective way to accomplish tasks you want to repeat. When you execute a script, the commands in the script are performed in the order in which they appear.

Electronics Desktop can record scripts in VBScript or IronPython, and can run external scripts written in VBScript, IronPython, CPython, or JavaScript. Additionally, it contains an IronPython command shell for executing scripts.

When running Ansys Electronics Desktop from the command line, scripts can be written in any language that provides Microsoft COM methods.

The following sections contain more information about scripting:

- [Scripting Help Conventions](#) – explains the layout of the scripting help.
- [Introduction to VBScript](#) – provides a broad overview of VBScript
- [Introduction to IronPython](#) – provides a broad overview of IronPython.
- [Introduction to C-Python](#) – provides guidance on using C-Python for Ansys Electronics Desktop scripts.
- [Ansys Electronics Desktop Scripting](#) – details instructions and tips for running, recording, and working with scripts in Electronics Desktop.
- [PyAEDT](#) (Beta) – a Python library that interacts directly with the AEDT API to make scripting simpler for the end user.

Scripting Help Conventions

The majority of this guide lists individual script commands using the following format.

[ScriptName]

[Description of script use.]

UI Access	[UI commands corresponding to the script command, if any.]
Parameters	[List of arguments taken by the script command, if any. Includes argument types and brief descriptions.]
Return Value	[The script's return value, if any.]

Python Syntax	[Correct syntax for the command in Python. Arguments are enclosed in angle brackets (<>).]
Python Example	[Sample script]

VB Syntax	[Correct syntax for the command in VBScript. Arguments are enclosed in angle brackets (<>)]
VB Example	[Sample script]

Variable Types

The following data types are used throughout the help:

- **<string>** – use within quotation marks.
- **<bool>** – boolean value; should be set to either True or False.
- **<int>** – an integer. For example, 1.
- **<double>** – a double precision value. For example, 1.2.
- **<array>** – in VBScript, an array; in IronPython, a list contained in square brackets.
- **<value>** – can be an integer, string, or VBScript variable, depending on context.

Introduction to VBScript

Ansys Electronics Desktop can use Microsoft® Visual Basic® Scripting (VBScript) to record macros. VBScript is based on the Microsoft Visual Basic programming language.

This chapter provides an overview of key VBScript components.

[Simple and Composite Names](#)

[VBScript Variables](#)

[VBScript Operators](#)

[Controlling Program Execution](#)

[Looping Through Code](#)

[VBScript Procedures](#)

[Converting Between Data Types](#)

[Including Scripts](#)

[Aborting Scripts](#)

[Interacting with a Script](#)

[Recommended VBScript References](#)

[Sample HFSS Script](#)

[Sample Circuit Script](#)

[Sample Q3D Script](#)

For more details about VBScript, please see the *Recommended VBScript References* section at the end of this chapter.

Simple and Composite Names

Components, symbols, footprints, models, and padstacks possess either "simple" names or "composite" names. Composite names are used to distinguish items from libraries that may possess the same simple name. A composite name is created by combining an item's library name with its simple name. Composite names for definitions are unique, but simple names are not.

- Composite names are used by definition manager script commands to uniquely identify script definitions.
- Materials and scripts do not have composite names, so project definitions for these items must possess a unique simple name.
- The format of a composite name is `LibraryName:SimpleDefinitionName`. For example, the composite name for the component "CAP_in" the system library Nexxim Circuit Elements\Capacitors is "Nexxim Circuit Elements\Capacitors:CAP_in."
- The format of a composite name in a project is `OriginLibraryName:SimpleDefinitionName`. For example, the composite name for the project component "CAP_" that was originally from the system library Nexxim Circuit Elements\Capacitors is "Nexxim Circuit Elements\Capacitors:CAP_".
- Not all definitions in a project have a library of origin. Newly added definitions do not have a library of origin, and project definitions whose names are changed do not have a library of origin (even if they did before the name change). As a result, the composite name for items without a library of origin is the item's simple name itself. For example, the composite name for the project component "CAP_" that came from a system library and was renamed to "MyCAP_" is "MyCAP_".

To construct a composite name, select **Tools > Edit Configured Libraries > Components** to open the **Edit Libraries** dialog box. The subnames used to construct a composite name can be found in the **Name** and **Origin** columns that correspond to a particular component. The **Origin** column contains the library portion of the composite name, while the **Name** column contains the simple portion of the composite name.

VBScript Variables

A VBScript variable is a placeholder representing information that may change during the time your script is running. Variables are useful because they let you assign a short and easy to remember name to each piece of data you plan to use. Use a variable name in a script to view or modify its value.

Declaring Variables

To declare variables explicitly in a script, use the `Dim`, `Public`, or `Private` statements. For example:

```
Dim box_xsize
```

After declaring a variable, you can assign information to it. For example:

```
box_xsize = "3mm"
```

You can declare multiple variables by separating each variable name with a comma. For example:

```
Dim Top, Bottom, Left, Right
```

You can also declare a variable implicitly by simply using its name in your script. Doing so is not generally a good practice because you could misspell the variable name in one or more places, causing unexpected results when your script is run. For that reason, the **Option Explicit** statement is available to require explicit declaration of all variables. The **Option Explicit** statement should be the first statement in your script.

Declaring Variables in Python

Python does not require you to declare variables before you assign a value to them.

You can directly assign information to it. For example:

```
box_xsize = "3mm"
```

Variable Naming Conventions

You should use names that are short but intuitive and easy to remember. Use the following conventions for naming variables in VBScript:

- Begin with an alphabetic character.
- Cannot contain an embedded period.
- Must not exceed 255 characters.
- Must be unique in the scope in which it is declared.
- Do not use VBScript keywords.

Scope and Lifetime of Variables

Variables at the script level are available to all procedures within the script. At the procedure level, variables are available only within the procedure. It has local scope and is a procedure-level variable.

The lifetime of a variable depends on how long it exists. The script-level variables exist from declaration until the end of the script. A procedure-level variable exists only as long as you are in the procedure and is destroyed when the procedure exits.

Array Variables

Create an array variable when you want to assign more than one related value to a single variable. An array variable contains a series of values. For example:

```
Dim Primitives(2)
```

All arrays in VBScript are zero-based, so the array above actually contains 3 elements. You assign data to each of the array's elements using an index into the array. Data can be assigned to the elements of an array as follows:

```
Primitives(0) = "Box1"
```

```
Primitives(1) = "Cone1"
```

```
Primitives(2) = "Cylinder1"
```

Similarly, the data can be retrieved from any element using an index into a particular array element. For example:

```
one_prim = Primitives(1)
```

You can also use the Array function to assign an array of elements to a variable. For example:

```
Dim Primitives
Primitives = Array ("Box1", "cone1", "Cylinder1")
```

Note:

When using the Array function, do not use parentheses on the variable when it is declared. For example, use Dim myarray, not Dim myarray().

If you do not know the size of the array at declaration or the size changes during the time your script is running, you can use dynamic arrays. They are declared without size or number of dimensions inside the parentheses. For example:

```
Dim FirstArray()
ReDim SecondArray()
```

To use a dynamic array, you must subsequently use **ReDim** to determine the number of dimensions and the size of each dimension. You can also use the **Preserve** keyword to preserve the contents of the array as the resizing takes place.

```
ReDim FirstArray(25)
ReDim Preserve FirstArray(30)
```

VBScript Operators

VBScript provides operators, which are grouped into these categories: arithmetic operators, comparison operators, and logical operators.

Please see the online *VBScript User's Guide* for more details.

Operator Precedence

When several operations occur in an expression, each part is evaluated and resolved in a pre-determined order, called operator precedence. You can use parentheses to override the order of precedence and force some parts of an expression to be evaluated before others. Operations within parentheses are always performed before those outside the parentheses. Within parentheses, however, standard operator precedence is maintained.

When an expression contains operators from more than one category, they are evaluated in the following order:

1. [Arithmetic Operators](#)
2. [String Concatenation Operator](#) (&)
3. [Comparison Operators](#)
4. [Logical Operators](#)

Arithmetic operators within a single expression are evaluated in the following order of precedence:

1. Exponentiation (^)
2. Multiplication and Division (*,/): These two operators are of equal precedence and are evaluated in the left-to-right order in which they appear within the expression.
3. Integer Division (\)
4. Modulus Arithmetic (Mod)
5. Addition and Subtraction (+,-): These two operators are of equal precedence and are evaluated in the order in which they appear within the expression.

If the same arithmetic operator appears multiple times within a single expression, they are evaluated in the left-to-right order in which they appear.

Comparison operators all have equal precedence and are evaluated in the left-to-right order in which they appear within the expression.

Logical operators all have equal precedence and are evaluated in the left-to-right order in which they appear within the expression.

Arithmetic Operators

Following is a list of VBScript's arithmetic operators:

Symbol	Description
^	Exponentiation
-	Unary negation
*	Multiplication
/	Division
\	Integer division
Mod	Modulus arithmetic
+	Addition
-	Subtraction

Note:

For the order of precedence for these operators, see [Operator Precedence](#).

Comparison Operators

Following is a list of VBScript's comparison operators:

Symbol	Description
=	Equality
<>	Inequality
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to
Is	Object equivalence

Note:

All comparison operators have the same precedence. When multiple comparisons exist in a single expression, evaluate them in the left-to-right order in which they appear.

Logical Operators

Following is a list of VBScript's logical operators:

Symbol	Description
Not	Logical negation
And	Logical conjunction
Or	Logical disjunction
Xor	Logical exclusion
Eqv	Logical equivalence
Imp	Logical implication

Note:

All logical operators have the same precedence. When multiple logical operators exist in a single expression, evaluate them in the left-to-right order in which they appear.

Controlling Program Execution

You can use conditional statements to control the flow of a script. There are two types of conditional statements in VBScript:

- [If...Then...Else](#)
- [Select Case](#)

Using If...Then...Else

Following is an example that demonstrates the If...Then...Else conditional statement:

```
If obj = "Box1" Then
    <statements to execute>
ElseIf obj = "Cylinder1" Then
    <statements to execute>
Else
    <statements to execute>
End If
```

Using Select Case

Following is an example that demonstrates the Select Case conditional statement:

```
Select Case primitive_name
    Case "Box1"
        <statements to execute>
    Case "Cylinder1"
        <statements to execute>
    Case Else
        <statements to execute>
End Select
```

Looping Through Code

Looping allows you to run a group of statements repeatedly. There are two types of loops:

- [For...Next](#): Uses a counter to run statements a specified number of times.
- [Do...Loop](#): Loops while or until a condition is True.

When using conditional statements that test for zero voltage/current, it is important to note that a real voltage or current should not be trusted to be exactly zero, even when it should be.

Typically, the voltage or current is often on the order of 'epsilon' (1e-16) or smaller; hence, it is nonzero in value.

Using a For...Next Loop

The For...Next type of loop allows you to run a group of statements repeatedly. It uses a counter to run statements a specified number of times. Following is an example that demonstrates the For...Next loop:

```
For variable = start To end
    <statements to execute>
Next
```

You can exit early from a For...Next loop with the Exit For statement.

Using a Do Loop

You can use **Do...Loop** statements to run a block of statements until (or while) a condition is true.

Repeating Statements While a Condition is True

Use the While keyword to check a condition in a Do...Loop statement. The syntax is as follows:

```
Do While condition
    <statements to execute>
Loop
```

Repeating a Statement Until a Condition Becomes True

Following is the syntax:

```
Do Until condition
    <statements to execute>
Loop
```

You can exit early from a loop by using the Exit For statement.

VBScript Procedures

In VBScript, there are two kinds of procedures, [Sub](#) and [Function](#). These procedures are called by name, they can receive arguments, and each performs a specific task with a group of VBScript statements. If there is no argument, then the Sub or Function statement must include an empty set of parentheses.

Function Procedures

A Function returns a value by assigning a value to its name in one or more statements. Following is the syntax of a Function:

```
Function FunctionName([arguments])
```

```
<Function statements>
```

```
End Function
```

Sub Procedures

A Sub procedure is like a function procedure, except that it does not return a value through its name. Following is the syntax of a Sub:

```
Sub ProcedureName([arguments])
```

```
<Procedure statements>
```

```
End Sub
```

Converting Between Data Types

To convert data from one subtype to another, use the following VBScript functions:

CStr	Syntax: CStr(variablename). Converts variablename to a string. For example, it can be used to convert the number 2.5 to the string "2.5".
CBool	Syntax: CBool(variablename). Converts variablename to a boolean. If variablename is 0 or "0", CBool returns False. Otherwise it returns True.
CDBl	Syntax: CDBl(variablename). Converts variablename to a double precision number. For example, it can be used to convert the string "2.5" to the number 2.5.
CInt	Syntax: CInt(variablename). Converts variablename to an integer.

Including Scripts

You can include one script within another using the following command:

```
#include "<scriptfilename>"
```

Where scriptfilename is the full path name to a file that contains script text, or is the name of a script in the project library or script library (listed in the project window under the Definitions/Scripts directory).

The command works for VBScript, JScript, and for the following:

- Scripts in the project library that are run by right-clicking the script icon in the project window and choosing **Run Script**
- Scripts in files that are external are run by choosing **Tools> Run Script**

- Scripts that are specified as callbacks in the **Property** dialog box
- Scripts that are run to draw parameterized footprints in layout

An include command can be placed anywhere in a script, but for readability it is recommended that commands be placed at the beginning of a file. The same script can be included multiple times without error, and circular inclusions will be ignored.

Aborting Scripts

You can abort a script that is running in the desktop simply by pressing the ESC key. Terminating a script in this manner works for each of the following:

- Scripts in the project library that are run by right-clicking the script icon in the project window and choosing **Run Script**.
- Scripts in files that are external can be run by choosing **Tools > Run Script**.
- Scripts that are specified as callbacks in the **Property** dialog box.
- Scripts that are run to draw parameterized footprints in layout.

Interacting with a Script

VBScript provides two functions that enable you to interact with a script while it is running:

the `InputBox` function and the `MsgBox` function.

The `InputBox` function displays a dialog box with an input field. The value that is typed into the input field is returned. For example:

```
Dim users_string  
users_string = InputBox ("text prompt", "title of the pop-up dialog _  
    box", "default text for the input box")
```

The last two arguments to the function are optional.

The `MsgBox` function shows a message and returns a number based on the button the user presses. For example:

```
MsgBox ("message text")
```

Recommended VBScript References

Microsoft Corporation. *VBScript User's Guide*.

Available <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/script56/html/vbstutor.asp>.

Childs, M., Lomax, P., and Petrusha, R. *VBScript in a Nutshell: A Desktop Quick Reference*.

May 2002. O'Reilly & Associates. ISBN: 1-56592-720-6.

Introduction to IronPython

IronPython is an implementation of the Python programming language targeting the .NET runtime. What this means in practical terms is that IronPython uses the Python programming language syntax and standard python libraries and can additionally use .NET classes and objects to give one the best of both worlds. This usage of .NET classes is fairly seamless in that a class defined in a .NET assembly can be used as a base class of a python class.

Scope

Functioning as a tutorial on Python or IronPython is way out of the scope of this document. There are several excellent resources online that do a very good job in that regard. This document only attempts to provide a limited introduction to IronPython as used to script Ansys EM products.

This document is also not a tutorial on the scripting of Ansys EM products. It complements the existing scripting guide (available from a product's Help menu) and provides a pythonic interpretation of that information. The reader might have to refer to either the scripting guide or recorded samples of VBScript to follow some of the sections.

Python compatibility

The version of IronPython in use is **2.7** and built on the .NET framework version 4.0: this version targets **Python 2.7** language compatibility. While most python files will execute under IronPython with no changes, python libraries that make use of extensions written in the C programming language (NumPy or SciPy for instance), are not expected to work under IronPython. In such cases, it might be possible to locate .NET implementation of such libraries or explore the use of IronClad.

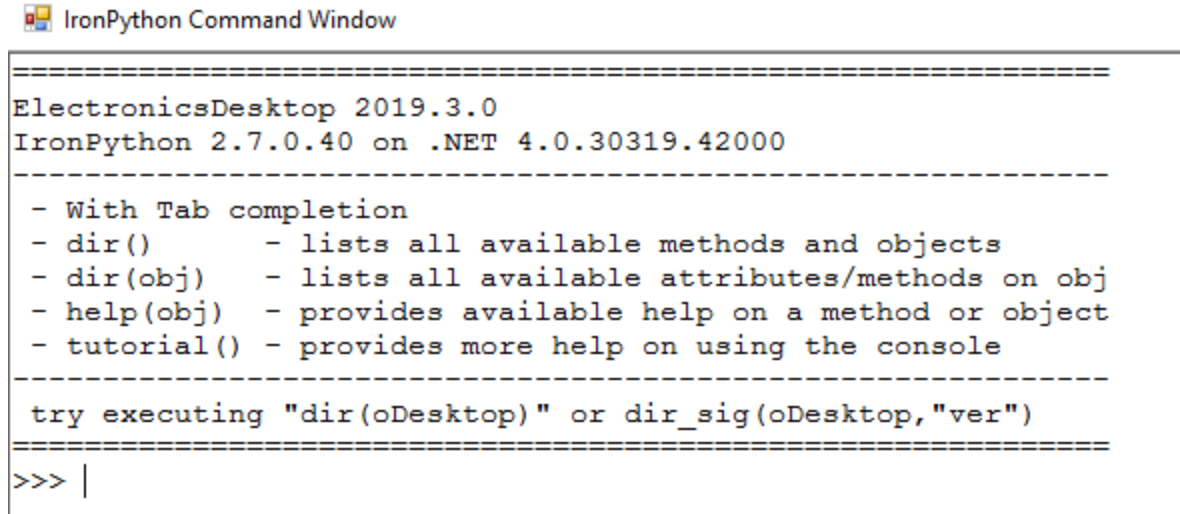
[\(http://code.google.com/p/ironclad/\)](http://code.google.com/p/ironclad/).

Advantages of IronPython

The advantages that IronPython use provides are significant:

- Python has a large eco-system with plenty of supporting libraries, Visual IDEs and debuggers. It is actively developed and enhanced.
- IronPython, in addition, has access to the entire .NET eco system. This allows us, for instance, to create a modern GUI using the **System.Windows.Forms** assembly from IronPython code and call any other .NET assembly for that matter.
- The use of IronPython's technologies enables the ability to interactively script Desktop (feature in development). This allows better discovery of the scripting APIs as well as directly programming to the scripting API in python, a language more tractable and platform independent compared with VBScript.
- The Python syntax of dictionaries is somewhat easier to read and write when supplying arguments to the scripting methods.

This document describes IronPython briefly and then goes on to describe the desktop provided IronPython scripting console and scripting with IronPython. You can open an IronPython Command Window by clicking **Tools > Open Command Window**.



```

IronPython Command Window

=====
ElectronicsDesktop 2019.3.0
IronPython 2.7.0.40 on .NET 4.0.30319.42000
=====
- With Tab completion
- dir()           - lists all available methods and objects
- dir(obj)        - lists all available attributes/methods on obj
- help(obj)       - provides available help on a method or object
- tutorial()      - provides more help on using the console
=====
try executing "dir(oDesktop)" or dir_sig(oDesktop,"ver")
=====
>>> |

```

The document assumes that you know how desktop scripting works using VBScript or JavaScript.

[Introduction to IronPython](#)

[IronPython Mini Cookbook](#)

[Translating Script Commands from VBScript to IronPython](#)

[Scripting Using Iron Python](#)

[Standalone IronPython and Desktop IronPython](#)

[IronPython Examples](#)

[Creating User Defined Primitives and User Defined Models in Python Scripts](#)

Introduction to IronPython

IronPython is an implementation of the Python programming language targeting the .NET runtime. What this means in practical terms is that IronPython uses the Python programming language syntax and standard python libraries and can additionally use .NET classes and objects to give one the best of both worlds. This usage of .NET classes is fairly seamless in that a class defined in a .NET assembly can be used as a base class of a python class.

Scope

Functioning as a tutorial on Python or IronPython is way out of the scope of this document. There are several excellent resources online that do a very good job in that regard. This doc-

ument only attempts to provide a limited introduction to IronPython as used to script Ansys EM products.

This document is also not a tutorial on the scripting of Ansys EM products. It complements the existing scripting guide (available from a product's Help menu) and provides a pythonic interpretation of that information. The reader might have to refer to either the scripting guide or recorded samples of VBScript to follow some of the sections.

Python compatibility

The version of IronPython in use is **2.7** and built on the .NET framework version 4.0: this version targets **Python 2.7** language compatibility. While most python files will execute under IronPython with no changes, python libraries that make use of extensions written in the C programming language (NumPy or SciPy for instance), are not expected to work under IronPython. In such cases, it might be possible to locate .NET implementation of such libraries or explore the use of IronClad.

[\(http://code.google.com/p/ironclad/\)](http://code.google.com/p/ironclad/).

Advantages of IronPython

The advantages that IronPython use provides are significant:

- Python has a large eco-system with plenty of supporting libraries, Visual IDEs and debuggers. It is actively developed and enhanced.
- IronPython, in addition, has access to the entire .NET eco system. This allows us, for instance, to create a modern GUI using the **System.Windows.Forms** assembly from IronPython code and call any other .NET assembly for that matter.
- The use of IronPython's technologies enables the ability to interactively script Desktop (feature in development). This allows better discovery of the scripting APIs as well as directly programming to the scripting API in python, a language more tractable and platform independent compared with VBScript.
- The Python syntax of dictionaries is somewhat easier to read and write when supplying arguments to the scripting methods.

This document describes IronPython briefly and then goes on to describe the desktop provided IronPython scripting console and scripting with IronPython. You can open an IronPython Command Window by clicking **Tools > Open Command Window**.



```

IronPython Command Window

=====
ElectronicsDesktop 2019.3.0
IronPython 2.7.0.40 on .NET 4.0.30319.42000
=====
- With Tab completion
- dir()      - lists all available methods and objects
- dir(obj)   - lists all available attributes/methods on obj
- help(obj)  - provides available help on a method or object
- tutorial() - provides more help on using the console
=====
try executing "dir(oDesktop)" or dir_sig(oDesktop,"ver")
=====
>>> |

```

The document assumes that you know how desktop scripting works using VBScript or JavaScript.

[Introduction to IronPython](#)

[IronPython Mini Cookbook](#)

[Translating Script Commands from VBScript to IronPython](#)

[Scripting Using Iron Python](#)

[Standalone IronPython and Desktop IronPython](#)

[IronPython Examples](#)

[Creating User Defined Primitives and User Defined Models in Python Scripts](#)

IronPython Mini Cookbook

This topic presents simple counterparts between IronPython and VBScript. It does not provide a full tutorial on IronPython syntax. Because IronPython is a Python implementation, you can consult Python documentation for additional information.

Comments

VBScript	IronPython
Comments start with a single quote: ' comment	Comments start with a hash: # comment

Assigning/Creating Variables

VBScript	IronPython
Declare with a Dim: Dim doc Assignment then needs a Set instruction: Set doc = app.GetActiveProject()	No Set syntax. Simply create and assign: doc = app.GetActiveProject()

Create Lists/Arrays

VBScript	IronPython
Declare as array of String with 11 indices, from 0 through 10: Dim myArray(0 to 10) as String myArray(0) = "Hello" myArray(1) = "bye" Declare an array with no size: Dim array2() as String Re-dimension an array once size is known: ReDim array2(0 to 2) as String array2(0) = "this" array2(1) = "also"	Declare an empty array: myEmptyArray = [] Declare an array and initialize it with 5 ints: myInitdArray = [1, 2, 3, 4, 5] Python lists can have items of any type and there is no pre-declaration. Declare an array and init with mixed types: mixed = ["hello", 1, 2, ["nested"]] Append to an array: mixed.append(3.5)

Create Dictionaries/Maps

VBScript	IronPython
Declare with a Dim: Dim dict Use the CreateObject function with ProgID Scripting.Dictionary: Set dict = CreateObject _ ("Scripting.Dictionary") Add items using the object, key, item syntax: dict.Add key, item	An IronPython dictionary is a collection of name value pairs. Just like arrays, there is no restriction on the keys or the values. <u>For purposes of Ansys EM scripting, however, all keys must be strings</u> Delimiters are curly braces. Use a colon between the key and the value. Separate key value pairs with a comma: <pre>myDict = { "a" : 1, "b" : "hello there", "c" : [1, 2, "abc"] }</pre>

Boolean Values

VBScript	IronPython
Boolean literals are in lower case: true false	The first letter is capitalized: True False

Converting Numbers to Strings and Vice Versa

VBScript	IronPython
Use CInt, CDbl, CBool, CLng to convert the string representation to the number representation. Use IsNumber to check before conversion: Dim nStr = "100" Dim n = CInt(nStr) Use CStr to convert a number to its string representation: Dim v, vStr v = 100 vStr = CStr(v)	Use integer() or float() or double() functions to cast a string CONTAINING the string representation of whatever you are casting to: <pre>strInt = "3" intVal = int(strVal) floatVal = float(strVal)</pre> Invoke the str() function with the int/float values as needed. You can alternately use the string formatting method listed below: <pre>strVal = str(42) strVal = str(42.345)</pre>

String Formatting/Concatenation

VBScript	IronPython
String concatenation uses the & operator: <pre>Dim allStr, str1 str1 = " how are you" allStr = "Hello " & " There" & str1</pre> There seems to be no direct string formatting function in VBScript. Using string concatenation or using Replace are the two built-in options: <pre>Dim fmt = "{1} climbs stalk {2}" Dim str = Replace(fmt, "{1}", "jack") str = Replace(str, "{2}", 10)</pre>	If you have two strings, you can always concatenate them using the '+' operator: <pre>str1 = "hello" str2 = "world" str12 = str1 + " " + str2</pre> If you have different types (for instance a string and an int), you must use the string formatting commands. When formatting multiple arguments, they must be entered as a tuple (item1, item2,): <pre>num = 10 str3 = "%s climbs stalk %d" % ("jack", num) str4 = "%d stalks" % num</pre>

Looping over Lists

VBScript	IronPython
<pre>Dim myArray(0 to 2) as String myArray(0) = "alpha" myArray(1) = "bravo" myArray(2) = "charlie" For Each i in myArray Print i Next</pre>	<pre>vals = [1, 3, 3.456] def process(val): return 2*val for i in vals: print i print " -> " process(i)</pre>

Looping over a Range

VBScript	IronPython
To loop over a range, specify start, end, and step: <pre>For i = 0 To 10 Step 1 Print i Next</pre>	<pre>for i in range(0, 10): print i</pre>

Related Topics:[Indentation in IronPython](#)[Methods in IronPython](#)[Introduction to IronPython](#)[Translating Script commands from VBScript to IronPython](#)[Scripting Using Iron Python](#)[IronPython Samples](#)**Indentation in IronPython**

Python is a language where white space (spaces and tabs) is syntactically significant. You must understand the basics of indentation before scripting in python.

Any statement that introduces a block of code should be written so that every line of the block has the same indent (leading spaces or tabs) and the indent should be at least one more than the indent of the introducing statement.

Note:

Python recommends the use of spaces over tabs.

Indenting Functions

Define a function that starts at 0 indentation:

```
def multInt(a,b):
```

Every line following `def multInt` that is expected to be a part of the function, must be indented to line up with the function.

```
def multInt(a,b):
```

```
    return a
```

Indenting If Conditions

Each line that belongs to the body of this function should have an indent that is more than the indent used by the if statement.

```
def multInt(a,b):
```

```
If a%2 == 0:
```

```
    return (a * b) + 100
```

```
else:
```

```
    return (a * b) + 1000
```

Methods in IronPython

Finding Methods

To list all methods available in the string module, import the module:

```
import string
```

Then get the directory listing:

```
dir(string)
```

This returns a list of all the methods available (as well as some `__somenam__` internal names that can be ignored).

Help

Once you know a function name, you can get more help on it using the built-in `help` method.

Related Topics

[Introduction to IronPython](#)

[Translating Script commands from VBScript to IronPython](#)

[Scripting Using Iron Python: Putting it all Together](#)

[IronPython Samples](#)

Translating Script Commands from VBScript to IronPython

This topic briefly describes scripting methods and arguments via VBScript samples. The distinctions made here are significant and useful when translating scripts written in VBScript to IronPython.

Related Topics

[Script Method Argument](#)

[VBscript Method Call Types](#)

[Converting VBScript Function calls to IronPython Syntax](#)

[Introduction to IronPython](#)

[IronPython Mini Cookbook](#)

[Scripting Using Iron Python](#)

[IronPython Samples](#)

Script Method Argument

[Script method calls in VBScript](#) generally take the form:

```
objectName .methodName ( arg1, arg2, ...)
```

The function call syntax is a standard followed by several programming languages. However, the argument types in VBScript objects used for product scripting are restricted to the following:

- Primitive Types
- Named Arrays
- Named Functions

Primitive Types

Primitive types are the standard `bool`, `int`, `float`, `double`, and `string`.

Named Arrays

Named arrays are a special construct used very commonly and can be found in many recorded script samples.

A named array begins with **Array("NAME:someName"** followed by a collection of comma separated values which can be:

- Primitive values
- Arrays of primitive values
- Additional named arrays
- Keys, in the form **"keyName:="** followed by a primitive value or function

Named Functions

Named functions are arrays which start with **Array(** and *do not* have a leading "NAME:name" item. **They are always introduced by a key** and can contain comma separated values of the following type:

- A primitive value
- A key (of the form **"keyName:="**) followed by
 - A primitive value
 - Another function (nested function)

Related Topics

[Translating Script commands from VBScript to IronPython](#)

VBScript Method Call Types

VBScript method calls fall into two categories and the distinction between the two results in syntax differences. These syntax differences are significant when converting VBScript to IronPython.

VBScript Functions

In VBScript terminology, functions return values. The syntax for this is one shared with practically all programming languages:

```
Set oDesktop = oAnsoftApp.GetAppDesktop()
```

```
Set oProject = oDesktop.NewProject
```

Note:

If there are arguments, the method name is *always* followed by an argument list enclosed in parentheses. If the argument list is empty, as shown above for the *NewProject* call, the parentheses can be omitted.

VBScript Sub-Routines

VBScript subroutines are those that do not have any return value. VBScript allows these to be written without any parentheses even if they have a non-empty argument list.

```
oModule.CreateReport "XY Plot1", "Standard", "XY Plot", "optimtee :  
optimtee", _  
Array("Domain:=", "Sweep"), Array("Freq:=", Array("All"), "off-  
set:=", _  
Array("Ouin")), Array("X Component:=", "Freq", "Y Component:=", _  
Array("dB20(S(1,1))", "dB20(S(1,2))", "dB20(S(1,3))", _  
"dB20(S(2,1))", "dB20(S(2,2))", "dB20(S(2,3))", "dB20(S(3,1))",  
"dB20(S(3,2))", "dB20(S(3,3))"), Array()
```

Related Topics

[Translating Script commands from VBScript to IronPython](#)

Converting VBScript Function calls to IronPython Syntax

When used for scripting, IronPython function names are always followed by parentheses.

So:

- If you see a VBScript snippet that looks like a VBScript subroutine, remember to add parentheses.
- If you see a VBScript function that has no arguments and no parentheses, remember to add them around an empty argument list.

The parentheses change is the only one to keep in mind when converting VBScript function calls syntax to IronPython.

Return Values

VBscript return values are sometimes assigned via the Set declaration. IronPython return values are simple assignment (See: [Iron Python Mini Cookbook](#)).

Primitive Method Arguments

Replace each VBScript primitive with an equivalent IronPython primitive.

Boolean values in IronPython have their first letter capitalized (`True` instead of `true` and `False` instead of `false`).

For arrays, the recommended approach is to simply replace a VBScript array with a Python array. The mapping is simple:

- Change `Array(` to a square bracket `[` and close with another square bracket `]` instead of a parenthesis `)`
- Remove the line continuation underscore symbol: `_`
- Map Boolean values correctly

Named Array Arguments

Formatting helps readability immensely but is not required.

All that *must* be done is:

- Add the parentheses, since the VBScript subroutine omits them
- Replace the `Array( )` delimiters with `[ ]`
- Remove the `Char(34)` function (which introduced a double quote) and replace it with the escaped double quote literal: `\"`
- Replace `true` with `True`
- Remove the line continuation symbol: `_`

Named Array Values with All Key Value Pairs

While it is generally not allowed to replace arrays and nested arrays with Python dictionaries, in the case where the named array consists entirely of key value pairs, you can use a dictionary and avoid typing the trailing `:=` symbols after the keys. This further aids readability of the script.

- If all key value pairs
- Remove the trailing `:=` after each key
- Replace the `,` after the key with a `:`
- If the named array is the top level argument, ensure that the `NAME:name` is present and is split into `NAME : name` as a key value pair
- Enclose the converted array in a `{ }` pair to declare the dictionary.

Named Arrays with Nested Named Arrays

- Split the `NAME:name` field into a key value pair
- Translate array key value pair to a dictionary key value pair.
- Create a new key with the name of the nested array and keep the nested array (as an array or as a dictionary) as its value. If the nested array is being retained as an array, the `NAME:name` field should be retained in the array. If the nested array is being converted to a dictionary, the name is optional: if also retained in the nested array, it must match the outer key.

["NAME:name",

```
"key1:=", 1,  
"key2:=", 2,  
["NAME:name2", "R:=", 255]  
]
```

Sample Script: Named array with nested named array in array syntax

The above named array with a nested named array (after conversion to IronPython as named array) can be converted to a dictionary as well. The dictionary can take any of the following forms

```
{ "NAME" : "name",  
  "key1" : 1,  
  "key2" : 2,  
  "name2" : ["NAME:name2", "R:=", 255]  
}
```

Sample Script: Named array with nested named array as mixed dictionary + array

```
{ "NAME" : "name",  
  "key1" : 1,  
  "key2" : 2,  
  "name2" : {"R" : 255}  
}
```

Sample Script: Named array with nested named array in all dictionary syntax

```
{ "NAME" : "name",  
  "key1" : 1,  
  "key2" : 2,  
  "name2" : {  
    "NAME" : "name2",  
    "R" : 255  
  }  
}
```

Function Blocks

Function blocks in VBScript argument syntax are represented as arrays without the "NAME:..." field. However, functions are always introduced by a key in a parent structure. Function blocks

can therefore never exist as a top-level argument. They are only found as the value pairs inside a named array or inside another function block.

Important:

Function blocks and their items cannot be converted to dictionaries even though they might be composed entirely of key value pairs.

The reason for this is the need to main the user-entered order. Every item in a function block is expect to be transmitted to the script method in exactly the same order as typed out and this is impossible to achieve when a dictionary is used (as the keys get reordered according to the dictionary's internal tree/key sorting scheme).

When you see a function block, simply replace the Array() delimiters with python array delimiters []

Scripting Using Iron Python

If you have existing VBScript/Javascript scripts use existing scripts them as much as possible by either embedding the test into the IronPython script or invoking them.

Translating a script in VBScript to IronPython

Read the [chapter on translation](#) and study the samples in that chapter as well as those in the appendix. For python syntax and the differences, the [mini-cookbook chapter](#) will also be useful.

Writing an IronPython script from scratch

Read through the scripting guide available from the product's help menu and translate the VBScript methods described to IronPython using the information provided in the [chapter on translation](#). Studying the samples in the document will also prove helpful.

For python syntax and the differences, the [mini-cookbook chapter](#) will also be useful.

[IronPython Script Execution Environment](#)

[Scripting using Embedded VBScript or JavaScript](#)

[Scripting with IronPython](#)

[Standalone IronPython and Desktop IronPython](#)

Related Topics

[Introduction to IronPython](#)

[IronPython Mini-cookbook](#)

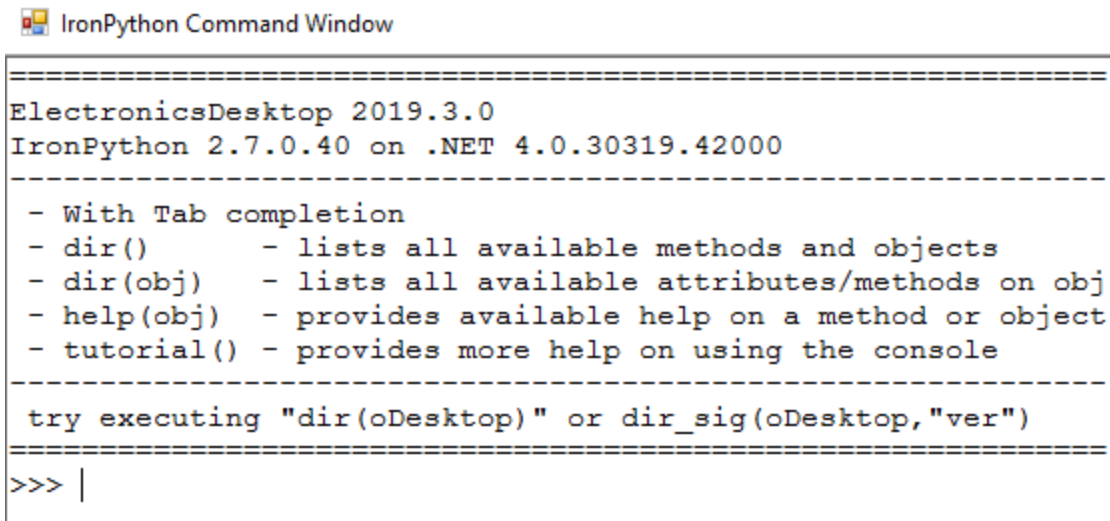
[Translating Script commands from VBScript to IronPython](#)

[Appendix: IronPython Samples](#)

IronPython Script Execution Environment

Scripts written in IronPython are executed by desktop in four different ways:

- **Tools > Open Command Window**, to open the **IronPython Command Window**:



```
IronPython Command Window

=====
ElectronicsDesktop 2019.3.0
IronPython 2.7.0.40 on .NET 4.0.30319.42000
=====
- With Tab completion
- dir()      - lists all available methods and objects
- dir(obj)   - lists all available attributes/methods on obj
- help(obj)  - provides available help on a method or object
- tutorial() - provides more help on using the console
=====
try executing "dir(oDesktop)" or dir_sig(oDesktop,"ver")
=====
>>> |
```

- **Tools > Run Script** menu item, select "IronPython" from the file type drop-down list.
- Launch the product with a script argument.
 - Maxwell -runscript someScript.py to keep Maxwell GUI open after completing script execution.
 - Maxwell -features=beta -ng -runscriptandexit someScript.py to run Maxwell in a non-graphical mode and exit after script completion. Note that this is a beta feature.
- Register an IronPython script as an external tool using the **Tools > External Tools** menu item.

When desktop executes a script, it does so in an execution environment setup with predefined variables and functions. These predefined variables and functions are how the script communicates with the desktop, and they come in four flavors addressed in the following subtopics:

[Script Argument for IronPython](#)

Script Argument for IronPython

When scripts are launched using the **Tools > Run Script** menu item, the dialog that pops up allows the user to specify arguments.

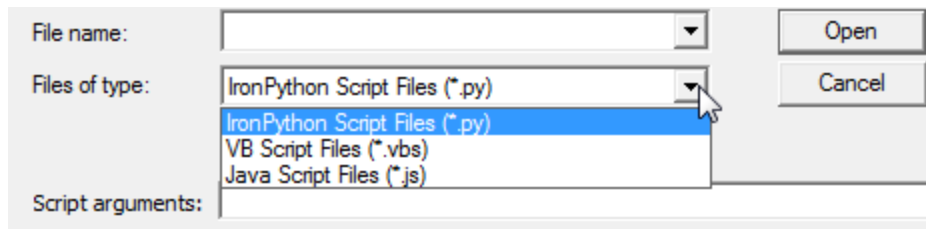


Figure 1: Run Script dialog and script arguments

Any argument specified here is communicated to the script being executed as the predefined variable **ScriptArgument**.

Related Topics

[IronPython Script Execution Environment](#)

Scripting using Embedded VBScript or JavaScript

Since script recording is still done in VBScript and users are expected to have a significant collection of VBScript or JavaScript assets, it is useful to continue to use existing script files and snippets even when scripting in IronPython. The various **Run<*>Command** methods have been designed for this purpose.

For instance: one can create a parameterized cone in HFSS by executing the following IronPython script from the **Tools > Run Script** menu.

```
# assign the VBScript snippet obtained from a script recording from
HFSS to

# coneScript and replace the BottomRadius recorded value with botRa-
dius

coneScript = """Dim oAnsoftApp

Dim oDesktop

Dim oProject

Dim oDesign

Dim oEditor

Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")

Set oDesktop = oAnsoftApp.GetAppDesktop()

oDesktop.RestoreWindow

Set oProject = oDesktop.GetActiveProject()

oProject.InsertDesign "HFSS", "HFSSPyTestDesign", "DrivenModal", ""
```

```
Set oDesign = oProject.SetActiveDesign("HFSSPyTestDesign")
Set oEditor = oDesign.SetActiveEditor("3D Modeler")
oEditor.CreateCone Array("NAME:ConeParameters", _
    "XCenter:=", "0mm", "YCenter:=", "0mm", "ZCenter:=", "0mm", _
    "WhichAxis:=", "Z", "Height:=", "2mm", _
    "BottomRadius:=", "3mm", _
    "TopRadius:=", "0mm"), Array("NAME:Attributes", "Name:=", _
    "Cone1", "Flags:=", "", "Color:=", "(132 132 193)", "Trans-
    parency:=", 0, _
    "PartCoordinateSystem:=", "Global", "UDMId:=", "", "Mater-
    ialValue:=", _
    "" & Chr(34) & "vacuum" & Chr(34) & "", "SolveInside:=", _
    true)
""

SetScriptingLanguageToVBScript()
RunScriptCommand(coneScript)
```

Sample Script 11: Hybrid VBScript + IronPython scripting: parameterized Cone Creation

Even though recorded VBScript is used for scripting, the incremental functionality that is provided using IronPython is the ability to write a GUI using IronPython/.NET, collect information from the user and then modify or generate the VBScript commands to actually script the Ansys EM desktop. This GUI functionality is cross platform and a significant positive. The following example demonstrates a contrived use of a .NET window form to display the argument supplied to the IronPython script (via the ***ScriptArgument*** variable).

```
#import the CLR references
import clr
clr.AddReference("System.Windows.Forms")

from System.Windows.Forms import Application, Form, Label, Button,
DockStyle

# the GUI form to show some text
# the class below derives from Form (System.Windows.Forms.Form)
```

```
# imported above from the .NET assembly.
class ShowPropertiesForm(Form):
    def __init__(self, name, text):
        self.Name = name

        self._label = Label()
        self._label.Text = text
        self._label.Dock = DockStyle.Fill

        _button = Button()
        _button.Text = "Close"
        _button.Dock = DockStyle.Bottom
        _button.Click += self._buttonPressed

        self.Controls.Add(self._label)
        self.Controls.Add(_button)

    def _buttonPressed(self, sender, args):
        self.Close()

#-----
# Main script code
#-----
#display the ScriptArgument variable as the text label
# in the form.
gui = ShowPropertiesForm("Sample Form", ScriptArgument)

# This makes it a modal dialog.
gui.ShowDialog()

# the following will make it a non-modal dialog
```

```
#Application.Run(gui)
```

Sample Script 12: Demonstrates the use of a .NET form from IronPython

While creating cross platform user interfaces from scripts is one of the main motivations driving the adoption of IronPython, any .NET assembly can be used with the caveat that Linux use requires Mono compatibility of any used assemblies.

While this hybrid approach is useful when you have existing VBScript commands that you want to reuse or when you want to quickly parameterize a recorded sample, the one significant limitation of this approach is the inability to capture return values from VBScript or JavaScript calls that do return something. Full two way communication with the product requires the use of pure IronPython to directly invoke the script objects as described below.

Related Topics

[IronPython Script Execution Environment](#)

Scripting with IronPython

While this section talks about directly interacting with the script objects, note that you can execute VBScript or Javascript at any point using any of the available Run*Command functions. using your existing script assets in this fashion and mixing with IronPython code for new functionality as needed is a viable and option.

Access to the application scripting objects is provided via the predefined **oDesktop** object (as listed in Script Objects). Interacting with the script objects is very natural, method calls are made just like in VBScript except that the argument syntax is somewhat simplified to follow natural Python syntax. All primitive types (string, integer, double) map to the natural primitive types in python. The only differences from the VBScript syntax are seen when specifying array type arguments. The differences are described in earlier chapters.

Note:

The typical VBScript calls to obtain the registered COM scripting interface via CreateObject calls and then obtain the oDesktop object from it using the GetAppDesktop() is not needed (or even supported on all platforms). Since all scripting occurs in the context of a running workbench, the available Desktop object is always provided and expected to be used directly.

Scripting using the IronPython scripting API is very much like scripting with VBScript except that

- Any argument is supplied via the built in **ScriptArgument** variable
- The **oDesktop** object is always available
- The scripting method names are identical to the ones used with VBScript
- Method calls, while the name is the same have to adhere to the rule of ensuring trailing parentheses irrespective of whether the function returns anything or has any arguments.

- Any compound/block arguments should be translated to the appropriate IronPython array or dictionary syntax.

The [samples section](#) lists a collection of pure IronPython snippets: these, along with the various script snippets listed in this document should serve as a guide and reference.

Related Topics

[IronPython Script Execution Environment](#)

[Standalone IronPython and Desktop IronPython](#)

Standalone IronPython

In general, it is easier to run a script directly from Electronics Desktop. Standalone IronPython does not implement all the functionality available when a script is run from Electronics Desktop. It only implements full support for COM functions.

Running Standalone IronPython

Standalone IronPython uses COM to get the handle to the AnsysEDT app. To run standalone IronPython, you'll need to call the IronPython interpreter `ipy64.exe`.

It is located in:

```
\\<AnsysEDTInstallationPath>\common\IronPython\ipy64.exe
```

For example, to run `myScript.py`, type the following in the command line:

```
"C:\Program Files\AnsysEM\v242\Win64\common\IronPython\ipy64.exe"  
"<filePath>\myScript.py"
```

You can set the interpreter to be the default program when double-clicking the `.py` script. You can use any recorded script as the basis for a standalone script and simply add an installation-internal path to the python module search path (as shown below) and end the script with a new shutdown call.

Using a Recorded Script

A python script recorded in AnsysEDT already has the required lines to be run as a standalone, except for the first two lines (path settings) and the final `Shutdown()` call. See the [example script](#) below.

Creating an External Script

When creating a script outside of Electronics Desktop, the following lines should be included at the beginning of your script:

- `import sys`

Imports the sys module containing system-specific functions native to IronPython.

- `sys.path.append("<InstallationPath>")`
`# Adds the Electronics Desktop installation path to the list of directories Python searches for modules and files.`
- `sys.path.append("<InstallationPath>/PythonFiles/DesktopPlugin")`
`# Adds the PythonFiles/DesktopPlugin subfolder to the list of directories Python searches for modules and files.`
- `import ScriptEnv`
`# This imports ScriptEnv.py from the installation path specified above. ScriptEnv.py performs an operating system check and defines functions used in Electronics Desktop scripts. See the annotations in the ScriptEnv.py file for more information.`
- `ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")`
`or ScriptEnv.InitializeNew(NonGraphical=True)`
`# Initialize and InitializeNew are functions within ScriptEnv.py. The first option launches Electronics Desktop. The second allows you to run a script without launching Electronics Desktop. See the annotations in the ScriptEnv.py file for more information.`

You must end the script with:

- `ScriptEnv.Shutdown()`
`# This stops ScriptEnv.py. If you are running multiple scripts, include this only at the end of the last script.`

Example Script

```
import sys

sys.path.append(r"C:\Program Files\AnsysEM\v242\Win64")

sys.path.append(r"C:\Program Files\An-
sysEM\v242\Win64\PythonFiles\DesktopPlugin")

import ScriptEnv

ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")

oDesktop.RestoreWindow()

oProject = oDesktop.NewProject()

oProject.InsertDesign("HFSS", "HFSSDesign1", "DrivenModal", "")

oDesign = oProject.SetActiveDesign("HFSSDesign1")

oEditor = oDesign.SetActiveEditor("3D Modeler")
```



```
oEditor.CreateRectangle(  
[  
    "NAME:RectangleParameters",  
    "IsCovered:= ", True,  
    "XStart:= ", "-0.2mm",  
    "YStart:= ", "-3mm",  
    "ZStart:= ", "0mm",  
    "Width:= ", "0.8mm",  
    "Height:= ", "1.2mm",  
    "WhichAxis:= ", "Z"  
],  
[  
    "NAME:Attributes",  
    "Name:= ", "Rectangle1",  
    "Flags:= ", "",  
    "Color:= ", "(132 132 193)",  
    "Transparency:= ", 0,  
    "PartCoordinateSystem:=", "Global",  
    "UDMId:= ", "",  
    "MaterialValue:= ", "\"vacuum\"",  
    "SolveInside:= ", True  
])  
oDesign.SetDesignSettings(['NAME:Design Settings Data', 'Allow Material  
Override:=' , True, 'Calculate Lossy Dielectrics:=' , True])  
oEditor.SetModelUnits(['NAME:Units Parameter', 'Units:=' , 'mil', 'Res-  
cale:=' , False ])  
ScriptEnv.Shutdown()
```

IronPython Samples

Change property

The following snippets show how a change property command (in this case, to change the color of a cone) looks in VBScript and its two possible IronPython variants.

```
oEditor.ChangeProperty Array("NAME:AllTabs", Array("NAME:Geometry3DAttributeTab", _  
    Array("NAME:PropServers", "Cone1"), _  
    Array("NAME:ChangedProps", _  
        Array("NAME:Color", "R=", 255, "G=", 255, "B=", 0))))
```

Sample Script 13: ChangeProperty command to change color of a cone in VBScript

```
oEditor.ChangeProperty(  
["NAME:AllTabs",  
    ["NAME:Geometry3DAttributeTab",  
    ["NAME:PropServers", "Cone1"],  
    ["NAME:ChangedProps",  
    ["NAME:Color", "R=", 0, "G=", 0, "B=", 64]  
    ]  
]  
    ])
```

Sample Script 14: ChangeProperty command to change color of cone using Python arrays

Any time there are named arrays composed purely of key-value pairs, they can always be represented using a Python dictionary, irrespective of the nesting of said named array.

```
oEditor.ChangeProperty(  
    ["NAME:AllTabs",  
    ["NAME:Geometry3DAttributeTab",  
        ["NAME:PropServers", "Cone1"],  
        ["NAME:ChangedProps",  
            {  
                "NAME": "Color",  
                "R" : 0,  
                "G" : 64,  
                "B" : 0
```

```
    }]]  
  ])
```

Sample Script 15: ChangeProperty command to change the color of a cone using Python arrays and dictionaries

Create a Cone using IronPython

Most scripting tasks using IronPython are expected to be formatted as the following example. One starts with the predefined **oDesktop** object and drills down to the design, editors, modules etc and issues any required commands on the object while formatting the script command arguments in natural python syntax.

```
oProject = oDesktop.GetActiveProject()  
oDesign = oProject.InsertDesign("HFSS","Random","DrivenModal","")  
oEditor = oDesign.SetActiveEditor("3D Modeler")  
oEditor.CreateCone(  
{  
    "NAME" : "ConeParameters",  
    "XCenter" : "0mm",  
    "YCenter" : "0mm",  
    "ZCenter" : "0mm",  
    "WhichAxis" : "Z",  
    "Height" : "2mm",  
    "BottomRadius" : "1.56204993518133mm",  
    "TopRadius" : "0mm"  
},  
{  
    "NAME" : "Attributes",  
    "Name" : "Cone1",  
    "Flags" : "",  
    "Color" : "(132 132 193)",  
    "Transparency" : 0,  
    "PartCoordinateSystem": "Global",  
    "UDMId" : "",
```

```
"MaterialValue" : "\"vacuum\"",  
"SolveInside" : True  
}  
)
```

Sample Script 16: IronPython script to create a cone

Create geometry and then create a grid from it using copy/paste/move

The following script demonstrates slightly more advanced use of scripting and the use of return values from script methods. It creates a 5x5 grid of cones and also demonstrates the adding of information messages to the application's message window.

```
oProject = oDesktop.GetActiveProject()  
  
oDesign = oProject.InsertDesign("HFSS", "Hersheys Kisses", "DrivenModal", "")  
  
oEditor = oDesign.SetActiveEditor("3D Modeler")  
  
# create the first cone  
AddInfoMessage("Creating first cone")  
  
firstConeName = "firstCone"  
coneBotRad = "1.5mm"  
oEditor.CreateCone(  
    {  
        "NAME" : "ConeParameters",  
        "XCenter" : "0mm",  
        "YCenter" : "0mm",  
        "ZCenter" : "0mm",  
        "WhichAxis" : "Z",  
        "Height" : "2mm",  
        "BottomRadius": coneBotRad,  
        "TopRadius" : "0mm"  
    },  
    {  
        "NAME" : "Attributes",
```

```
"Name" : firstConeName,
"Flags" : "",
"Color" : "(132 132 193)",
"Transparency" : 0,
"PartCoordinateSystem": "Global",
"UDMId" : "",
"MaterialValue" : "\"vacuum\"",
"SolveInside" : True
}
)

# Now replicate this a few times and create an array out of it
AddInfoMessage("Replicating it 24 times")
for x in range(5):
    for y in range(5):
        # leave the first one alone in it's created
        # position
        if x == 0 and y == 0:
            continue

# all other grid positions, replicate from the
# first one

# copy first
oEditor.Copy(
    {
        "NAME" : "Selections",
        "Selections" : firstConeName
    }
)
```

```
# paste it and capture the pasted name
# the pasted names come in an array as we could
# be pasting a selection composed of multiple objects
pasteName = oEditor.Paste()[0]

# now move the pasted item to it's final position
oEditor.Move(
    {
        "NAME" : "Selections",
        "Selections" : pasteName
    },
    {
        "NAME" : "TransalateParameters",
        "CoordinateSystemID" : -1,
        "TranslateVectorX" : "%d * 3 * %s" % (x, coneBotRad),
        "TranslateVectorY" : "%d * 3 * %s" % (y, coneBotRad),
        "TranslateVectorZ" : "0mm"
    }
)

# Now fit the display to the created grid
oEditor.FitAll()
```

Sample Script 17: Sample script to create a cone and then use copy/paste/move to replicate it.

Related Topics

[Introduction to IronPython](#)

[IronPython Mini-cookbook](#)

[Translating Script commands from VBScript to IronPython](#)

[Scripting Using Iron Python: Putting it all Together](#)

Creating User Defined Primitives and User Defined Models in Python Scripts

You can create User Defined Primitives and User Defined Models in Python scripts (based on the IronPython implementation).

Advantages Compared to C++

- No need to create and build project; all you need to do is create a Python script
- Python script is platform independent
- Scripts can inherit functionality from existing scripts
- Garbage collector - no need to free memory
- Easy debugging

Changes compared to C

Though methods, constants and structures are kept as close to the C implementation as possible, some changes had to be made to make code Python-compatible.

Structures

- Structures have the same names as in C implementation.
- Structures fields names are capitalized.
- Arrays in structures become lists in Python (Technically a .NET IList container)
- Structure instances are created using the supplied constructors and members are accessed using the provided access methods.

For a complete list of structures and examples please see [UDP/UDM Structures](#).

Return Values for UDM and UDP Functions

For information on return values for each UDM and UDP function, see the [Return Values](#) section.

Constants

Enumeration/Enum constants have almost the same names as in C but the enum must be qualified by the type. Additionally, redundant "UDP", "UDM" or type prefixes have been removed. This allows for better human-readability.

```
# Example of specifying the LengthUnit enum by qualifying it
# with the type of the enum: UnitType
unitType = UnitType.LengthUnit
```

For a complete list of enum constants please see [UDP/UDM Constants](#).

Methods

Methods are described in [IUDPExtension methods](#), [IUDMExtension methods](#), and [UDMFunctionLibrary](#) listed further in this document. A separate chapter includes a [UDP IronPython example of fillet and chamfer](#).

The main differences in functions parameters (from C implementation):

- functions names in UDPFunctionLibrary and UDMFunctionLibrary are capitalized
- arrays become a python list of objects
- `void * callback parameter` is dropped from the parameter list
- output parameters (pointer types that are filled during the function call) usually become return values
- 'list size' parameter usually will be omitted as redundant

Output Parameters

The rule for the output parameters is as follows:

- If the function has one output parameter variable and no return value, the variable will become function's return value. The same will happen if the return value is a 'success/failure' boolean ('None' will be returned on failure and parameter variable - on success).
- If the function has one output parameter and a return value, the function will return a Python tuple where function return value will be the first one in the tuple.
- If there is more than one out variable, the function will return a Python tuple with all output parameters in the specified order. If function has a return value, it must always be the first in the tuple.

one output parameter; return value is ignored

```
udmDefinition = udmFunctionLibrary.GetDefinition()
```

**# one output parameter; return value must be preserved. return
and output values are packed into the return tupe, in order**

```
(lRet, partIdsList) = udpFunctionLibrary.DetachFaces(nPartIds, faceIdsList)
```

Two output parameter; return value must be preserved

the return tuple is (returnVal, output1, output2)

```
(bRet, udpPositionLow, udpPositionHigh) = udmFunctionLibrary.GetBoundingBox(partId, exact);
```


Comparison with C function:

C	Python
<pre>bool getDefinition(UDMDefinition* udmDefinition, void* callbackData);</pre> where udmDefinition is an output parameter	<pre>udmDefinition = udmFunctionLibrary.GetDefinition()</pre> (Note: callbackData is omitted in py interface)
<pre>long detachIFaces(int nFacesAndPartIds, long* faceIds, long* partIds, void* callbackData);</pre> where partIds is an output parameter	<pre>(bRet, partIds) = udmFunctionLibrary.DetachIFaces (nFacesAndPartIds, faceIds)</pre> (Note: callbackData is omitted in py interface)

'List Size' Parameters

The rule for the 'list size' is as follows:

- If function has input 'List' parameter and input 'list size' parameter, 'list size' parameter will be omitted.
- If function has output 'List' parameter and output 'list size' parameter, 'list size' parameter will be omitted.
- If function has output 'List' parameter and input 'list size' parameter, 'list size' parameter won't be omitted as it's needed for memory allocation in the corresponding C++ function from the UDP/UDM function library.

Example:

input list, input list size

```
lret = udpFunctionLibrary.Unite(objectIds)
```

output list, output list size

```
faceIdList = udmFunctionLibrary.GetAllFaces(PartId)
```

output list, input list size

```
(lret, partIdList) = udpFunctionLibrary.DetachFaces(listSize,  
faceIdList)
```

Comparison with C function:

C	Python
<pre>bool getAllFaces(long partId, long* numFaces, long** facelds, void* callbackData);</pre> where numFaces and facelds are output parameters and numFaces is the size of faceld.	<pre>facelds = udmFunc- tionLibrary.GetAllFaces(partId)</pre> (ignore numFaces as redundant: fol- ded into facelds, return value is omitted: folded into the facelds is None check callbackData is omitted)
<pre>long unite(long numObjects, long* objectIds, void* callbackData);</pre> where numObjects and objectIds are input parameters and numObjects is the size of objectIds.	<pre>lret = udpFunctionLibrary.Unite (objectIds)</pre> (ignore numObjects as redundant: fol- ded into objectIds callbackData is omitted)
<pre>long detachFaces(long nSize, long* facelds, long* partIds, void* callbackData);</pre> where partIds is and output list and nSize is an input parameters and nSize is the size of partIds.	<pre>(lret, partIdList) = udpFunc- tionLibrary.DetachFaces(nSize, facelds)</pre> (nSize is not ignored, callbackData is omitted)

Added Parameters

There is a special case in UDPFunctionLibrary: two functions - DuplicateAlongLine and DuplicateAroundAxis - have new integer listSize parameter added to their signatures.

This parameter defines the size of the output List. This is done for compliance with C++ geometry library as the size of the List must be predefined and this size is different from the existing parameter's values.

Example:

```
(ret, cloneIDs) = funcLib.DuplicateAlongLine(partID, transVec,
numCubes, cloneIDsSize)

(ret, cloneIDs) = funcLib.DuplicateAroundAxis(partID, axis, angle,
nClones, cloneIDsSize)
```

Here cloneIDsSize is a new integer parameter.

Comparison with C function:

C	Python
<pre>long duplicateAlongLine(long partId, UDPVector transVector, int nClones, long* nClones, void* callbackData);</pre>	<pre>(lret, cloneIds) = udmFunctionLibrary.DuplicateAlongLine(partId, transVec, nClones, cloneIdsSize) (callbackData is omitted cloneIdsSize is a new parameter)</pre>
<pre>long duplicateAroundAxis(long partId, UDPCoordinateSystemAxis axis, double angle, int nClones, long* nClones, void* callbackData);</pre>	<pre>(lret, cloneIds) = udmFunctionLibrary.DuplicateAroundAxis (partId, axis, angle, nClones, cloneIdsSize) (callbackData is omitted cloneIdsSize is a new parameter)</pre>

Developing a UDM/UDP

Creation

To create a User Defined Primitive in Python you write a Python script that implements [UDPExtension class](#). To create a User Defined Model in Python you write a Python script that implements [UDMExtension](#) class (see links for full description).

Location

The scripts are located the same way the C based UDM/UDP are. They are expected to be under the UserDefinedParts or UserDefinedModels sub-directories of one of the library folders (SysLib, UserLib or Personallib). They will then appear under the appropriate menu items:

Draw > User Defined Primitives for UDP or **Draw > User Defined Model for UDM**.

The sub-directories structure created in one of the specified directory will be displayed in the UDP/UDM menu.

Keep in mind that there is no difference between the menu display for C and Python implementations of UDM or UDP - only the file names without extensions are displayed

Organize

"Lib" sub-directory is a special directory. The contents of this directory is not shown in the menu. In the "Lib" directory you can create Python scripts with base classes and utilities to be used in UDP/UDM Python scripts. All the Lib directories upstream of a script (till the UserDefinedModels or UserDefinedPrimitives) are included in the Python search path and this allows for easy import of helper modules in such directories.

To use UDM data structures, constants, and/or classes in your Lib sub-directory scripts you have to add import statement to the scripts:

For UDM:extension:

```
from UDM import *
```

For UDP:extension:

```
from UDP import *
```

Edit/Reload

Python is a scripting language, so if you have errors in your script, you will see them at the time you try to run the script. The errors will be displayed in the Message Manager Window. If you need more information, you might be able to get it from log files. See: Debug Logging.

You can always change your script, call **Update Menu** command from **Draw > User Defined Model > menu** or **Draw > User Defined Primitives > menu** and run the script again. If you delete script you might want to restart the application instead of calling **Update Menu**.

UDPExtension

Import

You do not have to add import statements for the predefined classes, structures, and constants - it is done for you and all data types described in this document can be used in your Python script.

However you have to add import statements to your helper scripts in your Lib sub-directory.

```
from UDP import *
```

Main class: UDPExtension

You must write a class derived from IUDPExtension with a mandatory name UDPExtension:

```
class UDPExtension(IUDPExtension):
```

The class should implement [IUDPExtension methods](#) described below.

UDPExtension methods

All methods are same as the methods in the C UDP implementation. The changes to the methods signatures are just to conform to the Python style.

Mandatory methods.

These methods must be implemented in the UDP Python script as methods of UDPExtension class.

GetLengthParameterUnits()

- returns string.

GetPrimitiveTypeInfo()

- returns UDPPrimitiveTypeInfo.

GetPrimitiveParametersDefinition2()

- returns a list of UDPPrimitiveParameterDefinition2 or None on failure

AreParameterValuesValid2(errorMsg, udpParams)

- errorMsg is a c# list of strings
- udpParams is a c# list of UDPParam
- returns True if udpParams are valid, False otherwise.

CreatePrimitive2(funcLib, udpParams)

- funcLib is [UDMFunction library](#)
- udpParams is a c# list of UDPParam
- returns True on success, False on failure.

Optional methods

These methods, which have default implementations, can be implemented as methods of UDPExtension class as needed. Default methods will return NULL or FALSE depending on the return type.

GetPrimitiveParameters()

- returns Python list of strings or NULL

GetRegisteredFaceNames()

- returns Python list of strings or NULL

GetRegisteredEdgeNames()

- returns Python list of strings or NULL

GetRegisteredVertexNames()

- returns Python list of strings or NULL

ProjectParametersOnToValidPlane2(currentUDPParams, projectedUDPParams)

- currentUDPParams is a list of UDPPParam
- projectedUDPParams is a list of UDPPParam
- returns True on success, False on failure.

MapParametersDefinitionVersions2(oldVersion, oldUDPParams)

- oldVersion is a string
- oldUDPParamsis a list of UDPPParam
- returns Python list of UDPPParam or NULL

GetOldPrimitiveParametersDefinition2(version)

- version is a string
- returns a list of UDPPPrimitiveParameterDefinition2 or None on failure.

Example UDP

```
import sys

class UDPExtension(IUDPExtension):

    def GetLengthParameterUnits(self):
        return "mm"

    def GetPrimitiveTypeInfo(self)
        typeInfo = UDPPPrimitiveTypeInfo(
            name = "SampleUDP",
            purpose = "example",
```

```
        company="Ansys",  
        date="12.21.12",  
        version = "1.0")  
  
    return typeInfo  
  
    ...  
  
    ...
```

UDMExtension

Import

You do not have to add import statements for the predefined classes and structures - it is done for you, and all data types described in this document can be used in your Python script.

However you have to add import statements to your helper scripts in your Lib sun-directory.

```
from UDM import *
```

Main class: UDMExtension

You must write a class derived from IUDMExtension with a mandatory name UDMExtension:

```
class UDMExtension(IUDMExtension):
```

The class should implement [IUDMExtension methods](#) described below.

IUDMExtension methods

All methods are the same as the methods in the C UDM implementation. The changes to the methods signatures are just to conform to the Python style.

Mandatory methods.

These methods must be implemented in the UDM Python script as methods of UDMExtension class.

GetInfo()

- returns UDMInfo object populated with appropriate UDM information.

IsAttachedToExternalEditor()

- returns True if UDM dll is attached to external editor.
- In case of python UDMs, this should typically return False

CreateInstance(funcLib)

- funcLib is UDMFunctionLibrary
- returns UDMPParameters.

GetUnits(instanceld)

- instanceld is a long
- returns string containing units for the instance.

Refresh(funcLib, udmlnParams, updatedParams, refreshModifiedPartsOnly, nonEditedPartRefIds)

This Method is called every time a UDM is refreshed. Geometry creation/refresh should happen in this method.

- funcLib is UDMFunctionLibrary
- udmlnParams is a list of UDMPParameters that comes from desktop
- updatedParams: UDM script can change the UDM parameters it receives. Updated parameters need to be sent back to desktop. If the UDM script is not going to change any of the parameters that it received, it needs to copy udmlnParams to updatedParams.
- refreshModifiedPartsOnly is a boolean

Supporting this flag is optional. For UDMs where the refresh performance is not an issue, it is recommended to ignore this flag and update all parts every time.

This flag can be used to optimize performance of Refresh method when the model created by UDM is large. If the UDM consists of multiple parts, and new parameters change only a few parts amongst them, UDM script can only modify parts that are changed by the new parameters.

- nonEditedPartRefIds: If RefreshModifiedPartsOnly is true and the UDM script supports partial update, Refresh method needs to return ids of parts that are unchanged.

returns True on success, False on failure.

ReleaseInstance(instanceld)

- instanceld is a long
- This should release any resources assigned to this particular instance of UDM.
- returns True on success, False on failure.

GetAttribNameForEntityId()

- Returns string that acts as a the name of the attribute containing entity IDs.
- For example, it can return a unique string such as "ATTRIB_XACIS_ID"
- Python UDMs should implement this method.

GetAttribNameForPartId()

- Returns string that acts as a the name of the attribute containing entity IDs.
- For example, it can return a unique string such as "ATTRIB_XACIS_ID" (Can be same as GetAttribNameForEntityId())
- Python UDMs should implement this method.

Optional methods

These methods have default implementations (default is to return NULL or FALSE depending on the return type) but can be overridden by the user as needed as methods of UDMExtension class.

DialogForDefinitionOptionsAndParams(self, defData, optData, params):

Replaces the old UDMDialogForDefinitionAndOptions method, which is still supported, but users are urged to use UDMDialogForDefinitionOptionsAndParams. If both methods are present, application will use UDMDialogForDefinitionOptionsAndParams.

- UDM can pop up dialog for UDM definition, options, parameters in this method. Definition, options, and parameters are set/modified by user and returned to application. DII can also just give default definition, options and parameters.
- Returns two booleans and a string
- First boolean returns whether the method was successful or not.
- Second boolean returns whether the application should popup a dialog box. If it is True, application will populate a dialog with definition, options, parameters that are returned.
- String returned contains length units for parameters.

DialogForDefinitionAndOptions(self, defData, optData) [Deprecated]

UDM can pop up dialog for UDM definition and options in this method. Definition, and options are set/modified by user and returned to application. DII can also just give default definition and options.

- Returns two booleans.
- First boolean provides whether the call to this method was successful or not.
- Second boolean determines whether the application should pop up a dialog box. If this is true, application will populate the dialog with the definitions and options that are returned. As no parameters are returned, no parameters are shown in this dialog.

GetInstanceSourceInfo(instanceId)

- instanceId is a long
- returns string containing source information of UDM instance. It is used to create initial name for UDM instance.

ShouldAttachDefinitionFilesToProject()

- returns true if any of definition files needs to be attached to project
- returns python list of string containing definition names of files or NULL

Example UDM

```
class UDMExtension(IUDMExtension):

    def IsAttachedToExternalEditor(self):
        return False

    def GetInfo(self)
        udmInfo = UDMInfo(
            name = "SampleUDM",
            purpose = "udm example",
            company="Ansys",
            date="12.21.12",
            version = "1.0")

        return udmInfo

    ...

    ...
```

UDMFunctionLibrary

UDMFunctionLibrary implements IUDMFunctionLib interface. The IUDMFunctionLib object is passed as a parameter to Python script in the following functions

- CreateInstance
- Refresh

You can call any of the functions from the functions list (shown below).

```
partRefId = udmFunctionLib.GetPartRefId(partId)
```

For example sample code that calls GetBoundingBox in Python script can look like this:

```
partId = 10
exact = True
udpPosition = UDPPosition(0,0,0)
```

```
(bret, udpPositionLow, udpPositionHigh) = udmFunctionLibrary.GetBoundingBox(partId, exact);
```

```
if bret:
    udpPosition.X = udpPositionLow.X
```

As you can see udpPositionLow and udpPositionHigh output parameters are defined in the call to GetBoundingBox function. There is no need to define them before the function call.

Functions list:

1. **List_of_UDMDefinition:** udmDefinitionList = **GetDefinition()**
2. **List_of_UDMOption:** udmOptionList = **GetOptions()**
3. **bool:** bret = **SetMaterialName(string: matName, int: partId)**
4. **bool:** bret = **SetMaterialName2(string: matName, string: partName)**
5. **bool:** bret = **SetPartName(string: partName, int: partId)**
6. **int:** iret = **GetInstanceId()**
7. **string:** str = **GetPartRefId(int: partId)**
8. **bool:** bret = **SetPartRefId(int: partId, string: refId)**
9. **List_of_int:** faceIds = **GetAllFaces(int: partId)**
10. **List_of_int:** edgeIds = **GetAllEdges(int: partId)**
11. **List_of_int:** vertexIds = **GetAllVertices(int: partId)**
12. **bool:** bret = **SetFaceAttribs(List_of_int: faceIds, List_of_string: attribs)**
13. **bool:** bret = **SetEdgeAttribs(List_of_int: edgeIds, List_of_string: attribs)**
14. **bool:** bret = **SetVertexAttribs(List_of_int: vertexIds, List_of_string: attribs)**
15. **string:** str = **GetModelerUnit()**
16. **string:** str = **GetCacheFileForUDMResume()**
17. **bool:** bret = **SetPartColor(int: partId, int: nColor)**
18. **bool:** bret = **SetPartFlags(int: partId, int: nFlags)**
19. (**bool:** bret, **UDPPosition:** low, **UDPPosition:** high) = **GetBoundingBox(int: partId, bool: exact)**
20. **bool:** bret = **IsParametricUpdate()**
21. **bool:** bret = **SetMaterialNameByRefId(string: partRefId, string: matName)**
22. **bool:** bret = **SetPartNameByRefId(string: partRefId, string: partName)**
23. **bool:** bret = **SetPartColorByRefId(string: partRefId, int: nColor)**
24. **bool:** bret = **SetPartFlagsByRefId(string: partRefId, int: nFlags)**

In addition to the above functions all functions defined in the UDPFunctionLib are available in the IUDMFunctionLib and can be called directly exactly the same way as the IUDMFunctionLib functions.

Example:

```
udmFunctionLib.CreateCircle(center, radius, ratio, isCovered)
```

UDM/UDP Functions

Return Values for Each UDM and UDP Function

ID – ID of created Object

SI – Success Indicator. Identifies whether or not operation was successful.

Functions list:

1. **bool:** SI = **AddMessage(MessageSeverity:** messageSeverity, **string:** message)
2. **bool:** SI = **NameAFace(UDPPosition:** pointOnFace, **string:** faceName)
3. **bool:** SI = **NameAEdge(UDPPosition:** pointOnEdge, **string:** edgeName)
4. **bool:** SI = **NameAVertex(UDPPosition:** pointOnVertex, **string:** vertexName)
5. **int:** ID = **GetFacelDFromPosition(UDPPosition:** pointOnFace)
6. **int:** ID = **GetEdgeIDFromPosition(UDPPosition:** pointOnEdge)
7. **int:** ID = **CreatePolyline(UDPPolylineDefinition:** polylineDefinition)
8. **int:** ID = **CreateRectangle(CoordinateSystemPlane:** whichPlane, **UDPPosition:** center-Point, **List_of_double:** widthAndHeight, **int:** isCovered)
9. **int:** ID = **CreateArc(CoordinateSystemPlane:** whichPlane, **UDPPosition:** centerPoint, **UDPPosition:** startPoint, **double:** fAngle)
10. **int:** ID = **CreateCircle(CoordinateSystemPlane:** whichPlane, **UDPPosition:** center-Point, **double:** fRadius, **int:** isCovered)
11. **int:** ID = **CreateEllipse(CoordinateSystemPlane:** whichPlane, **UDPPosition:** center-Point, **double:** fMajorRadius, **double:** fRadiusRatio, **int:** isCovered)
12. **int:** ID = **CreateRegularPolygon(CoordinateSystemPlane:** whichPlane, **UDPPosition:** centerPoint, **UDPPosition:** startPoint, **int:** numOfSides, **int:** isCovered)
13. **int:** ID = **CreateEquationBasedCurve(UDPEquationBasedCurveDefinition:** curveDefinition)
14. **int:** ID = **CreateEquationBasedSurface(UDPEquationBasedSurfaceDefinition:** surfaceDefinition)
15. **int:** ID = **CreateSpiral(UDPSpiralDefinition:** spiralDefinition)
16. **int:** ID = **CreateBox(UDPPosition:** startPoint, **List_of_double:** boxXYZsize)
17. **int:** ID = **CreateSphere(UDPPosition:** centerPoint, **double:** fRadius)
18. **int:** ID = **CreateCylinder(CoordinateSystemAxis:** whichAxis, **UDPPosition:** center-Point, **double:** fRadius, **double:** fHeight)

19. **int:** ID = **CreateCone**(**CoordinateSystemAxis:** whichAxis, **UDPPosition:** centerPoint, **double:** fBottomRadius, **double:** fTopRadius, **double:** fHeight)
20. **int:** ID = **CreateTorus**(**CoordinateSystemAxis:** whichAxis, **UDPPosition:** centerPoint, **double:** fMajorRadius, **double:** fMinorRadius)
21. **int:** ID = **CreatePolyhedron**(**CoordinateSystemAxis:** whichAxis, **UDPPosition:** centerPoint, **UDPPosition:** startPosition, **int:** numOfSides, **double:** fHeight)
22. **int:** ID = **CreateHelix**(**UDPHelixDefinition:** helixDefinition)
23. **bool:** SI = **Unite**(**List_of_int:** pObjectIDArray)
24. **bool:** SI = **Subtract**(**List_of_int:** pBlankObjectIDArray, **List_of_int:** pToolObjectIDArray)
25. **bool:** SI = **Intersect**(**List_of_int:** pObjectIDArray)
26. **bool:** SI = **Imprint**(**List_of_int:** pBlankObjectIDArray, **List_of_int:** pToolObjectIDArray)
27. **bool:** SI = **SweepAlongVector**(**int:** profileID, **UDPVector:** sweepVector, **UDPSweepOptions:** sweepOptions)
28. **bool:** SI = **SweepAroundAxis**(**int:** profileID, **CoordinateSystemAxis:** whichAxis, **double:** sweepAngle, **UDPSweepOptions:** sweepOptions)
29. **bool:** SI = **SweepAlongPath**(**int:** profileID, **int:** pathID, **UDPSweepOptions:** sweepOptions)
30. **bool:** SI = **Translate**(**int:** partID, **UDPVector:** translateVector)
31. **bool:** SI = **Rotate**(**int:** partID, **CoordinateSystemAxis:** whichAxis, **double:** rotateAngle)
32. **bool:** SI = **Mirror**(**int:** partID, **UDPPosition:** mirrorPlaneBasePosition, **UDPVector:** mirrorPlaneNormalVector)
33. **bool:** SI = **Transform**(**int:** partID, **List_of_double:** rotationMatrix, **UDPVector:** translateVector)
34. **bool:** SI = **Scale**(**int:** partID, **double:** xScale, **double:** yScale, **double:** zScale)
35. (**bool:** SI, **List_of_int:** cloneIDs) = **DuplicateAlongLine**(**int:** partID, **UDPVector:** translateVector, **int:** numTotalObjs, **int:** cloneIDsListSize)
36. (**bool:** SI, **List_of_int:** cloneIDs) = **DuplicateAroundAxis**(**int:** partID, **CoordinateSystemAxis:** whichAxis, **double:** rotateAngle, **int:** numTotalObjs, **int:** cloneIDsListSize)
37. **int:** ID = **DuplicateAndMirror**(**int:** partID, **UDPPosition:** mirrorPlaneBasePosition, **UDPVector:** mirrorPlaneNormalVector)
38. **bool:** SI = **Connect**(**List_of_int:** objectIDArray)
39. **bool:** SI = **Offset**(**int:** partID, **double:** offsetDistance)
40. **int:** ID? = **Section**(**int:** partID, **CoordinateSystemPlane:** sectionPlane)
41. (**bool:** SI, **int:** ID) = **Split**(**int:** partID, **CoordinateSystemPlane:** splitPlane, **SplitWhichSideToKeep:** whichSideToKeep, **bool:** bSplitCrossingObjectsOnly)
42. (**bool:** SI, **List_of_int:** importedObjectIDs) = **ImportNativeBody2**(**string:** fileNameWithFullPath)

- 43. (**bool**: SI, **List_of_int**: importedObjectIDs) = **ImportAnsoftGeometry**(**string**: fileNameWithFullPath, **List_of_string**: overridingParamsNameArray, **List_of_UDPPParam**: overridingParamsArray)
- 44. **int**:ID = **Clone**(**int**: partID)
- 45. **bool**: SI = **DeletePart**(**int**: partID)
- 46. **int**: ID = **CreateObjectFromFace**(**int**: faceID)
- 47. **bool**: SI = **Fillet**(**UDPBLNDElements**: entitiesToFillet, **UDPBLNDFilletOptions**: filletOptions)
- 48. **bool**: SI = **Chamfer**(**UDPBLNDElements**: entitiesToChamfer, **UDPBLNDChamferOptions**: chamferOptions)
- 49. (**bool**: SI, **List_of_int**: newPartIDs) = **DetachFaces**(**int**: newPartIDArraySize, **List_of_int**: faceIDs)
- 50. (**bool**: SI, **List_of_int**: newPartIDs) = **DetachEdges**(**int**: newPartIDArraySize, **List_of_int**: edgeIDs)
- 51. **int**: ID = **CreateObjectFromEdge**(**int**: edgeID)
- 52. **bool**: SI = **SheetThicken**(**int**: partID, **double**: fThickness, **bool**: bThickenBothSides)
- 53. (**bool**: SI, **List_of_int**: newPartIDArray) = **SweepFaceAlongNormal**(**int**: newPartIDArraySize, **List_of_int**: faceIDArray, **double**: sweepLength)
- 54. **bool**: SI = **CoverLine**(**int**: partID)
- 55. **bool**: SI = **CoverSurface**(**int**: partID)
- 56. **bool**:SI= **UncoverFaces**(**List_of_int**: faceIDArray)
- 57. (**bool**: SI, **int**: numPartsCreated, **List_of_int**: faceIDArray) = **SeparateBodies**(**int**: partID, **int**: numPartsCreated)
- 58. **bool**: SI = **MoveFaces**(**List_of_int**: faceIDArray, **bool**: bMoveAlongNormal, **double**: fOffsetDistance, **UDPVector**: moveVector)
- 59. **bool**: SI = **WrapSheet**(**int**: sheetBodyID, **int**: targetBodyID)
- 60. **bool**: SI = **ImprintProjection**(**int**: blankBodyID, **List_of_int**: toolBodyIDArray, **bool**: bNormalProjection, **UDPVector**: projectDirection, **double**: projectDistance)
- 61. **string**: path = **GetTempDirPath**()
- 62. **string**: path = **GetSysLibDirPath**()
- 63. **string**: path = **GetUserLibDirPath**()
- 64. **string**: path = **GetPersonalLibDirPath**()
- 65. **string**: path = **GetInstallDirPath**()
- 66. **string**: path = **GetProjectPath**()
- 67. (**bool**: SI, **bool**: abort) = **SetProgress**(**UDPPProgress**: progress)

UDP/UDM Structures and Constants

The following sections describe:

- [UDP/UDM Structures](#)
- [UDP/UDM Constants](#)

UDP/UDM Structures

Differences compared to C API

- **UDMDefinition**
- **UDMOptions**
- **UDMParameters**

Instead of containing arrays of data, the structures contain single fields where each field corresponds to an item in a different array from the original C API. The structure objects thus constructed are added to the Python list. Alternately the Python list can be initialized using the structure objects.

Example (creating UDMParameter list):

```
udmParamList = [  
    UDMParameter(  
        "cubeSizeName", UnitType.LengthUnit,  
        UDPParam(ParamDataType.Double, cubeSize),  
        ParamPropType.Value,  
        ParamPropFlag.MustBeReal),  
    UDMParameter(  
        "cubeDistanceName", UnitType.LengthUnit,  
        UDPParam(ParamDataType.Double, cubeDistance),  
        ParamPropType.Value,  
        ParamPropFlag.MustBeReal),  
    UDMParameter("numCubesName", UnitType.LengthUnit,  
        UDPParam(ParamDataType.Int, numCubes),  
        ParamPropType.Number,  
        ParamPropFlag.MustBeInt]
```

- **UDPParam**

- **UDPPParamData**

Data field in UDPPParam is now an object - the same for all types of data used - as Python can work with any type of data.

UDPPParamData is obsolete, thus not implemented. Be sure to set proper data type to UDPPParam.DataType when setting UDPPParam.Data.

Example:

```
nCubesParam = UDPPParam(ParamDataType.Int, numCubes)
```

```
nCubes = nCubesParam.Data
```

```
distanceParam = UDPPParam()
```

```
distanceParam.setDouble(10.5)
```

```
doubleDistance = distanceParam.Data * 2
```

- **UDP3x3Matrix**

The structure is not implemented. Use size 9 Python List of doubles instead.

Example:

```
rotationMatrix =[0,0,1, 1,0,0, 0,0,1]
```

```
udpFunctionLib.Transform(partId, rotationMatrix, translationVector)
```

List of structures

You can use constructors to create a structure. You can also modify fields - directly or by provided methods.

Example:

```
pos1 = UDPPosition(1,2,3)
```

```
pos2 = UDPPosition(x=1,y=10,z=0)
```

```
pos2.Z = pos1.Z
```

```
udpParam = UDPPParam(ParamDataType.Double,1)
```

```
value = udpParam.Data
```

Structure	Construction	Members
UDPPrimitiveTypeInfo	UDPPrimitiveTypeInfo(string name,	string Name string Purpose

Structure	Construction	Members
	string purpose, string company, string date, string version)	string Company string Date string Version
UDPPrim- itiveParameterDefinition	UDPPrim- itiveParameterDefinition(string name, string description, UnitType unitType, double defaultValue)	string Name string Description UnitType UnitType double DefaultValue
UDPPParam	UDPPParam() UDPPParam(ParamDataType dataType, object data) object can be int, double, string, bool or UDPPosition methods: setInt(int val) setBool(bool val) setString(string val) setDouble(double val) setPosition(UDPPosition val)	ParamDataType DataType object Data object can be int, double , string, bool or UDPPosition
UDPPrim- itiveParameterDefinition2	UDPPrim- itiveParameterDefinition2(string name, string description, UnitType unitType,	string Name string Description UnitType UnitType ParamPropType PropType ParamPropFlag PropFlag

Structure	Construction	Members
	ParamPropType propType, ParamPropFlag propFlag, UDPParam defaultValue)	UDPParam DefaultValue
UDPPosition	UDPPosition(double x, double y, double z)	double X double Y double Z
UDPVector	UDPVector(double x, double y, double z)	double X double Y double Z
UDPSweepOptions	UDPSweepOptions(SweepDraftType draftType, double draftAngle, double twistAngle)	SweepDraftType DraftType double DraftAngle double TwistAngle
UDPPolylineSegmentDefinition	UDPPolylineSegmentDefinition(PolylineSegmentType segmentType, int segmentStartIndex, int numberOfPoints, double angle, UDPPosition centerPoint, CoordinateSystemPlane arcPlane)	PolylineSegmentType SegmentType int segmentStartIndex, int numberOfPoints, double angle, UDPPosition centerPoint, CoordinateSystemPlane arcPlane)
UDPPolylineDefinition	UDPPolylineDefinition() UDPPolylineDefinition(List_of_UDPPosition positions, List_of_UDPPolylineSegmentDefinition segDefs, int closed,	int IsClosed int IsCovered List_of_UDPPosition ArrayOfPosition List_of_UDPPolylineSegmentDefinition ArrayOfSegmentDefinition

Structure	Construction	Members
	int covered)	
UDPEqua- tionBasedCurveDefinition	UDPEqua- tionBasedCurveDefinition(string functionXt, string functionYt, string functionZt, double tStart, double tEnd, int numOfPointsOnCurve)	string FunctionXt string FunctionYt string FunctionZt double TStart double TEnd int NumOfPointsOnCurve
UDPEqua- tionBasedSurfaceDefinition	UDPEqua- tionBasedSurfaceDefinition(string functionXuv, string functionYuv, string functionZuv, double uStart, double uEnd, double vStart, double vEnd int reserved1 int reserved2)	string FunctionXuv string FunctionYuv string FunctionZuv double UStart double UEnd double VStart double VEnd two integer arguments that are reserved for future use. They need to be provided, for example as 0. For example: theSurfaceDefinition = UDPEqua- tionBasedSur- faceDefinition ("u", "v", "1", 0, 1, 0, 1, 0, 0)
UDPHelixDefinition	UDPHelixDefinition(int profileID, UDPPosition ptOnAxis, UDPPosition axisDir, double noOfTurns, bool isRightHanded, double radi- usChangePerTurn,	int ProfileID UDPPosition PtOnAxis UDPPosition AxisDir double NoOfTurns bool IsRightHanded double RadiusChangePerTurn double Pitch

Structure	Construction	Members
	double pitch)	
UDPSpiralDefinition	UDPSpiralDefinition(int profileID, UDPPosition ptOnAxis, UDPPosition axisDir, double noOfTurns, bool isRightHanded, double radiusChangePerTurn)	int ProfileID UDPPosition PtOnAxis UDPPosition AxisDir double NoOfTurns bool IsRightHanded double RadiusChangePerTurn
UDPBLNDElements	UDPBLNDElements(int partID, int noOfEdges; int* listOfEdges;) UDPBLNDElements(int partID, int noOfVertices; int* listOfVertices;)	UDPBLNDElements can hold either edges or vertices, but not both at the same time. Edges should be applied to solids, and vertices should be applied to sheets. int PartID /* part to be blended i.e. filleted/chamfered */ int noOfEdges; int* listOfEdges; /* edges to be blended */ int noOfVertices; int* listOfVertices; /* vertices to be blended */
UDPBLNDFilletOptions	UDPBLNDFilletOptions(bool suppressFillet, BLNDFilletRadiusLaw filletRadiusLaw, double filletStartRadius, double filletEndRadius, bool followSmoothEdgeSequence, BLNDFilletType filletType, double setbackDistance,	bool SuppressFillet /* Reserved for future */ BLNDFilletRadiusLaw FilletRadiusLaw double FilletStartRadius double FilletEndRadius bool FollowSmoothEdgeSequence /* Reserved for future */ BLNDFilletType FilletType double SetbackDistance

Structure	Construction	Members
	double bulgeFactor)	double BulgeFactor /* Reserved for future */
UDPBLNDChamferOptions	UDPBLNDChamferOptions(bool supressChamfer, BLNDChamferRangeLaw chamferRangeLaw, double chamferLeftRange, double chamferRightRange)	bool SupressChamfer BLNDChamferRangeLaw ChamferRangeLaw double ChamferLeftRange double ChamferRightRange
UDPPProgress	UDPPProgress(int prog, int subProg, string mesg, string subMesg)	int Prog int SubProg string Mesg string SubMesg
UDMInfo	UDMInfo(string name, string purpose, string company, string date, string version)	string Name string Purpose string Company string Date string Version
UDMDefinition	UDMDefinition() UDMDefinition(string name, UDParam value, ParamPropType propType, ParamPropFlag propFlag)	string DefName UDPPParam DefValue ParamPropType PropType ParamPropFlag PropFlag
UDMOption	UDMOption() UDMOption(string name, UDParam value,	string OptName UDPPParam OptValue ParamPropType PropType ParamPropFlag PropFlag

Structure	Construction	Members
	ParamPropType propType, ParamPropFlag propFlag)	
UDMPParameter	UDMPParameter() UDMPParameter(string name, UDParam value, UnitType unitType, ParamPropType propType, ParamPropFlag propFlag)	string ParamName UDPParam ParamValue UnitType UnitType ParamPropType PropType ParamPropFlag PropFlag

UDP/UDM Constants

Full names of enum constants must be used in scripts.

Example:

```
unitType = UnitType.LengthUnit
```

```
dataType = ParamDataType.Int
```

Enum constants:

enum Constant	Parameters
UnitType	NoUnit LengthUnit AngleUnit
ParamDataType	Int Double String Bool Position Unknown
ParamPropType	Text Menu Number

enum Constant	Parameters
	Value FileName Checkbox Position Unknown
ParamPropFlag	NoFlag ReadOnly MustBeInt MustBeReal Hidden Unknown
CoordinateSystemAxis	XAxis YAxis ZAxis
CoordinateSystemPlane	XYPlane YZPlane ZXPlane
SweepDraftType	ExtendedDraft RoundDraft NaturalDraft MixedDraft
SplitWhichSideToKeep	SplitKeepBoth SplitKeepPositiveOnly SplitKeepNegativeOnly
PolylineSegmentType	LineSegment ArcSegment SplineSegment AngularArcSegment
MessageSeverity	WarningMessage ErrorMessage

enum Constant	Parameters
	InfoMessage IncompleteMessage FatalMessage
BLNDFilletRadiusLaw	BLNDConstantRadius BLNDVariableRadius
BLNDFilletType	BLNDRound /* The outward surface of the fillet is curved.*/ BLNDMitered /* The outward surface of the fillet is flat and cut at an angle.*/
BLNDChamferRangeLaw	BLNDConstantRange BLNDVariableRange
PartPropertyFlags	PropNonModel PropDisplayWireFrame PropReadOnly PostprocessingGeometry PropInvisible PropShowDirection PropDummy

UDP Python Example

This Python script example demonstrates how to use the UDPBLNDElements structure and the UDP chamfer and fillet functions.

```
import sys

primitive_info = UDPPrimitiveTypeInfo(
    name="Fillet_Chamfer",
    purpose="Fillet Chamfer Example",
    company="Ansys",
    date="09/11/2020",
    version="1.0")

primitive_param_definitions = [
    UDPPrimitiveParameterDefinition2(
        "x_size",
```



```

    "",
    UnitType.LengthUnit,
    ParamPropType.Value,
    ParamPropFlag.MustBeReal,
    UDPParam(ParamDataType.Double, 10)),
UDPPrimitiveParameterDefinition2(
    "y_size",
    "",
    UnitType.LengthUnit,
    ParamPropType.Value,
    ParamPropFlag.MustBeReal,
    UDPParam(ParamDataType.Double, 5)),
UDPPrimitiveParameterDefinition2(
    "z_size",
    "",
    UnitType.LengthUnit,
    ParamPropType.Value,
    ParamPropFlag.MustBeReal,
    UDPParam(ParamDataType.Double, 2))
]
length_units = "mm"

#####
# Class Implementation
#####
class UDPExtension(IUDPExtension):
    def CreatePrimitive2(self, func_lib, param_values):
        """
        Inbuilt function that is called to generate a UDP after suc-
        cessful validation

```

```
:param func_lib: drawing inbuilt class, see in Help: UDMFunctionLibrary

:param param_values: list of udp parameter values (user input)
generated by UDP Core

:return: None

"""

param_dict = self.get_param_dict(param_values)

start_point = UDPPosition(0, 0, 0)
box = func_lib.CreateBox(start_point, [
    param_dict["x_size"],
    param_dict["y_size"],
    param_dict["z_size"]
])

# points on the middle of 4 vertical edges
points = [
    [0, 0, param_dict["z_size"]/2],
    [param_dict["x_size"], 0, param_dict["z_size"]/2],
    [param_dict["x_size"], param_dict["y_size"], param_dict["z_size"]/2],
    [0, param_dict["y_size"], param_dict["z_size"]/2]
]

edges = [func_lib.GetEdgeIDFromPosition(UDPPosition(point[0],
point[1], point[2])) for point in points]

fillet_rad = 0.1 * param_dict["x_size"] # 10% of X size
fillet_opt = UDPBLNDFilletOptions(True, BLNDFilletRadiusLaw.BLNDConstantRadius, fillet_rad, 0.0, True, BLNDFilletType.BLNDRound, 0.0, 0.0)
```

```
chamfer_length = 0.1 * param_dict["x_size"] # 10% of X size

chamfer_opt = UDPBLNDChamferOptions(False, BLNDChamferRangeLaw.BLNDConstantRange, chamfer_length, 0.0)

# select your geometry to which to apply operations
blend_element = UDPBLNDElements(box)

# specify attribute ListOfEdges to which edges to apply fillet operation
blend_element.ListOfEdges = edges[0:2]
func_lib.Fillet(blend_element, fillet_opt)

# redeclare attribute ListOfEdges to which edges to apply chamfer operation
blend_element = UDPBLNDElements(box)
blend_element.ListOfEdges = edges[2:4]
func_lib.Chamfer(blend_element, chamfer_opt)

# Provide to the user Info message indicating success
func_lib.AddMessage(MessageSeverity.InfoMessage, "Completed!")

def GetPrimitiveTypeInfo(self):
    return primitive_info

def GetLengthParameterUnits(self):
    return length_units

def GetPrimitiveParametersDefinition2(self):
    return primitive_param_definitions

def AreParameterValuesValid2(self, error, udp_params):
```

```
    return True

# Custom Functions

def get_param_value_by_name(self, param_values, param_name):
    """
    Function to get a value of a single parameter accessing it by
    name

    :param param_values: list of udp parameter values (user input)
    generated by UDP Core

    :param param_name: name of the parameter as specified in defin-
    ition list

    :return: Value of the parameter or None if parameter does not
    exist
    """
    param_dict = self.get_param_dict(param_values)
    value = param_dict.get(param_name, None)
    return value

def get_param_dict(self, param_values):
    """
    Function to return a dictionary of UDP parameter name and value
    (key: value) pairs

    :param param_values: list of udp parameter values (user input)
    generated by UDP Core

    :return: dict of parameter name and values
    """
    udm_param_def = self.GetPrimitiveParametersDefinition2()
    param_dict = {}
    for i, param in enumerate(udm_param_def):
        param_value = param_values[i].Data
        if str(param.PropType) != "Menu":
            param_dict[param.Name] = param_value
```

```

else:

    param_dict[param.Name] = param_value.replace("'", '').split
    ("",") [0]

return param_dict

```

Introduction to CPython

An Ansys Electronics Desktop 2024R1 feature allows CPython to be used for standalone scripting, as an alternative to IronPython, to:

- Launch Ansys Electronics Desktop ([InitializeNew](#))
- Connect with a running instance of Ansys Electronics Desktop ([Initialize](#))
- Execute Ansys Electronics Desktop script functions

One advantage of CPython is the large set of libraries and tools that are available. See below for instructions on modifying a script so that it can be launched with CPython interpreters.

CPython Script Engine and ansysedt process are communicated using GRPC, the ansysedt process started as GRPC server, the Script Script Engine acted as the GRPC client.

Important:

(Supported on both Windows and Linux in 24R1)

Launching Ansys Electronics Desktop on a remote machine is not supported in this release. Initialize() will connect to a remote instance that is already running on the remote machine, but cannot launch a new instance on the remote machine.

Connect() to an opened project in remote machine needs to use a share network path.

Creating an External Script

While [the same as IronPython](#) when run externally, a CPython recorded script must be modified by adding the following lines to the beginning of your script *before* `import ScriptEnv`

```

import sys

# Imports the sys module containing system-specific functions native to Python.

sys.path.append(r"<InstallationPath>/PythonFiles/DesktopPlugin")

# Adds the PythonFiles/DesktopPlugin subfolder to the list of directories Python searches for modules and files.

```

Those lines are followed by:

```

import ScriptEnv

```

```
# This imports ScriptEnv.py from the installation path specified above.
```

Follow that with:

- Either `InitializeNew()` or `Initialize()`, as described below.
- Any desired Electronics Desktop scripting commands.
- Closing command, as described below.

Start ansysedt as GRPC server

But default ansysedt process will be started as GRPC server. When there is no argument provided ansysedt.exe it start listening on the first available port from 50051.

-grpcsrv Flag

`ansysedt -grpcsrv <optional port number or port range>.`

`ansysedt -grpcsrv`

If no Port Number specified, same as the default like no grpcsrv flag

`ansysedt -grpcsrv portNumber`

ansysedt process will be listening on the port number, but report error as the port is used by other application.

`ansysedt -grpcsrv FirstPortNumber:LastPortNumber`

ansysedt process will be listening on the first available port within the port range, but report error if all ports in the range used.

`ansysedt -grpcsrv FirstPortNumber:NumberOfPorts`

ansysedt process will be listening on the first available port within the port range, but report error if all ports in the range used.

Note: If the last number is smaller than the first number the last number will be treated as the number of ports. (50051: 50250 has the same range as 50051:200)

Connect Functions:

1. `Connect(projectPath)`
 - a. Connect to the opened project.
 - b. Launch `ansysedt.exe`, open the project, then connect to it.
 - c. Error if the project does not exist.
2. `Connect(portNumber)` Connect to running ansysedt process in the local machine. Error if no ansysedt process listening on this port.
3. `Connect(machine, projectPath)` connect to an open project in remote Machine. `projectPath` must be a network path. Error if no ansysedt process opened the project.

4. `Connect(machine, portNumber)` connect to a running `ansysedt` process on the remote machine. Error if no `ansysedt` process listening on this port.

Example:

```
import sys

sys.path.append(r"<InstallationPath>/PythonFiles/DesktopPlugin")

import ScriptEnv

ScriptEnv.Connect(r"c:\myProjects\Project1.aedt")

oProject = oDesktop.GetActiveProject()

print(oProject.GetName())
```

Launching Electronics Desktop

To launch a new instance of Electronics Desktop and connect `oApplication` and `oDesktop` to it:

```
InitializeNew(NonGraphical = <True|False>, Module = None, Machine =
"", Port = <Port#>)
```

Where:

- **NonGraphical** – Specifies whether to launch Electronics Desktop in non-graphical mode.
- **Module** – Behavior remains unchanged from [Iron Python](#) and should be left defaulted to "None." See the code in `ScriptEnv.py` for more details.
- **Machine** – Currently an empty string, as `InitializeNew()` will only launch Electronics Desktop on the current machine.
- **Port** – Electronics Desktop will launch using the first unused port it finds starting at `<Port#>`. If `Port = 0`, the starting port will be 50051.

Note:

`InitializeNew()` will *always* launch a new instance of Electronics Desktop. Please use `Initialize()` to connect to an existing instance. See below.

Connecting with a Running Instance of Electronics Desktop

To connect `oApplication` and `oDesktop` to an existing Electronics Desktop instance, or launch a new instance and connect to it if necessary:

```
Initialize(name, NG = <True|False>, machine = "", port = <Port#>)
```

Where:

- **Name** – Ignored.
- **NG** – If launch is necessary, specifies whether to launch Electronics Desktop in non-graphical mode.

- **Machine** – The machine on which to launch/connect. For current machine, pass empty string or use localhost.
- **Port** – If port is nonzero, the script tries to connect to an existing instance on <port#> running on <machine>. If there is no instance running on that <port>, a new instance of Electronics Desktop launches on that port and then connects to it. If port = 0, the new instance is launched on the first free port, starting at 50051.

Closing Electronics Desktop/Ending the Script

To close Electronics Desktop, add the following line to the end of the script:

```
ScriptEnv.Shutdown()  
  
# This stops ScriptEnv.py. If you are running multiple scripts,  
include this only at the end of the last script.
```

-grpcsrv Flag

This flag will launch the application in a mode where the executable serves as a scripting server that can be used for CPython scripting in conjunction with the CPython stand alone scripting instructions that were mentioned earlier. The -ng flag can be combined with -grpcsrv.

On Windows:

```
ansysedt.exe -grpcsrv <optional port number>.
```

On Linux:

```
ansysedt -grpcsrv <optional port number>.
```

If the port number is omitted, the default of 50051 will be used.

With -grpcsrv, a message will be displayed in the **Messages** window indicating that the server was started. If the requested port is in use by another application, starting the sever may fail.

Related Topics:

[Standalone Scripting in Iron Python](#)

Ansyes Electronics Desktop Scripting

This chapter provides an overview of scripting in Ansys Electronics Desktop.

[Overview of Ansys Electronics Desktop Scripting Objects](#)

[Running a Script](#)

[Recording a Script](#)

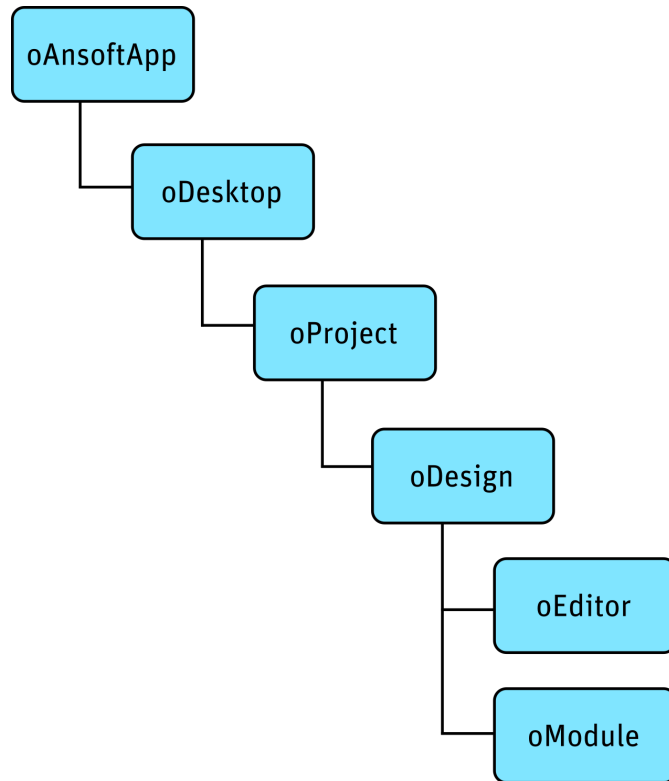
[Working with Project Scripts](#)

[Executing a Script from Within a Script](#)

[Ansys Electronics Desktop Scripting Conventions](#)

Overview of Electronics Desktop Scripting Objects

When you record a script using Ansys Electronics Desktop, the beginning of the script must contain some standard commands, as illustrated in the following chart. The commands in the chart define the objects used by Electronics Desktop in the script and assign values to these objects. The objects are used in the hierarchical order shown.



The commands are described below, followed by examples.

oAnsoftApp

The **oAnsoftApp** object provides a handle for Iron Python or VBScript to access the `Ansoft.ElectronicsDesktop` product.

In Iron Python, for example:

```
oAnsoftApp = CreateObject('Ansoft.ElectronicsDesktop')
```

In VBScript, for example:

```
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
```

oDesktop

The **oDesktop** object is used to perform desktop-level operations, including project management.

In Iron Python, for example:

```
oDesktop = oAnsoftApp.GetAppDesktop()
```

In VBScript, for example:

```
Set oDesktop = oAnsoftApp.GetAppDesktop()
```

Consult the following for details about script commands recognized by `oDesktop`:

- [Desktop Object Script Commands](#).

oProject

The **oProject** object corresponds to one project open in Electronics Desktop. It is used to manipulate the project and its data. Its data includes variables, material definitions, and one or more designs.

In Iron Python, for example:

```
oProject = oDesktop.GetActiveProject()
```

In VBScript, for example:

```
Set oProject = oDesktop.GetActiveProject()
```

Consult the following for details about script commands recognized by `oProject`:

- [Project Object Script Commands](#)

oDesign

The **oDesign** object corresponds to a design in the project. This object is used to manipulate the design and its data, including variables, modules, and editors.

In Iron Python, for example:

```
oDesign = oProject.GetActiveDesign()
```

In VBScript, for example:

```
Set oDesign = oProject.GetActiveDesign()
```

Consult the following for details about script commands recognized by `oDesign`:

- [Design Object Script Commands](#)
- [Output Variable Script Commands](#)
- [Reporter Editor Script Commands](#)

oEditor

The **oEditor** object corresponds to an editor, such as the 3D Modeler, Layout, or Schematic editor. This object is used to add and modify data in the editor.

In Iron Python, for example:

```
oEditor = oDesign.SetActiveEditor('3D Modeler')
```

In VBscript, for example:

```
Set oEditor = oDesign.SetActiveEditor("3D Modeler")
```

Consult the following for details about script commands recognized by `oEditor`:

- [3D Modeler Editor Script Commands](#)
- [Schematic Editor Script Commands](#)

Important:

There is no Reporter Editor object for `oEditor`. Reporter Editor commands are executed by `oDesign`.

See: [Reporter Editor Script Commands](#).

oModule

The `oModule` object corresponds to a module in the design. Modules are used to handle a set of related functionalities.

In IronPython, for example:

```
oModule = oDesign.GetModule('BoundarySetup')
```

In VBscript, for example:

```
Set oModule = oDesign.GetModule("BoundarySetup")
```

Consult the following for details about script commands recognized by `oModule`:

- Analysis Module Script Commands
- Boundary and Excitation Module Script Commands
- Field Overlays Module Script Commands
- Mesh Operations Module Script Commands
- Optimetrics Module Script Commands
- Radiation Module Script Commands
- Reduce Matrix Module Script Commands
- Solutions Module Script Commands

Example Script Opening

Combining the above objects, a script in Iron Python could begin like the following:

```
oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
oDesktop = oAnsoftApp.GetAppDesktop()
oProject = oDesktop.SetActiveProject("Project1")
oDesign = oProject.SetActiveDesign("Design1")
```

```
oEditor = oDesign.SetActiveEditor("3D Modeler")  
oModule = oDesign.GetModule("BoundarySetup")
```

In VBscript, this would be:

```
Dim oAnsoftApp  
Dim oDesktop  
Dim oProject  
Dim oDesign  
Dim oEditor  
Dim oModule  
  
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")  
Set oDesktop = oAnsoftApp.GetAppDesktop()  
Set oProject = oDesktop.SetActiveProject("Project1")  
Set oDesign = oProject.SetActiveDesign("Design1")  
Set oEditor = oDesign.SetActiveEditor("3D Modeler")  
Set oModule = oDesign.GetModule("BoundarySetup")
```

GetActiveProject and GetActiveDesign for Wider Use

The sample script above only works for "Design1" within "Project1". To create a script that is usable for any open project, you can use one or both of `GetActiveProject` and `GetActiveDesign`.

In IronPython:

```
oProject = oDesktop.GetActiveProject()  
oDesign = oProject.GetActiveDesign()
```

In VBscript:

```
Set oProject = oDesktop.GetActiveProject()  
Set oDesign = oProject.GetActiveDesign()
```

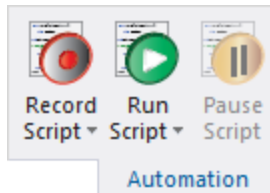
Running a Script

Electronics Desktop scripts can be run from within the software or from the command line.

Within Electronics Desktop

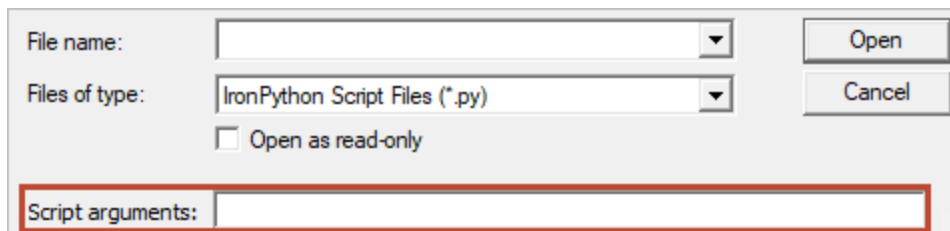
To run scripts in Electronics Desktop:

1. Click **Tools > Run Script**, or select the **Automation** tab and click the **Run Script** icon:



The **Run Script** file browser appears.

2. Use the file browser to locate the script file (*.vbs, *.py, or *.js).
3. If desired, type script arguments in the Script Arguments field:



You can access script arguments using the **AnsoftScriptHost.arguments** collection from VBScript. This is a standard COM collection.

4. Click **Open**.

Electronics Desktop executes the script.

While script execution is in progress, the **Run Script** button transforms into a **Stop Script** button. Click **Stop Script** to stop the script execution.

To temporarily pause a running script, click **Pause Script**. This button transforms into a **Resume Script** button, which you can click to resume script execution.

From the Command Line

To run a script from a command line, add the `-runscriptandexit` or `-runscript` argument to the Electronics Desktop command line syntax.

To use script arguments, add the `-scriptargs` parameter and specify the arguments. For example:

```
ansyedt.exe -scriptargs "hello there"
```

In Iron Python, the command line parameter following `-scriptargs` is passed without modification as a single string in the `ScriptArgument` python variable.

In VBScript, the command line parameter following `-scriptargs` is split into multiple strings and converted to a VBScript collection which is accessible via the **AnsoftScript.Arguments** collection. To access these arguments, for example:

```
msgbox AnsoftScript.Arguments(0) // Returns 'hello'

msgbox AnsoftScript.Arguments(1) // Returns 'there'
```

For more information about running a script from the command line, consult the Maxwell help topic "Running Maxwell from the Command Line".

Direct Launch

If you run a script directly from the command line without launching Electronics Desktop, script arguments will be in the **WSH.arguments** collection instead of the **AnsoftScriptHost.arguments** collection. The following script ensures the arguments are accessed regardless of the collection in which they reside:

```
on error resume next

dim args

Set args = AnsoftScript.arguments

if(IsEmpty(args)) then

Set args = WSH.arguments

End if

on error goto 0

// At this point, args has the arguments regardless of collection

msgbox "Count is " & args.Count

for i = 0 to args.Count - 1

msgbox args(i)

next
```

Recording a Script

Electronics Desktop can record a script based on UI actions and save this script in either IronPython (*.py) or VBScript (*.vbs) format.

Scripts can be saved to an [external file](#), or [to the project](#).

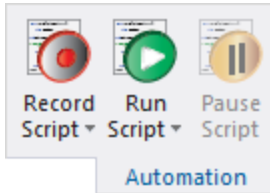
Important:

When you record a script, every subsequent action you take is recorded. You must manually stop recording.

Recording a Script to File

To record a script to file:

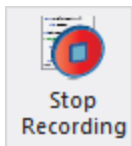
1. Click **Tools > Record Script to File**, or select the **Automation** tab and click the **Record Script** icon:



A **Save As** file browser appears.

2. Navigate to the location where you want to save the script.
3. In the **File Name** field, type a name for the script file.
4. Use the **Save as Type** drop-down menu to select either IronPython or VBScript.
5. Click **Save**.

The **Record Script** button transforms into a **Stop Recording** button, and Electronics Desktop begins recording your actions.



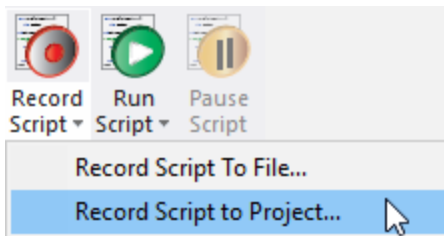
6. Perform the steps you want to record.
7. When you have finished recording the script, click **Stop Recording**, or select **Tools > Stop Script Recording**.

The recorded script is saved to the folder you specified.

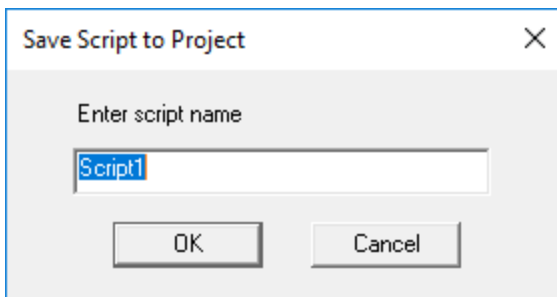
Recording a Script to a Project

To record a script to a project:

1. Click **Tools > Record Script to Project**, or select the **Automation** tab and use the **Record Script** drop-down menu to select **Record Script to Project**.



The **Save Script to Project** dialog box appears:



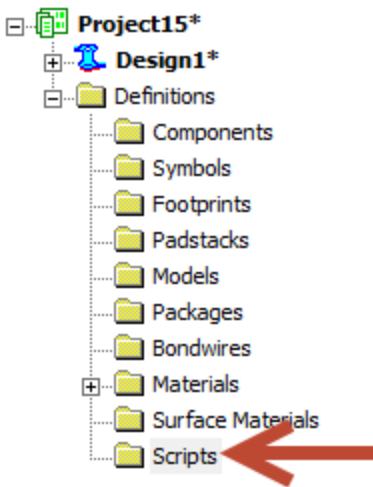
2. Enter a name for the script in the text box, then click **OK**.
3. Perform the steps you want to record.
4. When you have finished, click **Stop Recording**, or select **Tools > Stop Script Recording**.

The recorded script is saved to *scriptname.vbs* in the Scripts library and can be accessed from the Project Manager. See: [Working with Project Scripts](#).

Working with Project Scripts

Scripts can be [recorded to a project](#).

Once a script has been recorded to the project, you can manage it in the **Project Manager** from the **Definitions** folder:



Individual scripts appear in this folder. Right-click a script to edit or run it:



You can also run project scripts from the **Automation** tab by selecting **Run Script > Project Scripts > [Script Name]**.

Note:

Project scripts are stored in the project scripts library. Refer to the topic "Managing Library Contents" for information on working with libraries.

Executing a Script from Within a Script

Electronics Desktop provides a script command that enables you to launch another script from within the script that is being executed.

In IronPython:

```
oDesktop.RunScript(<ScriptName>)
```

In VBScript:

```
oDesktop.RunScript <ScriptName>
```

If the full path to the script is not specified, Electronics Desktop searches for the specified script in the following locations, in order:

1. Personal Library Directory (PersonalLib)
2. User Library Directory (UserLib)
3. System Library Directory (SysLib)
4. Installation Directory

Each of the library directories can be specified in Electronics Desktop under **Tools > Options > General Options**, on the **Project Options** tab.

Electronics Desktop Scripting Conventions

A number of scripting conventions exist for Electronics Desktop regarding syntax, arguments, and numerical values.

Consult the following topics:

- [Named Arguments](#)
- [Setting Numerical Values](#)
- [Event Callback Scripting](#)

Named Arguments

Many Electronics Desktop script commands use named arguments. The names can appear in three ways:

1. Named data, where name precedes data.

For example:

```
..., "SolveInside:=", true, ...
```

2. Named Array, where name precedes array.

For example:

```
..., "Attributes:=", Array(...), ...
```

3. Named Array, where name is inside an array.

For example:

```
..., Array("NAME:Attributes", ...), ...
```

In the first and second examples, the name is formatted as "**<Name> : =**". This signals to Electronics Desktop that this is a name for the next argument in the script command. In the third example, the name is formatted as "**NAME : <name>**" and is the first element of the array.

The names are used both to identify what the data means to you and to inform Electronics Desktop which data is being given. The names must be included or the script will not play back correctly. However, if you are writing a script, you do not need to pass in every piece of data that

the command can take. For example, if you are modifying a boundary, the script will be recorded to include every piece of data needed for the boundary, whether or not it was modified. If you are writing a script by hand, you can add only the data that changed and omit anything that you do not want to change. Electronics Desktop will use the names to determine which data you provided.

VBscript Example

For example, when editing an impedance boundary, Electronics Desktop records the `EditImpedance` command as follows in VBscript:

```
oModule.EditImpedance "Imped1", Array("NAME:Imped1",  
    "Resistance:=", "100", "Reactance:=", "50",  
    "InfGroundPlane:=", false)
```

If you only wish to change the resistance, you can leave out the other data arguments when manually writing a script:

```
oModule.EditImpedance "Imped1", Array("NAME:Imped1",  
    "Resistance:=", "100")
```

IronPython Example

When editing a port excitation, Electronics Desktop records the `Edit` command as follows in IronPython:

```
oModule.Edit("Port1",  
    ["NAME:Port1",  
        ["NAME:Properties",  
            "PortSolver:=", "true",  
            "Phase:=", "0deg",  
            "Magnitude:=", "2mA",  
            "Impedance:=", "50Ohm",  
            "Theta:=", "0deg",  
            "Phi:=", "0deg",  
            "PostProcess:=", "false",  
            "Renormalize:=", "50Ohm + 0i Ohm",  
            "Deembed:=", "0mm",  
            "RefToGround:=", "false"  
        ],  
        "Type:=", "EdgePort",
```

```
"IsGapSource:=", true,  
"UpperProbe:=", false,  
"LayoutObject:=", "Port1",  
"Pin:=", "",  
"ReferencePort:=", ""  
]  
)
```

If you only wish to change the magnitude, you can leave out the other data arguments when manually writing a script:

```
oModule.Edit("Port1",  
    ["NAME:Port1",  
        ["Magnitude:=", "1mA"]  
    ]  
)
```

Setting Numerical Values

For script arguments that expect a number, the following options are possible:

- Pass in the number directly.

In IronPython:

```
oModule.EditVoltage('Voltage1', ['NAME:Voltage1', 'Voltage:=',  
3.5])
```

In VBScript:

```
oModule.EditVoltage "Voltage1", Array("NAME:Voltage1",  
"Voltage:=", 3.5)
```

- Pass in a string containing the number with units.

In IronPython:

```
oModule.EditVoltage('Voltage1', ['NAME:Voltage1', 'Voltage:=',  
'3.5V'])
```

In VBScript:

```
oModule.EditVoltage "Voltage1", Array("NAME:Voltage1",  
"Voltage:=", "3.5V" )
```

- Pass in a variable name.

In IronPython:

```
var = 3.5

oModule.EditVoltage('Voltage1', ['NAME:Voltage1', 'Voltage:=',
var])
```

In VBScript:

```
dim var

var = "3.5V"

oModule.EditVoltage "Voltage1", Array("NAME:Voltage1",
"Voltage:=", var)
```

Event Callback Scripting

Event Callback scripting allows you to define custom JavaScript and VBScript routines that will run automatically after detecting a triggering event, such as placing a component or running a simulation. When defining an Event Callback script, specify one or more scripts that will be run after a particular event is detected.

A callback script can only access functions and other scripts defined by its callback definition. For example, a Simplorer callback script can call PropHost.GetValue — and all other PropHost functions — but only from scripts defined in the Property Dialog callback. As a result, "PropHost" is a Simplorer script item that is only visible in "Property" callback scripts. For more information, see [Callback Scripting Using PropHost Object](#) and [Callback Scripting Using Compliance Object](#).

The following table lists allowable callback events, items that are visible from the associated callback script, and the set of accessible functions that can be called.

Callback Event	Scripts Visible from the Event Callback Script	Functions Callable from the Visible Script
Place Component	Compliance	Compliance.GetParentDesign() — Returns a oDesign item that can be used to call Design functions. Compliance.GetPropservername() – Returns a Compliance identification name that can be used in oEditor property-method scripts, such as GetPropertyValue(), SetPropertyValue(), etc. Compliance.GetComponentName() — Returns the component name, e.g. "MS_TRL".

Simulate Component	ComplInstance	ComplInstance.GetDesign() — Returns the interface to the specified design simulation. ComplInstance.GetProgress() — Returns the completion percentage (from 0 to 100) of the specified design simulation. ComplInstance.GetRunStatus() — Returns the status number of the specified design simulation. ComplInstance.Abort() — Aborts the specified design simulation.

The function, **ExecuteAnsoftScript(<ScriptName>)**, searches the configured Ansys Electronics Desktop script libraries by name for the script passed to it, and invokes the found ScriptName. The invoked script will run with the same set of visible script items as the originally called script. That is, **ComplInstance** is visible from the invoked sub-script, ScriptName, and ComplInstance's functions can be called from ScriptName.

PyAEDT (Beta)

PyAEDT is a Python library that interacts with the AEDT API to make scripting simpler for the end user. It supports all AEDT 3D products (HFSS, Icepak, Maxwell 3D, and Q3D Extractor), 2D tools, Ansys Mechanical, EMIT, Circuit tools like Nexxim, system simulation tools like Twin Builder, and layout tools like HFSS 3D Layout and EDB. Additionally, it enables the end user to have a CPython interface with AEDT. Its class and method structures simplify operation for the end user, enabling more Pythonic code while reusing information as much as possible across the various APIs.

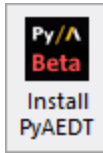
Documentation for PyAEDT can be found online at: <https://aedt-docs.pyansys.com/version/stable/>

Installing PyAEDT adds three items to the **Tools > Toolkit > PersonalLib** menu and to the **Automation** tab:

- **Console** – launches the PyAEDT console.
- **Jupyter Notebook** – launches Jupyter Notebook (a computational notebook) in an internet browser.
- **Run PyAEDT Script** – launches a file browser allowing you to select a Python script to run via PyAEDT.

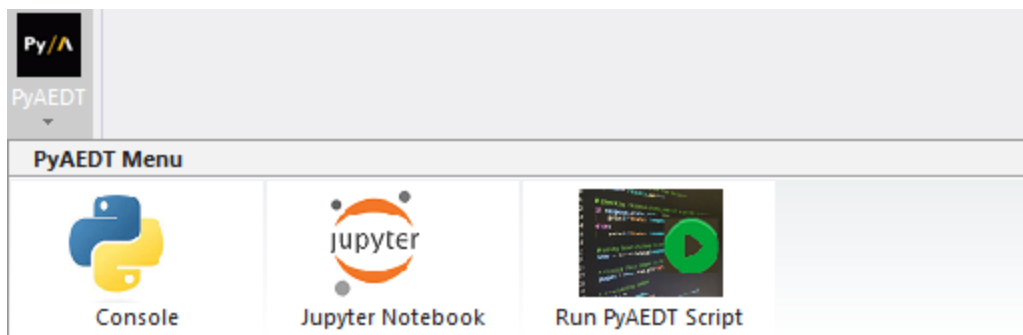
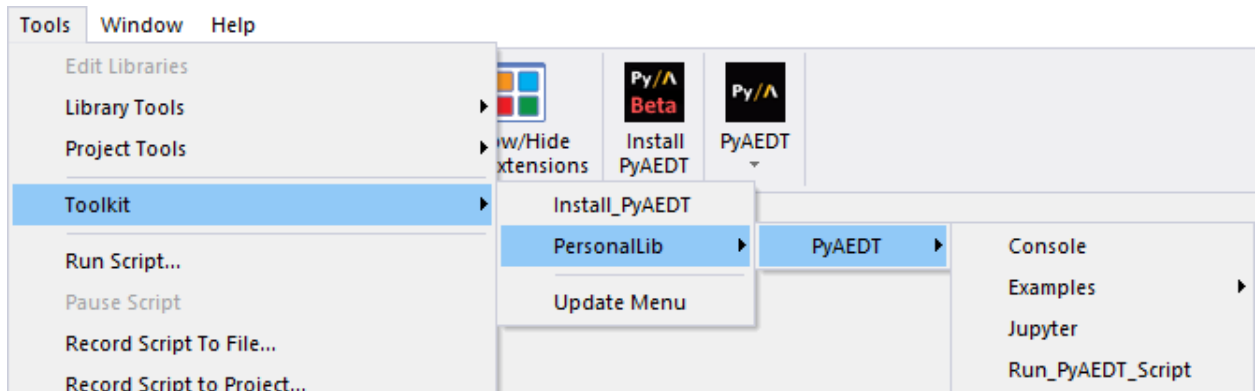
To install PyAEDT:

- From the **Automation** tab, click **Install PyAEDT**.



A web browser launches, and takes you to detailed installation instructions.

When installation is complete, the **Tools > Toolkit > PersonalLib** menu and the **Automation** tab update to display PyAEDT menu options:



This page intentionally
left blank.

2 - Object-Oriented Property Scripting

Object-oriented scripting enhances scripting in AEDT by allowing object-oriented access to retrieve or modify object properties. The primary gain is the ease with which properties of various existing objects in an AEDT project or design can be read, modified, and set. This feature also allows for much less code to be written to access object properties and enables much more readable code for users, avoiding complex array input.

This topic covers the following:

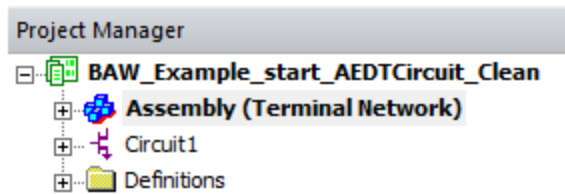
- Logic/syntax
- Basic attributes of project and design properties
- Example Scripts

Object-Oriented Scripting

There are five basic functions in object-oriented scripting for retrieving and setting properties:

1. GetChildNames()
2. GetChildObjects()
3. GetPropNames()
4. GetPropValue()
5. SetPropValue()

At a high level, use GetChildNames() to determine what object instances exist for a given object. An example is shown below to demonstrate for an AEDT Project shown that has two Designs.



If you open the Command Window that allows for executing python code, you first define the Project Object, oProject, and the Design Object, oDesign, as shown:

```
>>> oProject = oDesktop.GetActiveProject()  
>>> oDesign = oProject.GetActiveDesign()
```

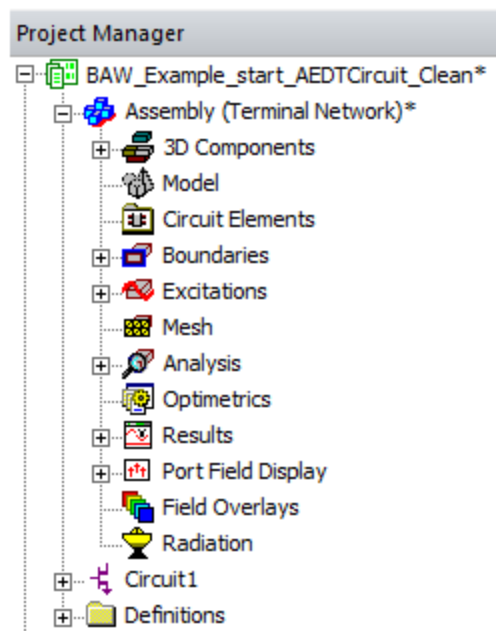
Once the objects have been defined, you can use GetChildNames() to learn what object instances exist for each. As an example, observe the Child Names of oProject, and you see a list of the Designs in the AEDT Project, per the GUI.

```
>>> oProject.GetChildNames()  
['Assembly', 'Circuit1']
```

As another example, retrieve the names of the Object Instances available in oDesign, to see the various objects associated with a Design setup:

```
>>> oDesign.GetChildNames()  
['Boundaries', 'Excitations', 'Circuit Elements', 'Model', 'Mesh', 'Analysis', 'Optimetrics', 'Port  
Field Display', 'Field Overlays', 'Radiation', 'Results', '3D Modeler']
```

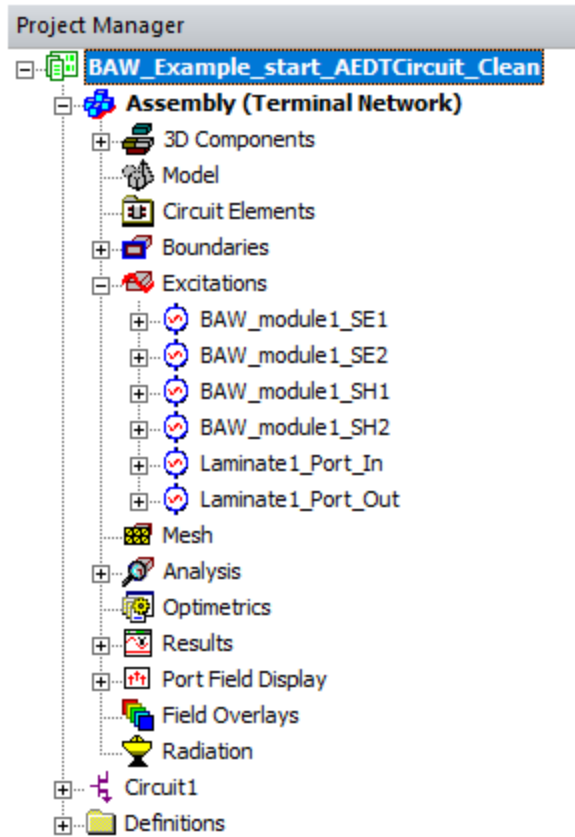
These names are what you would expect based on the Project Manager Layout:



In the **Project Manager** above, any object that has the '+' symbol is populated with children that you can query. Once you know the name of the object you want, you can instance it via the `GetChildObject()` command. This defines the instance to the desired object. As an example, set an instance for the Excitations:

```
>>> oExcitations = oDesign.GetChildObject('Excitations')
```

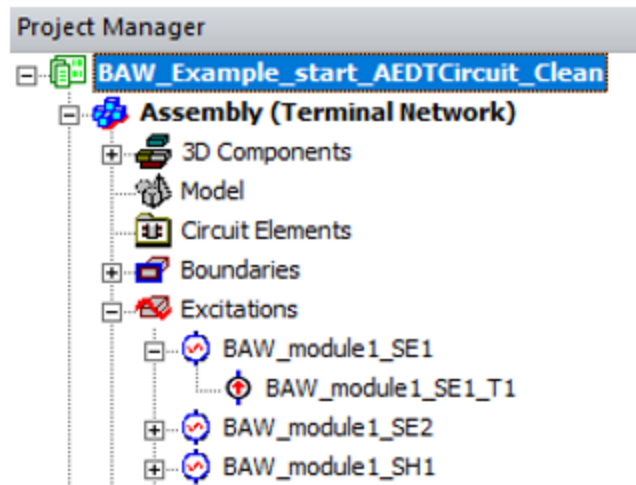
You now have an object, oExcitations defined to be the oDesign Child Object 'Excitations'. What does this mean? If you expand the Excitations dialogue in the Project Manager, you expect that the Child Names of this object would be the names of the Excitations as defined, in this case six ports:



```
>>> oExcitations.GetChildNames()  
['Laminat1_Port_In', 'Laminat1_Port_Out', 'BAW_module1_SE1', 'BAW_module1_SE2', 'BAW_module1_SH1',  
'BAW module1 SH2']
```

There is clear logic to the Object Child Names as the children of the oExcitations object as the ports that have been defined in HFSS. Looking at the Project tree can help you to conceive and retrieve desired information.

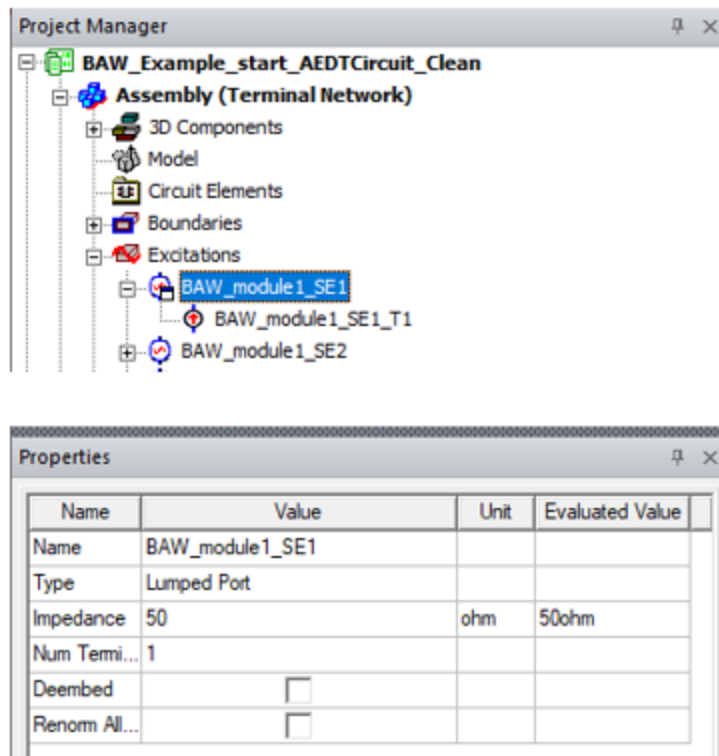
Expand the first port to see its expected Child Object, its Terminal, in the **Project Manager** Window:



Through scripting, first define the Port Object (in this example, oPort) using the 'GetChildObject()' command for the first port, 'BAW_module1_SE1.' Then determine its Object Child Name, the terminal definition:

```
>>> oPort = oExcitations.GetChildObject('BAW_module1_SE1')
>>> oPort.GetChildNames()
['BAW_module1_SE1_T1']
```

As the use and logic for `GetChildNames()` and `GetChildObject()` have been demonstrated, you can now explore the properties of each of these objects, if they exist. The function to determine what properties exist is `GetPropNames()`. Use this to determine what properties exist to be retrieved or modified for a given object. The properties available are readily identifiable in the **Property** window, by default located beneath the **Project Manager** window. For example, if you select a given port object, 'BAW_module1_SE1' the Property window populates as shown:



If you execute the `GetPropNames()` function on the previously defined object, `oPort`, you see the same Property Names as available in the **Properties** window:

```
>>> oPort.GetPropNames()  
['Name', 'Type', 'Impedance', 'Num Terminals', 'Deembed', 'Renorm All Terminals']
```

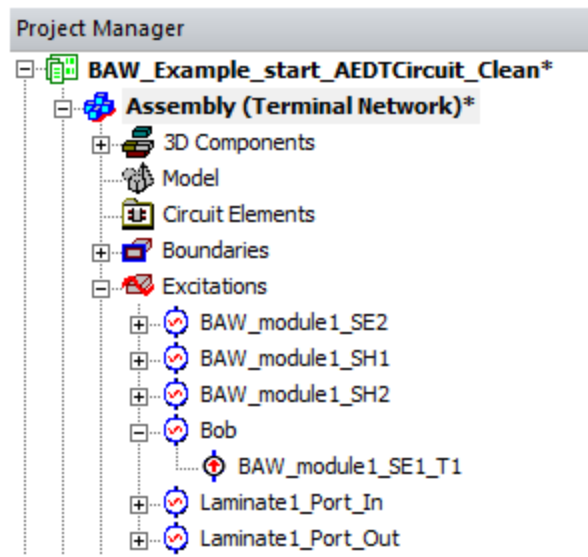
Once you identify the desired object and you know the desired property, you can access the value via `GetPropValue()`. For example, if you want to retrieve the name of the object `oPort`:

```
>>> oPort.GetPropValue('Name')  
'BAW_module1_SE1'
```

To change the value of the property, use the `SetPropValue()` function. The arguments for this function are (*Property Name*, *New Value*). For example, to change the name of the port to 'Bob':

```
>>> oPort.SetPropValue('Name', 'Bob')  
True
```

This function returns a Boolean 'True' if successful. The Project Manager window updates accordingly:



This approach for retrieving and setting properties is general and can be used for many aspects of an Ansys Electronics Desktop simulation. This Object-Oriented method of property identification and modification operates only on existing objects. Object-Oriented scripting cannot create new instances; you must revert to the functions in a given Module to do that. Not all Children of a given object may be accessible via the `GetChildNames()` command just yet. An example is given for Material property modification later in this App Note. However, if you need specific objects you can reference details in the Scripting Help or reach out to an Application Engineer.

Material Properties and Examples

This section discusses the material properties and how to access and modify them. Because materials are globally defined, the objects are children of the Project, `oProject`, as shown below:

```
>>> oProject = oDesktop.GetActiveProject()
>>> oMaterials = oProject.GetChildObject('Materials')
>>> oMaterials.GetChildNames()
['vacuum', 'Cap_Mat', 'Outline_Mat', 'SolderMask_Mat', 'copper', 'pec']
```

All materials with a Project Definition, or assigned to an object, in the Project are accessible. For example, assume you want to see the conductivity of 'copper.' Follow the same flow as in the previous section:

```
>>> oCopper = oMaterials.GetChildObject('copper')
>>> oCopper.GetPropNames()
['Coordinate System Type', 'Coordinate System Type/Choices', 'Relative Permittivity Type', 'Relative Permittivity Type/Choices', 'Relative Permittivity', 'Relative Permeability Type', 'Relative Permeability Type/Choices', 'Relative Permeability', 'Bulk Conductivity Type', 'Bulk Conductivity Type/Choices', 'Bulk Conductivity', 'Dielectric Loss Tangent Type', 'Dielectric Loss Tangent Type/Choices', 'Dielectric Loss Tangent', 'Magnetic Loss Tangent Type', 'Magnetic Loss Tangent Type/Choices', 'Magnetic Loss Tangent', 'Electric Coercivity Type', 'Electric Coercivity Magnitude', 'Magnetic Coercivity Type', 'Magnetic Coercivity Magnitude', 'Thermal Conductivity Type', 'Thermal Conductivity Type/Choices', 'Thermal Conductivity', 'Magnetic Saturation Type', 'Magnetic Saturation', 'Lande G Factor Type', 'Lande G Factor', 'Delta H Type', 'Delta H', '- Measured Frequency Type', '- Measured Frequency', 'Core Loss Model', 'Core Loss Model/Choices', 'Mass Density Type', 'Mass Density', 'Composition', 'Composition/Choices', 'Specific Heat Type', 'Specific Heat', 'Young's Modulus Type', 'Young's Modulus Type/Choices', 'Young's Modulus', 'Poisson's Ratio Type', 'Poisson's Ratio Type/Choices', 'Poisson's Ratio', 'Thermal Expansion Coefficient Type', 'Thermal Expansion Coefficient Type/Choices', 'Thermal Expansion Coefficient', 'Magnetostriiction Type', 'Inverse Magnetostriiction Type', 'Thermal Material Type', 'Thermal Material Type/Choices', 'Solar Behavior Type', 'Solar Behavior Type/Choices', 'Solar Behavior']
```

The Material Property of interest is "Bulk Conductivity." So you create a "Cond" object to store the value and use the GetPropValue function to obtain it. Then name the Cond object to see the value:

```
>>> Cond = oCopper.GetPropValue('Bulk Conductivity')
>>> Cond
'58000000'
```

To change the conductivity, use the `SetPropValue()` function as shown below:

```
>>> oCopper.SetPropValue('Bulk Conductivity', '100')
True
>>> NewCond = oCopper.GetPropValue('Bulk Conductivity')
>>> NewCond
'100'
```

Body Properties and Modification

The following example shows how to retrieve the properties of a Body in the model, in this case a Region object. Once you identify the desired property, you can modify it as needed.

```
>>> oModel = oDesign.GetChildObject('3D Modeler')
>>> oModel.GetChildNames()
['RadBox_Region_1']
>>> oRegion = oModel.GetChildObject('RadBox_Region_1')
>>> oRegion.GetPropNames()
['Name', 'Material', 'Solve Inside', 'Orientation', 'Orientation/Choices', 'Model', 'Group', 'Display
Wireframe', 'Material Appearance', 'Color', 'Color/Red', 'Color/Green', 'Color/Blue', 'Transparent']
```

Retrieving Variables

Retrieving defined variables in a Design or Project is a common effort for automation. There are two types of variables, Design and Project. Project variables are preceded with a '\$' symbol and are retrieved in the Project object as it is globally defined to all Designs. Design variables do not have any preceding symbols and are retrieved in the Design object as their scope is limited to a given Design. The following example demonstrates the retrieval of Project Variable names and values, and then Design variable names and values.:

```
>>> oProjVar = oProject.GetChildObject("Variables")
>>> oProjVar.GetPropNames()
['$test']
>>> oProjVar.GetPropValue('$test')
'0'
>>> oDesVar = oDesign.GetChildObject("Variables")
>>> oDesVar.GetPropNames()
['test']
>>> oDesVar.GetPropValue('test')
'0'
```

Retrieve Datasets and Values

The GetChildObject, GetChildTypes and GetChildNames functions operate on the oDesktop objects. This allows you to retrieve and view datasets and values. The dataset script wrapper store all values internally in SI units, and converts them back to user-supplied units when you request non-SI property values. For example, if you assigned a dataset to the example OptimTee project in HFSS, you could use these functions in the command window:

```
>>>oDesktop.GetChildTypes()

['Projects']

>>>oDesktop.GetChildNames()

['OptimTee']

>>>arrProjectNames = oDesktop.GetChildNames()

>>>tp = oDesktop.GetChildObject('OptimTee')

>>>tp.GetChildTypes()
```

```
['Design', 'Project Data']
>>>tp.GetChildNames('Project Data')
['Variables', 'Materials', 'Surface Materials', 'Datasets']
>>>ds = tp.GetChildObject('datasets')
>>>ds.GetChildNames()
['$ds1']
>>>ds1=ds.GetChildObject('$ds1')
>>>ds1.GetPropValue('[:,:]')
[[1.0, 4.0], [2.0, 5.0], [3.0, 6.0]]
>>>ds1.GetPropSIValue()
[[1.0, 4.0], [2.0, 5.0], [3.0, 6.0]]
>>>ds1.DimUnits
>>>ds1.DimUnits = ['mm','mm']
>>>ds1.DimUnits
['mm', 'mm']
>>>ds1.GetPropSIValue()
[[0.001, 0.00400000000000000001], [0.002, 0.00500000000000000001], [0.00300000000000000001,
0.00600000000000000001]]
>>>ds1.GetPropValue('[:,:]')
[[1.0, 4.0], [2.0, 5.0], [3.0, 6.0]]
```

GetSolutionData API

Many users want to use scripts to extract solution data from Ansys Electronics Desktop for custom Post Processing. Scripting includes a new method to do this without having to export data to a file and then re-import it for use in a script. The new function is accessible via the “ReportSetup” Module. The function call is “GetSolutionDataPerVariation()”. A code snippet to extract Terminal S Parameter data is shown:

```
8 oModule=.oDesign.GetModule("ReportSetup")
9 Results=.oModule.GetSolutionDataPerVariation("Terminal.Solution.Data", "Setup1::Sweep1:",
10 ..... [
11 ..... "Domain:=", "Sweep"
12 ..... ],
13 ..... [
14 ..... "Freq:=", ["All"]
15 ..... ],
16 ..... [
17 ..... "dB(St(Terminal_1,Terminal_1))"
18 ..... ]
19 ##.Get.Dependent.and.Independent.Variable.data.for.Nominal.Variation
20 NominalData=.Results[0]
21 ##.Get.Independent.Variable.Data
22 ##.For.second.argument,.if.pass.True.then.data.is.in.SI.Units
23 ##.if.pass.False.then.data.is.in.default.scale.units.instead.of.SI
24 SweepValues=.NominalData.GetSweepValues("Freq", True)
25 ##.Get.Dependent.Variable.DataValues
26 ##.Note:.Can.pass.any.'Y.Component'.Name
27 DataValues=.NominalData.GetRealDataValues("dB(St(Terminal_1))")
```

The above code shows how you can extract the Dependent and Independent data to variables for easy manipulation. For more information on other functions available for this, see [GetSolutionDataPerVariation](#).

Summary

Scripting has been advancing in Ansys Electronics Desktop to better allow you to customize and automate their repetitive or complex simulations. The ability to easily retrieve and set property values via the Object-Oriented scripting allows for ease of both writing and

reading. The ability to extract solution data within a script execution is a new functionality that markedly enables more advanced post processing.

Object oriented property scripting presents an easy to use, intuitive and object oriented representation of the data model. The framework supports query of objects and their properties including the edits of the data model in an object oriented fashion. With the new scripting framework, data exposure is intuitive and provides maximum coverage.

Each exposed script object supports the following COM functions:

- **GetName**
 - Return name of the object as text string
 - e.g. name of a design, solve setup, boundary, etc
- **GetChildTypes**
 - An object can have different types of children.
 - Return array of text string. Can be empty if the object's children are NOT categorized into different types.

For example, a design object has 3 children types. The following examples show how the commands run in the **Tools > Open Command Window** for IronPython.

```
>>> design.GetChildTypes()

['Module', 'Editor', 'Design Data']
```

- **GetChildNames**
 - Input: [String – Type]. Default = “Module” and “Editor” for design script object. ‘All’ for other script objects.
 - Return an array of immediate children's names, of a given type if specified

For example, a Mechanical design object has these children.

```
>>> design.GetChildNames()

['Boundaries', 'Excitations', 'Optimetrics', 'Results', '3D Modeler']
```

Four of the children are of “Module” type

```
>>> design.GetChildNames("module")
```

```
['Boundaries', 'Excitations', 'Optimetrics', 'Results']
```

- **GetChildObject**

- Input: String -- Object path. The path may include multiple generations.
- Return the child object if found

For example,

```
>>> d = project.GetChildObject("hfss")
>>> d.GetChildObject("3d modeler").GetChildNames()
['Box1', 'Box1_1', 'Box1_1_1']
>>> project.GetChildObject("hfss/3d modeler").GetChildNames()
['Box1', 'Box1_1', 'Box1_1_1']
```

- **GetPropNames**

- Input: [BOOL - IncludeReadOnly] -- default to true
- Return an array of the object's properties

For example,

```
>>> geom = project.GetChildObject("hfss/3d modeler").GetChildObject("Box1")
>>> geom.GetPropNames()
['Name', 'Material', 'Material/SIValue', 'Material/EvaluatedValue', 'Solve Inside', 'Orientation', 'Orientation/Choices', 'Model', 'Group', 'Display Wireframe', 'Material Appearance', 'Color', 'Color/Red', 'Color/Green', 'Color/Blue', 'Transparent']
```

- **GetPropValue**

- Input: String – Property Path. The path may include multiple generations.
- Return the property value as VARIANT

For example,

```
>>> geom.GetPropValue("material")
' "vacuum" '

>>> geom.GetPropValue("xsize")
' 3mm '

>>> op.GetPropValue("attach to original object")
False
```

- **SetPropValue**

- Input: String – Property Path. The path may include multiple generations.
- Input: String – Data. New value of the property.
- Return -- True if property data is updated successfully. False if failed to assign the new value.

For example,

```
>>> geom.SetPropValue("model", False)

>>> boxcmd.SetPropValue("ysize", "4mm")
```

- **GetPropEvaluatedValue (<PropName>)**

For example,

```
oVar = oDesign.GetChildObject(" Variables/var")
oVar.GetPropEvaluatedValue()
```

- **GetPropSIValue (<PropName>)**

For example,

```
oCreateBox = oDesign.GetChildObject("3D Modeler/Box1/CreateBox:1")
oCreateBox.GetPropValue("xSize")
    return "length / 2"
oCreateBox.GetPropEvaluatedValue("xSize")
    return '0.4mm'
oCreateBox.GetPropSIValue("xSize")
    return 0.0004
```

Additional Details Specific to AEDT Solvers

“3D Modeler” of 3D products and “Machine” of Rmxprt are exposed as “Editor” type children of a design script object.

“Variables” and “Design Settings” are exposed as “Design Data” type children of a design script object.

The following “Module” types are exposed as “Module” type children of a design script object.

HFSS

- Boundaries, Excitations, Circuit Elements, Hybrid Regions, Analysis, Radiation, Field Overlays, Optimetrics, Results

HFSS 3D Layout

- Boundaries, Excitations, Circuit Elements, Analysis, Radiation, Field Overlays, Optimetrics, Results

Maxwell 3D/2D

- Boundaries, Excitations, Analysis, Field Overlays, Optimetrics, Results

Rmxprt

- Analysis, Field Overlays, Optimetrics, Results

Q3D

- Boundaries, Nets, Analysis, Optimetrics,
Q2D
- Boundaries, Conductors, Analysis, Field Overlays, Optimetrics,
Icepak
- Thermal, Monitor, Mesh, Analysis, Field Overlays, Optimetrics, Results
Mechanical
- Boundaries, Excitations, Analysis, Field Overlays, Optimetrics, Results
Circuit
- Optimetrics, Results
Circuit Netlist
- Results
EMIT
- Coupling
Simplorer/Twin Builder
- Analysis, Optimetrics, Results

Additional details on Boundaries/Excitations

Each design type presents its boundaries/excitations data in the project tree as different groups. For example, an HFSS design has Boundaries, Excitations, Circuit Elements and Hybrid Regions while a Icepak design has just a “Thermal” project tree folder.

These module script objects do not have properties

```
>>> project.GetChildObject("icepak/thermal").GetPropNames()  
  
[]
```

GetChildTypes of these module script objects returns the types of its immediate children

```
>>> d = p.GetChildObject("q2d")
>>> d.GetChildObject("conductors").GetChildTypes()
['NonIdealGround', 'SignalLine']
```

GetChildNames of these module object returns its immediate children

```
>>> p.GetChildObject("icepak/thermal").GetChildNames()
['Source1', 'Resistance1', 'ConductingPlate1', 'Source2', 'Resistance2', 'ConductingPlate2',
'Source3', 'Resistance3', 'ConductingPlate3']
```

GetChildNames can be invoked with a “type” and the returns will be filtered by that given type.

```
>>> p.GetChildObject("icepak/thermal").GetChildNames("resistance")
['Resistance1', 'Resistance2', 'Resistance3']
```

Children of a module object are scriptable objects and have properties.

```
>>> port = p.GetChildObject("hfss/excitations/1")
>>> port.GetPropNames(False)
['Name', 'Deembed', 'Deembed Dist', 'Renorm All Terminals']
```

You can query/edit these properties

```
>>> port.GetPropValue("deembed")
False
>>> port.SetPropValue("deembed", True)
True
>>> port.GetPropValue("deembed")
```

True

A boundary/excitation script object can also have children. For example, HFSS terminal is a child of its port. Q3D source/sink can be children of a net.

```
>>> port.GetChildNames()
['Box1_T1']
>>> port.GetChildTypes()
['Terminal']
>>> p.GetChildObject("q3d/nets/s2").GetChildNames()
['Source2', 'Sink2']
>>> p.GetChildObject("q3d/nets/s2").GetChildTypes()
['Sink', 'Source']
```

3D component encapsulation

These script interfaces are compliant with encapsulation. For example,

- Design.GetChildObject("boundaries").GetChildNames() will not return component boundaries
- SetPropValues of component excitations can only be used to edit post processing settings such as 'Deembed', 'Deembed Dist' of a HFSS port.

Additional details on Solve setup

All solve setups are children of the "Analysis" script object. This parent script object is also of the type "Module".

```
>>> d = oDesktop.GetActiveProject().GetChildObject("hfss")
>>> d.GetChildNames()
['Boundaries', 'Excitations', 'Hybrid Regions', 'Circuit Elements', 'Analysis', 'Optimetrics',
'RadField', 'Results', '3D Modeler']
```

```
>>> setups = d.GetChildObject("analysis")
```

This module script object has no property

```
>>> setups.GetPropNames()  
  
[]
```

The children of this module script object in a 3D design is not categorized into different types because the solve setup type is one-to-one to the solution type of a 3D design.

```
>>> setups.GetChildTypes()  
  
[]
```

The children of this module script object in a 3DLayout and Simplorer/TwinBuilder design is categorized into different solve setup types, such as “Transient”, “AC” and “DC” in a Simplorer/TwinBuilder design and “HFSS”, “PlanarEM”, “SIwave” in a 3D layout design.

A solve setup script object can also have children. Children are typically frequency sweeps.

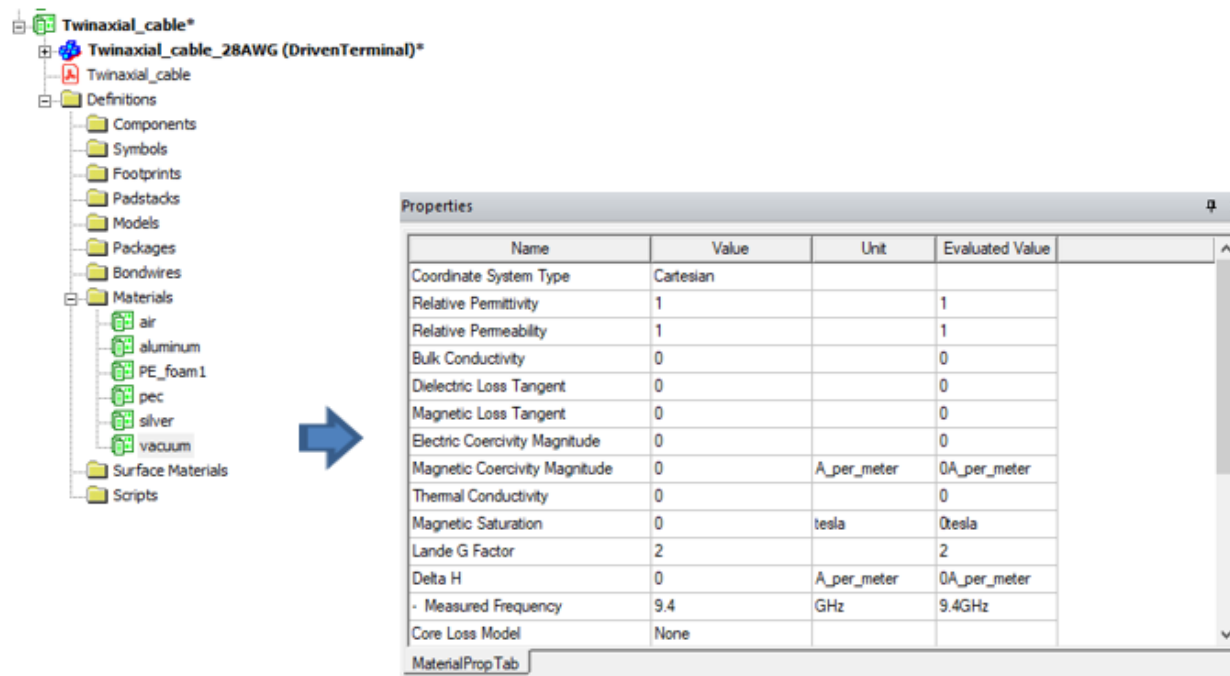
```
>>> setup.GetChildNames()  
  
['Sweep', 'Sweep1', 'Sweep2']  
  
>>> setup.GetChildTypes()  
  
['Discrete', 'Interpolating']  
  
>>> sweep1 = setup.GetChildObject("sweep")
```

Related Topics

[Example: GetPropNames and GetPropValues for Layered Impedance Boundary Script](#)

Materials Scripting Support

Supported material properties are shown in a Property window for the material item.



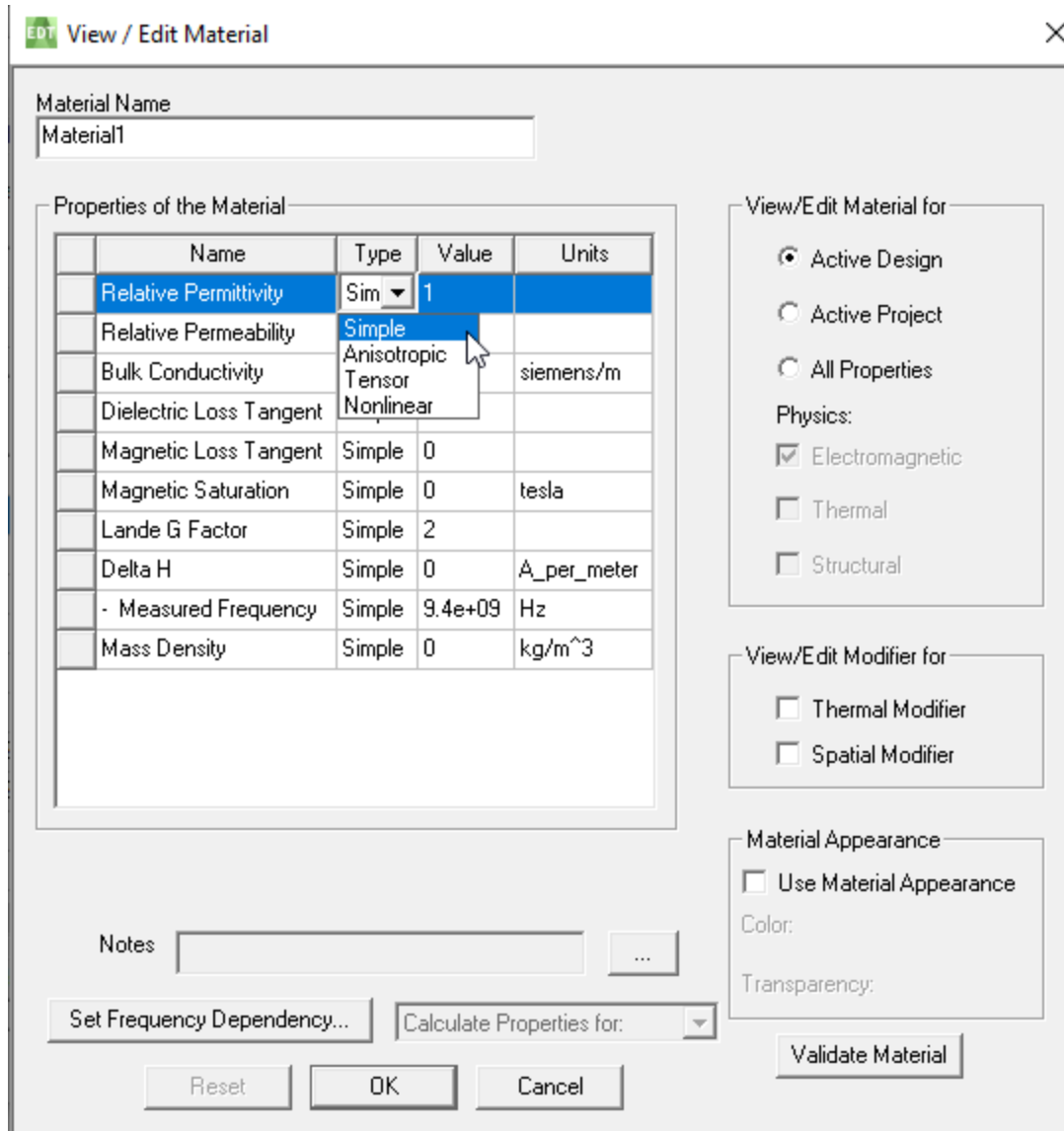
The screenshot displays the Maxwell Scripting Guide interface. On the left, a project tree shows a hierarchy starting with 'Twinaxial_cable*', followed by 'Twinaxial_cable_28AWG (DrivenTerminal)*', and then 'Twinaxial_cable'. Under 'Twinaxial_cable', there is a 'Definitions' folder containing 'Components', 'Symbols', 'Footprints', 'Padstacks', 'Models', 'Packages', 'Bondwires', and 'Materials'. The 'Materials' folder is expanded, showing a list of materials: 'air', 'aluminum', 'PE_foam1', 'pec', 'silver', and 'vacuum'. A blue arrow points from the 'Materials' folder to the 'Properties' window on the right.

The 'Properties' window displays a table of material properties for the selected material. The table has four columns: Name, Value, Unit, and Evaluated Value. The properties listed are:

Name	Value	Unit	Evaluated Value
Coordinate System Type	Cartesian		
Relative Permittivity	1		1
Relative Permeability	1		1
Bulk Conductivity	0		0
Dielectric Loss Tangent	0		0
Magnetic Loss Tangent	0		0
Electric Coercivity Magnitude	0		0
Magnetic Coercivity Magnitude	0	A_per_meter	0A_per_meter
Thermal Conductivity	0		0
Magnetic Saturation	0	tesla	0tesla
Lande G Factor	2		2
Delta H	0	A_per_meter	0A_per_meter
- Measured Frequency	9.4	GHz	9.4GHz
Core Loss Model	None		

The 'MaterialPropTab' tab is selected at the bottom of the Properties window.

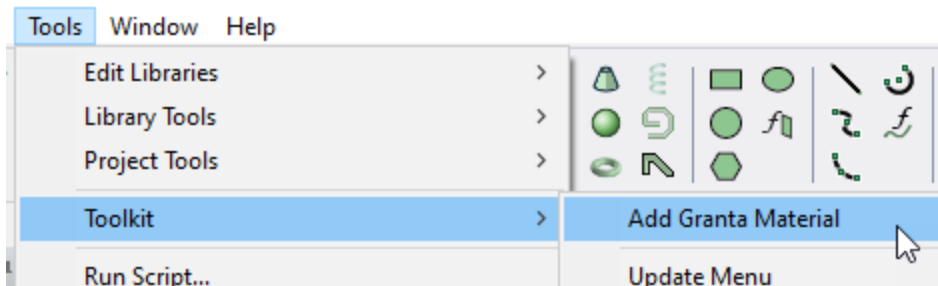
Some Material Properties like Relatively Permittivity may have values assigned as BH Curves or Tensors, as discussed in the Assigning Materials chapter of the online help.

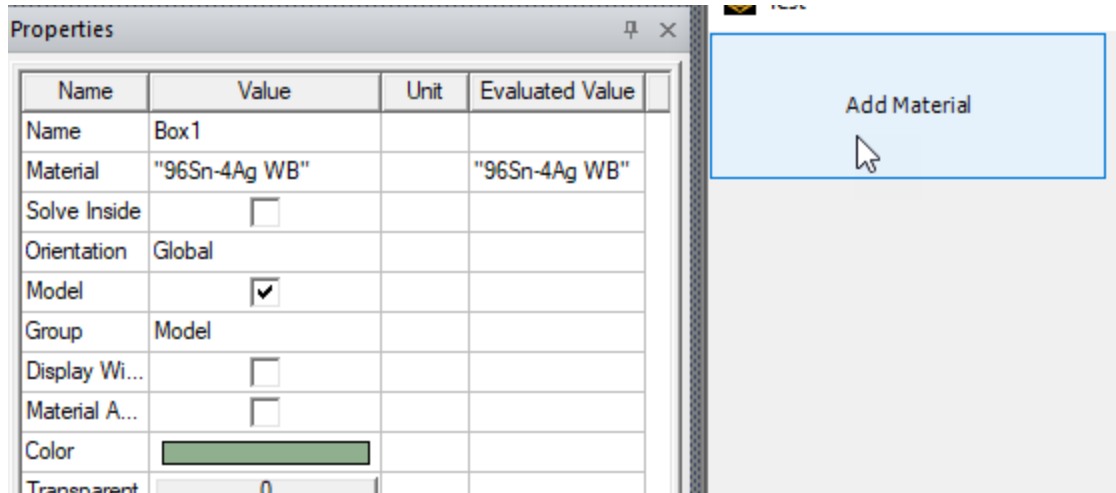


With this feature enabled you can then:

- Get/Set Simple material property
- Get/Set Anisotropic material property
- Get/Set Nonlinear material property
- Get/Set Vector material property
 - Components hide/shown as needed
- Get/Set Tensor material property
- Get/Set Choice material property
- [AddDefinitionFromBlock](#)
- [AddDefiniitonFromLibFile](#)
- [GetExtendedDefinitionObect](#)

A new Toolkit allows you to select materials from the Granta materials gateway, such that project materials will automatically be added when you select a material from the gateway, and that the gateway itself is easily accessed from the materials.





What is not supported:

- Change of property type
- Custom material property, due to its complexity

Object Oriented Scripting for Materials

Materials are Child objects of the Active Project. In the IronPython command window, you can execute `GetPropNames()` for a specified material as follows:

```
>>> omats = oDesktop.GetActiveProject().GetChildObject("Materials")
>>> omat = omats.GetChildObject("vacuum")
>>> omat.GetPropNames()
```

```
['Coordinate System Type', 'Coordinate System Type/Choices', 'Relative Permeability Type',  
'Relative Permeability Type/Choices', 'Relative Permeability', 'Relative Per-  
meability/SIValue', 'Relative Permeability/EvaluatedValue', 'Bulk Conductivity Type', 'Bulk  
Conductivity Type/Choices', 'Bulk Conductivity', 'Bulk Conductivity/SIValue', 'Bulk Con-  
ductivity/EvaluatedValue', 'Magnetic Coercivity Type', 'Magnetic Coercivity Magnitude', 'Mag-  
netic Coercivity Magnitude/SIValue', 'Magnetic Coercivity Magnitude/EvaluatedValue',  
'Composition', 'Composition/Choices', 'Young's Modulus Type', 'Young's Modulus Type/Choices',  
'Young's Modulus', 'Young's Modulus/SIValue', 'Young's Modulus/EvaluatedValue', 'Poisson's  
Ratio Type', 'Poisson's Ratio Type/Choices', 'Poisson's Ratio', 'Poisson's Ratio/SIValue',  
'Poisson's Ratio/EvaluatedValue']
```

Examples showing change to material property type:

```
>>> omat.GetPropValue("Relative Permeability Type/Choices")  
  
['Simple', 'Anisotropic', 'Tensor', 'Nonlinear']  
  
>>> omat.GetPropValue("Relative Permeability Type")  
  
'Nonlinear'  
  
>>> omat.SetPropValue("Relative Permeability Type", "Simple")  
  
True  
  
>>> omat.GetPropValue("Relative Permeability Type")  
  
'Simple'  
  
>>> omat.SetPropValue("Relative Permeability", 10)  
  
True
```

Examples showing change to a vector component value

```
>>> omat.GetPropValue("Magnetic Coercivity Magnitude")  
  
'0A_per_meter'  
  
>>> omat.SetPropValue("Magnetic Coercivity Magnitude", "-1A_per_meter")
```

```
True
```

```
>>> omat.GetPropNames()
```

```
['Coordinate System Type', 'Coordinate System Type/Choices', 'Relative Permeability Type',  
'Relative Permeability Type/Choices', 'Relative Permeability', 'Relative Per-  
meability/SIValue', 'Relative Permeability/EvaluatedValue', 'Bulk Conductivity Type', 'Bulk  
Conductivity Type/Choices', 'Bulk Conductivity', 'Bulk Conductivity/SIValue', 'Bulk Con-  
ductivity/EvaluatedValue', 'Magnetic Coercivity Type', 'Magnetic Coercivity Magnitude', 'Mag-  
netic Coercivity Magnitude/SIValue', 'Magnetic Coercivity Magnitude/EvaluatedValue', 'Magnetic  
Coercivity Components', 'Magnetic Coercivity Components/Component1', 'Magnetic Coercivity Com-  
ponents/Component2', 'Magnetic Coercivity Components/Component3', 'Composition', 'Com-  
position/Choices', '- Stacking Factor Type', '- Stacking Factor', '- Stacking Factor/SIValue',  
'- Stacking Factor/EvaluatedValue', '- Stacking Direction', '- Stacking Direction/Choices',  
"Young's Modulus Type", "Young's Modulus Type/Choices", "Young's Modulus", "Young's Mod-  
ulus/SIValue", "Young's Modulus/EvaluatedValue", "Poisson's Ratio Type", "Poisson's Ratio  
Type/Choices", "Poisson's Ratio", "Poisson's Ratio/SIValue", "Poisson's Ratio/EvaluatedValue"]
```

```
>>> omat.SetPropValue("Magnetic Coercivity Components/Component2", 2)
```

```
True
```

```
>>> omat.GetPropValue("Magnetic Coercivity Components")
```

```
['Component1:=', '2', 'Component2:=', '2', 'Component3:=', '0']
```

Change choice property value

```
>>> omat.GetPropValue("Composition/Choices")
```

```
['Solid', 'Lamination', 'Litz Wire']
```

```
>>> omat.SetPropValue("Composition", "Lamination")
```

```
True
```

```
>>> omat.GetPropNames()
```

```
[ 'Coordinate System Type', 'Coordinate System Type/Choices', 'Relative Permeability Type',
'Relative Permeability Type/Choices', 'Relative Permeability', 'Relative Per-
meability/SIValue', 'Relative Permeability/EvaluatedValue', 'Bulk Conductivity Type', 'Bulk
Conductivity Type/Choices', 'Bulk Conductivity', 'Bulk Conductivity/SIValue', 'Bulk Con-
ductivity/EvaluatedValue', 'Magnetic Coercivity Type', 'Magnetic Coercivity Magnitude', 'Mag-
netic Coercivity Magnitude/SIValue', 'Magnetic Coercivity Magnitude/EvaluatedValue', 'Magnetic
Coercivity Components', 'Magnetic Coercivity Components/Component1', 'Magnetic Coercivity Com-
ponents/Component2', 'Magnetic Coercivity Components/Component3', 'Composition', 'Com-
position/Choices', '- Stacking Factor Type', '- Stacking Factor', '- Stacking Factor/SIValue',
'- Stacking Factor/EvaluatedValue', '- Stacking Direction', '- Stacking Direction/Choices',
"Young's Modulus Type", "Young's Modulus Type/Choices", "Young's Modulus", "Young's Mod-
ulus/SIValue", "Young's Modulus/EvaluatedValue", "Poisson's Ratio Type", "Poisson's Ratio
Type/Choices", "Poisson's Ratio", "Poisson's Ratio/SIValue", "Poisson's Ratio/EvaluatedValue"]
```

```
>>> omat.SetPropValue("Magnetic Coercivity Components/Component2", 2)
```

```
True
```

```
>>> omat.GetPropValue("Magnetic Coercivity Components")
```

```
['Component1:=', '2', 'Component2:=', '2', 'Component3:=', '0']
```

Change choice property value

```
>>> omat.GetPropValue("Composition/Choices")
```

```
['Solid', 'Lamination', 'Litz Wire']
```

```
>>> omat.SetPropValue("Composition", "Lamination")
```

```
True
```

```
>>> omat.GetPropNames()
```

```
[ 'Coordinate System Type', 'Coordinate System Type/Choices', 'Relative Permeability Type',
'Relative Permeability Type/Choices', 'Relative Permeability', 'Relative Per-
meability/SIValue', 'Relative Permeability/EvaluatedValue', 'Bulk Conductivity Type', 'Bulk
Conductivity Type/Choices', 'Bulk Conductivity', 'Bulk Conductivity/SIValue', 'Bulk
```

```
Conductivity/EvaluatedValue', 'Magnetic Coercivity Type', 'Magnetic Coercivity Magnitude',  
'Magnetic Coercivity Magnitude/SIValue', 'Magnetic Coercivity Magnitude/EvaluatedValue', 'Mag-  
netic Coercivity Components', 'Magnetic Coercivity Components/Component1', 'Magnetic Coer-  
civity Components/Component2', 'Magnetic Coercivity Components/Component3', 'Composition',  
'Composition/Choices', '- Stacking Factor Type', '- Stacking Factor', '- Stacking Fact-  
or/SIValue', '- Stacking Factor/EvaluatedValue', '- Stacking Direction', '- Stacking Dir-  
ection/Choices', 'Young's Modulus Type', 'Young's Modulus Type/Choices', 'Young's Modulus',  
'Young's Modulus/SIValue', 'Young's Modulus/EvaluatedValue', 'Poisson's Ratio Type', 'Pois-  
son's Ratio Type/Choices', 'Poisson's Ratio', 'Poisson's Ratio/SIValue', 'Poisson's Ratio/E-  
valuatedValue']
```

```
>>>
```

Project Object Script Commands

Project commands should be executed by the oProject object.

One example of accessing this object is:

```
Set oProject = oDesktop.GetActiveProject()
```

Dataset Script Commands:

[AddDataset](#)

[DeleteDataset](#)

[EditDataset](#)

[ExportDataset](#)

[HasDataset](#)

[ImportDataset](#)

Other Project Object Script Commands:

[AddMaterial \(Maxwell\)](#)

[AnalyzeAll](#)

[ChangeProperty](#)

[ClearMessages](#)

[CloneMaterial](#)

[Close](#)

[CopyDesign](#)

[CutDesign](#)

[DeleteDesign](#)

[DeleteToolObject](#)

[EditMaterial \(Maxwell\)](#)

[ExportMaterial](#)

[GetActiveDesign](#)

[GetArrayVariables](#)

[GetChildNames \[Project\]](#)

[GetChildObject \[Project\]](#)

[GetChildTypes \[Project\]](#)

[GetConfigurableData](#)

[GetDefinitionManager](#)

[GetDependentFiles](#)

[GetDesign](#)

[GetDesign](#)

[GetDesigns](#)

[GetEDBHandle](#)

[GetLegacyName](#)

[GetName \[Project\]](#)

[GetObjPath \[Project\]](#)

[GetPath](#)

[GetPropEvaluatedValue](#)

[GetPropNames \[Project\]](#)

[GetPropSIValue](#)

[GetPropValue \[Project\]](#)

[GetProperties](#)

[GetPropertyValue](#)

[GetTopDesignList](#)

[GetVariableValue](#)

[GetVariables](#)

[InsertDesign](#)

[InsertDesignWithWorkflow](#)

[InsertToolObject](#)

[Paste \[Project Object\]](#)

[Redo \[Project Level\]](#)

[RemoveAllUnusedDefinitions](#)

[RemoveMaterial](#)

[RemoveUnusedDefinitions](#)

[Rename](#)

[RunToolkit](#)

[Save](#)

[SaveAs](#)

[SaveAsStandAloneProject](#)

[SaveProjectArchive](#)

[SetActiveDefinitionEditor](#)

[SetActiveDesign](#)

[SetPropValue \[Project\]](#)

[SetPropertyValue](#)

[SetVariableValue](#)

[SimulateAll](#)

[Undo \[Project\]](#)

[UpdateDefinitions](#)

AnalyzeAll [project]

Runs the project-level script command from the script, which simulates all solution setups and Optimetrics setups for all design instances in the project. The UI waits until simulation is finished before continuing with the script.

UI Access	Project > Analyze All.
Parameters	None.
Return Value	None.

Python Syntax	AnalyzeAll()
Python Example	<code>oProject.AnalyzeAll()</code>

VB Syntax	AnalyzeAll
VB Example	<code>oProject.AnalyzeAll</code>

ClearMessages

Clears information in the **Messages** window.

Note:

Additional options are available when using the Desktop-level [ClearMessages](#) command.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	ClearMessages()
Python Example	<code>oProject.ClearMessages()</code>

VB Syntax	ClearMessages
VB Example	<code>oProject.ClearMessages</code>

Close

Closes the active project.

Warning:

Unsaved changes will be lost.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	Close()
Python Example	<code>oProject.Close()</code>

VB Syntax	Close
VB Example	<code>oProject.Close</code>

CopyDesign

Copies a specified design.

UI Access	Edit > Copy.		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design to copy from.
Return Value	None.		

Python Syntax	<code>CopyDesign (<DesignName>)</code>
Python Example	<code>oProject.CopyDesign ("HFSSDesign1")</code>

VB Syntax	<code>CopyDesign <DesignName></code>
VB Example	<code>oProject.CopyDesign "HFSSDesign1"</code>

CutDesign

Cuts a design from the active project. The design is stored in memory and can be pasted.

Warning:

This is a legacy command that is no longer supported and should not be used as it may have unintended effects on solved designs.

UI Access	Edit > Cut.		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design.
Return Value	None.		

Python Syntax	CutDesign (<DesignName>)		
Python Example	oProject.CutDesign("SimplorerDesign1")		

VB Syntax	CutDesign <DesignName>		
VB Example	oProject.CutDesign "SimplorerDesign1"		

DeleteDesign

Deletes a specified design in the project.

UI Access	Edit > Delete, or Delete in the ribbon.		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design.

Return Value	None.

Python Syntax	DeleteDesign (<DesignName>)
Python Example	<code>oProject.DeleteDesign("MaxwellDesign2")</code>

VB Syntax	DeleteDesign <DesignName>
VB Example	<code>oProject.DeleteDesign "MaxwellDesign2"</code>

DeleteToolObject

Note:

This command is for internal Ansys use only.

UI Access	N/A		
Parameters	Name	Type	Description
	<ObjectName>	String	Name of the tool object.
Return Value	None.		

Python Syntax	DeleteToolObject(<ObjectName>)

Python Example	<code>oProject.DeleteToolObject ("object1")</code>

VB Syntax	<code>DeleteToolObject <ObjectName></code>
VB Example	<code>oProject.DeleteToolObject "object1"</code>

GetActiveDesign

Returns the design in the active project

Note: GetActiveDesign will return normally if there are no active objects.

UI Access	N/A
Parameters	None.
Return Value	Object of the active design.
Python Syntax	<code>GetActiveDesign()</code>
Python Example	<code>oDesign = oProject.GetActiveDesign()</code>

VB Syntax	<code>GetActiveDesign</code>
VB Example	<code>Set oDesign = oProject.GetActiveDesign</code>

GetConfigurableData (Project)

Note:
This command is for internal Ansys use only.

Python Syntax	<code>GetConfigurableData()</code>
Python Example	<code>oProject.GetConfigurableData()</code>

GetDefinitionManager

Gets the `DefinitionManager` object.

UI Access	N/A
Parameters	None.
Return Value	<code>DefinitionManager</code> object.

Python Syntax	<code>GetDefinitionManager()</code>
Python Example	<code>oDefinitionManager = oProject.GetDefinitionManager()</code>

VB Syntax	<code>GetDefinitionManager</code>
VB Example	<code>Set oDefinitionManager = oProject.GetDefinitionManager</code>

Note: For more information on commands for the `DefinitionManager`, see [Definition Manager Script Commands](#).

GetDependentFiles

Provides a list of the external files referenced in the project, including characteristic (for example, MDX) and coupled project files.

UI Access	N/A
Parameters	None.
Return Value	List of referenced files.

Python Syntax	GetDependentFiles()
Python Example	<pre>files = oProject.GetDependentFiles()</pre>

VB Syntax	GetDependentFiles
VB Example	<pre>files = oProject.GetDependentFiles</pre>

GetDesign

Use: Returns the specified design.

Command: None

Syntax: GetDesign <DesignName>

Return Value: The specified design.

Parameters: <DesignName>

Type: <string>

Name of the design to return.

VB Example: `Set oDesign = oProject.GetDesign ("Maxwell13DDesign1")`

GetEDBHandle

Returns the EDB handle for the project.

Important:

This script is for internal Ansys use only.

UI Access	N/A
Parameters	None.
Return Value	String indicating the EDB handle for the project.

Python Syntax	<code>GetEDBHandle()</code>
Python Example	<code>oProject.GetEDBHandle()</code>

VB Syntax	<code>GetEDBHandle</code>
VB Example	<code>oProject.GetEDBHandle</code>

GetLegacyName

Obtains the legacy name of a project.

Note:

This command is for internal Ansys use only.

UI Access	N/A
Parameters	None.
Return Value	String containing the legacy project name.

Python Syntax	GetLegacyName()
Python Example	<code>oProject.GetLegacyName()</code>

VB Syntax	GetLegacyName
VB Example	<code>oProject.GetLegacyName</code>

GetName [Project]

Obtains the project name

UI Access	N/A
Parameters	None.
Return Value	String containing the project name, not including the path or extension.

Python Syntax	GetName()
Python Example	<code>oProject.GetName()</code>

VB Syntax	GetName
VB Example	<code>oProject.GetName</code>

GetPath

Returns the location of the project on disk.

UI Access	N/A
Parameters	None.
Return Value	String containing the path to the project, not including the project name.

Python Syntax	GetPath()
Python Example	<code>oProject.GetPath()</code>

VB Syntax	GetPath

VB Example	<code>oProject.GetPath</code>

GetTopDesignList

Returns a list of top-level design names.

UI Access	N/A
Parameters	None.
Return Value	List of strings containing name of top-level designs.

Python Syntax	<code>GetTopDesignList()</code>
Python Example	<code>oProject.GetTopDesignList()</code>

VB Syntax	<code>GetTopDesignList</code>
VB Example	<code>oProject.GetTopDesignList</code>

InsertDesign

Inserts a new design in the project. For Maxwell scripts, the last argument will always be empty.

UI Access	Project > [Insert Maxwell 3D Design / Insert Maxwell 2D Design / Insert RMxpert Design].		
Parameters	Name	Type	Description
	<code><DesignType></code>	String	Design type. "Maxwell 2D", "Maxwell 3D", or "RMxpert".
	<code><DesignName></code>	String	Design name.
	<code><SolutionType></code>	String	Dependent on design type.

			For Maxwell 3D, can be: "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "DCConduction", "ElectroDCConduction", or "ElectricTransient". For Maxwell 2D, can be: "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "ACConduction", or "DCConduction". For RMXprt, can be: "Three-Phase Induction Motor", "Single-Phase Induction Motor", "Three-Phase Synchronous Machine", "Brushless Permanent-Magnet DC Motor", "Adjust-Speed Synchronous Machine", "Permanent-Magnet DC Motor", "Switched Reluctance Motor", "Line-Start Permanent-Magnet Synchronous Motor", "Universal Motor", "DC Machine", "Claw-Pole Alternator", "Three-Phase Non-Salient Synchronous Machine", or "Generic Rotating Machine".
	""	None	Empty argument.
Return Value	None.		

Python Syntax	<code>InsertDesign (<DesignType>, <DesignName>, <SolutionType>,"")</code>
Python Example	<code>oProject.InsertDesign('Maxwell 3D', 'MyDesign', 'ElectricTransient', '')</code>

VB Syntax	<code>InsertDesign <DesignType>, <DesignName>, <SolutionType>,""</code>
VB Example	<code>oProject.InsertDesign "Maxwell 3D", "MyDesign", "ElectricTransient", ""</code>

InsertDesignWithWorkflow

Inserts a design with a named workflow and returns an IDispatch string.

UI Access	N/A		
Parameters	Name	Type	Description
	<type>	String	Type of design.
	<workflowName>	String	Name of the workflow.
	<specName>	String	Name of the spec.
	<fileName>	String	Name of the file.
	<libLoc>	String	Type of library, such as SysLib.
	<stationaryPath>	String	Path.
Return Value	IDispatch string, such as 'IDispatch(IAltraSimScript)'		

Python Syntax	InsertDesignWithWorkflow(<type>, <workflowName>, <specName>, <fileName>, <libLoc>, <stationaryPath>)
Python Example	<pre>oProject.InsertDesignWithWorkflow ("Circuit Design", "Serial Design", "PCIE3 Stressed", "LongChannel", "SysLib", "C:\\Program Files\\AnsysEM\\v242\\Win64\\syslib\\MS - RT_duroid 6010 (Er=10.2) 0.010 inch, 0.5 oz copper.asty")</pre>

VB Syntax	InsertDesignWithWorkflow <type>, <workflowName>, <specName>, <fileName>, <libLoc>, <stationaryPath>
VB Example	<pre>oProject.InsertDesignWithWorkflow "Circuit Design", "Serial Design", _ "PCIE3 Stressed", "LongChannel", "SysLib", _ "C:\\Program Files\\AnsysEM\\v242\\Win64\\syslib\\MS - RT_duroid 6010" & _ " (Er=10.2) 0.010 inch, 0.5 oz copper.asty"</pre>

InsertToolObject

Note:

This command is for internal Ansys use only.

Python Syntax	InsertToolObject()
Python Example	<code>oProject.InsertToolObject()</code>

Paste (Project Object)

Pastes a design in the active project.

UI Access	Edit > Paste.
Parameters	None.
Return Value	None

Python Syntax	Paste()
Python Example	<code>oProject.Paste()</code>

VB Syntax	Paste
VB Example	<code>oProject.Paste</code>

Redo [Project Level]

Reapplies the last project-level command.

UI Access	Edit > Redo.
Parameters	None.
Return Value	None.

Python Syntax	Redo()
Python Example	<code>oProject.Redo()</code>

VB Syntax	Redo
VB Example	<code>oProject.Redo</code>

Rename

Renames the project and saves it. Similar to [SaveAs\(\)](#).

UI Access	Edit > Rename.		
Parameters	Name	Type	Description
	<NewName>	String	Desired name of the project. The path is optional.
	<OverWriteOk>	Boolean	• True - overwrite the file on disk if it exists. • False - prevent overwrite.
Return Value	None.		

Python Syntax	<code>Rename(<NewName>,<OverWriteOK>)</code>
Python Example	<code>oProject.Rename("c:\projects\MyProject.aedt", True)</code>

VB Syntax	<code>Rename <NewName>,<OverWriteOK></code>
VB Example	<code>oProject.Rename "c:\projects\MyProject.aedt", true</code>

RestoreProjectArchive

Restores a previously archived project to a specified path.

UI Access	File > Restore Archive.		
Parameters	Name	Type	Description
	<code><ArchiveFilePath></code>	String	Path to archived file
	<code><ProjectFilePath></code>	String	Path to restore location
	<code><OverwriteExistingFiles></code>	Boolean	True to overwrite an existing file of the same name; else False.
	<code><OpenProjectAfterRestore></code>	Boolean	True to open the project after it is restored; else False.
Return Value	None.		

Python Syntax	<code>RestoreProjectArchive (<Archivefilepath>, <ProjectFilePath>, <OverwriteExistingFiles>, <OpenProjectAfterRestore>)</code>

Python Example	<code>oDesktop.RestoreProjectArchive("C:\Users\jdoe\Documents\OptimTee.aedt", "C:\Documents\OptimTee.aedt", False, True)</code>

VB Syntax	<code>RestoreProjectArchive <Archivefilepath>, <ProjectFilePath>, <OverwriteExistingFiles>, <OpenProjectAfterRestore></code>
VB Example	<code>oDesktop.RestoreProjectArchive "C:\Users\jdoe\Documents\OptimTee.aedt", _ "C:\Documents\OptimTee.aedt", false, true</code>

Save

Saves the active project.

UI Access	File > Save.
Parameters	None.
Return Value	None.

Python Syntax	<code>Save()</code>
Python Example	<code>oProject.Save()</code>

VB Syntax	<code>Save</code>
VB Example	<code>oProject.Save</code>

SaveAs

Saves the project under a new name. Requires a full path.

Note:

This script takes two parameters for non-schematic/layout designs and four parameters for schematic/layout designs.

UI Access	File > Save As.		
Parameters	Name	Type	Description
	<NewName>	String	The desired name of the project, with directory and extension.
	<OverWriteOK>	Boolean	True to overwrite the file of the same name, if it exists. False to prevent over-write.
	<DefaultAction>	String	For Schematic/Layout projects only. Otherwise omit. See note below. Valid actions: ef_overwrite , ef_copy_no_overwrite, ef_make_path_absolute, or empty string.
	<OverwriteActions>	Array	For Schematic/Layout projects only. Otherwise omit. See note below. Structured array: Array("Name: <Action>", <FileName>, <FileName>, ...) Valid actions: ef_overwrite , ef_copy_no_overwrite, ef_make_path_absolute, or empty string.
Return Value	None.		

Python Syntax	For non-Schematic/Layout project: SaveAs (<NewName> <OverWriteOK>) For Schematic/Layout project: SaveAs (<NewName> <OverWriteOK> <DefaultAction> <OverrideActions>)

Python Example	<pre>oProject.SaveAs('D:/projects/project1.aedt', True) ---- oProject.SaveAs('D:/Projects/Project1.aedt', True, 'ef_overwrite', ['NAME:OverrideActions', ['NAME:ef_copy_no_overwrite', ['NAME:Files', '\$PROJECTDIR/circuit_models.inc']], ['NAME:ef_make_path_absolute', ['NAME:Files', '\$PROJECTDIR/SL_6s.sp']]])</pre>

VB Syntax	For non-Schematic/Layout project: SaveAs <NewName> <OverWriteOK> For Schematic/Layout project: SaveAs <NewName> <OverWriteOK> <DefaultAction> <OverrideActions>
VB Examples	<pre>oProject.SaveAs "D:/projects/project1.aedt", true ---- oProject.SaveAs "F:\Designer Projects\TA33097\HighSpeedChannel.aedt", true, "ef_overwrite", Array ("NAME:OverrideActions", Array("NAME:ef_copy_no_overwrite", Array("NAME:Files", "\$PROJECTDIR/circuit_models.inc")), Array("NAME:ef_make_path_absolute", Array("NAME:Files", "\$PROJECTDIR\SL_6s.sp")))</pre>

Important:

The DefaultAction and OverrideActions strings correspond to the following actions:

- **ef_overwrite** – Copy file to new project directory and overwrite.
- **ef_copy_no_overwrite** – Copy file to new project directory and don't overwrite.
- **ef_make_path_absolute** – Change reference to point to file in old project directory.
- **Empty String** – Do nothing.

The DefaultAction is applied to all files that are NOT explicitly listed in the OverrideActions array. Those in the OverrideActions array are separate arrays for actions that are different from the default action; those actions are applied to the files listed in the same array:

- If OverrideActions are not specified, DefaultAction is applied to ALL files in project directory.

SaveAsStandAloneProject

Saves the project as a standalone copy.

Note:

This script is not supported when the application is being controlled by Ansys Workbench.

UI Access	N/A		
Parameters	Name	Type	Description
	<projectName>	String	The desired name of the project, with directory and extension.
Return Value	None.		

Python Syntax	SaveAsStandAloneProject(<projectName>)
Python Example	<code>oProject.SaveAsStandAloneProject('D:/projects/project1.aedt')</code>

VB Syntax	SaveAsStandAloneProject <projectName>
VB Examples	<code>oProject.SaveAsStandAloneProject "D:/projects/project1.aedt"</code>

SaveProjectArchive

Saves the active project as an archive to the specified file path.

UI Access	File > Archive.		
Parameters	Name	Type	Description
	<archiveFilePath>	String	Path to archived file.
	<IncludeExternalFiles>	Boolean	True to include external files; False to exclude.
	<IncludeResultsFiles>	Boolean	True to include simulation files associated with the project; False to exclude.
	<AdditionalFiles>	Array	Additional specified files to include.
	<ArchiveNotes>	String	String describe the archive.
Return Value	None.		

Python Syntax	SaveProjectArchive(<archivefilepath>, <IncludeExternalFiles>, <IncludeResultsFiles>, <AdditionalFiles>, <ArchiveNotes>)
Python	<code>oProject.SaveProjectArchive("C:\\Users\\Documents\\Ansoft\\Project27.aedt", True,</code>

Example	<code>False, [], "")</code>

VB Syntax	<code>SaveProjectArchive <archivefilepath>, <IncludeExternalFiles>, <IncludeResultsFiles>, <AdditionalFiles>, <ArchiveNotes></code>
VB Example	<code>oProject.SaveProjectArchive "C:\Documents\OptimTee.aedt", true, false, Array(), "My notes"</code>

SetActiveDefinitionEditor

Obtains a specified definition editor.

UI Access	N/A		
Parameters	Name	Type	Description
	<code><EditorName></code>	String	Name of the definition editor to set active, one of "SymbolEditor", "FootprintEditor".
	<code><DefinitionName></code>	String	The combination name for the symbol or footprint, <code><libname>:<def-name></code>
Return Value	Object for the definition to be edited.		

Python Syntax	<code>SetActiveDefinitionEditor(<EditorName>, <DefinitionName>)</code>
Python Example	<code>oProject.SetActiveDefinitionEditor("SymbolEditor", "Simplorer Elements\Basic Elements\Circuit\Passive Elements:R")</code>

VB Syntax	SetActiveDefinitionEditor <EditorName>, <DefinitionName>
VB Example	oProject.SetActiveDefinitionEditor "SymbolEditor",_ "Simplorer Elements\Basic Elements\Circuit\Passive Elements:R"

SetActiveDesign

Sets a design to be the active design.

UI Access	N/A		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design to set as the active design.
Return Value	None.		

Python Syntax	SetActiveDesign (<DesignName>)
Python Example	oDesign = oProject.SetActiveDesign("SimplorerDesign2")

VB Syntax	SetActiveDesign <DesignName>
VB Example	Set oDesign = oProject.SetActiveDesign "SimplorerDesign2"

SimulateAll

Simulates all solution setups and Optimetrics setups for all design instances in the project. Script processing only continues when all analyses are finished.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	SimulateAll()
Python Example	<code>oProject.SimulateAll()</code>

VB Syntax	SimulateAll
VB Example	<code>oProject.SimulateAll</code>

Undo [Project]

Cancels the last project-level command.

UI Access	Edit > Undo.
Parameters	None.
Return Value	None.

Python Syntax	Undo()
Python Example	<code>oProject.Undo()</code>

VB Syntax	Undo
VB Example	<code>oProject.Undo</code>

UpdateDefinitions

Updates all definitions. The **Messages** window reports when definitions are updated, or warns when definitions cannot be found.

UI Access	Tools > Project Tools > Update Definitions. Click Select All , then Update .
Parameters	None.
Return Value	None.

Python Syntax	UpdateDefinitions()
Python Example	<code>oProject.UpdateDefinitions()</code>

VB Syntax	UpdateDefinitions
VB Example	<code>oProject.UpdateDefinitions</code>

This page intentionally
left blank.

3 - Application Object Script Commands

The Application object commands permit you to get the AppDesktop. Application object commands should be executed by the oAnsoftApp object.

`oAnsoftApp.<CommandName> <args>`

General Application Script Commands

The following are general script commands recognized by the **oAnsoftApp** object:

- [GetAppDesktop](#)

The following deprecated commands are no longer supported and produce an error if used.

- GetDesiredRamMBLimit (deprecated)
- GetHPCLicenseType (deprecated)
- GetMaximumRamMBLimit (deprecated)
- GetMPISpawnCmd(deprecated)
- GetMPIVendor (deprecated)
- GetNumberOfProcessors (deprecated)
- GetUseHPCForMP (deprecated)
- SetDesiredRamMBLimit (deprecated)
- SetHPCLicenseType (deprecated)
- SetMaximumRamMBLimit (deprecated)
- SetMPISpawnCmd (deprecated)
- SetMPIVendor (deprecated)
- SetNumberOfProcessors (deprecated)
- SetUseHPCForMP (deprecated)

GetAppDesktop

GetAppDesktop is a function of oAnsoftApp. This function does not take an input and it returns an object. The object is assigned to the variable oDesktop.

UI Access	NA		
Parameters	Name	Type	Description
	None		
Return Value	Object		

Python Syntax	GetAppDesktop()		
Python Example	<code>oDesktop = oAnsoftApp.GetAppDesktop()</code>		

VB Syntax	GetAppDesktop()		
VB Example	<code>Set oDesktop = oAnsoftApp.GetAppDesktop()</code>		

GetApp

Obtains the product application object.

UI Access	NA		
Parameters	Name	Type	Description
	<productname>	String	One of "Twin Builder", "HFSS", "Maxwell", "Q3D Extractor". "Maxwell Circuit

	<table><tr><td></td><td></td><td>Editor", "Designer"</td></tr></table>			Editor", "Designer"
		Editor", "Designer"		
Return Value	Object			

Python Syntax	<code>GetApp(<productname>)</code>
Python Example	<code>oHFSSApp = oAnsoftApp.GetApp("HFSS")</code>

VB Syntax	<code>GetApp(<productname>)</code>
VB Example	<pre>Dim oHFSSApp oHFSSApp = oAnsoftApp.GetApp("HFSS")</pre>

This page intentionally
left blank.

4 - Desktop Object Script Commands

Desktop commands should be executed by the oDesktop object. Some new commands permit you to query objects when you do not know the names. See: [Object Oriented Property Scripting](#).

```
Set oDesktop =  
    CreateObject("Ansoft.ElectronicsDesktop")  
    oDesktop.CommandName <args>
```

[AddMessage](#)

[AreThereSimulationsRunning](#)

[ClearMessages](#)

[CloseAllWindows](#)

[CloseProject](#)

[CloseProjectNoForce](#)

[Count](#)

[DeleteProject](#)

[DeleteRegistryEntry](#)

[DoesRegistryValueExist](#)

[DownloadJobResults](#)

[EnableAutoSave](#)

[ExportOptionsFiles](#)

[GetActiveProject](#)

[GetActiveScheduler](#)

[GetActiveSchedulerInfo](#)

[GetAutoSaveEnabled](#)

[GetBuildDateTimeString](#)

[GetCustomMenuSet](#)

[GetDefaultUnit](#)

[GetDesktopConfiguration](#)

[GetDistributedAnalysisMachines](#)

[GetDistributedAnalysisMachinesForDesignType](#)

[GetExeDir](#)

[GetGDIObjectCount](#)

[GetLibraryDirectory](#)

[GetLocalizationHelper](#)

[GetMessages](#)

[GetMonitorData](#)

[GetPersonalLibDirectory](#)

[GetProcessID](#)

[GetProjectDirectory](#)

[GetProjectList](#)

[GetProjects](#)

[GetRegistryInt](#)

[GetRegistryString](#)

[GetRunningInstancesMgr](#)

[GetSchematicEnvironment](#)

[GetScriptingToolsHelper](#)

[GetSysLibDirectory](#)

[GetTempDirectory](#)

[GetUserLibDirectory](#)

[GetVersion](#)

[IsFeatureEnabled](#)

[KeepDesktopResponsive](#)

[LaunchJobMonitor](#)

[NewProject](#)

[OpenAndConvertProject](#)

[OpenMultipleProjects](#)

[OpenProject](#)

[OpenProjectWithConversion](#)

[PageSetup](#)

[PauseRecording](#)

[PauseScript](#)

[Print](#)

[QuitApplication](#)

[RefreshJobMonitor](#)

[ResetLogging](#)

[RestoreProjectArchive](#)

[RestoreWindow](#)

[ResumeRecording](#)

[RunACTWizardScript](#)

[RunProgram](#)

[RunScript](#)

[RunScriptWithArguments](#)

[SelectScheduler](#)

[SetActiveProject](#)

[SetActiveProjectByPath](#)

[SetCustomMenuSet](#)

[SetDesktopConfiguration](#)

[SetLibraryDirectory](#)

[SetProjectDirectory](#)

[SetRegistryFromFile](#)

[SetRegistryInt](#)

[SetRegistryString](#)

[SetSchematicEnvironment](#)

[SetTempDirectory](#)[ShowDockingWindow](#)[Sleep](#)[StopSimulations](#)[SubmitJob](#)[TileWindows](#)**Related Topics:**[Desktop Commands For Registry Values](#)[ImportExport Tool Commands](#)

AddMessage

Add a message with severity and context to message window.

UI Access	N/A		
Parameters	Name	Type	Description
	<projectName>	String	Project name. Passing an empty string adds the message as the desktop global message.
	<designName>	String	Design name. Ignored if project name is empty. Passing an empty string adds the message to project node in the message tree.
	<severity>	Integer	One of "Error", "Warning" or "Info". Anything other than the first two is treated as "Info" 0 = Informational, 1 = Warning, 2 = Error, 3 = Fatal
	<msg>	String	The message for the message window.
	<category>	String	Optional. The category is created with the message under the design tree node if the category does not exist. If the category already exists, the new mes-

			sage is added to the end of the existing category. It is ignored if the project or design is empty. If missing or empty, the message is added to the Design node in the message tree.
Return Value	None.		

Python Syntax	AddMessage(<projectName>, <designName>, <severity>, <msg>, <category>)
Python Example	oDesktop.AddMessage("Project1", "Maxwell", 0, "This is a test message", "")

VB Syntax	AddMessage <projectName>, <designName>, <severity>, <msg>, <category>
VB Example	oDesktop.AddMessage "Project1", "Maxwell1", 0, "This is a test message", ""

ClearMessages

For a specified project and design, this command clears all messages in a specified severity range. The user-specified integral severity level is interpreted as:

0 => info, 1 => warning, 2 => error, and 3 => fatal error.

The ClearMessages function accepts four input arguments. The first two specifies project and design, respectively. This function clears messages in the range specified by the third (interpreted as start severity) and fourth (interpreted as stop severity) arguments. The fourth argument has a default value of 0. The last two arguments need not be specified in any given order, i.e., they can indicate either ascending or descending trend. The function clears all messages having severity level within this specified range. The start and stop severity need not be specified in any given order (i.e., they can indicate either ascending or descending severity).

UI Access	In Message Manager , right-click and select Clear messages for [ProjectName]...		
Parameters	Name	Type	Description
	<projectName>	String	Name of the project from which to clear messages.
	<designName>	String	Name of the design under the <projectName> from which to clear messages.
	<startSeverity>	Integer	User-specified severity level at which messages start getting cleared: • 0 = informational • 1 = warning • 2 = error • 3 = fatal error
	<stopSeverity>	Integer	<i>Optional.</i> User-specified stop severity level until which messages will be cleared. • 0 = informational • 1 = warning • 2 = error • 3 = fatal error If not specified, <stopSeverity> has a default value of 0.
Return Value	None.		

Python Syntax	ClearMessages(<projectName>, <designName>, <startSeverity>, <stopSeverity>)
Python Example	Preserve Current Functionality; stopSeverity = 0 (DEFAULT VALUE) <pre># startSeverity = 0; clears just the Info messages oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell13DDesign1", 0) # startSeverity = 1; clears all the Info and Warning messages</pre>

```
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 1)
# startSeverity = 2; clears all the Info, Warning, Error
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 2)
# startSeverity = 3; clears all messages, i.e., Info, Warning, Error, Fatal
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 3)
Extend the current functionality by enabling range-based deletion
# startSeverity = 0; clears the Info messages;
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 0)
# startSeverity = 0 and stopSeverity = 0; clears all the Info messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 0,
0)
# startSeverity = 0 and stopSeverity = 1; clears all the Info and Warning mes-
sages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 0,
1)
# startSeverity = 0 and stopSeverity = 2; clears all the Info, Warning, and Error
messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 0,
2)
# startSeverity = 0 and stopSeverity = 3; clears all the Info, Warning, Error,
and Fatal messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor", "Maxwell3DDesign1", 0,
3)
```



```
# startSeverity = 1; clears all the Info and Warning messages;
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",1)
# startSeverity = 1 and stopSeverity = 1; clears all the Warning messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",1, 1)
# startSeverity = 1 and stopSeverity = 2; clears all the Warning and Error messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",1, 2)
# startSeverity = 1 and stopSeverity = 3; clears all the Warning, Error, and Fatal messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",1, 3)

# startSeverity = 2; clears all the Info, Warning, and Error messages;
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",2)
# startSeverity = 2 and stopSeverity = 2; clears all the Error messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",2, 2)
# startSeverity = 2 and stopSeverity = 3; clears all the Error and Fatal messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",2, 3)
# startSeverity = 2 and stopSeverity = 0; clears all the Info, Warning, and Error messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",2, 0)
```

```
# startSeverity = 3; clears all the Info, Warning, Error, and Fatal messages;
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",3)
# startSeverity = 3 and stopSeverity = 3; clears all the Fatal error messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",3, 3)
# startSeverity = 3 and stopSeverity = 0; clears all the Info, Warning, Error,
and Fatal messages
oDesktop.ClearMessages("Ex_EC_1_Asymmetrical_Conductor","Maxwell3DDesign1",3, 0)
```

VB Syntax	ClearMessages <projectName>, <designName>, <startSeverity>, <stopSeverity>
VB Example	<pre>Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop") Set oDesktop = oAnsoftApp.GetAppDesktop() oDesktop.ClearMessages "MyProject", "MyDesign", 0, 3</pre>

CloseAllWindows

Closes all MDI child windows on the desktop.

UI Access	From main menu, Window > CloseAll .
Parameters	None.
Return Value	None.

Python Syntax	<code>CloseAllWindows()</code>
Python Example	<code>oDesktop.CloseAllWindows()</code>

VB Syntax	<code>CloseAllWindows</code>
VB Example	<code>oDesktop.CloseAllWindows</code>

CloseProject

Closes a specified project. Changes to the project are not saved. Save the project using the Project command **Save** or **Save As** before closing to save changes.

UI Access	File > Close		
Parameters	Name	Type	Description
	<ProjectName>	String	The name of the project already in the Desktop that is to be closed, without path or extension
Return Value	None		

Python Syntax	<code>CloseProject (<ProjectName>)</code>
Python Example	<code>oDesktop.CloseProject ("MyProject")</code>

VB Syntax	<code>CloseProject <ProjectName></code>
VB Example	<code>oDesktop.CloseProject "MyProject"</code>

CloseProjectNoForce

Use: Close a named project currently open in the Desktop, unless a simulation is running. Changes to the project will not be saved. Save the project using the Project command **Save** or **Save As** before closing to save changes. To determine if the project has been closed, use `GetProjectList` and see if the named project is present.

UI Access	File > Close		
Parameters	Name	Type	Description
	<ProjectName>	String	The name of the project already on the Desktop that is to be closed, without path or extension
Return Value	None		

Python Syntax	CloseProjectNoForce (<ProjectName>)		
Python Example	<code>oDesktop.CloseProjectNoForce ("MyProject")</code>		

VB Syntax	CloseProjectNoForce <ProjectName>		
VB Example	<code>oDesktop.CloseProjectNoForce "MyProject"</code>		

Count

Gets the total number of queried projects or designs obtained by `GetProjects()` and `GetDesigns()` commands.

UI Access	N/A
Parameters	None.
Return Value	Integer
Python Syntax	Count
Python Example	<pre>projects = oDesktop.GetProjects() numprojects = projects.Count</pre>

VB Syntax	Count
VB Example	<pre>Dim oAnsoftApp Dim oDesktop Dim oProject Dim oDesign Dim oEditor Dim oModule Dim oProjects Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop") Set oDesktop = oAnsoftApp.GetAppDesktop() oDesktop.RestoreWindow Dim projects set projects = oDesktop.GetProjects() for i = 0 to projects.Count - 1</pre>

	<pre>msgbox projects(i).GetName() dim designs set designs = projects(i).GetDesigns() for j = 0 to designs.Count - 1 msgbox designs(j).GetName() next next</pre>

DeleteProject

Deletes a project from disk.

UI Access	Edit > Delete		
Parameters	Name	Type	Description
	<ProjectName>	String	Name of the project
Return Value	None		

Python Syntax	DeleteProject (<ProjectName>)
Python Example	oDesktop.DeleteProject ("MyProject")

VB Syntax	DeleteProject <ProjectName>
VB Example	oDesktop.DeleteProject "MyProject"

EnableAutoSave

Enable or disable autosave feature.

UI Access	N/A		
Parameters	Name	Type	Description
	<enable>	Boolean	True to enable autosave; False disables it.
Return Value	None.		

Python Syntax	EnableAutoSave(<enable>)
Python Example	oDesktop.EnableAutoSave (True)

VB Syntax	EnableAutoSave <enable>
VB Example	oDesktop.EnableAutoSave True

ExportOptionsFiles

Copies the options config files to the *DestinationDirectory*.

UI Access	Tools > Options > Export Options Files ...

Parameters	Name	Type	Description
	<DestinationDirectory>	String	The path to the destination directory.
Return Value	None		

Python Syntax	ExportOptionsFiles(<DestinationDirectory>)
Python Example	<code>oDesktop.ExportOptionsFiles("D:/test/export/")</code>

VB Syntax	ExportOptionsFiles <DestinationDirectory>
VB Example	<code>oDesktop.ExportOptionsFiles"D:/test/export/"</code>

GetActiveProject

Obtains the project currently active in the Desktop, as an object.

Note:

GetActiveProject returns normally if there are no active objects.

UI Access	N/A
Parameters	None.
Return Value	Object: the project that is currently active in the desktop.

Python Syntax	GetActiveProject()
Python Example	<code>oProject = oDesktop.GetActiveProject()</code>

VB Syntax	GetActiveProject
VB Example	<code>Set oProject = oDesktop.GetActiveProject</code>

GetAutoSaveEnabled

Checks whether the autosave feature is enabled.

UI Access	N/A
Parameters	None.
Return Value	Integer: • 1 – Autosave is enabled. • 0 – Autosave is not enabled.

Python Syntax	GetAutoSaveEnabled()
Python Example	<code>Enabled = oDesktop.GetAutoSaveEnabled()</code>

VB Syntax	GetAutoSaveEnabled

VB Example	<code>Enabled = oDesktop.GetAutoSaveEnabled()</code>

GetBuildDateTimeString

Returns a string representing the build date and time of the product;

UI Access	N/A
Parameters	None.
Return Value	String build date and time, in the format: year-month-day hour:minute:second. Example: 2019-01-18 21:59:33

Python Syntax	<code>GetBuildDateTimeString()</code>
Python Example	<code>oDesktop.GetBuildDateTimeString()</code>

VB Syntax	<code>GetBuildDateTimeString</code>
VB Example	<code>dnt = oDesktop.GetBuildDateTimeString</code>

GetCustomMenuSet

Returns the name of the current selected menu set in **Tools > Options > General Options > General > Desktop Configuration**.

UI Access	N/A

Parameters	None.
Return Value	String containing current menu set. For example, 'Default', 'EM', 'Twin Builder'.

Python Syntax	GetCustomMenuSet ()
Python Example	<code>oDesktop.GetCustomMenuSet ()</code>

VB Syntax	GetCustomMenuSet
VB Example	<code>Set oProject = oDesktop.GetCustomMenuSet</code>

GetDefaultUnit

Returns the default unit for a physical quantity.

UI Access	Tools > Options > General Options > Default Units. Note that this menu only displays units that can be changed, while the script can be used to view additional default units.		
Parameters	Name	Type	Description
	<type>	String	String containing a type of measurement. Valid strings are (case insensitive): "Acceleration" "Angle" "AngularAcceleration" "AngularDamping" "AngularSpeed"

			• "Capacitance" • "Conductance" • "Current" • "CurrentChangeRate" • "DataRate" • "DeltaH" (Magnetic Field Strength) • "Density" • "Flux" • "Force" • "Frequency" • "Inductance" • "Length" • "MagneticReluctance" • "Mass" • "MassFlowRate" • "MomentInertia" • "Power" • "Pressure" • "PressureCoefficient" • "Resistance" • "SaturateMagnetization" (Magnetic Inductance) • "Speed" • "Temperature" • "Time"

	<table><tr><td></td><td></td><td> • "Torque" • "Voltage" • "VoltageChangeRate" • "Volume" • "VolumeFlowPerPressureRoot" • "VolumeFlowRate" </td></tr></table>			• "Torque" • "Voltage" • "VoltageChangeRate" • "Volume" • "VolumeFlowPerPressureRoot" • "VolumeFlowRate"
		• "Torque" • "Voltage" • "VoltageChangeRate" • "Volume" • "VolumeFlowPerPressureRoot" • "VolumeFlowRate"		
Return Value	String containing the default unit (for example, "mm").			

Python Syntax	GetDefaultUnit(<type>)
Python Example	<code>oDesktop.GetDefaultUnit("Length")</code>

VB Syntax	GetDefaultUnit <type>
VB Example	<code>oDesktop.GetDefaultUnit("Length")</code>

GetDesigns

Obtains all designs in the current project.

UI Access	N/A
Parameters	None.
Return Value	List of objects for all designs in the project.

Python Syntax	GetDesigns()

Python Example	<code>oProject.GetDesigns()</code>

VB Syntax	<code>GetDesigns</code>
VB Example	<code>oProject.GetDesigns</code>

GetDesktopConfiguration

Returns the name of the current selected configuration in **Tools > Options > General Options > General > Desktop Configuration**.

UI Access	N/A
Parameters	None.
Return Value	String containing current Desktop configuration. For example, 'All', 'Twin Builder'.

Python Syntax	<code>GetDesktopConfiguration()</code>
Python Example	<code>oDesktop.GetDesktopConfiguration()</code>

VB Syntax	<code>GetDesktopConfiguration</code>
VB Example	<code>Set oProject = oDesktop.GetDesktopConfiguration</code>

GetDistributedAnalysisMachines

Gets a list of machines used for distributed analysis. You can iterate through the list using standard VBScript methods.

UI Access	N/A
Parameters	None.
Return Value	Returns a collection of names of machines used for distributed analysis.

Python Syntax	GetDistributedAnalysisMachines()
Python Example	<code>oDesktop.GetDistributedAnalysisMachines()</code>

VB Syntax	GetDistributedAnalysisMachines
VB Example	<code>oDesktop.GetDistributedAnalysisMachines()</code>

GetDistributedAnalysisMachinesForDesignType

To obtain a list of the machines set up for analysis of the specified design type.

UI Access	NA		
Parameters	Name	Type	Description
	<designTypeName>	String	The name of the type of design, such as "Twin Builder", "HFSS", "HFSS-IE", "Maxwell 3D", "Maxwell 2D", "RMxpert", "EM Design", "Circuit", "System", "Q3D Extractor", "2D Extractor"

Return Value	Object; returns a collection of machine names.

Python Syntax	GetDistributedAnalysisMachinesForDesignType (<designTypeName>)
Python Example	<pre>machineNames =oDesktop. GetDistributedAnalysisMachinesForDesignType ("Maxwell")</pre>

VB Syntax	GetDistributedAnalysisMachinesForDesignType <designTypeName>
VB Example	<pre>Set machineNames =oDesktop. GetDistributedAnalysisMachinesForDesignType ("Maxwell")</pre>

GetExeDir

Returns the path where the executable is located.

UI Access	N/A
Parameters	None.
Return Value	String path where executable is located. Example: 'C:/Program Files/AnsysEM/v242/Win64/'

Python Syntax	GetExeDir()
Python Example	<code>oDesktop.GetExeDir()</code>

VB Syntax	GetExeDir
VB Example	<code>oDesktop.GetExeDir</code>

GetGDIObjectCount

Note:

This command is for internal Ansys use only.

Python Syntax	GetGDIObjectCount()
Python Example	<code>oDesktop.GetGDIObjectCount()</code>

VB Syntax	GetGDIObjectCount()
VB Example	<code>oDesktop.GetGDIObjectCount()</code>

GetLibraryDirectory

Get the path to the `SysLib` directory.

UI Access	NA

Parameters	Name	Type	Description
	None		
Return Value	String The path to the SysLib directory.		

Python Syntax	GetLibraryDirectory()
Python Example	AddInfoMessage (str (oDesktop.GetLibraryDirectory()))

VB Syntax	GetLibraryDirectory
VB Example	MsgBox oDesktop.GetLibraryDirectory

VB Example:

message box returns the path in this example

```
Dim oAnsoftApp
```

```
Dim oDesktop
```

```
Dim oProject
```

```
Dim oDesign
```

```
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()

oDesktop.RestoreWindow

libdir = oDesktop.GetLibraryDirectory
msgbox(oDesktop.GetLibraryDirectory())
```

GetLocalizationHelper

Note:
This command is for internal Ansys use only.

Returns the object for the localization helper.

UI Access	NA		
Parameters	Name	Type	Description
	None		
Return Value	Object Localization helper object, such as "IDispatch(ILocalizationHelper)"		

Python Syntax	GetLocalizationHelper()
Python Example	oDesktop.GetLocalizationHelper()

VB Syntax	GetLocalizationHelper
VB Example	<code>oDesktop.GetLocalizationHelper()</code>

GetMessages

Get the messages from a specified project and design.

UI Access	NA		
Parameters	Name	Type	Description
	<ProjectName>	String	Name of the project for which to collect messages. An incorrect project name results in no messages (design is ignored). An empty project name results in all messages (design is ignored)
	<DesignName>	String	Name of the design in the named project for which to collect messages. An incorrect design name results in no messages for the named project. An empty design name results in all messages for the named project
	<Severity>	Integer	Severity is 0-3, and is tied in to info/warning/error/fatal types as follows: • 0 – info and above • 1 – warning and above • 2 – error and fatal • 3 – fatal only (rarely used)
Return Value	Array of string messages.		

Python Syntax	<code>GetMessages (<ProjectName>, <DesignName>, <Severity>)</code>
Python Example	<code>Messages = oDesktop.GetMessages("MyProject", "Maxwell11", 1)</code>

VB Syntax	<code>GetMessages <ProjectName>, <DesignName>, <Severity></code>
VB Examples	<code>Messages = oDesktop.GetMessages "MyProject", "Maxwell11", 1</code>

GetName [Desktop]

Gets names of queried projects or designs obtained by GetProjects() and GetDesigns() commands. See the example query.

UI Access	N/A
Parameters	N-
Return Value	String containing name.

Python Syntax	<code>GetName()</code>
Python Example	<pre>set projects = oDesktop.GetProjects() project_name = projects(0).GetName()</pre>

VB Syntax	<code>GetName()</code>
VB Example	-----

	In this example, message box returns project name. projects(0) is the first of the several projects. Similarly projects(1) displays name of second project ----- <pre>Dim oAnsoftApp Dim oDesktop Dim oProject Dim oDesign Dim oEditor Dim oModule Dim oProjects Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop") Set oDesktop = oAnsoftApp.GetAppDesktop() oDesktop.RestoreWindow set projects = oDesktop.GetProjects() project_name = projects(0).GetName() msgbox(project_name)</pre>

GetPersonalLibDirectory

Get the path to the PersonalLib directory.

UI Access	N/A

Parameters	None.
Return Value	String path to the <code>PersonalLib</code> directory.

Python Syntax	<code>GetPersonalLibDirectory()</code>
Python Example	<code>oDesktop.GetPersonalLibDirectory()</code>

VB Syntax	<code>GetPersonalLibDirectory</code>
VB Example	<code>oDesktop.GetPersonalLibDirectory</code>

GetProcessID

Returns the process ID of `ansysedt.exe`.

UI Access	N/A
Parameters	None.
Return Value	Integer process ID of <code>ansysedt.exe</code> . For example, 12716.

Python Syntax	<code>GetProcessID ()</code>
Python Example	<code>oDesktop.GetProcessID ()</code>

VB Syntax	GetProcessID
VB Example	<code>oDesktop.GetProcessID</code>

GetProjects

Returns a list of all the projects that are currently open in Electronics Desktop. Once you have the projects, you can iterate through them using standard VBScript methods.

UI Access	N/A
Parameters	None.
Return Value	Returns a collection containing objects for all open projects in Electronics Desktop.

Python Syntax	GetProjects()
Python Example	<code>oDesktop.GetProjects()</code>

VB Syntax	GetProjects
VB Example	<code>oDesktop.GetProjects</code>

GetProjectDirectory

Gets the path to the Project directory.

UI Access	N/A
Parameters	None.
Return Value	String path to the Project directory.

Python Syntax	GetProjectDirectory()
Python Example	<code>oDesktop.GetProjectDirectory()</code>

VB Syntax	GetProjectDirectory
VB Example	<code>oDesktop.GetProjectDirectory</code>

GetProjectList

Returns a list of all projects that are open in Electronics Desktop.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing the names of all open projects in Electronics Desktop.

Python Syntax	GetProjectList()
Python Example	<code>list_of_projects = oDesktop.GetProjectList()</code>

VB Syntax	GetProjectList
VB Example	<code>list_of_projects = oDesktop.GetProjectList</code>

GetSchematicEnvironment

Returns the name of the current schematic environment set in **Tools > Options > General Options > General > Desktop Configuration**.

UI Access	N/A
Parameters	None.
Return Value	Integer representing a schematic environment: • 0 = Circuit • 1 = Twin Builder • 2 = Maxwell Circuit

Python Syntax	GetSchematicEnvironment()
Python Example	<code>oDesktop.GetSchematicEnvironment()</code>

VB Syntax	GetSchematicEnvironment

VB Example	<code>Set oProject = oDesktop.GetSchematicEnvironment</code>

GetScriptingToolsHelper

Note:

This command is for internal Ansys use only.

Returns the object for the scripting tools helper.

UI Access	NA		
Parameters	Name	Type	Description
	None		
Return Value	Object ScriptingTools helper object		

Python Syntax	<code>GetScriptingToolsHelper()</code>
Python Example	<code>oDesktop.GetScriptingToolsHelper()</code>

VB Syntax	<code>GetScriptingToolsHelper</code>
VB Example	<code>oDesktop.GetScriptingToolsHelper()</code>

GetSysLibDirectory

Get the path to the `SysLib` directory.

UI Access	N/A
Parameters	None.
Return Value	String path to the <code>SysLib</code> directory.

Python Syntax	<code>GetSysLibDirectory()</code>
Python Example	<code>oDesktop.GetSysLibDirectory()</code>

VB Syntax	<code>GetSysLibDirectory</code>
VB Example	<code>oDesktop.GetSysLibDirectory</code>

GetTempDirectory

Gets the path to the `Temp` directory.

UI Access	N/A
Parameters	None.
Return Value	String path to the <code>Temp</code> directory.

Python Syntax	<code>GetTempDirectory()</code>

Python Example	<code>oDesktop.GetTempDirectory()</code>

VB Syntax	<code>GetTempDirectory</code>
VB Example	<code>oDesktop.GetTempDirectory</code>

GetUserLibDirectory

Gets the path to the UserLib directory.

UI Access	N/A
Parameters	None.
Return Value	Stringpath to the <code>UserLib</code> directory.

Python Syntax	<code>GetUserLibDirectory()</code>
Python Example	<code>oDesktop.GetUserLibDirectory()</code>

VB Syntax	<code>GetUserLibDirectory</code>
VB Example	<code>oDesktop.GetUserLibDirectory</code>

GetVersion

Returns a string representing the version.

UI Access	N/A
Parameters	None.
Return Value	String containing version of the product.

Python Syntax	GetVersion()
Python Example	<code>oDesktop.GetVersion()</code>

VB Syntax	GetVersion()
VB Example	<code>oDesktop.GetVersion</code>

LaunchJobMonitor

Use: For use in starting job monitoring. This brings up the **Monitor Job** dialog box.

UI Access	Launch Job Monitor		
Parameters	Name	Type	Description
	<projectPath>	String	Path to the project file to be monitored
Return Value	None		

Python Syntax	LaunchJobMonitor()
Python Example	<code>oDesktop.LaunchJobMonitor("C:\\projects\\basic.aedt")</code>

VB Syntax	LaunchJobMonitor()
VB Example	<code>oDesktop.LaunchJobMonitor("C:\\projects\\basic.aedt")</code>

NewProject

Creates a new project. The new project becomes the active project.

UI Access	File > New.
Parameters	None.
Return Value	Object, the project that is added.

Python Syntax	NewProject()
Python Example	<code>oProject = oDesktop.NewProject()</code>

VB Syntax	NewProject
VB Example	<code>Set oProject = oDesktop.NewProject</code>

OpenAndConvertProject

Opens a legacy project and converts or copies it to .aedt format.

UI Access	Click File > Open , and choose a legacy project		
Parameters			
	Name	Type	Description
	<itemPath>	String	full project path of the legacy project
	<legacyChoice>	Integer	0: show conversion dialog box, (same as File > Open of a legacy file) 1: rename (changes extension to .aedt, the original file and results are renamed) 2: copy (creates new file with .aedt extension, and the original file and results remain available)
Return Value	An object reference to the newly opened project which has the .aedt extension		

Warning: If project file/results with the same name and .aedt extension already exist in the same directory, they will be overwritten.

Python Syntax	OpenAndConvertProject(<i>filePath</i> , <i>legacyChoice</i>)
Python Example	<pre>oProject = oDesktop.OpenAndConvertProject("c:\files\optimtee.hfss", 1)</pre> Note: optimtee.hfss is gone after this code executes <pre>oProject = oDesktop.OpenAndConvertProject("c:\files\optimtee.hfss", 2)</pre> Note: optimtee.hfss remains after this code executes

VB Syntax	OpenAndConvertProject(<i>filePath</i> , <i>legacyChoice</i>)

VB Example	<pre>Set oProject = oDesktop.OpenAndConvertProject("c:\files\optimtee.hfss", 1)</pre> Note: optimtee.hfss is gone after this code executes

OpenMultipleProjects

Opens all files of a specified type in a specified directory.

UI Access	N/A		
Parameters	Name	Type	Description
	<directory>	String	Path to the projects.
	<file Type>	String	Type of projects to open.
Return Value	None.		

Python Syntax	OpenAndConvertProject(<filePath>, <legacyChoice>)
Python Example	<pre>oProject = oDesktop.OpenAndConvertProject("c:\files\optimtee.", "*.aedt")</pre>

VB Syntax	OpenAndConvertProject <filePath>, <legacyChoice>
VB Example	<pre>Set oProject = oDesktop.OpenAndConvertProject "c:\files\optimtee.", "*.aedt"</pre>

OpenProject

Opens a specified project.

UI Access	Click File > Open .

Parameters	Name	Type	Description
	<FileName>	String	Full path of the project to open.
Return Value	An object reference to the newly opened project.		

Python Syntax	OpenProject(<FileName>)
Python Example	<code>oDesktop.OpenProject("D:/Projects/Project1.aedt")</code>

VB Syntax	OpenProject <FileName>
VB Example	<code>oDesktop.OpenProject "D:/Projects/Project1.aedt"</code>

OpenProjectWithConversion

Note:

This command is for internal Ansys use only.

Python Syntax	OpenProjectWithConversion()
Python Example	<code>oDesktop.OpenProjectWithConversion()</code>

Paste (Project Object)

Pastes a design in the active project.

UI Access	Edit > Paste
Parameters	None
Return Value	None

Python Syntax	Paste()
Python Example	<code>oProject.Paste()</code>

VB Syntax	Paste
VB Example	<code>oProject.Paste</code>

Paste (Project Object)

Pastes a design in the active project.

UI Access	Edit > Paste.
Parameters	None.
Return Value	None

Python Syntax	Paste()
Python Example	<code>oProject.Paste()</code>

VB Syntax	Paste
VB Example	<code>oProject.Paste</code>

PauseRecording

Temporarily stop script recording.

UI Access	NA		
Parameters	Name	Type	Description
	None		
Return Value	None		

Python Syntax	PauseRecording()
Python Example	<code>oDesktop.PauseRecording()</code>

VB Syntax	PauseRecording

VB Example	<code>oDesktop.PauseRecording</code>

PauseScript

Pause the execution of the script and pop up a message to the user. The script execution will not resume until the user chooses.

UI Access	Tools > Pause Script		
Parameters	Name	Type	Description
	<Message>	String	Any Text.
Return Value	None		

Python Syntax	<code>PauseScript (<Message>)</code>
Python Example	<code>oDesktop.PauseScript("Text to display in pop-up dialog box.")</code>

VB Syntax	<code>PauseScript <Message></code>
VB Example	<code>oDesktop.PauseScript "Text to display in pop-up dialog box."</code>

Print

Prints the contents of the active view window.

UI Access	File > Print.
Parameters	None.
Return Value	None.

Python Syntax	Print()
Python Example	<code>oDesktop.Print()</code>

VB Syntax	Print
VB Example	<code>oDesktop.Print</code>

QuitApplication

Exits the desktop.

UI Access	File > Exit.
Parameters	None.
Return Value	None.

Python Syntax	QuitApplication()
Python Example	<code>oDesktop.QuitApplication()</code>

VB Syntax	QuitApplication

VB Example	<code>oDesktop.QuitApplication</code>

RefreshJobMonitor

For use in monitoring a job.

UI Access	Tools > Job Management > Monitor Jobs.
Parameters	None.
Return Value	A string specifying the job state. The result can be any of the following strings: • "Monitor Not Visible" • "Queued" • "Running" • "Shutting Down" • "Unknown" • "Completed" • "Not Monitoring" • "Starting Monitoring"

Python Syntax	<code>RefreshJobMonitor()</code>
Python Example	<code>oDesktop.RefreshJobMonitor()</code>

VB Syntax	<code>RefreshJobMonitor</code>

VB Example	<code>oDesktop.RefreshJobMonitor</code>

ResetLogging

Redirects simulation log file to a specified directory and log level.

UI Access	N/A		
Parameters	Name	Type	Description
	<code><logFile></code>	String	Path to log file.
	<code><logLevel></code>	Integer	Specified log level.
Return Value	None.		

Python Syntax	<code>ResetLogging(<logFile>, <logLevel>)</code>
Python Example	<code>oDesktop.ResetLogging("C:/Project1.aedtresults/", 1)</code>

VB Syntax	<code>ResetLogging <logFile>, <logLevel></code>
VB Examples	<code>oDesktop.ResetLogging "C:/Project1.aedtresults/", 1</code>

RestoreProjectArchive

Restores a previously archived project to a specified path.

UI Access	File > Restore Archive.		
Parameters	Name	Type	Description
	<ArchiveFilePath>	String	Path to archived file
	<ProjectFilePath>	String	Path to restore location
	<OverwriteExistingFiles>	Boolean	True to overwrite an existing file of the same name; else False.
	<OpenProjectAfterRestore>	Boolean	True to open the project after it is restored; else False.
Return Value	None.		

Python Syntax	RestoreProjectArchive (<Archivefilepath>, <ProjectFilePath>, <OverwriteExistingFiles>, <OpenProjectAfterRestore>)
Python Example	oDesktop.RestoreProjectArchive ("C:\Users\jdoe\Documents\OptimTee.aedt", "C:\Documents\OptimTee.aedt", False, True)

VB Syntax	RestoreProjectArchive <Archivefilepath>, <ProjectFilePath>, <OverwriteExistingFiles>, <OpenProjectAfterRestore>
VB Example	oDesktop.RestoreProjectArchive "C:\Users\jdoe\Documents\OptimTee.aedt", _ "C:\Documents\OptimTee.aedt", false, true

RestoreWindow

Restores a minimized Desktop window.

UI Access	N/A

Parameters	None.
Return Value	None.

Python Syntax	RestoreWindow()
Python Example	<code>oDesktop.RestoreWindow()</code>

VB Syntax	RestoreWindow
VB Example	<code>oDesktop.RestoreWindow</code>

ResumeRecording

Resume recording a script.

UI Access	N/A
Parameters	None.
Return Value	None

Python Syntax	ResumeRecording()
Python Example	<code>oDesktop.ResumeRecording()</code>

VB Syntax	ResumeRecording
VB Example	<code>oDesktop.ResumeRecording</code>

RunACTWizardScript

Note:

This command is for internal Ansys use only.

Python Syntax	RunACTWizardScript()
Python Example	<code>oDesktop.RunACTWizardScript()</code>

RunProgram

Runs an external program.

UI Access	NA		
Parameters	Name	Type	Description
	<ProgName>	String	Name of the program to run.
	<ProgPath>	String	Location of the program. Pass in an empty string to use the system path.
	<WorkPath>	String	Working directory in which program will start.
	<ArgArray>	Array of Strings	Arguments to pass to the program. If no arguments, pass in <code>None</code>

Return Value	None

Python Syntax	RunProgram (<ProgName>, <ProgPath>, <WorkPath>, <ArgArray>)
Python Example	<pre>oDesktop.RunProgram("winword.exe", _ "C:\Program Files\Microsoft Office\Office10", _ "", None)</pre>

VB Syntax	RunProgram <ProgName>, <ProgPath>, <WorkPath>, <ArgArray>
VB Example	<pre>oDesktop.RunProgram "winword.exe", _ "C:\Program Files\Microsoft Office\Office10", _ "", None</pre>

RunScript

Launches another script from within the script currently being executed.

UI Access	Tools>Run Script		
Parameters	Name	Type	Description
	<SCriptPath>	String	Name or full path of the script to execute. If the full path to the script is not specified, Twin Builder searches for the specified script in the following locations, in this order:

			1. Personal library directory. This is the PersonalLib subdirectory in the project directory. The project directory can be specified in the General Options dialog box (click >Tools > Options > General Options to open this dialog box) under the Project Options tab. 2. User library directory. This is the userlib subdirectory in the library directory. The library directory can be specified General Options dialog in the box (click Tools > Options > General Options to to open this dialog box) under the Project Options tab. 3. System library directory. This is the syslib subdirectory in the library directory. The library directory can be specified in the General Options dialog box (click Tools > Options > General Options to open this dialog box) under the Project Options tab. 4. HFSS installation directory.
Return Value	Long the return code for the script method.		

Python Syntax	<code>RunScript (<ScriptPath>)</code>
Python Example	<code>oDesktop.RunScript ("C:/Project/test1.vbs")</code>

VB Syntax	RunScript <ScriptPath>
VB Example	oDesktop.RunScript ("C:/Project/test1.vbs")

RunScriptWithArguments

Similar to RunScript, launch another script from within the currently executing script, but with arguments.

UI Access	NA		
Parameters	Name	Type	Description
	<ScriptPath>	String	The name or full path of the script to execute. If the full path to the script is not specified, the software looks for the script in the following locations: • Personal library directory: "PersonalLib" The PersonalLib directory can be specified in Tools > Options > General Options on the 'Project Options' tab. • User library directory: "userlib" The UserLib directory can be specified in Tools > Options > General Options on the 'Project Options' tab. • System library directory: "syslib" The SysLib directory can be specified in Tools > Options > General Options on the 'Project Options' tab. • Software installation directory
	<Arguments>	String	The arguments to supply to the specified script.
Return Value	Long the return code for the script method.		

Python Syntax	<code>RunScriptWithArguments (<ScriptPath>, <Arguments>)</code>
Python Example	<pre>oDesktop.RunScriptWithArguments ("C:/Project/test2.py", "foo")</pre>

VB Syntax	<code>RunScriptWithArguments <ScriptPath>, <Arguments></code>
VB Example	<pre>oDesktop.RunScriptWithArguments "C:/Project/test2.vbs", "foo"</pre>

SelectScheduler

Selects the scheduler used for batch job submission. It tries non-graphical selection of the scheduler, attempting to get version information from the scheduler in order to check for successful selection. If unable to get the information, it displays the **Select Scheduler** window and waits for the user to complete the settings.

UI Access	Tools > Job Management > Select Scheduler.		
Parameters	Name	Type	Description
	<option>	String	One of the following options (not case sensitive): • Empty string for remote RSM service • "RSM" for local RSM • "Windows HPC" for Windows HPC • "LSF" for Load-Sharing Facility • "SGE" for Grid Engine (GE, OGE, SGE, UGE, etc.) • "PBS" for Portable Batch Scheduler/System (PBSPPro, Torque, Maui, etc.)

			"Ansys Cloud" for Ansys Cloud
	<code><address (optional)></code>	String	String specifying the IP address or hostname of the head node or for the remote host running the RSM service.
	<code><username (optional)></code>	String	Username string to use for remote RSM service (or blank to use username stored in current submission host user settings). If the (non-blank) username doesn't match the username stored in current submission host user settings, then the Select Scheduler dialog is displayed to allow for password entry prior to job submission.
	<code><forcePasswordEntry (optional)></code>	String	Boolean used to force display of the Select Scheduler GUI to allow for password entry prior to job submission.
Return Value	The selected scheduler (if selection was successful, this string should match the input option string, although it could differ in upper/lowercase).		

Python Syntax	<code>Select Scheduler(<option>, <address>, <username>, <forcePasswordEntry>)</code>
Python Example	<pre>result = oDesktop.SelectScheduler("Windows HPC", "headnode.win.example.com")</pre>

VB Syntax	<code>Select Scheduler <option>, <address>, <username>, <forcePasswordEntry></code>
VB Example	<pre>result = oDesktop.SelectScheduler "Windows HPC", _ "headnode.win.example.com"</pre>

SetActiveProject

Specifies the name of the project that should become active in the desktop. Returns that project.

UI Access	N/A		
Parameters	Name	Type	Description
	<ProjectName>	String	The name of the project already in the Desktop that is to be activated.
Return Value	Object, the project that is activated.		

Python Syntax	SetActiveProject (<ProjectName>)		
Python Example	<pre>oProject = oDesktop.SetActiveProject ("Project1")</pre>		

VB Syntax	SetActiveProject <ProjectName>		
VB Example	<pre>Set oProject = oDesktop.SetActiveProject "Project1"</pre>		

SetActiveProjectByPath

Specifies the name of the project that should become active in the desktop. Returns that project. If a user has two projects open with the same name, the result of `SetActiveProject` is ambiguous (the first one listed in selected). This command permits unambiguous specification of the active project.

UI Access	N/A		
Parameters	Name	Type	Description
	<ProjectName>	String	The full path name of the project already in the Desktop that is to be activated.
Return Value	Object, the project that is activated.		

Python Syntax	SetActiveProjectByPath(<ProjectName>)
Python Example	<code>oProject = oDesktop.SetActiveProjectByPath("c:\Projects\MyProject.aedt")</code>

VB Syntax	SetActiveProjectByPath <ProjectName>
VB Example	<code>Set oProject = oDesktop.SetActiveProjectByPath "c:\Projects\MyProject.aedt"</code>

SetCustomMenuSet

Sets the custom menu set for Electronics Desktop.

UI Access	Navigate to Tools > Options > General Options > General > Desktop Configuration . Select a configuration using the Custom Menu Set drop-down menu.		
Parameters	Name	Type	Description
	<customMenuSet>	String	Name of desired menu set. Can be one of: 'Default' 'EM' 'RF' 'RF.0' 'SI' 'SI1.0' 'SI2.0' 'Twin Builder'

Return Value	None

Python Syntax	SetCustomMenuSet(<customMenuSet>)
Python Example	<code>oDesktop.SetCustomMenuSet('Default')</code>

VB Syntax	SetCustomMenuSet(<customMenuSet>)
VB Example	<code>Set oProject = oDesktop.SetCustomMenuSet "EM"</code>

SetDesktopConfiguration

Sets the desktop configuration.

UI Access	Navigate to Tools > Options > General Options > General > Desktop Configuration . Select a configuration using the Set targeted configuration drop-down menu.		
Parameters	Name	Type	Description
	<configName>	String	Name of desired Desktop configuration. Can be one of: • 'All' • 'EM' • 'RF' • 'SI' • 'Twin Builder'
Return Value	None		

Python Syntax	SetDesktopConfiguration (<configName>)
Python Example	<code>oDesktop.SetDesktopConfiguration('RF')</code>

VB Syntax	SetDesktopConfiguration(<configName>)
VB Example	<code>Set oProject = oDesktop.SetDesktopConfiguration "SI"</code>

SetLibraryDirectory

Sets the library directory path. The specified directory must already exist and contain a syslib folder.

UI Access	NA		
Parameters	Name	Type	Description
	<DirectoryPath>	String	The path to the SysLib Directory
Return Value	None		

Python Syntax	SetLibraryDirectory (<DirectoryPath>)
Python Example	<code>oDesktop.SetLibraryDirectory("c:\libraries")</code>

VB Syntax	SetLibraryDirectory <DirectoryPath>
VB Example	oDesktop.SetLibraryDirectory"c:\libraries"

SetProjectDirectory

Sets the project directory path.

UI Access	N/A		
Parameters	Name	Type	Description
	<DirectoryPath>	String	The path to the project directory. This should be writeable by the user.
Return Value	None.		

Python Syntax	SetProjectDirectory (<DirectoryPath>)
Python Example	oDesktop.SetProjectDirectory("c:\projects")

VB Syntax	SetProjectDirectory <DirectoryPath>
VB Example	oDesktop.SetProjectDirectory "c:\projects"

SetSchematicEnvironment

Sets the schematic environment for Electronics Desktop.

UI Access	Navigate to Tools > Options > General Options > General > Desktop Configuration . Select a schematic environment using the Custom Menu Set drop-down menu.

Parameters	Name	Type	Description
	<schEnv>	Integer	Desired schematic environment. Can be one of: • 0 (Circuit) • 1 (Twin Builder) • 2 (Maxwell Circuit)
Return Value	None		

Python Syntax	SetSchematicEnvironment(<schEnv>)
Python Example	<code>oDesktop.SetSchematicEnvironment(1)</code>

VB Syntax	SetSchematicEnvironment(<schEnv>)
VB Example	<code>Set oProject = oDesktop.SetSchematicEnvironment 2</code>

SetTempDirectory

Sets the temp directory path. The directory will be automatically created if it does not already exist.

UI Access	N/A		
Parameters	Name	Type	Description
	<DirectoryPath>	String	The path to the Temp directory. This should be writeable by the user.
Return Value	None.		

Python Syntax	SetTempDirectory (<DirectoryPath>)
Python Example	<code>oDesktop.SetTempDirectory("c:\tmp")</code>

VB Syntax	SetTempDirectory <DirectoryPath>
VB Example	<code>oDesktop.SetTempDirectory "c:\tmp"</code>

ShowDockingWindow

Shows or hides a docking window.

UI Access	Right click docking window > Show/Hide.		
Parameters	Name	Type	Description
	<windowName>	String	The window name (for example, "Message Manager", "Component Libraries", "Properties")
	<show>	Boolean	True to show; False to hide.
Return Value	None.		

Python Syntax	ShowDockingWindow (<windowName>, <show>)
Python Example	<code>oDesktop.ShowDockingWindow('Message Manager', False)</code>

VB Syntax	ShowDockingWindow <windowName> <show>
VB Example	oDesktop.ShowDockingWindow "Message Manager" False

Sleep

Suspends execution of HFSS for the specified number of milliseconds, up to 60,000 milliseconds (1 minute).

UI Access	NA		
Parameters	Name	Type	Description
	<TimeInMil- liseconds>	Integer	The time that the execution should be suspended in milliseconds
Return Value	None		

Python Syntax	Sleep (<TimeInMilliseconds>)
Python Example	oDesktop.Sleep(1000)

VB Syntax	Sleep <TimeInMilliseconds>
VB Example	oDesktop.Sleep 1000

SubmitJob

Submits a batch job to a scheduler. When submitting the same project file multiple times, you should have the script wait for each job (or jobs for multi-step) to finish, which can be done via the monitoring functions LaunchJobMonitor() and RefreshJobMonitor(), checking the result of RefreshJobMonitor() in a loop until it returns completed ("Monitor Not Visible") status.

UI Access	Tools > Job Management > Submit Job.		
Parameters	Name	Type	Description
	<settingsPath>	String	Path to the settings file (exported from the Submit Job GUI) to use for submission.
	<projectPath>	String	Path to the project file to use in the batch job. This could be an archive (.aedtz file) or an un-archived project.
	<design (optional)>	String	Name of the design to use for batch solve.
	<setup (optional)>	String	Name of the specific setup to solve.
Return Value	Array of job ID strings (empty if no jobs submitted).		

Python Syntax	SubmitJob(<settingsPath>, <projectPath>, <design>, <setup>)
Python Example	<pre> jobIDs = oDesktop.SubmitJob("C:\\hpc-settings\\Submit_ Job_Settings.ared", "C:\\projects\\basic.aedt") moreIDs = oDesktop.SubmitJob("C:\\hpc-settings\\Submit_ Job_Settings.ared", "C:\\projects\\basic.aedt", "Design1", "Setup1") </pre>

VB Syntax	SubmitJob <settingsPath>, <projectPath>, <design>, <setup>

VB Example	<pre>jobIDs = oDesktop.SubmitJob "C:\\hpc-settings\\Submit_ Job_Settings.aredt", "C:\\projects\\basic.aedt" moreIDs = oDesktop.SubmitJob "C:\\hpc-settings\\Submit_ Job_Settings.aredt", "C:\\projects\\basic.aedt", "Design1", "Setup1"</pre>

TileWindows

Arrange all open windows in a tiled format.

UI Access	From main menu, Window >Tile Horizontally or Window >Tile Vertically .		
Parameters	Name	Type	Description
	<TilingFlag>	Integer	• 0 – Tile vertically. • 1 – Tile horizontally.
Return Value	None.		

Python Syntax	TileWindows(<TilingFlag>)
Python Example	oDesktop.TileWindows(0)

VB Syntax	TileWindows <TilingFlag>
VB Example	oDesktop.TileWindows 0

Desktop Commands For Registry Values

The Ansys Registry is stored as XML format file. By default it is located at C:\User-s\<UserName>\Documents\Ansoft\<AnsysProductNameversion>\config\<PC_NAME>_user.XML. Most of the Ansys product configuration information is stored in this XML file. These methods allow you to change the product configuration in VB or Python.

For example, to set the DSO & HPC analysis setup for HFSS using a Python script:

1. Start Maxwell.
2. Go to the DSO and HPC options and create a setup named "test".
3. Export the setup to a file (for example, c:\temp\test.acf).
4. Copy the exported file to a target PC (for example, f:\temp\test.acf).
5. Run the following script:

```
#import the setup

oDesktop.SetRegistryFromFile("f:\\temp\\test.acf")

# Set Active Setup to "test"

oDesktop.SetRegistryString("Desktop/ActiveDSOConfigurations/Maxwell", "test")
```

See the following subtopics:

[DeleteRegistryEntry](#)

[DoesRegistryValueExist](#)

[GetRegistryInt](#)

[GetRegistryString](#)

[SetRegistryFromFile](#)

[SetRegistryInt](#)

[SetRegistryString](#)

DoesRegistryValueExist

Determines whether a registry value exists.

UI Access	N/A		
Parameters	Name	Type	Description
	<KeyName>	String	Full name of registry key, including path.
Return Value	Boolean: • True – Key exists. • False – Key does not exist.		

Python Syntax	DoesRegistryValueExist(<KeyName>)
Python Example	<pre>Exist = oDesktop.DoesRegistryValueExist('Desktop/ActiveDSOConfigurations/Maxwell')</pre>

VB Syntax	DoesRegistryValueExist(<KeyName>)
VB Example	<pre>bExist = oDesktop.DoesRegistryValueExist("Desktop/ActiveDSOConfigurations/Maxwell")</pre>

GetRegistryInt

Obtains registry key integer value.

UI Access	N/A		
Parameters	Name	Type	Description
	<KeyName>	String	Full name of registry key, including path.
Return Value	Integer if the integer value is found. Return as Bad-Argument-Value if registry key does not exist or it is not an integer value.		

Python Syntax	GetRegistryInt(<KeyName>)		
Python Example	<pre>num = oDesktop.GetRegistryInt('Desktop/Settings/ProjectOptions/Maxwell/UpdateReportsDynamicallyOnEdits')</pre>		

VB Syntax	GetRegistryInt(<KeyName>)		
VB Example	<pre>num = oDesktop.GetRegistryInt("Desktop/Settings/ProjectOptions/Maxwell/UpdateReportsDynamicallyOnEdits")</pre>		

GetRegistryString

Obtains registry key string value.

UI Access	N/A		
Parameters	Name	Type	Description
	<KeyName>	String	Full name of registry key, including path.

Return Value	String if the string value is found. Return as Bad-Argument-Value if registry key does not exist or it is not a string value.

Python Syntax	GetRegistryString(<KeyName>)
Python Example	<pre>activeDSO = oDesktop.GetRegistryString('Desktop/ActiveDSOConfigurations/Maxwell')</pre>

VB Syntax	GetRegistryString(<KeyName>)
VB Example	<pre>activeDSO = oDesktop.GetRegistryString("Desktop/ActiveDSOConfigurations/Maxwell")</pre>

SetRegistryFromFile

Configures registry by specifying an Analysis Configuration file which must have been exported from the HPC and Analysis panel.

UI Access	N/A		
Parameters	Name	Type	Description
	<filePath>	String	Full file path of registry file.
Return Value	Success if configuration is imported. Bad argument value if the file is not found or does not contain valid analysis configuration data.		

Python Syntax	SetRegistryFromFile(<filePath>)
Python Example	<pre>oDesktop.SetRegistryFromFile('c:/temp/test.acf')</pre>

VB Syntax	SetRegistryFromFile <filePath>
VB Example	oDesktop.SetRegistryFromFile "c:/temp/test.acf"

SetRegistryInt

Sets registry key to an integer value.

UI Access	N/A		
Parameters	Name	Type	Description
	<KeyName>	String	Full name of registry key, including path.
	<int>	Integer	Integer value to be assigned to registry key.
Return Value	Success if the key is defined as an integer. Bad argument value if a key is not defined, or if the value is a text string.		

Python Syntax	SetRegistryInt(<KeyName>, <int>)
Python Example	oDesktop.SetRegistryInt('Desktop/Settings/ProjectOptions/Maxwell/UpdateReportsDynamicallyOnEdits', 0)

VB Syntax	SetRegistryInt <KeyName> <int>
VB	oDesktop.SetRegistryInt "Desktop/Settings/ProjectOptions/Maxwell/Up-

Example	<code>dateReportsDynamicallyOnEdits", 0</code>

SetRegistryString

Sets registry key to a string value.

UI Access	N/A		
Parameters	Name	Type	Description
	<KeyName>	String	Full name of registry key, including path.
	<value>	String	String value to be assigned to registry key.
Return Value	Success if the key is defined as a text string. Bad argument value if the key is not defined or requires an integer value.		

Python Syntax	<code>SetRegistryString(<KeyName>, <value>)</code>
Python Example	<code>oDesktop.SetRegistryString('Desktop/ActiveDSOConfigurations/Maxwell', 'Local')</code>

VB Syntax	<code>SetRegistryString <KeyName>, <value></code>
VB Example	<code>oDesktop.SetRegistryString "Desktop/ActiveDSOConfigurations/Maxwell", "Local"</code>

ImportExport Tool Commands

These oDesktop commands are run by using the GetTool script to call the ImportExport Tool.

```
oTool = oDesktop.GetTool("ImportExport")
```


Scripts run via the ImportExport Tool include:

[ImportANF](#)

[ImportANFV2](#)

[ImportAutoCAD](#)

[ImportAWRMicrowaveOffice](#)

[ImportEDB](#)

[ImportExtracta](#)

[ImportGDSII](#)

[ImportGerber](#)

[ImportIDF](#)

[ImportIDFandMerge](#)

[ImportIPC](#)

[ImportODB](#)

[ImportXFL](#)

ImportANF

Imports an ANF file into a new project. For older ANFv2 files, use [ImportANFv2](#).

UI Access	File > Import > ANF.		
Parameters	Name	Type	Description
	<ANFfilename>	String	Full path of ANF file.
Return Value	None.		

Python Syntax	<code>ImportANF(<ANFfilename>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportANF('C:\\\\AnsTranslator\\\\results\\\\package4.anf')</pre>

VB Syntax	<code>ImportANF <ANFfilename></code>
VB Example	<pre>Dim oAnsoftApp Dim oDesktop Dim oProject Dim oDesign Dim oEditor Dim oModule Dim oProjects Dim omachine Dim oTool Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop") Set oDesktop = oAnsoftApp.GetAppDesktop() oDesktop.RestoreWindow Set oTool = oDesktop.GetTool("ImportExport")</pre>

```
oTool.ImportANF("%UserProfile%\Documents\HFSS Examples\package.anf")
```

ImportANFv2

Imports an ANFv2 file into a new project. For newer ANF files, use [ImportANF](#).

UI Access	File > Import > ANF.		
Parameters	Name	Type	Description
	<ANFfilename>	String	Full path of ANF file.
	<outputPathName>	String	Full path of *.aedb output file.
	<controlFileName>	String	Full path of XML control file.
	<cmpFileName>	String	Full path of CMP file. Pass empty string if none.
Return Value	None.		

Python Syntax	ImportANFv2 (<ANFfilename>, <outputPathName>, <controlFileName>, <cmpFileName>)
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportANFV2("C:/Users/jdoe/Documents/SAMPLEFILES/my_model.anf", "C:/Users/jdoe/Documents/Ansoft/my_model.aedb", "C:/Users/jdoe/Documents/Ansoft/my_model.xml", "")</pre>

VB Syntax	ImportANFv2 <ANFfilename>, <outputPathName>, <controlFileName>, <cmpFileName>
VB Example	<pre>Dim oAnsoftApp</pre>

	<pre>Dim oDesktop Dim oProject Dim oDesign Dim oEditor Dim oModule Dim oProjects Dim omachine Dim oTool Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop") Set oDesktop = oAnsoftApp.GetAppDesktop() oDesktop.RestoreWindow Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportANFV2 "C:/Users/jdoe/Documents/SAMPLEFILES/my_model.anf", _ "C:/Users/jdoe/Documents/Ansoft/my_model.aedb", _ "C:/Users/jdoe/Documents/Ansoft/my_model.xml", ""</pre>

ImportAutoCAD

Imports an AutoCAD file into a new project.

UI Access	File > Import > AutoCAD.		
Parameters	Name	Type	Description
	<dxfileName>	String	Full path of DXF file.

	<outputPathFileName>	String	Full path of EDB file to create during import.
	<controlFileName>	String	Full path of XML control file. Pass empty string if none.
Return Value	None.		

Python Syntax	ImportAutoCAD (<dxfFileName>, <outputPathFileName>, <controlFileName>)		
Python Example	<pre>Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportAutoCAD('C:/MyPath/a4lines.dxf', 'C:/MyPath/a4lines.aedb.edb', 'C:/MyPath/a4lines.xml')</pre>		

VB Syntax	ImportAutoCAD <dxfFileName>, <outputPathFileName>, <controlFileName>		
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportAutoCAD "C:/MyPath/a4lines.dxf", "C:/MyPath/a4lines.aedb.edb", "C:/MyPath/a4lines.xml"</pre>		

ImportAWRMicrowaveOffice

Imports an AWRMicrowaveOffice file into a new project.

UI Access	File > Import > AWRMicrowaveOffice.		
Parameters	Name	Type	Description
	<xmlFileName>	String	Full path of XML file.
	<outputPathName>	String	Full path of output file.
	<logFileName>	String	Full path of log file.
Return Value	None.		

Python Syntax	<code>ImportAWRMicrowaveOffice (<xmlFileName>, <outputPathName>, <logFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportAWRMicrowaveOffice('C:/MyFiles/package4.xml', 'C:/MyFiles/package4.aedb', 'C:/MyFiles/package4.log')</pre>

VB Syntax	<code>ImportAWRMicrowaveOffice <xmlFileName>, <outputPathName>, <logFileName></code>
VB Example	<pre>Dim oAnsoftApp Dim oDesktop Dim oProject Dim oDesign Dim oEditor Dim oModule Dim oProjects Dim omachine Dim oTool</pre>

```

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oTool = oDesktop.GetTool("ImportExport")
oTool.ImportAWRMicrowaveOffice "C:/MyFiles/package4.xml", "C:/MyFiles/package4.aedb",
-
"C:/MyFiles/package4.log"

```

ImportEDB

Imports an EDB file into a new project.

UI Access	File > Import > EDB.		
Parameters	Name	Type	Description
	<edbFileName>	String	Full path of EDB file.
Return Value	None.		

Python Syntax	ImportEDB (<edbFileName>)
Python Example	<pre> oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportEDB('C:/MyFiles/projectforimport.aedb/edb.def') </pre>

VB Syntax	ImportEDB <edbFileName>

VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportEDB "C:/MyFiles/projectforimport.aedb/edb.def"</pre>

ImportExtracta

Imports a Cadence Extracta file into a new project.

Note:
In order for this script to work, you must have the Cadence-supplied executable Extracta.exe installed on your machine.

UI Access	File > Import > Cadence APD / Allegro / SiP.		
Parameters	Name	Type	Description
	<extractaFileName>	String	Full path of Extracta file.
	<outputPathName>	String	Full path of output file to be created.
	<controlFileName>	String	Optional. Full path to XML control file.
Return Value	None.		

Python Syntax	ImportExtracta (<extractaFileName>, <outputPathName>, <controlFileName>)
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportExtracta('C:/MyFiles/projectforimport.brd', 'C:/MyFiles/project.edb', '')</pre>

VB Syntax	<code>ImportExtracta <extractaFileName>, <outputPathName>, <controlFileName></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportExtracta "C:/MyFiles/projectforimport.brd", "C:/MyFiles/project.edb", ""</pre>

ImportGDSII

Imports a GDSII file into a new project.

UI Access	File > Import > GDSII.		
Parameters	Name	Type	Description
	<code><gdsiiFileName></code>	String	Full path of GDSII file.
	<code><outputPathName></code>	String	Full path of EDB file to create during import.
	<code><controlFileName></code>	String	Optional. Full path of XML control file. Full path of “technology” (corresponding to <code>-t=technologyfile</code> argument in <code>anstranslator</code>). Pass empty string if none.
	<code><mapFileName></code>	String	If technology file was used in place of the control file. Full path to either <code>gdsii</code> mapping file (corresponding to <code>-g=gdsMappingFile</code> argument in <code>anstranslator</code>) or XML control file. Otherwise, pass empty string.
Return Value	None.		

Python Syntax	<code>ImportGDSII(<gdsiiFileName>, <outputPathName>, <controlFileName>, <mapFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportGDSII('C:/Files/test.gds', 'C:/Files/test.aedb',</pre>

	<code>'C:/Files/test.ircx', 'C:/Files/test.xml')</code>

VB Syntax	<code>ImportGDSII <gdsiiFileName>, <outputPathName>, <controlFileName>, <mapFileName></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportGDSII "C:/Files/test.gds", "C:/Files/test.aedb.edb", _ "C:/Files/test.xml"</pre>

ImportGerber

Imports a GERBER file into a new project.

UI Access	File > Import > GERBER.		
Parameters	Name	Type	Description
	<code><gerberFileName></code>	String	Full path of GERBER file.
	<code><outputPathName></code>	String	Full path of EDB file to create during import.
	<code><controlFileName></code>	String	Optional. Full path of XML control file. Pass empty string if none.
Return Value	None.		

Python Syntax	<code>ImportGerber(<gerberFileName>, <outputPathName>, <controlFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport')</pre>

	<pre>oTool.ImportGERBER('C:/Files/test.gbr', 'C:/Files/test.aedb.edb', 'C:/Files/test.xml')</pre>

VB Syntax	ImportGerber <gerberFileName>, <outputPathName>, <controlFileName>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportGERBER "C:/Files/test.gbr", "C:/Files/test.aedb.edb", _ "C:/Files/test.xml"</pre>

ImportIDF

Imports an IDF file into a new project. To merge with an existing design, use [ImportIDFandMerge](#).

UI Access	placeholder		
Parameters	Name	Type	Description
	<NAME>	string	Settings
	<Board>	bool	Full path of board file
	<Library>	int	Full path of library file
	<Control>	int	Full path of control file
	<Filters>	list	["Cap","Height","HeightExclude2D","Ind","Power","Res"]
	<CreateFilteredAsNonModel>	bool	True or False
	<FootPrint>	string	Footprint size and unit
	<NAME>	string	definitionOverridesMap
	<NAME>	string	instanceOverridesMap
	<HighSurfThickness>	string	High side surface thickness and unit
	<LowSurfThickness>	string	Low side surface thickness and unit
	<InternalLayerThickness>	string	Internal layer thickness and unit

	<NumInternalLayer>	int	Number of internal layers
	<HighSurfaceCopper>	int	High side surface coverage percentage
	<LowSurfaceCopper>	int	Low side surface coverage percentage
	<InternalLayerCopper>	int	Internal layer coverage percentage
	<TraceMaterial>	string	Trace material name
	<SubstrateMaterial>	int	Substrate material name
	<CreateBoard>	bool	True or False
	<ModelBoardAsRect>	bool	True or False
	<ModelDeviceAsRect>	bool	True or False
	<Cutoff>	bool	True or False
	<IncludeDrilledHoles>	bool	True or False
	<HoleDiameterCutoff>	string	Hole diameter size and unit
	<ReplaceDevices>	bool	True or False
	<CreatePointsOnBoardForInstances>	bool	True or False
Return Value	None.		

Python Syntax	ImportIDF_1 (<NAME>, <Board>, <Library>, <Control>, <Filters>, <CreateFilteredAsNonModel>, <NAME>, <NAME>, <HighSurfThickness>, <LowSurfThickness>, <InternalLayerThickness>, <NumInternalLayer>, <HighSurfaceCopper>, <LowSurfaceCopper>, <InternalLayerCopper>, <TraceMaterial>, <SubstrateMaterial>, <CreateBoard>, <ModelBoardAsRect>, <ModelDeviceAsRect>, <Cutoff>, <IncludeDrilledHoles>, <ReplaceDevices>)
Python Example	<pre>oDesign.ImportIDF(["NAME:Settings",</pre>

```

"Board:="                , "C:\\Users\\Model Files\\brd_board.emn",
"Library:="              , "C:\\Users\\Model Files\\brd_board.emp",
"Control:="              , "C:\\Users\\Model Files\\brd_board.xml",
"Filters:="              , ["HeightExclude2D"],
"CreateFilteredAsNonModel:=", False,
"FootPrint:="            , "0.1mm2",
[
    "NAME:definitionOverridesMap"
],
[
    "NAME:instanceOverridesMap"
],
"HighSurfThickness:="    , "0.07mm",
"LowSurfThickness:="     , "0.07mm",
"InternalLayerThickness:=", "0.07mm",
"NumInternalLayer:="     , 2,
"HighSurfaceCopper:="    , 30,
"LowSurfaceCopper:="     , 30,
"InternalLayerCopper:="  , 30,
"TraceMaterial:="        , "Cu-Pure",
"SubstrateMaterial:="    , "FR-4",
"CreateBoard:="          , True,
"ModelBoardAsRect:="     , True,
"ModelDeviceAsRect:="    , False,
"Cutoff:="               , False,
"IncludeDrilledHoles:="  , True,
"HoleDiameterCutoff:="   , "0.1mm",
"ReplaceDevices:="       , False,
"CreatePointsOnBoardForInstances:=", False
] )

```

VB Syntax	ImportIDF <idfFileName>, <outputPathName>, <controlFileName>, <libFileName>
VB Example	<pre>oDesign.ImportIDF Array("NAME:Settings", "Board:=", _"C:\Users\Model Files" "\brd_ board.emn", "Library:=", _"C:\Users\Model Files" "\brd_board.emp", "Control:=", _"C:\Users\Model Files" "\brd_board.xml", "Filters:=", Array("HeightExclude2D"), "CreateFilteredAsNonModel:=", _false, Array("NAME:definitionOverridesMap"), Array("NAME:instanceOverridesMap"), "HighSurfThickness:=", _"0.07mm", "LowSurfThickness:=", "0.07mm", "InternalLayerThickness:=", "0.07mm", "NumInternalLayer:=", _2, "HighSurfaceCopper:=", 30, "LowSurfaceCopper:=", 30, "InternalLayerCopper:=", _30, "TraceMaterial:=", "Cu-Pure", "SubstrateMaterial:=", "FR-4", "CreateBoard:=", _true, "ModelBoardAsRect:=", true, "ModelDeviceAsRect:=", false, "Cutoff:=", _false, "IncludeDrilledHoles:=", true, "ReplaceDevices:=", false)</pre>

ImportIDFandMerge

Imports an IDF file into a new project. To import without merging, use [ImportIDF](#).

UI Access	File > Import > IDF. Enable Merge with existing and select a project/design.		
Parameters	Name	Type	Description

	<idfFileName>	String	Full path of GERBER file.
	<libFileName>	String	Optional. Full path of library file. Pass empty string if none.
	<controlFileName>	String	Optional. Full path of XML control file. Pass empty string if none.
	<project>	String	Name of project to merge with.
	<design>	String	Name of design to merge with.
Return Value	None.		

Python Syntax	<code>ImportIDFandMerge(<idfFileName>, <libPathName>, <controlFileName>, <project>, <design>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportIDFandMerge('C:/Files/test.brd', 'C:/Files/test.aedb.lib', 'C:/Files/test.xml', 'Project1', 'Design12')</pre>

VB Syntax	<code>ImportIDFandMerge<idfFileName>, <libPathName>, <controlFileName>, <project>, <design></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportIDFandMerge "C:/Files/test.brd", "C:/Files/test.lib", _ "C:/Files/test.xml", "Project1", "Design12"</pre>

ImportIDX

Imports and IDX model into a new Icepak project.

UI Access	Icepak > Import IDX

Parameters	Name	Type	Description
	<NAME>	string	Settings
	<Board>	bool	Full path of .idx file
	<Library>	string	NA
	<Control>	string	NA
	<Filters>	list	HeightExclude2D
	<CreateFilteredAsNonModel>	bool	True or False
	<NAME>	string	definitionOverridesMap
	<NAME>	string	instanceOverridesMap
	<HighSurfThickness>	string	High surface layer thickness value and unit
	<LowSurfThickness>	string	Low surface layer thickness value and unit
	<InternalLayerThickness>	string	Internal surface layer thickness value and unit
	<NumInternalLayer>	int	Number of internal layers value
	<HighSurfaceCopper>	int	High surface percent coverage value
	<LowSurfaceCopper>	int	Low surface percent coverage value
	<InternalLayerCopper>	int	Internal layer percent coverage value
	<TraceMaterial>	string	Trace material name
	<SubstrateMaterial>	int	Substrate material name
	<CreateBoard>	bool	True or False
	<ModelBoardAsRect>	bool	True or False
	<ModelDeviceAsRect>	bool	True or False
	<Cutoff>	bool	True or False
	<ReplaceDevices>	bool	True or False
Return Value	None		

Python Syntax	ImportIDX (<NAME>, <Board>, <Library>, <Control>, <Filters>, <CreateFilteredAsNonModel>, <NAME>, <NAME>, <HighSurfThickness>, <LowSurfThickness>, <InternalLayerThickness>, <NumInternalLayer>,

	<code><HighSurfaceCopper>, <LowSurfaceCopper>, <InternalLayerCopper>, <TraceMaterial>, <SubstrateMaterial>, <CreateBoard>, <ModelBoardAsRect>, <ModelDeviceAsRect>, <Cutoff>, <ReplaceDevices>)</code>
Python Example	<pre> oDesign.ImportIDX(["NAME:Settings", "Board:=" , "C:\\Users\\Model files\\PCB-00278_A.idx", "Library:=" , "", "Control:=" , "", "Filters:=" , ["HeightExclude2D"], "CreateFilteredAsNonModel:=", False, ["NAME:definitionOverridesMap"], ["NAME:instanceOverridesMap"], "HighSurfThickness:=" , "0.07mm", "LowSurfThickness:=" , "0.07mm", "InternalLayerThickness:=", "0.07mm", "NumInternalLayer:=" , 2, "HighSurfaceCopper:=" , 30, "LowSurfaceCopper:=" , 30, "InternalLayerCopper:=" , 30, "TraceMaterial:=" , "Cu-Pure", "SubstrateMaterial:=" , "FR-4", "CreateBoard:=" , True, "ModelBoardAsRect:=" , False, "ModelDeviceAsRect:=" , False, "Cutoff:=" , False, "ReplaceDevices:=" , False]) </pre>

VB Syntax	ImportIDX (<NAME>, <Board>, <Library>, <Control>, <Filters>, <CreateFilteredAsNonModel>, <NAME>, <NAME>, <HighSurfThickness>, <LowSurfThickness>, <InternalLayerThickness>, <NumInternalLayer>, <HighSurfaceCopper>, <LowSurfaceCopper>, <InternalLayerCopper>, <TraceMaterial>, <SubstrateMaterial>, <CreateBoard>, <ModelBoardAsRect>, <ModelDeviceAsRect>, <Cutoff>, <ReplaceDevices>)
VB Example	<pre>oDesign.ImportIDX Array("NAME:Settings", "Board:=", _"C:\Users\Model files" _ "\PCB-00278_A.idx", "Library:=", "", "Control:=", "", "Filters:=", Array(_"HeightExclude2D"), "CreateFilteredAsNonModel:=", false, Array("NAME:definitionOverridesMap"), Array("NAME:instanceOverridesMap"), "HighSurfThickness:=", _0.07mm", "LowSurfThickness:=", "0.07mm", "InternalLayerThickness:=", "0.07mm", "NumInternalLayer:=", _2, "HighSur- faceCopper:=", 30, "LowSurfaceCopper:=", 30, "InternalLayerCopper:=", _30, "TraceMaterial:=", "Cu-Pure", "SubstrateMaterial:=", "FR-4", "CreateBoard:=", _true, "ModelBoardAsRect:=", false, "ModelDeviceAsRect:=", false, "Cutoff:=", _false, "ReplaceDevices:=", false)</pre>

ImportIPC

Imports an IPC2581 file into a new project.

UI Access	File > Import > IPC2581.		
Parameters	Name	Type	Description
	<ipcFileName>	String	Full path of IPC2581 file.
	<outputPathName>	String	Full path of EDB file to create during import.
	<controlFileName>	String	Optional. Full path of XML control file. Pass empty string if none.

Return Value	None.

Python Syntax	<code>ImportIPC(<ipcFileName>, <outputPathName>, <controlFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportIPC('C:/Files/test.cvg', 'C:/Files/test.aedb', 'C:/Files/test.xml', 'C:/Files/test.txt')</pre>

VB Syntax	<code>ImportIPC <ipcFileName>, <outputPathName>, <controlFileName></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportIPC "C:/Files/test.cvg", "C:/Files/test.aedb", _ "C:/Files/test.xml"</pre>

ImportODB

Imports an ODB++ file into a new project.

UI Access	File > Import > ODB++.		
Parameters	Name	Type	Description
	<odbFileName>	String	Full path of ODB++ file.
	<outputPathName>	String	Full path of EDB file to create during import.
	<controlFileName>	String	Optional. Full path of XML control file. Pass empty string if none.
Return Value	None.		

Python Syntax	<code>ImportODB(<odbFileName>, <outputPathName>, <controlFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportODB('C:/Files/test.odb', 'C:/Files/test.aedb', 'C:/Files/test.xml')</pre>

VB Syntax	<code>ImportODB <odbFileName>, <outputPathName>, <controlFileName></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportODB "C:/Files/test.odb", "C:/Files/test.aedb", _ "C:/Files/test.xml"</pre>

ImportXFL

Imports an XFL file into a new project.

UI Access	File > Import > XFL.		
Parameters	Name	Type	Description
	<code><xflFileName></code>	String	Full path of XFL file.
	<code><outputPathName></code>	String	Full path of EDB file to create during import.
	<code><controlFileName></code>	String	Optional. Full path of XML control file. Pass empty string if none.

Return Value	None.

Python Syntax	<code>ImportXFL(<xflFileName>, <outputPathName>, <controlFileName>)</code>
Python Example	<pre>oDesktop.RestoreWindow() Set oTool = oDesktop.GetTool('ImportExport') oTool.ImportXFL('C:/Files/test.xfl', 'C:/Files/test.aedb', 'C:/Files/test.xml')</pre>

VB Syntax	<code>ImportXFL <xflFileName>, <outputPathName>, <controlFileName></code>
VB Example	<pre>Set oTool = oDesktop.GetTool("ImportExport") oTool.ImportXFL "C:/Files/test.xfl", "C:/Files/test.aedb", _ "C:/Files/test.xml"</pre>

This page intentionally
left blank.

5 - Running Instances Manager Script Commands

The Running Instances Manager is a scripting object that lets you identify and connect to all running instances of Electronics Desktop. oDesktop objects that are returned provide full scripting functionality. Running Instances Manager commands should be executed by the oDesktop object. For example:

```
Set oRunningInstances = oDesktop.GetRunningInstancesMgr()
```

[GetAllRunningInstances](#)

[GetRunningInstanceByProcessID](#)

[GetRunningInstanceByProject](#)

GetAllRunningInstances

Returns a list of running instances of Ansys Electronics Desktop.

UI Access	N/A
Parameters	None.
Return Value	Array containing list of Ansys Electronics Desktop instances.

Python Syntax	GetAllRunningInstances()
Python Example	<code>obj = oRunningInstances.GetAllRunningInstances()</code>

VB Syntax	GetAllRunningInstances
VB Example	<code>set obj = oRunningInstances.GetAllRunningInstances()</code>

GetRunningInstanceByProcessID

Returns the instance of Ansys Electronics Desktop that is running a specified process.

UI Access	N/A		
Parameters	Name	Type	Description
	<processID>	Integer	Process ID
Return Value	String of the returned instance.		

Python Syntax	GetRunningInstanceByProcessID(<processID>)
Python Example	<code>obj = oRunningInstances.GetRunningInstanceByProcessID(12345)</code>

VB Syntax	GetRunningInstanceByProcessID <processID>
VB Example	<code>set obj = oRunningInstances.GetRunningInstanceByProcessID(12345)</code>

GetRunningInstancesMgr

Returns the object of the Running Instances Manager.

UI Access	N/A		
Parameters	Name	Type	Description
	None		

Return Value	Object Running instances manager object

Python Syntax	<code>GetRunningInstancesMgr()</code>
Python Example	<code>oRunningInstances = oDesktop.GetRunningInstancesMgr()</code>

VB Syntax	<code>GetRunningInstancesMgr()</code>
VB Example	<code>Set oRunningInstances = oDesktop.GetRunningInstancesMgr()</code>

This page intentionally
left blank.

6 - Project Object Script Commands

Project commands should be executed by the oProject object.

One example of accessing this object is:

```
Set oProject = oDesktop.GetActiveProject()
```

Dataset Script Commands:

[AddDataset](#)

[DeleteDataset](#)

[EditDataset](#)

[ExportDataset](#)

[HasDataset](#)

[ImportDataset](#)

Other Project Object Script Commands:

[AddMaterial \(Maxwell\)](#)

[AnalyzeAll](#)

[ChangeProperty](#)

[ClearMessages](#)

[CloneMaterial](#)

[Close](#)

[CopyDesign](#)

[CutDesign](#)

[DeleteDesign](#)

[DeleteToolObject](#)

[EditMaterial \(Maxwell\)](#)

[ExportMaterial](#)

[GetActiveDesign](#)

[GetArrayVariables](#)

[GetChildNames \[Project\]](#)

[GetChildObject \[Project\]](#)

[GetChildTypes \[Project\]](#)

[GetConfigurableData](#)

[GetDefinitionManager](#)

[GetDependentFiles](#)

[GetDesign](#)

[GetDesign](#)

[GetDesigns](#)

[GetEDBHandle](#)

[GetLegacyName](#)

[GetName \[Project\]](#)

[GetObjPath \[Project\]](#)

[GetPath](#)

[GetPropEvaluatedValue](#)

[GetPropNames \[Project\]](#)

[GetPropSIValue](#)

[GetPropValue \[Project\]](#)

[GetProperties](#)

[GetPropertyValue](#)

[GetTopDesignList](#)

[GetVariableValue](#)

[GetVariables](#)

[InsertDesign](#)

[InsertDesignWithWorkflow](#)

[InsertToolObject](#)

[Paste \[Project Object\]](#)

[Redo \[Project Level\]](#)

[RemoveAllUnusedDefinitions](#)

[RemoveMaterial](#)

[RemoveUnusedDefinitions](#)

[Rename](#)

[RunToolkit](#)

[Save](#)

[SaveAs](#)

[SaveAsStandAloneProject](#)

[SaveProjectArchive](#)

[SetActiveDefinitionEditor](#)

[SetActiveDesign](#)

[SetPropValue \[Project\]](#)

[SetPropertyValue](#)

[SetVariableValue](#)

[SimulateAll](#)

[Undo \[Project\]](#)

[UpdateDefinitions](#)

AddDataset

Adds a dataset. This can be executed by the oProject, or oDesign variables.

UI Access	Project > Datasets > Add.		
Parameters	Name	Type	Description
	< <i>DatasetDataArray</i> >	Array	Array("NAME:<DatasetName>", Array("NAME:Coordinates", <CoordinateArray>, <CoordinateArray>, ...)
	< <i>DatasetName</i> >	String	Name of the dataset.
	< <i>CoordinateArray</i> >	Array	Array("NAME:Coordinate", "X:=", <double>, "Y:=",<double>)
Return Value	None.		

Python Syntax	AddDataset <DatasetDataArray>
Python Example	<pre>oProject.AddDataset(["NAME:\$ds1", ["NAME:Coordinates", ["NAME:Coordinate", "X:=", 2, "Y:=", 4], ["NAME:Coordinate", "X:=", 6, "Y:=", 8]]]) oDesign.AddDataset(</pre>

```
[
  "NAME:$ds1",
  [
    "NAME:Coordinates",
    [
      "NAME:Coordinate",
      "X:=", 2,
      "Y:=", 4
    ],
    [
      "NAME:Coordinate",
      "X:=", 6,
      "Y:=", 8
    ]
  ]
]
```

VB Syntax

AddDataset <DatasetDataArray>

VB Example	<pre>oProject.AddDatasetArray("NAME:ds1", Array("NAME:Coordinates", Array("NAME:Coordinate", "X:=", 1, "Y:=", 2, Array("NAME:Coordinate", "X:=", 3, "Y:=", 4), Array("NAME:Coordinate", "X:=", 5, "Y:=", 7), Array("NAME:Coordinate", "X:=", 6, "Y:=", 20))) oDesign.AddDatasetArray("NAME:ds1", Array("NAME:Coordinates", Array("NAME:Coordinate", "X:=", 1, "Y:=", 2, Array("NAME:Coordinate", "X:=", 3, "Y:=", 4), Array("NAME:Coordinate", "X:=", 5, "Y:=", 7), Array("NAME:Coordinate", "X:=", 6, "Y:=", 20)))</pre>

AnalyzeAll [project]

Runs the project-level script command from the script, which simulates all solution setups and Optimetrics setups for all design instances in the project. The UI waits until simulation is finished before continuing with the script.

UI Access	Project > Analyze All.
Parameters	None.
Return Value	None.

Python Syntax	AnalyzeAll()
Python Example	<code>oProject.AnalyzeAll()</code>

VB Syntax	AnalyzeAll
VB Example	<code>oProject.AnalyzeAll</code>

ChangeProperty

Changes the properties of an object in the history tree.

UI Access	Right-click an object in the History Tree and select Properties .		
Parameters	Name	Type	Description
	<code><propertyArgs></code>	Array	Structured array. The properties vary depending on the object. Due to the number of potential configurations, it is recommended that you generate this script using the UI's Automation tab.
Return Value	None.		

Python Syntax	ChangeProperty(<code><propertyArgs></code>)
Python Example	Example Changing the Position of a Box: <code>oEditor.ChangeProperty(</code>

```
[
  "NAME:AllTabs",
  [
    "NAME:Geometry3DCmdTab",
    [
      "NAME:PropServers" ,
      "Box1:CreateBox:1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Position",
        "X:="          , "0.35in",
        "Y:="          , "0.55in",
        "Z:="          , "0in"
      ]
    ]
  ]
])
```

Example Changing a Box's Material and Wireframe Display:

```
oEditor.ChangeProperty(
[
  "NAME:AllTabs",
  [
    "NAME:Geometry3DAttributeTab",
    [
      "NAME:PropServers" ,
      "Box1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Material",
```

```
        "Value:="          , "\"vacuum\""
    ],
    [
        "NAME:Display Wireframe",
        "Value:="          , True
    ]
]
]
])
```

Example Changing the Model Depth:

```
oEditor.ChangeProperty(
[
    "NAME:AllTabs",
    [
        "NAME:Maxwell2D",
        [
            "NAME:PropServers",
            "Design Settings"
        ],
        [
            "NAME:ChangedProps",
            [
                "NAME:Model settings/Depth",
                "Value:="          , "2.4mm"
            ]
        ]
    ]
])
```

Example Changing 2D Transient Use of Band Mapping Angle:

```

oEditor.ChangeProperty(
[
  "NAME:AllTabs",
  [
    "NAME:MeshSetupTab",
    [
      "NAME:PropServers",
      "MeshSetup:CylindricalGap1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Use Band Mapping Angle",
        "Value:="          , True
      ]
    ]
  ]
])

oEditor.ChangeProperty(
[
  "NAME:AllTabs",
  [
    "NAME:MeshSetupTab",
    [
      "NAME:PropServers",
      "MeshSetup:CylindricalGap1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Band Mapping Angle",
        "Value:="          , "2deg"
      ]
    ]
  ]
]

```

	<pre>]) Example Edit VirtualPort Excitation by changing component properties: oEditor.ChangeProperty(["NAME:AllTabs", ["NAME:Component Data", ["NAME:PropServers", "BoundarySetup:LC1_1_Port1"], ["NAME:ChangedProps", ["NAME:Voltage", "Value:=" , "10mV"]]]]) ...</pre>

VB Syntax	ChangeProperty <propertyArgs>
VB Example	Example Changing the Position of a Box: <pre>oEditor.ChangeProperty Array("NAME:AllTabs", Array("NAME:Geometry3DCmdTab", Array("NAME:PropServers" ,</pre>

```

        "Box1:CreateBox:1"
    ),
    Array("NAME:ChangedProps",
        Array("NAME:Position",
            "X:="      , "0.35in",
            "Y:="      , "0.55in",
            "Z:="      , "0in"
        )
    )
)
)
)

```

Example Changing a Box's Material and Wireframe Display:

```

oEditor.ChangeProperty(
Array("NAME:AllTabs",
    Array("NAME:Geometry3DAttributeTab",
        Array("NAME:PropServers",
            "Box1"
        ),
        Array("NAME:ChangedProps",
            Array("NAME:Material",
                "Value:=", "\"vacuum\""
            ),
            Array("NAME:Display Wireframe",
                "Value:=", True
            )
        )
    )
)
)

```

Example Changing the Model Depth:

```

oEditor.ChangeProperty
Array("NAME:AllTabs",

```

```
        Array("NAME:Maxwell2D",
            Array("NAME:PropServers"
                "Design Settings"
            ),
            Array("NAME:ChangedProps",
                Array("NAME:Model settings/Depth",
                    "Value:=", "2.4mm"
                )
            )
        )
    )
)
```

Example Changing 2D Transient Use of Band Mapping Angle:

```
oEditor.ChangeProperty
Array("NAME:AllTabs",
    Array("NAME:MeshSetupTab",
        Array("NAME:PropServers"
            "MeshSetup:CylindricalGap1"
        ),
        Array("NAME:ChangedProps",
            Array("NAME:Use Band Mapping Angle",
                "Value:=", True
            )
        )
    )
)

oEditor.ChangeProperty
Array("NAME:AllTabs",
    Array("NAME:MeshSetupTab",
        Array("NAME:PropServers"
            "MeshSetup:CylindricalGap1"
```



```
),  
Array("NAME:ChangedProps",  
      Array("NAME:Band Mapping Angle",  
            "Value:=", "2deg"  
            )  
      )  
    )  
  )  
)
```

ClearMessages

Clears information in the **Messages** window.

Note:

Additional options are available when using the Desktop-level [ClearMessages](#) command.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	ClearMessages()
Python Example	<code>oProject.ClearMessages()</code>

VB Syntax	ClearMessages
VB Example	<code>oProject.ClearMessages</code>

Close

Closes the active project.

Warning:

Unsaved changes will be lost.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	Close()
Python Example	<code>oProject.Close()</code>

VB Syntax	Close
VB Example	<code>oProject.Close</code>

CopyDesign

Copies a specified design.

UI Access	Edit > Copy.

Parameters	Name	Type	Description
	<DesignName>	String	Name of the design to copy from.
Return Value	None.		

Python Syntax	CopyDesign (<DesignName>)
Python Example	<code>oProject.CopyDesign ("HFSSDesign1")</code>

VB Syntax	CopyDesign <DesignName>
VB Example	<code>oProject.CopyDesign "HFSSDesign1"</code>

CutDesign

Cuts a design from the active project. The design is stored in memory and can be pasted.

Warning:

This is a legacy command that is no longer supported and should not be used as it may have unintended effects on solved designs.

UI Access	Edit > Cut.		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design.
Return Value	None.		

Python Syntax	CutDesign (<DesignName>)
Python Example	<code>oProject.CutDesign("SimplorerDesign1")</code>

VB Syntax	CutDesign <DesignName>
VB Example	<code>oProject.CutDesign "SimplorerDesign1"</code>

DeleteDataset

Deletes a specified dataset. This can be executed by the oProject, or oDesign variables.

UI Access	Project > Datasets > Remove.		
Parameters	Name	Type	Description
	<DatasetName>	String	Name of the dataset found in the project.
Return Value	None.		

Python Syntax	DeleteDataset (<DatasetName>)
Python Example	<code>oProject.DeleteDataset('\$ds1')</code>
	<code>oDesign.DeleteDataset('\$ds1')</code>

VB Syntax	DeleteDataset <DatasetName>
VB Example	<pre>oProject.DeleteDataset "\$ds1" oDesign.DeleteDataset "\$ds1"</pre>

DeleteDesign

Deletes a specified design in the project.

UI Access	Edit > Delete , or Delete in the ribbon.		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design.
Return Value	None.		

Python Syntax	DeleteDesign (<DesignName>)
Python Example	<pre>oProject.DeleteDesign ("MaxwellDesign2")</pre>

VB Syntax	DeleteDesign <DesignName>
VB Example	<pre>oProject.DeleteDesign "MaxwellDesign2"</pre>

DeleteToolObject

Note:
This command is for internal Ansys use only.

UI Access	N/A		
Parameters	Name	Type	Description
	<ObjectName>	String	Name of the tool object.
Return Value	None.		

Python Syntax	DeleteToolObject(<ObjectName>)
Python Example	<code>oProject.DeleteToolObject ("object1")</code>

VB Syntax	DeleteToolObject <ObjectName>
VB Example	<code>oProject.DeleteToolObject "object1"</code>

EditDataset

Modifies a dataset. This can be executed by the oProject, or oDesign variables.

UI Access	Project > Datasets > Edit.

Parameters	Name	Type	Description
	<OriginalName>	String	Name of the original dataset.
	<DatasetDataArray>	Array	Data for the modified dataset.
Return Value	None.		

Python Syntax	EditDataset (<OriginalName> <DatasetDataArray>)
Python Example	<pre>oProject.EditDataset ("ds1" ["NAME:ds2", ["NAME:Coordinates", ["NAME:Coordinate", "X:=", 1, "Y:=", 2], ["NAME:Coordinate", "X:=", 3, "Y:=", 4]]]) oDesign.EditDataset ("ds1"</pre>

	<pre>["NAME:ds2", ["NAME:Coordinates", ["NAME:Coordinate", "X:=", 1, "Y:=", 2], ["NAME:Coordinate", "X:=", 3, "Y:=", 4]]]</pre>

VB Syntax	EditDataset <OriginalName> <DatasetDataArray>
VB Example	<pre>oProject.EditDataset "ds1" Array("NAME:ds2",_ Array("NAME:Coordinates",Array("NAME:Coordinate", "X:=", 1, "Y:=", 2), Array("NAME:Coordinate", "X:=", 3, "Y:=", 4)))</pre>


```
oDesign.EditDataset "ds1" Array("NAME:ds2",_
Array("NAME:Coordinates",Array("NAME:Coordinate",
"X:=", 1, "Y:=", 2), Array("NAME:Coordinate",
"X:=", 3, "Y:=", 4)))
```

ExportDataset

Exports a dataset to a named file. This can be executed by the oProject, or oDesign variables.

UI Access	Project > Datasets > Export.		
Parameters	Name	Type	Description
	<datasetFilePath>	String	The full path to the file.
Return Value	None.		

Python Syntax	ExportDataset (<datasetFilePath>)
Python Example	oProject.ExportDataset('e:/tmp/dsdata.txt')
	oDesign.ExportDataset('e:/tmp/dsdata.txt')

VB Syntax	ExportDataset <datasetFilePath>
VB Example	oProject.ExportDataset "e:/tmp/dsdata.txt"
	oDesign.ExportDataset "e:/tmp/dsdata.txt"

GetActiveDesign

Returns the design in the active project

Note: `GetActiveDesign` will return normally if there are no active objects.

UI Access	N/A
Parameters	None.
Return Value	Object of the active design.
Python Syntax	<code>GetActiveDesign()</code>
Python Example	<code>oDesign = oProject.GetActiveDesign()</code>

VB Syntax	<code>GetActiveDesign</code>
VB Example	<code>Set oDesign = oProject.GetActiveDesign</code>

GetArrayVariables

Returns a list of array variables. To get a list of indexed project variables, execute with `oProject`. To get a list of indexed local variables, use `oDesign`.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing names of variables.

Python Syntax	GetArrayVariables()
Python Example	<pre>oProject.GetArrayVariables() oDesign.GetArrayVariables()</pre>

VB Syntax	GetArrayVariables
VB Example	<pre>oProject.GetArrayVariables oDesign.GetArrayVariables</pre>

GetChildNames [Project]

Returns the names of the project's child objects.

UI Access	N/A		
Parameters	Name	Type	Description
	<type>	String	(Optional) Default is "design"; Any value returned by GetChildTypes() can be used.
Return Value	Array of names of children for the queried object.		

Python Syntax	GetChildNames (<type>)
Python Example	<pre>arrDesignNames = oProject.GetChildNames() arrVarbleNames = oProject.GetChildNames("Variable")</pre>

VB Syntax	GetChildNames (<type>)
VB Example	<pre>arrDesignNames = oProject.GetChildNames() arrVarbleNames = oProject.GetChildNames ("Variable")</pre>

GetChildObject [Project]

Returns the project's child objects.

Note:

GetChildObject will return normally if there are no active objects.

UI Access	N/A		
Parameters	Name	Type	Description
	<path>	String	The path may include multiple generations (for example, "designObject/moduleObj/SetupObject"). See Object Path .
Return Value	Object of a found child.		

Python Syntax	GetChildObject(<path>)
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDesign = oProject.GetChildObject ("TeeModel")</pre>

	<pre>oVariable = oProject.GetChildObject("VariableName") oReport = oProject.GetChildObject("TeeModel/Results/S Parameter Plot 1")</pre>

VB Syntax	<code>GetChildObject(<path>)</code>
VB Example	<pre>Set oProject = oDesktop.GetActiveproject() Set oDesign = oProject.GetChildObject("TeeModel") Set oVariable = oProject.GetChildObject("VariableName") Set oReport = oProject.GetChildObject("TeeModel/Results/S Parameter Plot 1")</pre>

GetChildTypes [Project]

Returns the types of the project's child objects.

UI Access	N/A
Parameters	None.
Return Value	Array of string represents types of the child objects.

Python Syntax	<code>GetChildTypes()</code>
Python Example	<code>oProject.GetChildTypes()</code>

VB Syntax	<code>GetChildTypes</code>

VB Example	<code>oProject.GetChildTypes</code>

GetConfigurableData (Project)

Note:

This command is for internal Ansys use only.

Python Syntax	<code>GetConfigurableData()</code>
Python Example	<code>oProject.GetConfigurableData()</code>

GetDefinitionManager

Gets the `DefinitionManager` object.

UI Access	N/A
Parameters	None.
Return Value	<code>DefinitionManager</code> object.

Python Syntax	<code>GetDefinitionManager()</code>
Python Example	<code>oDefinitionManager = oProject.GetDefinitionManager()</code>

VB Syntax	GetDefinitionManager
VB Example	<code>Set oDefinitionManager = oProject.GetDefinitionManager</code>

Note: For more information on commands for the DefinitionManager, see [Definition Manager Script Commands](#).

GetDependentFiles

Provides a list of the external files referenced in the project, including characteristic (for example, MDX) and coupled project files.

UI Access	N/A
Parameters	None.
Return Value	List of referenced files.

Python Syntax	GetDependentFiles()
Python Example	<code>files = oProject.GetDependentFiles()</code>

VB Syntax	GetDependentFiles
VB Example	<code>files = oProject.GetDependentFiles</code>

GetDesign

Returns the interface to a specific design in a given project.

UI Access	N/A		
Parameters	Name	Type	Description
	< <i>designName</i> >	String	Name of the design.
Return Value	Object of the specified design.		

Python Syntax	GetDesign (< <i>designName</i> >)		
Python Example	<code>oProject.GetDesign ("HFSSDesign1")</code>		

VB Syntax	GetDesign < <i>designName</i> >		
VB Example	<code>oProject.GetDesign "HFSSDesign1"</code>		

GetDesigns

Obtains all designs in the current project.

UI Access	N/A		
Parameters	None.		
Return Value	List of objects for all designs in the project.		

Python Syntax	GetDesigns()
Python Example	<code>oProject.GetDesigns()</code>

VB Syntax	GetDesigns
VB Example	<code>oProject.GetDesigns</code>

GetEDBHandle

Returns the EDB handle for the project.

Important:

This script is for internal Ansys use only.

UI Access	N/A
Parameters	None.
Return Value	String indicating the EDB handle for the project.

Python Syntax	GetEDBHandle()
Python Example	<code>oProject.GetEDBHandle()</code>

VB Syntax	GetEDBHandle
VB Example	<code>oProject.GetEDBHandle</code>

GetLegacyName

Obtains the legacy name of a project.

Note:

This command is for internal Ansys use only.

UI Access	N/A
Parameters	None.
Return Value	String containing the legacy project name.

Python Syntax	GetLegacyName()
Python Example	<code>oProject.GetLegacyName()</code>

VB Syntax	GetLegacyName
VB Example	<code>oProject.GetLegacyName</code>

GetName [Project]

Obtains the project name

UI Access	N/A
Parameters	None.
Return Value	String containing the project name, not including the path or extension.

Python Syntax	GetName()
Python Example	<code>oProject.GetName()</code>

VB Syntax	GetName
VB Example	<code>oProject.GetName</code>

GetObjPath [Project]

Obtains the project name from the full path.

UI Access	N/A
Parameters	None.
Return Value	String containing only the project name.

Python Syntax	GetObjPath()
Python Example	<code>oProject.GetObjPath()</code>

VB Syntax	GetObjPath
VB Example	<code>oProject.GetObjPath</code>

GetPath

Returns the location of the project on disk.

UI Access	N/A
Parameters	None.
Return Value	String containing the path to the project, not including the project name.

Python Syntax	GetPath()
Python Example	<code>oProject.GetPath()</code>

VB Syntax	GetPath
VB Example	<code>oProject.GetPath</code>

GetPropEvaluatedValue

Returns the Evaluated-Value for Value-Property and Variable. Returns the Property-value as text string for other property types

Note:

This command is not supported by the EMIT and Circuit design types.

UI Access	N/A		
Parameters	Name	Type	Description
	<PropName>	String	Name of the property.
Return Value	String value of the evaluated value.		

Python Syntax	GetPropEvaluatedValue (<PropName>)		
Python Example	<pre>oVar = oDesign.GetChildObject(" Variables/var") oVar.GetPropEvaluatedValue()</pre>		

GetPropNames [Project]

Obtains the property name of the object. At the project level, GetPropNames always returns empty because the project is not associated with any property.

UI Access	N/A		
Parameters	Name	Type	Description
	<includeReadOnly>	Boolean	(Optional) • True – Include read only props.

	<table><tr><td></td><td></td><td> False – Do not include read only props. </td></tr></table>			False – Do not include read only props.
		False – Do not include read only props.		
Return Value	Empty array.			

Python Syntax	GetPropNames ()
Python Example	<code>oProject.GetPropNames ()</code>

VB Syntax	GetPropNames
VB Example	<code>oProject.GetPropNames</code>

GetPropSIValue

Returns the SI-Value for Value-Property and Variable. Return NAN for other property type if its value is not able to convert to be a double-floating point value.

Note:

This command is not supported by the EMIT and Circuit design types.

UI Access	N/A		
Parameters	Name	Type	Description
	<PropName>	String	Name of the property.
Return Value	Property value as a double floating value, or NAN if the property value cannot be converted to double floating		

	point.

Python Syntax	<code>GetPropSIValue (<PropName>)</code>
Python Example	<pre> oCreateBox = oDesign.GetChildObject("3D Modeler/Box1/CreateBox:1") oCreateBox.GetPropValue("xSize") return "length / 2" oCreateBox.GetPropEvaluatedValue("xSize") return '0.4mm' oCreateBox.GetPropSIValue("xSize") return 0.0004 </pre>

GetPropValue [Project]

Returns the property value for the active project object, or specified property values.

UI Access	N/A		
Parameters	Name	Type	Description
	<propPath>	String	A child object's property path. See: Property Function Summary .
Return Value	String of property value.		

Python Syntax	<code>GetPropValue(<propPath>)</code>
Python Example	<code>oProject.GetPropValue("TeeModel/offset")</code>

VB Syntax	GetPropValue <propPath>
VB Example	oProject.GetPropValue "TeeModel/offset"

GetProperties

Gets a list of all the properties belonging to a specific <PropServer> and <PropTab>. This can be executed by the oProject, oDesign, or oEditor variables.

UI Access	N/A		
Parameters	Name	Type	Description
	<PropTab>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables") - ProjectVariableTab ("Project variables") - ConstantsTab ("Constants") - BaseElementTab ("Symbol" or "Footprint") - ComponentTab ("General") - Component("Component") - CustomTab ("Intrinsic Variables") - Quantities ("Quantities") - Signals ("Signals")
	<PropServer>	String	An object identifier, generally returned from another script method, such as CompInst@R;2;3

Return Value	Array of strings containing the names of the appropriate properties.

Python Syntax	<code>GetProperties(<PropTab>, <PropServer>)</code>
Python Example	<code>oEditor.GetProperties('PassedParameterTab', 'k')</code>

VB Syntax	<code>GetProperties <PropTab>, <PropServer></code>
VB Example	<code>oEditor.GetProperties "PassedParameterTab", "k"</code>

GetPropertyValue

Returns the value of a single property belonging to a specific *<PropServer>* and *<PropTab>*. This function is available with the Project, Design or Editor objects, including definition editors.

Tip: Use the script recording feature and edit a property, and then view the resulting script to see the format for that property.

UI Access	N/A		
Parameters	Name	Type	Description
	<i><PropTab></i>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables") - ProjectVariableTab ("Project variables") - ConstantsTab ("Constants") - BaseElementTab ("Symbol" or "Footprint")

			• ComponentTab ("General") • Component("Component") • CustomTab ("Intrinsic Variables") • Quantities ("Quantities") • Signals ("Signals")
	<i><PropServer></i>	String	An object identifier, generally returned from another script method, such as <code>CompInst@R;2;3</code>
	<i><PropName></i>	String	Name of the property.
Return Value	String value of the property.		

Python Syntax	<code>GetPropertyValue (<PropTab>, <PropServer>, <PropName>)</code>
Python Example	<pre>selectionArray = oEditor.GetSelections() for k in selectionArray: val = oEditor.GetPropertyValue("PassedParameterTab", k, "R") ... </pre>

VB Syntax	<code>GetPropertyValue (<PropTab>, <PropServer>, <PropName>)</code>
VB Example	<pre>selectionArray = oEditor.GetSelections for k in selectionArray: val = oEditor.GetPropertyValue("PassedParameterTab", k, "R") </pre>

	...

GetTopDesignList

Returns a list of top-level design names.

UI Access	N/A
Parameters	None.
Return Value	List of strings containing name of top-level designs.

Python Syntax	GetTopDesignList()
Python Example	<code>oProject.GetTopDesignList()</code>

VB Syntax	GetTopDesignList
VB Example	<code>oProject.GetTopDesignList</code>

GetValue

Gets the value of a single specified variable. To get the value of project variables, execute this command using `oProject`. To get the value of local variables, use `oDesign`.

UI Access	N/A

Parameters	Name	Type	Description
	<VarName>	String	Name of the variable to access.
Return Value	String represents the value of the variable.		

Python Syntax	GetVariableValue(<VarName>)
Python Example	<code>oProject.GetVariableValue("var_name")</code>

VB Syntax	GetVariableValue <VarName>
VB Example	<code>oProject.GetVariableValue "var_name"</code>

GetVariables

Returns a list of all defined variables. To get a list of project variables, execute this command using `oProject`. To get a list of local variables, use `oDesign`.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing the variables.

Python Syntax	GetVariables ()

Python Example	<pre>oProject.GetVariables() oDesign.GetVariables()</pre>

VB Syntax	GetVariables
VB Example	<pre>oProject.GetVariables oDesign.GetVariables</pre>

ImportDataset

Imports a dataset from a named file. This can be executed by the oProject, or oDesign variables. The name of the dataset is file-name+index number (e.g., dsdata1) unless the filename ends with a trailing number. When there is a trailing number at the end, we will remove the number and use first unused index. Alternatively, the name of the dataset can be explicitly defined by providing a string as an optional second argument.

UI Access	Project > Datasets > Import.		
Parameters	Name	Type	Description
	<datasetFilePath>	String	The full path to the file containing the dataset values. *.tab files recommended (see note below).
	<optionalDatasetName>	String	Optional. User-defined dataset name.
Return Value	None.		

Python Syntax	ImportDataset (<datasetFilePath>,<optionalDatasetName>)
Python Example	<pre>oProject.ImportDataset('e:\tmp\dsdata.tab') oDesign.ImportDataset('e:\tmp\dsdata.tab')</pre>

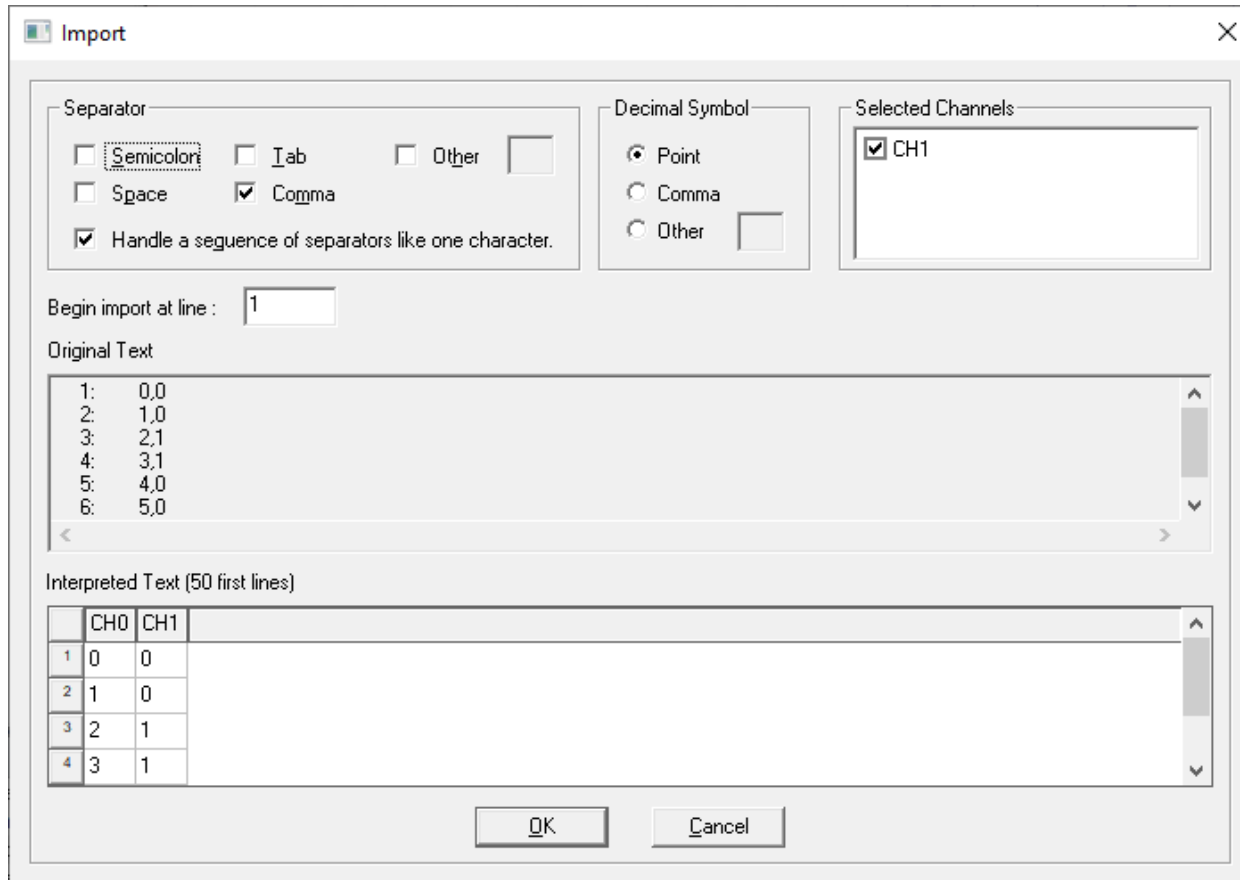
	<pre>oProject.ImportDataset('e:\tmp\dsdata.tab', 'MyDatasetName') oDesign.ImportDataset('e:\tmp\dsdata.tab', 'MyDatasetName')</pre>

VB Syntax	ImportDataset <datasetFilePath>, <optionalDatasetName>
VB Example	<pre>oProject.ImportDataset "e:\tmp\dsdata.tab" oDesign.ImportDataset "e:\tmp\dsdata.tab" oProject.ImportDataset "e:\tmp\dsdata.tab", "MyDatasetName" oDesign.ImportDataset "e:\tmp\dsdata.tab", "MyDatasetName"</pre>

Note About File Types:

Tab-delimited or space-delimited files with the extension *.tab are the recommended file type. When using ImportDataset at the Design level, *.tab is the only file type supported.

At the Project level, other file types are supported (for example, *.csv). However, after calling the command, you must configure the file import format manually through the Electronics Desktop GUI by selecting **Project > Datasets** and clicking **Import**.



InsertDesign

Inserts a new design in the project. For Maxwell scripts, the last argument will always be empty.

UI Access	Project > [Insert Maxwell 3D Design / Insert Maxwell 2D Design / Insert RMXprt Design].		
Parameters	Name	Type	Description
	<DesignType>	String	Design type. "Maxwell 2D", "Maxwell 3D", or "RMXprt".
	<DesignName>	String	Design name.
	<SolutionType>	String	Dependent on design type. For Maxwell 3D, can be: "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "DCConduction", "ElectroDCConduction", or "ElectricTransient". For Maxwell 2D, can be: "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "ACConduction", or "DCConduction". For RMXprt, can be: "Three-Phase Induction Motor", "Single-Phase Induction Motor", "Three-Phase Synchronous Machine", "Brushless Permanent-Magnet DC Motor", "Adjust-Speed Synchronous Machine", "Permanent-Magnet DC Motor", "Switched Reluctance Motor", "Line-Start Permanent-Magnet Synchronous Motor", "Universal Motor", "DC Machine", "Claw-Pole Alternator", "Three-Phase Non-Salient Synchronous Machine", or "Generic Rotating Machine".
	""	None	Empty argument.
Return Value	None.		

Python Syntax	InsertDesign (<DesignType>, <DesignName>, <SolutionType>,"")
Python Example	<pre>oProject.InsertDesign('Maxwell 3D', 'MyDesign', 'ElectricTransient', '')</pre>

VB Syntax	<code>InsertDesign <DesignType>, <DesignName>, <SolutionType>,""</code>
VB Example	<code>oProject.InsertDesign "Maxwell 3D", "MyDesign", "ElectricTransient", ""</code>

InsertDesignWithWorkflow

Inserts a design with a named workflow and returns an IDispatch string.

UI Access	N/A		
Parameters	Name	Type	Description
	<code><type></code>	String	Type of design.
	<code><workflowName></code>	String	Name of the workflow.
	<code><specName></code>	String	Name of the spec.
	<code><fileName></code>	String	Name of the file.
	<code><libLoc></code>	String	Type of library, such as SysLib.
	<code><stationaryPath></code>	String	Path.
Return Value	IDispatch string, such as 'IDispatch(IAltraSimScript)'		

Python Syntax	<code>InsertDesignWithWorkflow(<type>, <workflowName>, <specName>, <fileName>, <libLoc>, <stationaryPath>)</code>
Python Example	<code>oProject.InsertDesignWithWorkflow ("Circuit Design", "Serial Design", "PCIe3 Stressed", "LongChannel", "SysLib", "C:\\Program Files\\AnsysEM\\v242\\Win64\\syslib\\MS - RT_duroid 6010 (Er=10.2) 0.010 inch, 0.5 oz copper.asty")</code>

VB Syntax	<code>InsertDesignWithWorkflow <type>, <workflowName>, <specName>, <fileName>, <libLoc>, <stationaryPath></code>
VB Example	<pre>oProject.InsertDesignWithWorkflow "Circuit Design", "Serial Design", _ "PCIe3 Stressed", "LongChannel", "SysLib", _ "C:\Program Files\AnsysEM\v242\Win64\syslib\MS - RT_duroid 6010" & _ " (Er=10.2) 0.010 inch, 0.5 oz copper.asty"</pre>

InsertToolObject

Note:

This command is for internal Ansys use only.

Python Syntax	<code>InsertToolObject()</code>
Python Example	<code>oProject.InsertToolObject()</code>

Paste (Project Object)

Pastes a design in the active project.

UI Access	Edit > Paste.
Parameters	None.
Return Value	None

Python Syntax	Paste()
Python Example	<code>oProject.Paste()</code>

VB Syntax	Paste
VB Example	<code>oProject.Paste</code>

Redo [Project Level]

Reapplies the last project-level command.

UI Access	Edit > Redo.
Parameters	None.
Return Value	None.

Python Syntax	Redo()
Python Example	<code>oProject.Redo()</code>

VB Syntax	Redo
VB Example	<code>oProject.Redo</code>

RemoveAllUnusedDefinitions

Removes all unused project definitions.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	<code>RemoveAllUnusedDefinitions()</code>
Python Example	<code>oProject.RemoveAllUnusedDefinitions()</code>

VB Syntax	<code>RemoveAllUnusedDefinitions</code>
VB Example	<code>oProject.RemoveAllUnusedDefinitions</code>

RemoveMaterial

Removes a material from a library.

UI Access	Remove Material(s) command in the material editor		
Parameters	Name	Type	Description
	<code><MaterialName></code>	String	Name of the material to be removed.
	<code><IsProjectMaterial></code>	Boolean	If True, Maxwell assumes the material is a project material. The last two parameters will be ignored. If False, the material is not a project material.
	<code><LibraryName></code>	String	Name of the user or personal library where the material resides.
	<code><LibraryLocation></code>	String	Location of library. Valid options:"UserLib" or "PersonalLib".

Return Value	None.

Python Syntax	<code>RemoveMaterial (<MaterialName>, <IsProjectMaterial>, <LibraryName>, <LibraryLocation>)</code>
Python Example	<pre>oDefinitionManager.RemoveMaterial (["Material1", false, "mo0907", "UserLib"])</pre>

VB Syntax	<code>RemoveMaterial <MaterialName>, <IsProjectMaterial>, <LibraryName>, <LibraryLocation></code>
VB Example	<pre>oDefinitionManager.RemoveMaterial "Material1", false, "mo0907", "UserLib"</pre>

RemoveUnusedDefinitions

Removes any unused project definitions.

UI Access	Tools > Project Tools > Remove Unused Definitions.		
Parameters	Name	Type	Description
	<Definitions>	Array	Definitions to be removed, such as materials and surface materials.
Return Value	None.		

Python Syntax	RemoveUnusedDefinitions(<Definitions>)
Python Example	<pre>oProject.RemoveUnusedDefinitions([["NAME:Materials", "Al-Extruded"], ["NAME:SurfaceMaterials", "Steel-oxidised-surface"]])</pre>

VB Syntax	RemoveUnusedDefinitions <Definitions>
VB Example	<pre>oProject.RemoveUnusedDefinitions Array(Array("NAME:Materials", "Al-Extruded"), Array ("NAME:SurfaceMaterials", "Steel-oxidised-surface"))</pre>

Rename

Renames the project and saves it. Similar to [SaveAs\(\)](#).

UI Access	Edit > Rename.		
Parameters	Name	Type	Description
	<NewName>	String	Desired name of the project. The path is optional.
	<OverWriteOk>	Boolean	• True - overwrite the file on disk if it exists. • False - prevent overwrite.
Return Value	None.		

Python Syntax	Rename(<NewName>,<OverWriteOK>)
Python Example	<pre>oProject.Rename("c:\projects\MyProject.aedt", True)</pre>

VB Syntax	Rename <NewName>,<OverWriteOK>
VB Example	<pre>oProject.Rename "c:\projects\MyProject.aedt", true</pre>

RunToolkit

Runs a Python toolkit script, applying it to the active design. The toolkit script itself may have prerequisites, such as sweeps defined, or specific model characteristics, such as port definitions.

Important:

Full scripting for cable modeling is not supported and no arguments are allowed in the script. The **RunToolkit** command will not automatically create a cable bundle, but will simply open the **Cable Modeling** dialog box. You will have to manually input your parameters.

UI Access	[Maxwell] > Toolkit > [Script Name].		
Parameters	Name	Type	Description
	<LibraryType>	String	Name of the library in which the script is located.
	<ToolkitName>	String	Name of the Python script.
	<ToolkitArg>	Array	Structured array containing arguments required by the toolkit script.
Return Value	User-defined solution(s) and/or output(s).		

Python Syntax	RunToolkit(<LibraryType>, <ToolkitName>, <ToolkitArg>)		
Python Example	<pre>oDesign.RunToolkit("SysLib", "Cable Modeling/Automotive Cable Bundle", []) oDesign.RunToolkit("SysLib", "Cable Modeling/Oil-Gas Cable Bundle", [])</pre>		

VB Syntax	RunToolkit <LibraryType>, <ToolkitName>, <ToolkitArg>		
VB Example	<pre>oDesign.RunToolkit "SysLib", "Cable Modeling/Automotive Cable Bundle", Array() oDesign.RunToolkit "SysLib", "Cable Modeling/Oil-Gas Cable Bundle", Array()</pre>		

Save

Saves the active project.

UI Access	File > Save.
Parameters	None.
Return Value	None.

Python Syntax	Save()
Python Example	<code>oProject.Save()</code>

VB Syntax	Save
VB Example	<code>oProject.Save</code>

SaveAs

Saves the project under a new name. Requires a full path.

Note:

This script takes two parameters for non-schematic/layout designs and four parameters for schematic/layout designs.

UI Access	File > Save As.

Parameters	Name	Type	Description
	<NewName>	String	The desired name of the project, with directory and extension.
	<OverWriteOK>	Boolean	True to overwrite the file of the same name, if it exists. False to prevent overwrite.
	<DefaultAction>	String	For Schematic/Layout projects only. Otherwise omit. See note below. Valid actions: ef_overwrite , ef_copy_no_overwrite, ef_make_path_absolute, or empty string.
	<OverwriteActions>	Array	For Schematic/Layout projects only. Otherwise omit. See note below. Structured array: Array("Name: <Action>", <FileName>, <FileName>, ...) Valid actions: ef_overwrite , ef_copy_no_overwrite, ef_make_path_absolute, or empty string.
Return Value	None.		

Python Syntax	For non-Schematic/Layout project: SaveAs (<NewName> <OverWriteOK>) For Schematic/Layout project: SaveAs (<NewName> <OverWriteOK> <DefaultAction> <OverrideActions>)
Python Example	<pre>oProject.SaveAs('D:/projects/project1.aedt', True) ---- oProject.SaveAs('D:/Projects/Project1.aedt', True, 'ef_overwrite', ['NAME:OverrideActions', ['NAME:ef_copy_no_overwrite', ['NAME:Files', '\$PROJECTDIR/circuit_models.inc']], ['NAME:ef_make_path_absolute', ['NAME:Files', '\$PROJECTDIR/SL_6s.sp']]])</pre>

VB Syntax	For non-Schematic/Layout project: SaveAs <NewName> <OverWriteOK> For Schematic/Layout project: SaveAs <NewName> <OverWriteOK> <DefaultAction> <OverrideActions>
VB Examples	<pre>oProject.SaveAs "D:/projects/project1.aedt", true ---- oProject.SaveAs "F:\Designer Projects\TA33097\HighSpeedChannel.aedt", true, "ef_overwrite", Array ("NAME:OverrideActions", Array("NAME:ef_copy_no_overwrite", Array("NAME:Files", "\$PROJECTDIR/circuit_mod- els.inc")), Array("NAME:ef_make_path_absolute", Array("NAME:Files", "\$PROJECTDIR\SL_6s.sp")))</pre>

Important:

The DefaultAction and OverrideActions strings correspond to the following actions:

- **ef_overwrite** – Copy file to new project directory and overwrite.
- **ef_copy_no_overwrite** – Copy file to new project directory and don't overwrite.
- **ef_make_path_absolute** – Change reference to point to file in old project directory.
- **Empty String** – Do nothing.

The DefaultAction is applied to all files that are NOT explicitly listed in the OverrideActions array. Those in the OverrideActions array are separate arrays for actions that are different from the default action; those actions are applied to the files listed in the same array:

- If OverrideActions are not specified, DefaultAction is applied to ALL files in project directory.

SaveAsStandAloneProject

Saves the project as a standalone copy.

Note:

This script is not supported when the application is being controlled by Ansys Workbench.

UI Access	N/A		
Parameters	Name	Type	Description
	<projectName>	String	The desired name of the project, with directory and extension.
Return Value	None.		

Python Syntax	SaveAsStandAloneProject(<projectName>)
Python Example	oProject.SaveAsStandAloneProject('D:/projects/project1.aedt')

VB Syntax	SaveAsStandAloneProject <projectName>
VB Examples	oProject.SaveAsStandAloneProject "D:/projects/project1.aedt"

SaveProjectArchive

Saves the active project as an archive to the specified file path.

UI Access	File > Archive.		
Parameters	Name	Type	Description
	<archiveFilePath>	String	Path to archived file.
	<IncludeExternalFiles>	Boolean	True to include external files; False to exclude.
	<IncludeResultsFiles>	Boolean	True to include simulation files associated with the project; False to exclude.
	<AdditionalFiles>	Array	Additional specified files to include.
	<ArchiveNotes>	String	String describe the archive.
Return Value	None.		

Python Syntax	SaveProjectArchive(<archivefilepath>, <IncludeExternalFiles>, <IncludeResultsFiles>, <AdditionalFiles>, <ArchiveNotes>)
Python Example	<pre>oProject.SaveProjectArchive("C:\\Users\\Documents\\Ansoft\\Project27.aedt", True, False, [], "")</pre>

VB Syntax	SaveProjectArchive <archivefilepath>, <IncludeExternalFiles>, <IncludeResultsFiles>, <AdditionalFiles>, <ArchiveNotes>
VB Example	<pre>oProject.SaveProjectArchive "C:\Documents\OptimTee.aedt", true, false, Array(), "My notes"</pre>

SetActiveDefinitionEditor

Obtains a specified definition editor.

UI Access	N/A

Parameters	Name	Type	Description
	<EditorName>	String	Name of the definition editor to set active, one of "SymbolEditor", "FootprintEditor".
	<DefinitionName>	String	The combination name for the symbol or footprint, <libname>:<def-name>
Return Value	Object for the definition to be edited.		

Python Syntax	SetActiveDefinitionEditor(<EditorName>, <DefinitionName>)
Python Example	<pre>oProject.SetActiveDefinitionEditor("SymbolEditor", "Simplorer Elements\Basic Elements\Circuit\Passive Elements:R")</pre>

VB Syntax	SetActiveDefinitionEditor <EditorName>, <DefinitionName>
VB Example	<pre>oProject.SetActiveDefinitionEditor "SymbolEditor",_ "Simplorer Elements\Basic Elements\Circuit\Passive Elements:R"</pre>

SetActiveDesign

Sets a design to be the active design.

UI Access	N/A		
Parameters	Name	Type	Description
	<DesignName>	String	Name of the design to set as the active design.

Return Value	None.

Python Syntax	SetActiveDesign (<DesignName>)
Python Example	<code>oDesign = oProject.SetActiveDesign("SimplorerDesign2")</code>

VB Syntax	SetActiveDesign <DesignName>
VB Example	<code>Set oDesign = oProject.SetActiveDesign "SimplorerDesign2"</code>

SetPropValue [Project]

Sets a property value for an active project's child object.

UI Access	Edit Properties on ProjectTree objects.		
Parameters	Name	Type	Description
	<propPath>	String	A child object's property path. See: Property Function .
	<newValue>	String	New property value.
Return Value	Boolean: • True – property found. • False – property not found.		

Python Syntax	SetPropValue(<propPath>, <newValue>)
Python	<code>oProject.SetPropValue("TeeModel/offset", "2mm")</code>

Example	<code>oProject.SetPropValue("TeeModel/Results/S Parameter Plot 1/Display Type", "Data Table")</code>

VB Syntax	<code>SetPropValue <propPath>, <newValue></code>
VB Example	<code>oProject.SetPropValue "TeeModel/offset", "2mm"</code> <code>oProject.SetPropValue "TeeModel/Results/S Parameter Plot 1/Display Type", "Data Table"</code>

SetPropertyValue

Sets the value of a single property belonging to a specific PropServer and PropTab. This function is available with the Project, Design or Editor objects, including definition editors. This is not supported for properties of the following types: ButtonProp, PointProp, V3DPointProp, and VPointProp. Only the ChangeProperty command can be used to modify these properties.

Use the script recording feature and edit a property, and then view the resulting script entry or use GetPropertyValue for the desired property to see the expected format.

UI Access	N/A		
Parameters	Name	Type	Description
	<code><propTab></code>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables") - ProjectVariableTab ("Project variables") - ConstantsTab ("Constants") - BaseElementTab ("Symbol" or "Footprint")

			• ComponentTab ("General") • Component("Component") • CustomTab ("Intrinsic Variables") • Quantities ("Quantities") • Signals ("Signals")
	<code><propServer></code>	String	An object identifier, generally returned from another script method, such as <code>CompInst@R;2;3</code>
	<code><propName></code>	String	Name of the property.
	<code><propValue></code>	String	The value for the property
Return Value	None.		

Python Syntax	<code>SetPropertyValue(<propTab>, <propServer>, <propName>, <propValue>)</code>
Python Example	<code>oEditor.SetPropertyValue("PassedParameterTab", "k", "R", "2200")</code>

VB Syntax	<code>SetPropertyValue <propTab>, <propServer>, <propName>, <propValue></code>
VB Example	<code>oEditor.SetPropertyValue "PassedParameterTab", "k", "R", "2200"</code>

SetVariableValue

Sets the value of a variable. To set the value of a project variable, execute this command using `oProject`. To set the value of a local variable, use `oDesign`.

UI Access	N/A

Parameters	Name	Type	Description
	<VarName>	String	Variable name.
	<VarValue>	Value	New value for the variable.
Return Value	None.		

Python Syntax	SetVariableValue (<VarName>, <VarValue>)
Python Example	<code>oProject.SetVariableValue('\$Var1', '3mm')</code>

VB Syntax	SetVariableValue <VarName>, <VarValue>
VB Example	<code>oProject.SetVariableValue "\$Var1", "3mm"</code>

SimulateAll

Simulates all solution setups and Optimetrics setups for all design instances in the project. Script processing only continues when all analyses are finished.

UI Access	N/A
Parameters	None.
Return Value	None.

Python Syntax	SimulateAll()

Python Example	<code>oProject.SimulateAll()</code>

VB Syntax	<code>SimulateAll</code>
VB Example	<code>oProject.SimulateAll</code>

Undo [Project]

Cancels the last project-level command.

UI Access	Edit > Undo.
Parameters	None.
Return Value	None.

Python Syntax	<code>Undo()</code>
Python Example	<code>oProject.Undo()</code>

VB Syntax	<code>Undo</code>
VB Example	<code>oProject.Undo</code>

UpdateDefinitions

Updates all definitions. The **Messages** window reports when definitions are updated, or warns when definitions cannot be found.

UI Access	Tools > Project Tools > Update Definitions. Click Select All , then Update .
Parameters	None.
Return Value	None.

Python Syntax	UpdateDefinitions()
Python Example	<code>oProject.UpdateDefinitions()</code>

VB Syntax	UpdateDefinitions
VB Example	<code>oProject.UpdateDefinitions</code>

7 - Definition Manager Script Commands

The definition manager controls the use of materials and scripts in a project. It also provides access to the managers for symbols, footprints, padstacks, and components in a project.

```
Set oProject = oDesktop.SetActiveProject("Project1")
```

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

The topics for this section include:

[AddDefinitionFromBlock](#)

[AddMaterial](#)

[CloneMaterial](#)

[DoesMaterialExist](#)

[EditMaterial](#)

[ExportMaterial](#)

[RemoveMaterial](#)

[RemoveUnusedDefinitions](#)

Related Topics:

[Component Manager Script Commands](#)

[Material Manager Script Commands](#)

[Model Manager Script Commands](#)

[Network Data Explorer Manager Script Commands](#)

[Script and Library Scripts](#)

[Symbol Manager Script Commands](#)

AddMaterial (Maxwell)

Adds a local material.

UI Access	Add Material in the material editor.		
Parameters	Name	Type	Description
	<MaterialParams>	Array	["NAME: <material to be added>", <MatProperty>, <MatProperty>, ...]
	<MatProperty>	Array	Options will depend on what is specified. Some options include "AnisoProperty", "CustomProperty", and "nonlinear".

			For simple material: <pre>"<PropertyName>:=", <value></pre> For anisotropic material: <pre>["NAME:<PropertyName>", "property_type:=", "AnisoProperty", "unit:=", <Unit>", "component1:=", <value>, "component2:=", <value>, "component3:=", <value>))]</pre> Material properties for other property types are listed further in the topic.
	< <i>PropertyName</i> >	String	Should be one of the following (depending on the material, design, and solution types): <pre>"permittivity", "permeability", "RecoilPermeability", "conductivity", "dielectric_loss_tangent", "magnetic_loss_tangent", "magnetic_coercivity", "core_loss_type", "mass_density", "stacking_type", "magnetostriction", "inverse_magnetostriction", "youngs_modulus", "poissons_ratio"</pre>
	< <i>Unit</i> >	String	Possible values for magnetic properties: delta_H: "Oe", saturation_mag: "Gauss", "uGauss", "Tesla", "uTesla"
Return Value	None		

Python Syntax	AddMaterial (["NAME:<MaterialName>", <MatProperty>, <MatProperty>, ...])
Python Example	<pre> # Example: Magnetostrictive stress and deformation with full magnetostriction matrix [3x3] oDefinitionManager.AddMaterial(["NAME:Material2", "CoordinateSystemType:=", "Cartesian", "BulkOrSurfaceType:=" , 1, ["NAME:PhysicsTypes", "set:=" , ["Electromagnetic"]], ["NAME:magnetostriction", "property_type:=" , "CustomProperty", "PropertyModelType:=" , 4, # integer value of 1 (Linear), 2 (Nonlinear), 3 (Anisotropic), or 4 (Matrix option) "Dir1ModelType:=" , 0, "Dir2ModelType:=" , 0, "IsoCoeff:=" , "0", "IsoHMax:=" , "0A_per_meter", "IsoHMin:=" , "0A_per_meter", "Dir1Coeff:=" , "0", "Dir1HMax:=" , "0A_per_meter", "Dir1HMin:=" , "0A_per_meter", "Dir1VecX:=" , "1", "Dir1VecY:=" , "0", "Dir1VecZ:=" , "0", "Dir2Coeff:=" , "0", </pre>


```

"Dir2HMax:="          , "0A_per_meter",
"Dir2HMin:="          , "0A_per_meter",
"CoeffMatrixSize:="   , 3, # <int> [0, 3].
# If "PropertyModelType" is "4", size cannot be 0 (should not be empty)).
# CoeffMatrixSize = 3" means 3x3 matrix size, and matrix elements
#(C11, C12, .. C33) are specified in CoeffMatrix block.
"IsCoeffMatrixSymmetric:=", True,

[
  "NAME:CoeffMatrix", # if size is 3, has matrix elements C11, C12, ... C33 ,
  #no element if size is 0
  "C11:="              , "5",
  "C12:="              , "0",
  "C13:="              , "0",
  "C21:="              , "0",
  "C22:="              , "4",
  "C23:="              , "0",
  "C31:="              , "0",
  "C32:="              , "0",
  "C33:="              , "3"
],
"CoeffMatrixHMax:="    , "10A_per_meter", #Value of Hmax
[
  "NAME:Magnetostriction",
  [
    "NAME:Isotropic",
    [
      "NAME:OneVariation",
      "DatasetMapsName:=" , "Lambda-H curves versus Stress"
    ]
  ]
],
[
  "NAME:Direction1",

```

	<pre> ["NAME:OneVariation", "DatasetMapsName:=" , "Lambda-H curves versus Stress"]], ["NAME:Direction2", ["NAME:OneVariation", "DatasetMapsName:=" , "Lambda-H curves versus Stress"]], "SelectedDatasetMapsName:=" , "Lambda-H curves versus Stress"]])</pre>

VB Syntax	AddMaterial Array("NAME:<MaterialName>",<MatProperty>, <MatProperty>, ...)
VB Example	<pre>oDefinitionManager.AddMaterial Array("permittivity:=", "2.2", "0.002") oDefinitionManager.AddMaterial Array("NAME:Material2",_ "dielectric_loss_tangent:=", "44", Array("NAME:saturation_mag",_ "property_type:=", "AnisoProperty",_ "unit:=", "Gauss",_ </pre>

	<pre>"component1:=", "11", _ "component2:=", "22", _ "component3:=", "33"), _ "delta_H:=", "440e")</pre>

Additional Properties for Nonlinear Property Type

```
<MatProperty> (nonlinear material)  
  
  Array("NAME:permeability",  
    "property_type:=", "nonlinear",  
    "BTypeForSingleCurve:=", <string>,  
    "Hunit:=", <string>,"Bunit:=", <string>,  
    "IsTemperatureDependent:=", <bool>,  
    Array("NAME:BHCoordinates",  
      Array("NAME:Coordinate", "X:=", <value>, "Y:=", <value>, ...
```

BTypeForSingleCurve

Type: <string>

Should be one of the following: "Intrinsic", or "Normal" (defaults to Normal if nothing is specified). An error message displays in the Message Window if the string does not match either "Normal" or "Intrinsic"; advising the user of this and informing the user that this property has been set to "Normal".

Hunit

Type: <string>

Possible values: "Oe", "A_per_meter", "kA_per_meter"

Bunit

Type: <string>

Possible values: "tesla", "kTesla", "uGauss"

IsTemperatureDependent

Type: <bool>

possible values: True or False

Additional Properties for Vector Property Type

<MatProperty> (vector material)

Array("NAME:<PropertyName>",

"property_type:=", "VectorProperty", "Magnitude:=", <value>, "DirComp1:=", <value>, "DirComp2:=", <value>, "DirComp3:=", <value>))

<MatProperty> (choice material)

Array("NAME:<PropertyName>",

"property_type:=", "ChoiceProperty", "Choice:=", <string>")

Additional Python Examples

Python Example (Young's Modulus and Poisson's Ratio):

```
oDefinitionManager.AddMaterial("copper",
```

```
[
```

```
...
```

```
[
```

```
"NAME:youngs_modulus",
```

```

    "property_type:=" , "AnisoProperty",
    "unit:=" , "",
    "component1:=" , "1",
    "component2:=" , "1.5",
    "component3:=" , "3"
],
[
    "NAME:poissons_ratio",
    "property_type:=" , "AnisoProperty",
    "unit:=" , "",
    "component1:=" , "4",
    "component2:=" , "5",
    "component3:=" , "6"
],
...
])

```

Note: Young's Modulus and Poisson's Ratio properties are valid only for 2D and 3D Magnetostatic and Transient.

Python Example: Defining Anisotropic Hysteresis Model with Recoil-Line Relative Permeability

```

oDefinitionManager.AddMaterial(
[

```

```
"NAME:Alnico5_HalfLoop_HysteresisModel_Anisotropic",  
"CoordinateSystemType:=", "Cartesian",  
"BulkOrSurfaceType:=", 1,  
[  
"NAME:PhysicsTypes",  
"set:=", ["Electromagnetic","Thermal"]  
],  
[  
"NAME:AttachedData",  
[  
"NAME:MatAppearanceData",  
"property_data:=", "appearance_data",  
"Red:=", 218,  
"Green:=", 222,  
"Blue:=", 222  
]  
],  
"permittivity:=", "1",  
[  
"NAME:permeability",
```

```
"property_type:=" , "AnisoProperty",
"unit:=" , "",
[
  "NAME:component1",
  "property_type:=" , "nonlinear",
  "BTypeForSingleCurve:=" , "normal",
  "HUnit:=" , "A_per_meter",
  "BUnit:=" , "tesla",
  "IsTemperatureDependent:=", False,
  [
    "NAME:BHCoordinates",
    [
      "NAME:DimUnits",
      "",
      ""
    ],
  ],
  [
    "NAME:Point",
    -101859.1636,
    -1.44995289
  ],
],
```

```
    ...  
  ],  
  [  
    "NAME:Temperatures"  
  ],  
  "RecoilPermeability:=" , "1.5"  
],  
"component2:=" , "1",  
"component3:=" , "1"  
],  
"conductivity:=" , "2000000",  
"dielectric_loss_tangent:=", "0",  
"magnetic_loss_tangent:=", "0",  
[  
  "NAME:magnetic_coercivity",  
  "property_type:=" , "VectorProperty",  
  "Magnitude:=" , "-50910.9581943227A_per_meter",  
  "DirComp1:=" , "0",  
  "DirComp2:=" , "0",  
  "DirComp3:=" , "0"
```



```
],  
"thermal_conductivity:=", "50",  
"saturation_mag:=", "0gauss",  
"lande_g_factor:=", "2",  
"delta_H:=", "00e",  
"mass_density:=", "7300",  
"specific_heat:=", "400",  
"thermal_expansion_coefficient:=", "1.1e-05"  
])
```

Additional VB Examples

Example: (Square Litz Wire)

```
oDefinitionManager.AddMaterial Array("NAME:LitzMaterialSquare",  
...  
Array("NAME:stacking_type", "property_type:=", "ChoiceProperty", "Choice:=", "Litz Wire"),  
...  
Array("NAME:wire_type", "property_type:=",  
"ChoiceProperty", "Choice:=", "Square"), "strand_number:=", "16", "twisting_length_factor:=",  
"1.1", "wire_width:=", "2mm")
```

Example (Round Litz Wire):

```
oDefinitionManager.AddMaterial Array("NAME:LitzMaterialRound",_  
...  

```

```
Array("NAME:stacking_type", "property_type:=", "ChoiceProperty", "Choice:=", "Litz Wire"),  
...  
Array("NAME:wire_type", "property_type:=",  
"ChoiceProperty", "Choice:=", "Round"), "strand_number:=", "16", "twisting_length_factor:=" ,  
"1.1", "wire_diameter:=" "2mm")
```

Example (Rectangular Litz Wire):

```
oDefinitionManager.AddMaterial Array("NAME:LitzMaterialRectangular",_  
...  
Array("NAME:stacking_type", "property_type:=", "ChoiceProperty", "Choice:=", "Litz Wire"),  
...  
Array("NAME:wire_type", "property_type:=", _  
"ChoiceProperty", "Choice:=", "Rectangular"), "strand_number:=", "5", "twisting_length_  
factor:=" , "1.1",  
"wire_width:=", "5.5mm", "wire_thickness:=", "5.6",  
    Array("NAME:wire_thickness_direction", "property_type:=",  
    "ChoiceProperty", "Choice:=", "V(1)")  
    Array("NAME:wire_width_direction", "property_type:=",  
    "ChoiceProperty", "Choice:=", "V(2)")  
)
```

Note: Litz Wire properties are valid only for 2D/3D Transient and 2D/3D Eddy Current designs.

Example:

```
oDefinitionManager.AddMaterial Array("NAME:Material5",_
    "CoordinateSystemType:=", _"Cartesian",_
    Array("NAME:AttachedData"),_
    Array("NAME:magnetic_coercivity", "property_type:=", _
    "VectorProperty", "Magnitude:=", "-1A_per_meter",_
    "DirComp1:=", "1", "DirComp2:=", _"2", "DirComp3:=", _
    "3"))
```

Example:

```
oDefinitionManager.AddMaterial Array("NAME:Material4",_
    "CoordinateSystemType:=", _"Cartesian",_
    Array("NAME:AttachedData"),_
    Array("NAME:stacking_type", "property_type:=", _
    "ChoiceProperty", "Choice:=", "Lamination"),_
    "stacking_factor:=", "0",_
    Array("NAME:stacking_direction", "property_type:=", _
    "ChoiceProperty", "Choice:=", ""))
```

VB Example (Young's Modulus and Poisson's Ratio):

```
oDefinitionManager.AddMaterial "copper" ,
Array("NAME:copper",
...

```

```
Array("NAME:youngs_modulus",
"property_type:=", "AnisoProperty",
"unit:=", "",
"component1:=", "1",
"component2:=", "2",
"component3:=", "3"),

Array("NAME:poissons_ratio",
"property_type:=", "AnisoProperty",
"unit:=", "",
"component1:=", "4",
"component2:=", "5",
"component3:=", "6"),

...)
```

Note: Young's Modulus and Poisson's Ratio properties are valid only for 2D and 3D Magnetostatic and Transient.

CloneMaterial

Clones a local material.

UI Access	N/A		
Parameters	Name	Type	Description
	<matName>	String	Name of existing material.
	<newName>	String	Name for newly cloned material.
Return Value	Boolean: 1 - Material is cloned. 0 - Existing material not found or a conflict with the new material name.		

Python Syntax	CloneMaterial (<matName>, <newName>)
Python Example	oDefinitionManager.CloneMaterial("copper1", "copper3")

VB Syntax	CloneMaterial <matName>, <newName>
VB Example	oDefinitionManager.CloneMaterial "copper1", "copper3"

ComputeCoreLossCoefficients

Creates a current source.

UI Access	N/A		
Parameters	Name	Type	Description
	<CoreLossModelType>	string	One of: Electrical Steel or Power Ferrite
	<MassDensity>	double	Mass density of the core loss model type
	<AttachedData>	Array	Either <CoreLossData> or <CoreLossMultiCurveData>
Return Value	Array of 3 double values in SI unit: - For Electrical Steel, returns the values for [Kh, Kc, Ke] - For Power Ferrite, returns the values for [Cm, X, Y]		

Python Syntax	ComputeCoreLossCoefficients (<CoreLossModelType>, <MassDensity>,

	<AttachedData>)
Python Example	<pre>DefinitionManager.ComputeCoreLossCoefficients("Electrical Steel", 7000, ["NAME:CoefficientSetupData", "property_data:=" , "coreloss_data", "coefficient_setup:=" , "w_per_cubic_mete", "Frequency:=" , "60Hz", "Thickness:=" , "0.5mm", "Conductivity:=" , "0", ["NAME:Coordinates", ["NAME:DimUnits", "", ""], ["NAME:Points",0,0,3,6,4,18]]])</pre>

VB Example	Single Curve: <pre>oDefinitionManager.ComputeCoreLossCoefficients "Electrical Steel", 7670.5, Array("NAME:Coef- ficientSetupData", "property_data:=", _ "coreloss_data", "coefficient_setup:=", "w_per_cubic_meter", "Frequency:=", _ "60Hz", "Thickness:=", "0mm", "Conductivity:=", "0", Array("NAME:Coordinates", Array("NAME:DimUnits", _ "", "")), Array("NAME:Points", 0, 0, 3, 6, 4, 12)))</pre>

VB Example	Multi Curves: <pre>oDefinitionManager.ComputeCoreLossCoefficients "Electrical Steel", 7670.5, Array ("NAME:CoreLossMultiCurveData", "property_data:=", _ "coreloss_multi_curve_data", "coreloss_unit:=", "w_per_cubic_meter", Array("NAME:AllCurves", Array ("NAME:OneCurve", "Frequency:=", _ "50Hz", Array("NAME:Coordinates", Array("NAME:DimUnits", "", ""), Array("NAME:Points", 0, 0, _ 3, 6, 4, 12))), Array("NAME:OneCurve", "Frequency:=", "75Hz", Array("NAME:Coordinates", Array ("NAME:DimUnits", _ "", ""), Array("NAME:Points", 0, 0, 3, 9, 4, 16))))))</pre>

DoesMaterialExist

Checks for the presence of a material in the library by name

UI Access	None.		
Parameters	Name	Type	Description
	<MaterialName>	<String>	Name of the material to search for in the material database
Return Value	Boolean: • True – specified material exists. • False – specified material does not exist.		

Python Syntax	DoesMaterialExist(<MaterialName>)
Python Example	<code>oDefinitionManager.DoesMaterialExist("modified_epoxy")</code>

VB Syntax	DoesMaterialExist <MaterialName>
VB Example	oDefinitionManager.DoesMaterialExist "modified_epoxy"

EditMaterial (Maxwell)

Use: Modifies an existing material.

UI Access	View/Edit Materials command in the material editor		
Parameters	Name	Type	Description
	<OriginalName>	String	Name of the material before editing.
	<NewName>	String	New name for the material
	<MatProperties>	Array	Structured array containing material properties. Refer to AddMaterial (Maxwell) for information on material properties.
Return Value	None		

Python Syntax	EditMaterial (<OriginalName>, <NewName>,<MatProperties>)
Python Example	Example: Nonlinear Conductivity <pre>oDefinitionManager.EditMaterial("nonlinear_copper", [... ["NAME:conductivity",</pre>


```
"property_type:="          , "nonlinear",
[
  "NAME:JECordinates",
  [
    "NAME:DimUnits",
    "V_per_meter",
    "A_per_m2"
  ],
  [
    "NAME:Point",
    0,
    0
  ],
  [
    "NAME:Point",
    1,
    2
  ],
  [
    "NAME:Point",
    3,
    4
  ]
]
]
])
```

Example using a Thermal Modifier for Electrical Steel Core Loss properties

#applicable to Electrical Steel and Power Ferrite core loss models in 3D Eddy Current and 3D Transient designs)
oDefinitionManager.EditMaterial("copper",

```
[
  "NAME:copper",
  "CoordinateSystemType:=", "Cartesian",
  "BulkOrSurfaceType:=", 1,
  [
    "NAME:PhysicsTypes",
    "set:=", ["Electromagnetic","Thermal","Structural"]
  ],
  [
    "NAME:AttachedData",
    [
      "NAME:MatAppearanceData",
      "property_data:=", "appearance_data"
    ]
  ],
  [
    "NAME:ModifierData",
    [
      "NAME:ThermalModifierData",
      "modifier_data:=", "thermal_modifier_data",
      [
        "NAME:all_thermal_modifiers",
        [
          "NAME:one_thermal_modifier",
          "Property:=", "core_loss_kh",
          "Index:=", 0,
          "prop_modifier:=", "thermal_modifier",
          "use_free_form:=", True,
          "free_form_value:=", "if(Temp > 1000cel, 1, if(Temp -273.15cel, 1, 1))"
        ],
        [
          "NAME:one_thermal_modifier",
          "Property:=", "core_loss_kc",
```

```
emp > 1000cel, 1, if(Temp -273.15cel, 1, 1))"
],
[
  "NAME:one_thermal_modifier",
  "Property::=" , "core_loss_ke",
  "Index::=" , 0,
  "prop_modifier::=" , "thermal_modifier",
  "use_free_form::=" , True,
  "free_form_value::=" , "if(Temp > 1000cel, 1, if(Temp -273.15cel, 1, 1))"
],
[
  "NAME:one_thermal_modifier",
  "Property::=" , "core_loss_kdc",
  "Index::=" , 0,
  "prop_modifier::=" , "thermal_modifier",
  "use_free_form::=" , True,
  "free_form_value::=" , "if(Temp > 1000cel, 1, if(Temp -273.15cel, 1, 1))"
],
[
  "NAME:one_thermal_modifier",
  "Property::=" , "core_loss_equiv_cut_depth",
  "Index::=" , 0,
  "prop_modifier::=" , "thermal_modifier",
  "use_free_form::=" , True,
  "free_form_value::=" , "if(Temp > 1000cel, 1, if(Temp -273.15cel, 1, 1))"
]
]
]
]
```

VB Syntax	EditMaterial <OriginalName>, <NewName>,<MatProperties>
VB Example	Example: Set oProject = oDesktop.SetActiveProject("example_magnetostatic")

```
Set oDefinitionManager = oProject.GetDefinitionManager()

oDefinitionManager.EditMaterial "arm_steel", Array("NAME:arm_steel", "CoordinateSystemType:=",
"Cartesian", Array("NAME:AttachedData"), Array("NAME:ModifierData"), Array("NAME:permeability", "property_type:=",
"nonlinear", "BTypeForSingleCurve:=", "normal", "HUnit:=", "A_per_meter", "BUnit:=",
"tesla", "IsTemperatureDependent:=", "True", Array("NAME:BHCoordinates", Array
("NAME:Coordinate", "X:=",
0, "Y:=", 0), Array("NAME:Coordinate", "X:=", 4000, "Y:=", 1.413), Array("NAME:Coordinate", "X:=",
8010, "Y:=", 1.594), Array("NAME:Coordinate", "X:=", 16010, "Y:=", 1.751), Array
("NAME:Coordinate", "X:=",
24020, "Y:=", 1.839), Array("NAME:Coordinate", "X:=", 32030, "Y:=", 1.896), Array
("NAME:Coordinate", "X:=",
40030, "Y:=", 1.936), Array("NAME:Coordinate", "X:=", 48040, "Y:=", 1.967), Array
("NAME:Coordinate", "X:=",
64050, "Y:=", 2.008), Array("NAME:Coordinate", "X:=", 80070, "Y:=", 2.042), Array
("NAME:Coordinate", "X:=",
96080, "Y:=", 2.073), Array("NAME:Coordinate", "X:=", 112100, "Y:=", 2.101), Array
("NAME:Coordinate", "X:=",
128110, "Y:=", 2.127), Array("NAME:Coordinate", "X:=", 144120, "Y:=", 2.151), Array
("NAME:Coordinate", "X:=",
176150, "Y:=", 2.197), Array("NAME:Coordinate", "X:=", 208180, "Y:=", 2.24), Array
("NAME:Coordinate", "X:=",
272230, "Y:=", 2.325), Array("NAME:Coordinate", "X:=", 304260, "Y:=", 2.37), Array
("NAME:Coordinate", "X:=",
336290, "Y:=", 2.42), Array("NAME:Coordinate", "X:=", 396000, "Y:=", 2.5))), "con-
```

	<pre>ductivity:=", "2000000", Array("NAME:magnetic_coercivity", "property_type:=", "VectorProperty", "Mag- nitude:=", "0A_per_meter", "DirComp1:=", "1", "DirComp2:=", "0", "DirComp3:=", "0"), Array("NAME:- core_loss_type", "property_type:=", "ChoiceProperty", "Choice:=", "Hysteresis Model"), "core_loss_hci:=", "100000", "core_ loss_br:=", "88", "core_loss_hkc:=", "333", "core_loss_equiv_cut_depth:=", "0.002meter")</pre>
	VB Example (Young's Modulus and Poisson's Ratio): <pre>oDefinitionManager.EditMaterial "copper" , Array("NAME:copper", ... Array("NAME:youngs_modulus", "property_type:=", "AnisoProperty", "unit:=", "", "component1:=", "1", "component2:=", "2", "component3:=", "3"), Array("NAME:poissons_ratio", "property_type:=", "AnisoProperty", "unit:=", "", "component1:=", "4", "component2:=", "5", "component3:=", "6"), ...)</pre>

Additional Python Examples

Note: Young's Modulus and Poisson's Ratio properties are valid only for 2D & 3D Magnetostatic and Transient.

Python Example of Editing Young's Modulus and Poisson's Ratio:

```
oDefinitionManager.EditMaterial("copper",  
[  
...  
[  
    "NAME:youngs_modulus",  
    "property_type:" , "AnisoProperty",  
    "unit:" , "",  
    "component1:" , "1",  
    "component2:" , "1.5",  
    "component3:" , "3"  
],  
[  
    "NAME:poissons_ratio",  
    "property_type:" , "AnisoProperty",  
    "unit:" , "",  
    "component1:" , "4",
```

```

    "component2:=" , "5",
    "component3:=" , "6"
  ],
...
])

```

Python Example: Magnetostrictive stress and deformation with full magnetostriction matrix [3x3]

```

oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial("vacuum",
[
    "NAME:vacuum",
    "CoordinateSystemType:=", "Cartesian",
    "BulkOrSurfaceType:="      , 1,
    [
        "NAME:PhysicsTypes",
        "set:="                  , ["Electromagnetic"]
    ],
    [
        "NAME:AttachedData",
        [
            "NAME:MatAppearanceData",
            "property_data:="      , "appearance_data",
            "Red:="                 , 230,
            "Green:="               , 230,
            "Blue:="                , 230,
            "Transparency:="        , 0.949999988079071
        ]
    ],
    "permittivity:="      , "1",
    [
        "NAME:magnetostriction",

```

```

"property_type:="          , "CustomProperty",
"PropertyModelType:="      , 4, # Value of 1=Linear, 2=Nonlinear, 3=Anisotropic, 4=Matrix
"Dir1ModelType:="          , 0,
"Dir2ModelType:="          , 0,
"IsoCoeff:="               , "0",
"IsoHMax:="                , "0A_per_meter",
"IsoHMin:="                , "0A_per_meter",
"Dir1Coeff:="              , "0",
"Dir1HMax:="               , "0A_per_meter",
"Dir1HMin:="               , "0A_per_meter",
"Dir1VecX:="               , "1",
"Dir1VecY:="               , "0",
"Dir1VecZ:="               , "0",
"Dir2Coeff:="              , "0",
"Dir2HMax:="               , "0A_per_meter",
"Dir2HMin:="               , "0A_per_meter",
"CoeffMatrixSize:="        , 3, # <int> - [0, 3].
#If PropertyModelType is 4, (Matrix type), size cannot be 0
# CoeffMatrixSize = 3 defines a 3x3 matrix size,
#and matrix elements (C11, C12, .. C33) are specified in CoeffMatrix block
"IsCoeffMatrixSymmetric:=", False, # takes a Boolean value
[
  "NAME:CoeffMatrix",
  "C11:="                  , "1.05",
  "C12:="                  , "2",
  "C13:="                  , "3",
  "C21:="                  , "4",
  "C22:="                  , "5",
  "C23:="                  , "6",
  "C31:="                  , "7",
  "C32:="                  , "8",
  "C33:="                  , "9"

```



```

],
"CoeffMatrixHMax:" , "12A_per_meter", #Value of Hmax
[
  "NAME:Magnetostriction",
  [
    "NAME:Isotropic",
    [
      "NAME:OneVariation",
      "DatasetMapsName:" , "Lambda-H curves versus Stress"
    ]
  ],
  [
    "NAME:Direction1",
    [
      "NAME:OneVariation",
      "DatasetMapsName:" , "Lambda-H curves versus Stress"
    ]
  ],
  [
    "NAME:Direction2",
    [
      "NAME:OneVariation",
      "DatasetMapsName:" , "Lambda-H curves versus Stress"
    ]
  ],
  "SelectedDatasetMapsName:" , "Lambda-H curves versus Stress"
]
]
]
)

```

Note: Core Loss Y Parameter for Electrical Steel is valid only for 2D & 3D Eddy Current and Transient Designs.

Python Example: Defining Anisotropic Hysteresis Model with Recoil-Line Relative Permeability

```
oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial("Alnico5_HalfLoop_HysteresisModel_Anisotropic",
[
    "NAME:Alnico5_HalfLoop_HysteresisModel_Anisotropic",
    "CoordinateSystemType:=", "Cartesian",
    "BulkOrSurfaceType:=" , 1,
    [
        "NAME:PhysicsTypes",
        "set:=" , ["Electromagnetic","Thermal"]
    ],
    [
        "NAME:AttachedData",
        [
            "NAME:MatAppearanceData",
            "property_data:=" , "appearance_data",
            "Red:=" , 218,
            "Green:=" , 222,
            "Blue:=" , 222
        ]
    ]
]
```

```
],  
"permittivity:=" , "1",  
[  
  "NAME:permeability",  
  "property_type:=" , "AnisoProperty",  
  "unit:=" , "",  
  [  
    "NAME:component1",  
    "property_type:=" , "nonlinear",  
    "BTypeForSingleCurve:=" , "normal",  
    "HUnit:=" , "A_per_meter",  
    "BUnit:=" , "tesla",  
    "IsTemperatureDependent:=", False,  
    [  
      "NAME:BHCoordinates",  
      [  
        "NAME:DimUnits",  
        "",  
        ""  
      ],  
    ],  
  ]  
]
```

```
        "NAME:Point",
        -101859.1636,
        -1.44995289
    ],
    ...
],
[
    "NAME:Temperatures"
],
    "RecoilPermeability:=" , "1.5"
],
    "component2:=" , "1",
    "component3:=" , "1"
],
    "conductivity:=" , "2000000",
    "dielectric_loss_tangent:=" , "0",
    "magnetic_loss_tangent:=" , "0",
    [
        "NAME:magnetic_coercivity",
        "property_type:=" , "VectorProperty",
```

```

    "Magnitude:=" , "-50910.9581943227A_per_meter",
    "DirComp1:=" , "0",
    "DirComp2:=" , "0",
    "DirComp3:=" , "0"
],
    "thermal_conductivity:=" , "50",
    "saturation_mag:=" , "0gauss",
    "lande_g_factor:=" , "2",
    "delta_H:=" , "00e",
    "mass_density:=" , "7300",
    "specific_heat:=" , "400",
    "thermal_expansion_coefficient:=" , "1.1e-05"
])

```

Python Example: Core Loss Y Parameter for Electrical Steel (Simple Property)

```

oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial("steel_1008",
[
    "NAME:steel_1008",
    "CoordinateSystemType:=" , "Cartesian",
    "BulkOrSurfaceType:=" , 1,
    [

```

```
"NAME:PhysicsTypes",  
"set:=" , ["Electromagnetic","Thermal","Structural"]  
],  
...  
...  
[  
"NAME:core_loss_type",  
"property_type:=" , "ChoiceProperty",  
"Choice:=" , "Electrical Steel"  
],  
"core_loss_kh:=" , "105.359604403558",  
"core_loss_kc:=" , "0",  
"core_loss_ke:=" , "2.08295141230884",  
"core_loss_kdc:=" , "0",  
"mass_density:=" , "7872",  
[  
"NAME:stacking_type",  
"property_type:=" , "ChoiceProperty",  
"Choice:=" , "Litz Wire"  
],
```

```

"specific_heat:=" , "481",
"youngs_modulus:=" , "200000000000",
"poissons_ratio:=" , "0.25",
"thermal_expansion_coefficient:=" , "1.26e-05",
"core_loss_equiv_cut_depth:=" , "0.001meter",
[
"NAME:wire_type",
"property_type:=" , "ChoiceProperty",
"Choice:=" , "Round"
],
"strand_number:=" , "12",
"wire_diameter:=" , "3mm",
"core_loss_es_y:=" , "2.3"
])

```

Note: Core Loss Y Parameter for Electrical Steel is valid only for 2D & 3D Eddy Current and Transient Designs.

Python Example: Core Loss Y Parameter for Electrical Steel (Anisotropic Property)

```

oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial("steel_1008",
[

```

```
"NAME:steel_1008",
"CoordinateSystemType:=", "Cartesian",
"BulkOrSurfaceType:=" , 1,

[
"NAME:PhysicsTypes",
"set:=" , ["Electromagnetic","Thermal","Structural"]
],
...
...

[
"NAME:core_loss_type",
"property_type:=" , "ChoiceProperty",
"Choice:=" , "Electrical Steel"
],

"core_loss_kh:=" , "105.359604403558",
"core_loss_kc:=" , "0",
"core_loss_ke:=" , "2.08295141230884",
"core_loss_kdc:=" , "0",
"mass_density:=" , "7872",

[
"NAME:stacking_type",
"property_type:=" , "ChoiceProperty",
"Choice:=" , "Litz Wire"
],
```



```
"specific_heat:" , "481",
"youngs_modulus:" , "2000000000000",
"poissons_ratio:" , "0.25",
"thermal_expansion_coefficient:" , "1.26e-05",
"core_loss_equiv_cut_depth:" , "0.001meter",

[
"NAME:wire_type",
"property_type:" , "ChoiceProperty",
"Choice:" , "Round"
],

"strand_number:" , "12",

"wire_diameter:" , "3mm",

[
"NAME:core_loss_es_y",
"property_type:" , "AnisoProperty",
"unit:" , "",
"component1:" , "1",
"component2:" , "2",
"component3:" , "3" ]
])
```

Additional VB Examples

Example:Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")

```
Set oDesktop = oAnsoftApp.GetAppDesktop()

oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("pmsg-1_test")

Set oDefinitionManager = oProject.GetDefinitionManager()
```

```
oDefinitionManager.EditMaterial "M19-24G_3DSF0.950", Array("NAME:M19-24G_3DSF0.950", "CoordinateSystemType:=",  
"Cartesian", "BulkOrSurfaceType:=", 1, Array("NAME:PhysicsTypes", "set:=", Array(  
"Electromagnetic")), Array("NAME:permeability", "property_type:=", "nonlinear", "BType:=",  
"normal", "HUnit:=", "A_per_meter", "BUnit:=", "tesla", Array("NAME:BHCoordinates", Array  
("NAME:Coordinate", "X:=",  
0, "Y:=", 0), Array("NAME:Coordinate", "X:=", 22.28, "Y:=", 0.05), Array("NAME:Coordinate",  
"X:=",  
25.46, "Y:=", 0.1), Array("NAME:Coordinate", "X:=", 31.83, "Y:=", 0.15), Array("NAME:Coordinate",  
"X:=",  
47.74, "Y:=", 0.36), Array("NAME:Coordinate", "X:=", 63.66, "Y:=", 0.54), Array("NAME:Coordinate",  
"X:=",  
79.57, "Y:=", 0.65), Array("NAME:Coordinate", "X:=", 159.15, "Y:=", 0.99), Array("NAME:Coordinate",  
"X:=",  
318.3, "Y:=", 1.2), Array("NAME:Coordinate", "X:=", 477.46, "Y:=", 1.28), Array("NAME:Coordinate",  
"X:=",  
636.61, "Y:=", 1.33), Array("NAME:Coordinate", "X:=", 795.77, "Y:=", 1.36), Array("NAME:Coordinate",  
"X:=",  
1591.5, "Y:=", 1.44), Array("NAME:Coordinate", "X:=", 3183, "Y:=", 1.52), Array("NAME:Coordinate",  
"X:=",  
4774.6, "Y:=", 1.58), Array("NAME:Coordinate", "X:=", 6366.1, "Y:=", 1.63), Array("NAME:Coordinate",  
"X:=",  
7957.7, "Y:=", 1.67), Array("NAME:Coordinate", "X:=", 15915, "Y:=", 1.8), Array("NAME:Coordinate",  
"X:=",
```

```

31830, "Y:=", 1.9), Array("NAME:Coordinate", "X:=", 111407, "Y:=", 2), Array("NAME:Coordinate",
"X:=",
190984, "Y:=", 2.1), Array("NAME:Coordinate", "X:=", 350138, "Y:=", 2.3), Array("NAME:Coordin-
ate", "X:=",
509292, "Y:=", 2.5), Array("NAME:Coordinate", "X:=", 2100830, "Y:=", 4.5))), "conductivity:=",
"0", Array("NAME:core_loss_type", "property_type:=", "ChoiceProperty", "Choice:=",
"Electrical Steel"), "core_loss_kh:=", "0", "core_loss_kc:=", "4.32556", "core_loss_ke:=",
"0", "mass_density:=", "7650", Array("NAME:stacking_type", "property_type:=",
"ChoiceProperty", "Choice:=", "Lamination"), "stacking_factor:=", "0.95", Array("NAME:stacking_
direction", "property_type:=",
"ChoiceProperty", "Choice:=", "V(3)"), "core_loss_equiv_cut_depth:=",
"0.002meter")

```

Example:Litz Wire

Note: Litz Wire properties are valid only for 2D/3D Transient and 2D/3D Eddy Current designs.

```

oDefinitionManager.EditMaterial Array("NAME:LitzMaterialRound",
...
Array("NAME:wire_type", "property_type:=",
"ChoiceProperty", "Choice:=", "Round"), "strand_number:=", "16", "twisting_length_factor:=",
"1.1", "wire_diameter:=", "2mm")

```

Example: Thermal Modifier

Note: Example using a Thermal Modifier for Electrical Steel Core Loss properties (applicable to both Electrical Steel and Power Ferrite core loss models in 3D Eddy Current and 3D Transient designs)

```
Set oProject = oDesktop.SetActiveProject("3D_eddy_temperature_dependent_coreloss")
Set oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial "copper", Array("NAME:copper", "CoordinateSystemType:=",
"Cartesian", "BulkOrSurfaceType:=", 1, Array("NAME:PhysicsTypes", "set:=", Array(
"Electromagnetic", "Thermal", "Structural")), Array("NAME:AttachedData", Array("NAME:MatAppearanceData", "property_data:=",
"appearance_data")),
Array("NAME:ModifierData", Array("NAME:ThermalModifierData", "modifier_data:=",
"thermal_modifier_data", Array("NAME:all_thermal_modifiers", Array("NAME:one_thermal_modifier", "Property:=",
"core_loss_kh", "Index:=", 0, "prop_modifier:=", "thermal_modifier", "use_free_form:=",
true, "free_form_value:=", "if(Temp > 1000cel, 1, if(Temp < -273.15cel, 1, 1))"), Array
("NAME:one_thermal_modifier", "Property:=",
"core_loss_kc", "Index:=", 0, "prop_modifier:=", "thermal_modifier", "use_free_form:=",
true, "free_form_value:=", "if(Temp > 1000cel, 1, if(Temp < -273.15cel, 1, 1))"), Array
("NAME:one_thermal_modifier", "Property:=",
"core_loss_ke", "Index:=", 0, "prop_modifier:=", "thermal_modifier", "use_free_form:=",
```

```

true, "free_form_value:=", "if(Temp > 1000cel, 1, if(Temp < -273.15cel, 1, 1))"), Array
("NAME:one_thermal_modifier", "Property:=",
"core_loss_kdc", "Index:=", 0, "prop_modifier:=", "thermal_modifier", "use_free_form:=",
true, "free_form_value:=", "if(Temp > 1000cel, 1, if(Temp < -273.15cel, 1, 1))"), Array
("NAME:one_thermal_modifier", "Property:=",
"core_loss_equiv_cut_depth", "Index:=", 0, "prop_modifier:=",
"thermal_modifier", "use_free_form:=", true, "free_form_value:=",
"if(Temp > 1000cel, 1, if(Temp < -273.15cel, 1, 1))"))), "permeability:=",
"0.999991", "conductivity:=", "27840000", "thermal_conductivity:=", "400", Array("NAME:core_
loss_type", "property_type:=",
"ChoiceProperty", "Choice:=", "Electrical Steel"), "core_loss_kh:=", "0", "core_loss_kc:=",
"0", "core_loss_ke:=", "0", "core_loss_kdc:=", "0", "mass_density:=", "8933", "specific_
heat:=",
"385", "youngs_modulus:=", "120000000000", "poissons_ratio:=", "0.38", "thermal_expansion_coef-
ficient:=",
"1.77e-05", "core_loss_equiv_cut_depth:=", "0.001meter")

```

Example: B-P Curve Core Loss Model

Note: B-P Curve properties for Core Loss model are valid only for 2D/3D Transient and 2D/3D Eddy Current designs.

```

Set oProject = oDesktop.SetActiveProject("ThermalCoreLoss_2D_Eddy")
Set oDefinitionManager = oProject.GetDefinitionManager()
oDefinitionManager.EditMaterial "steel_1008_CL",
Array("NAME:steel_1008_CL", "CoordinateSystemType:=",
"Cartesian", "BulkOrSurfaceType:=", 1, Array("NAME:PhysicsTypes", "set:=", Array(

```

```
"Electromagnetic", "Thermal", "Structural")), Array("NAME:AttachedData", Array("NAME:MatAppearanceData", "property_data:=", "appearance_data", "Red:=", 161, "Green:=", 161, "Blue:=", 161)), Array("NAME:ModifierData", Array("NAME:ThermalModifierData", "modifier_data:=", "thermal_modifier_data", Array("NAME:all_thermal_modifiers"))), Array("NAME:permeability", "property_type:=", "non-linear", "BTypeForSingleCurve:=", "normal", "HUnit:=", "A_per_meter", "BUnit:=", "tesla", "IsTemperatureDependent:=", false, Array("NAME:BHCoordinates", Array("NAME:DimUnits", "", ""), Array("NAME:Point", 0, 0), Array("NAME:Point", 159.2, 0.2402), Array("NAME:Point", 318.3, 0.8654), Array("NAME:Point", 477.5, 1.1106), Array("NAME:Point", 636.6, 1.2458), Array("NAME:Point", 795.8, 1.331), Array("NAME:Point", 1591.5, 1.5), Array("NAME:Temperatures")), "conductivity:=", "2000000", "magnetic_loss_tangent:=", "0", Array("NAME:magnetic_coercivity", "property_type:=", "VectorProperty", "Magnitude:=", "0A_per_meter", "DirComp1:=", "1", "DirComp2:=", "0", "DirComp3:=", "0"), "thermal_conductivity:=", "45", Array("NAME:core_loss_type", "property_type:=", "ChoiceProperty", "Choice:=", "B-P Curve", "core_loss_kdc:=", "0", "mass_density:=", "7872", "specific_heat:=", "481", "youngs_modulus:=", "200000000000", "poissons_ratio:=", "0.25", "thermal_expansion_coefficient:=", "1.26e-05",
```

```

"core_loss_equiv_cut_depth=", "0.001meter",
Array("NAME:core_loss_curves",
"property_type=", "nonlinear",
"PUnit=", "kw/m^3",
"BUnit=", "tesla",
"Frequency=", "60Hz",
"Thickness=", "0.5mm",
"IsTemperatureDependent=", true,
Array("NAME:BPCoordinates",
Array("NAME:DimUnits", "", "")),
Array("NAME:Temperatures",
Array("NAME:Temperature", "TempRef=", "10cel",
Array("NAME:BPCoordinates",
Array("NAME:DimUnits", "", "")),
Array("NAME:Point", 0, 0),
Array("NAME:Point", 1, 2),
Array("NAME:Point", 3, 4))),
Array("NAME:Temperature", "TempRef=", "20cel",
Array("NAME:BPCoordinates",
Array("NAME:DimUnits", "", "")),
Array("NAME:Point", 0, 0),
Array("NAME:Point", 11, 21),
Array("NAME:Point", 31, 41))))))

```

GetProjxtMaterialNames

Returns the material names belonging to an active Project.

UI Access	N/A		
Parameters	Name	Type	Description
	None		
Return Value	String names of the materials in the active project.		

Python Syntax	GetProjectMaterialNames()
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDefinitionManager = oProject.GetDefinitionManager() materials = oDefinitionManager.GetProjectMaterialNames() AddWarningMessage(str(materials))</pre>

ExportMaterial

Exports a local material to a library.

UI Access	Export to Library command in the material editor.		
Parameters	Name	Type	Description
	<ExportData>	Array	["NAME:<LibraryName>", <MaterialName>, <MaterialName>, ...]
	<LibraryName>	String	Name of the exported library.
	<MaterialName>	String	Name of the material to be exported.
	<LibraryLocation>	String	Location to save the library. Only "PersonalLib" and "UserLib" are allowed.
Return Value	None.		

Python Syntax	<code>ExportMaterial (<ExportData>, <LibraryLocation>)</code>
Python Example	<pre>oDefinitionManager.ExportMaterial (["NAME:mylib",_ "Material1", "Material2", "Material3"], "PersonalLib")</pre>

VB Syntax	<code>ExportMaterial <ExportData>, <LibraryLocation></code>
VB Example	<pre>oDefinitionManager.ExportMaterial Array("NAME:mylib",_ "Material1", "Material2", "Material3"), "PersonalLib"</pre>

RemoveMaterial

Removes a material from a library.

UI Access	Remove Material(s) command in the material editor		
Parameters	Name	Type	Description
	<MaterialName>	String	Name of the material to be removed.
	<IsProjectMaterial>	Boolean	If True, Maxwell assumes the material is a project material. The last two parameters will be ignored. If False, the material is not a project material.
	<LibraryName>	String	Name of the user or personal library where the material resides.
	<LibraryLocation>	String	Location of library. Valid options:"UserLib" or "PersonalLib".
Return Value	None.		

Python Syntax	<code>RemoveMaterial (<MaterialName>, <IsProjectMaterial>, <LibraryName>, <LibraryLocation>)</code>
Python Example	<code>oDefinitionManager.RemoveMaterial (["Material1", false, "mo0907", "UserLib"])</code>

VB Syntax	<code>RemoveMaterial <MaterialName>, <IsProjectMaterial>, <LibraryName>, <LibraryLocation></code>
VB Example	<code>oDefinitionManager.RemoveMaterial "Material1", false, "mo0907", "UserLib"</code>

RemoveUnusedDefinitions

Removes any unused project definitions.

UI Access	Tools > Project Tools > Remove Unused Definitions.		
Parameters	Name	Type	Description
	<code><Definitions></code>	Array	Definitions to be removed, such as materials and surface materials.
Return Value	None.		

Python Syntax	<code>RemoveUnusedDefinitions(<Definitions>)</code>
Python Example	<code>oProject.RemoveUnusedDefinitions (</code>

	<pre>[["NAME:Materials", "Al-Extruded"], ["NAME:SurfaceMaterials", "Steel-oxidised-surface"]]</pre>

VB Syntax	RemoveUnusedDefinitions <Definitions>
VB Example	<pre>oProject.RemoveUnusedDefinitions Array(Array("NAME:Materials", "Al-Extruded"), Array ("NAME:SurfaceMaterials", "Steel-oxidised-surface"))</pre>

Component Manager Script Commands

The component manager provides access to components in a project. The manager object is accessed via the definition manager.

```
Set oDefinitionManager = oProject.GetDefinitionManager()  
Set oComponentManager = oDefinitionManager.GetManager("Component")
```

The topics for this section include:

[Add](#)

[AddDynamicNPortData](#)

[AddNPortData](#)

[AddSolverOnDemandModel](#)

[ClearSolutionCache](#)

[Edit](#)

[EditSolverOnDemandModel](#)

[EditWithComps](#)

[Export](#)

[GetNPortData](#)

[GetSolverOnDemandData](#)

[GetSolverOnDemandModelList](#)

[Remove](#)

[RemoveSolverOnDemandMode](#)

[UpdateDynamicLink](#)

Add [component manager]

*Use:*Add a component

Command: Tools > Edit Configured Libraries > Components > Add Component

*Syntax:*Add Array("NAME:<ComponentName>",
 "Info:=", <ComponentInfo>,

```
"RefBase:=", <string>, // reference designator
"NumParts:=", <int>, // parts per component
"OriginalComponent:=", <string>
"Terminal:=", <TerminalInfo>,
"Terminal:=", <TerminalInfo>, ...
// The remaining parameters are optional
Array("NAME:Parameters", // any combo of the following
"VariableProp:=", <VariableInfo>,
"CheckboxProp:=", <CheckBoxInfo>,
"ButtonProp:=", <ButtonInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>),
Array("NAME:Properties", // any combo of the following
"CheckboxProp:=", <CheckBoxInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
```

```
"MenuProp:=", <MenuInfo>),  
"VPointProp:=", <VPointInfo>,  
"PointProp:=", <PointInfo>),  
Array("Quantities",  
"QuantityProp:=", <QuantityPropInfo>...),  
Array("NAME:CosimDefinitions",  
  <CosimDefInfo>,  
  <CosimDefInfo>...)
```

Return Value: <string>

```
// composite name of the component.  
// If the name requested conflicts with the name of an existing  
// component, the requested name is altered to be unique.  
// The name returned reflects any change made to be unique.
```

Parameters: <ComponentName>:

```
<string> // simple name of the component
```

```
<ComponentInfo>:
```

```
Array("Type:=", <TypeInfo>,  
"NumTerminals:=", <int>,
```

```
"DataSource:=", <string>,  
"ModifiedOn:=", <ModifiedOnInfo>,  
"Manufacturer:=", "<string>,"  
"Symbol:=", <string>,  
"Footprint:=", <string>,  
"Description:=", <string>,  
"InfoTopic:=", <string>,  
"InfoHelpFile:=", <string>,  
"IconFile:=", <string>,  
"LibraryName:=", "",  
"OriginalLocation:=", "Project", // Project Location  
"Author:=", <string>,  
"OriginalAuthor:=", <string>,  
"CreationDate:= ", <int>)
```

<TypeInfo>:

An integer that is the or-ing of bits for each product listed below. The default setting is 0xffffffff (4294967295) which indicates valid for all products. In the component editing dialog, checking different boxes in the "Specify products for which this component is valid" grid control sets specific flags that correspond to the following hex/decimal settings:

Nexxim -- 100 binary, 4 decimal, 0x4

SIwaveDeNovo -- 1000 binary, 8 decimal, 0x8

Simplorer -- 10000 binary, 16 decimal, 0x10

MaxwellCircuit -- 100000 binary, 32 decimal, 0x20

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<TerminalInfo>:

Array(<string>, // symbol pin

<string> // footprint pin

<string>, // gate name

<bool>, // shared

<int>, // equivalence number

<int>, // what to do if unconnected: flag as error:0, ignore:1

<string> // description

<Nature>)

<Nature>:

<string> // content varies as follows

Nexxim/Circuit:

"Electrical" // the only choice

Simplorer:

// several choices

"Electrical", "Magnetic", "Fluidic", "Translational",

"Translational_V", "Rotational", "Rotational_V",

""Radiant", "Thermal", or <VHDLPackageName>

<VHDLPackageName>:

<string> // in the form <Library>.<Package>

<Library>:

<string> // name of the VHDL library

<Package>:

<string> // name of the VHDL package

<VariableInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back
<string>) // value: number, variable, or expression

<FlagLetters>:

<string> // "D" - has description parameter,
// "RD" - readonly & has description parameter,
// or "RHD" - readonly, hidden, & has description parameter

<CheckBoxInfo>:

Array(<string>, // name
<FlagLetters>,
<string>, // description
"CB:=", <string>, // optional - script for call back
<bool>) // value: true or false

<ButtonInfo>:

Array(<string>, // name
<FlagLetters>,
<string>, // description
<string>, // button title
<string>, // extra text

```
<ClientID>,  
"ButtonPropClientData:= ", <ClientDataArray>)
```

```
<ClientID>:  
<int> // specifies Button Prop Client  
// 0 - unknown, ButtonPropClientData  
// array will be empty  
// 1 - Netlist Prop Client  
// 2 - not used  
// 3 - File Name Prop Client
```

```
<ClientDataArray>:  
varies with <ClientID>
```

```
<ClientID> is 0 or 1: empty array  
Array()
```

```
<ClientID> is 3:  
Array("InternalFormatText:=", "<prefix><RelativePath>")
```

```
<prefix>:
```

<string> // "<Project>", "<PersonalLib>", "<UserLib>", or "<SysLib>"

<RelativePath>:

<string> // relative path to file from <prefix>

<TextInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back

<string>) // value: a text string

<NumberInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back

<real>, // value: a number

<string>) // units

<SeparatorInfo>:

```
Array(<string>, // name
<FlagLetters>,
<string>, // description
"CB:=", <string>, // optional - script for call back
<string>) // value: a text string
```

<ValueInfo>:

```
Array(<string>, // name
<FlagLetters>,
<string>, // description
"CB:=", <string>, // optional - script for call back
<string>) // value: a number, variable or expression
```

<MenuPropInfo>:

```
Array(<string>, // name
<FlagLetters>,
<string>, // description
<string>, // menu choices - separated by commas
<int>) // 0 based index of current menu choice
```

<VPointInfo>:

```
Array(<string>, // name
```

```
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>, // x value: number with length units  
<string>) // y value: number with length units
```

```
<PointInfo>:  
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<real>, // x value  
<real>) // y value
```

```
<QuantityPropInfo>:  
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
<string>, // value  
<TypeString>,
```

<TypeStringDependentInfo>)

<TypeString>:

<string> // "Across", "Through", or "Free"

<TypeStringDependentInfo>:

<TypeString> is "Free" :

<string>, // direction: "In", "Out", "InOut", or "DontCare"

// Following <string> is not present if direction is "DontCare"

<string> // when to calculate: "BeforeAnalogSolver",

// "BeforeStateGraph", "AfterStateGraph", or "DontCareWhen"

<TypeString> is "Across" or "Through":

<int>, // terminal 1

<int> // terminal 2

<CosimDefInfo>:

Array("NAME:CosimDefinition",

"CosimulatorType:=", <int> ,

"CosimDefName:=", <string> // "HFSS 3D Layout", "Circuit",

// "Custom", or "Netlist"

"IsDefinition:=", <bool>,
final array member(s) vary with CosimDefName)

final array members for HFSS 3D Layout:

"CosimStackup:=", <string>,
"CosimDmbedRatio:=", <int>

final array members for Circuit:

"ExportAsNport:=", <int>,
"UsePjt:=", <int>

final array member for Custom:

"DefinitionCompName:=", <string>

final array member for Netlist:

"NetlistString:=", <string>

VB Example:

Dim name

oComponentManager.Add (Array("NAME:MyComponent", _


```
"Info:=", Array("Type:=", 4294901767, _  
"NumTerminals:=", 2, _  
"DataSource:=", "", _  
"ModifiedOn:=", 1071096503, _  
"Manufacturer:=", "Ansys", _  
"Symbol:=", "bendo", _  
"Footprint:=", "BENDO", _  
"Description:=", "", _  
"InfoTopic:=", "", _  
"InfoHelpFile:=", "", _  
"IconFile:=", "", _  
"LibraryName:=", "", _  
"OriginalLocation:=", "Project", _  
"Author:=", "", _  
"OriginalAuthor:=", "", _  
"CreationDate:= ", 1147460679), _  
"Refbase:=", "U", _  
"NumParts:=", 1, _  
"OriginalComponent:=", "", _  
"Terminal:=", Array("n1", _  
"n1", _
```

```
"A", _  
false, _  
0, _  
1, _  
"", _  
"Electrical"), _  
"Terminal:=", Array("n2", _  
"n2", _  
"A", _  
false, _  
1, _  
0, _  
"", _  
"Electrical"), _  
Array("NAME:Parameters", _  
"MenuProp:=", Array("CoSimulator", _  
"D", _  
"", _  
"Default,HFSS 3D Layout,Circuit,Custom,Netlist", _  
0), _
```

```
"ButtonProp:=", Array("CosimDefinition", _  
"D", _  
"", _  
"", _  
"Edit", _  
0, _  
"ButtonPropClientData:=", Array()), _  
Array("NAME:CosimDefinitions", _  
Array("NAME:CosimDefinition", _  
"CosimulatorType:=", 0, _  
"CosimDefName:=", "HFSS 3D Layout", _  
"IsDefinition:=", true, _  
"CosimStackup:=", "Layout stackup", _  
"CosimDmbedRatio:=", 3), _  
Array("NAME:CosimDefinition", _  
"CosimulatorType:=", 1, _  
"CosimDefName:=", "Circuit", _  
"IsDefinition:=", true, _  
"ExportAsNport:=", 0, _  
"UsePjt:=", 0), _  
Array("NAME:CosimDefinition", _
```

```
"CosimulatorType:=", 2, _
"CosimDefName:=", "Custom", _
"IsDefinition:=", true, _
"DefinitionCompName:=", ""), _
Array("NAME:CosimDefinition", _
"CosimulatorType:=", 3, _
"CosimDefName:=", "Netlist", _
"IsDefinition:=", true, _
"NetlistString:=", "")))
```

Python Syntax	Add ["NAME:<ComponentName>", "Info:=", <ComponentInfo>, "RefBase:=", <string>, // reference designator "NumParts:=", <int>, // parts per component "OriginalComponent:=", <string> "Terminal:=", <TerminalInfo>, "Terminal:=", <TerminalInfo>, ... The remaining parameters are optional. ["NAME:Parameters", // any combo of the following "VariableProp:=", <VariableInfo>,

```

"CheckboxProp:=", <CheckBoxInfo>,
"ButtonProp:=", <ButtonInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>],
["NAME:Properties", Any combination of the following:
"CheckboxProp:=", <CheckBoxInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>,
"VPointProp:=", <VPointInfo>,
"PointProp:=", <PointInfo>],
["Quantities",
"QuantityProp:=", <QuantityPropInfo>...],
["NAME:CosimDefinitions",
<CosimDefInfo>,
<CosimDefInfo>...]]

```

Python Example	<pre>oComponentManager.Add(["NAME:Component", "Info:=", ["Type:=", 0, "NumTerminals:=", 0, "DataSource:=", "", "ModifiedOn:=", 1467910752, "Manufacturer:=", "", "Symbol:=", "Component", "ModelNames:=", "", "Footprint:=", "", "Description:=", "", "InfoTopic:=", "", "InfoHelpFile:=", "", "IconFile:=", "", "Library:=", "", "OriginalLocation:=", "Project", "IEEE:=", "", "Author:=", "",</pre>

	<pre>"OriginalAuthor:=", "", "CreationDate:=", 1467910746, "ExampleFile:=", ""], "Refbase:=", "U", "NumParts:=", 1, "ModSinceLib:=", True, "CompExtID:=", 2])</pre>

AddDynamicNPortData [component manager]

Adds a component using the specified data

Command: Project Menu > Add Model > Add 2DExtractor Model

Project Menu > Add Model > Add HFSS Model

Project Menu > Add Model > Add Nexsys Matlab Model

Project Menu > Add Model > Add Nport Model

Project Menu > Add Model > Add Parametric Model

Project Menu > Add Model > Add Q3D Model

Project Menu > Add Model > Add SIwave Model

Project Menu > Add Model > Add State-space Model

Syntax: AddDynamicNPortData Array("NAME:<ComponentDataName>",

"ComponentDataType:=", <DataType>,

"name:=", <string>, // Name of the item

```
"filename:=", <string>, // Path to the file to find the data>
"numberOfports:=", <int>,
"DesignName:=", <string>, // Name of the internal design
"SolutionName:=", <string>, // Name of the solution to reference
"Simulate:=", <bool>,
"CloseProject:=", <bool>,
"SaveProject:=", <bool>,
"RefNode:=", <bool>,
"InterpY:=", <bool>, // true to choose interpolating
"InterpAlg:=", <InterpolationAlgorithm>,
"NewToOldMap:=", <NewToOldMapPairs>,
"OldToNewMap:=", <OldToNewMapPairs>,
"PinNames:=", Array(<string>, <string>...))
```

Return Value: <string>

```
// composite name of the component.
// If the name requested conflicts with the name of an existing
// component, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.
```

Parameters: <ComponentDataName>:

```
<string> // simple name of the component
```


<DataType>:

<string> // "DesignerData" or "Q3DDData"

<InterpolationAlgorithm>:

<string> // "auto", "lin", "shadH", or "shadNH"

<NewToOldMapPairs>:

Array of name/value pairs such as "V1", "V2" that will have the new variable name first and the old variable name second.

<OldToNewMapPairs>:

Array of name/value pairs such as "V1", "V2" that will have the old variable name first and the new variable name second.

VB Example:

```
oComponentManager.AddDynamicNPortData  
Array("NAME:ComponentData", _  
"ComponentDataType:=", "DesignerData", _  
"name:=", "DesignerData", _  
"filename:=", _  
"C:/Program Files/AnsysEM/Designer/Examples/Projects/optiguide/optiguide.adn", _  
"numberofports:=", 2, _  
"DesignName:=", "DesignerModel1", _
```

```
"SolutionName:=", "Setup1 : Adaptive_1", _  
"Simulate:=", true, _  
"CloseProject:=", false, _  
"SaveProject:=", true, _  
"RefNode:=", false, _  
"InterpY:=", true, _  
"InterpAlg:=", "auto", _  
"NewToOldMap:=", Array("nport_height:=", "$height", _  
"nport_length:=", "$length", _  
"nport_width:=", "$width"), _  
"OldToNewMap:=", Array("$height:=", "nport_height", _  
"$length:=", "nport_length", _  
"$width:=", "nport_width"), _  
"PinNames:=", Array( "WavePort1:1", "WavePort2:1"))
```

AddNPortData [component manager]

Use: Adds a component using the specified data

Command: Project Menu > Add Model > Add Nport Model

Syntax: AddNPortData Array("NAME:<ComponentDataName>",

 "ComponentDataType:=", "NportData",

 "name:=", <string>, // Name of the item

```
"filename:=", <string>, // Path to the file to find the data
"numberOfports:=", <int>,
"filelocation:=", <LocationType>,
"domain:=", <string>, // "time" or "frequency"
"datamode:=", <string> // "EnterData", "Import", or "Link"
"devicename:=", <string>,
"ImpedanceTab:=", <bool>,
"NoiseDataTab:=", <bool>,
"DCBehaviorTab:=", <bool>,
"SolutionName:=", <string>,
"displayformat:=", <DisplayInfo>,
"datatype:=", <string>, // "SMatrix", "YMatrix", or "ZMatrix"
"ShowRefPin:=", <bool>,
"RefNodeCheckbox:=", <bool>, ...
<ProductOptionsInfo>)
```

Return Value: <string>

```
// composite name of the component.
// If the name requested conflicts with the name of an existing
// component, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.
```

Parameters: <ComponentDataName>:

<string> // [simple name](#) of the component

<LocationType>:

<string> // one of "UsePath", "PersonalLib", "UserLib", "SysLib",
// or "Project".

<dcInfo>:

<string> // one of "DCOpen", "DCShort", "DCShShort",
// "DCNone", or "DCEmpty".

<DisplayInfo>:

<string> // one of "MagnitudePhase", "ReallImaginary",
// or "DbPhase".

<ProductOptionsInfo>:

// The remaining parameters differ by product

// HFSS 3D Layout - doesn't support interpolation/DC behavior

"DCOption:=", -1,

```
"InterpOption:=", -1,
```

```
"ExtrapOption:=", -1,
```

```
"DataType:=", 0
```

```
// Nexxim
```

```
"DCOption:=", <NexximDCOption>,
```

```
"InterpOption:=", <NexximInterpOption>,
```

```
"ExtrapOption:=", <NexximExtrapOption>,
```

```
"DataType:=", 2
```

```
<NexximDCOption>:
```

```
<int> // 0 : Zero Padding
```

```
// 1 : Same as last point
```

```
// 2 : Linear extrapolation from last 2 points
```

```
// 3 : Constant magnitude, linear phase extrapolation
```

```
// 4 : Leave all signal lines open circuited
```

```
// 5 : Short all signal lines together
```

```
// 6 : Short all signal lines to ground
```

```
<NexximInterpOption>:
```

```
<int> // 0 : Step
```

// 1 : Linear

<NexximExtrapOption>:

<int> // 0 : Zero padding

// 1 : Same as last point

// 2 : Linear extrapolation from last 2 points

// 3 : Constant magnitude, linear phase extrapolation

<CircuitDCOption>:

<int> // 0 : Leave all signal lines open circuited

// 1 : Short all signal lines together

// 2 : Short all signal lines to ground

// 3 : Extrapolate from data provided (not recommended)

<CircuitInterpOption>:

<int> // 0 : Linear

// 1 : Cubic spline

// 2 : Rational polynomial

<CircuitExtrapOption>:

<int> // 0 : Same as interpolation

// 1 : Zero padding

// 2 : Same as last point

VB Example:

```
oComponentManager.AddNPortData Array("NAME:ComponentData", _  
"ComponentDataType=", "NPortData", _  
"name=", "NportData", _  
"filename=", "", _  
"numberofports=", 2, _  
"filelocation=", "UsePath", _  
"domain=", "frequency", _  
"datamode=", "Import", _  
"devicename=", "", _  
"ImpedenceTab=", true, _  
"NoiseDataTab=", true, _  
"DCBehaviorTab=", true, _  
"SolutionName=", "", _  
"displayformat=", "MagnitudePhase", _  
"datatype=", "SMatrix", _  
"ShowRefPin=", true, _  
"RefNodeCheckbox=", false, _
```

```
"DCOption:=", 3, _  
"InterpOption:=", 1, _  
"ExtrapOption:=", 3, _  
"DataType:=", 2, _  
"DCOption:=", 3, _  
"InterpOption:=", 1, _  
"ExtrapOption:=", 3, _  
"DataType:=", 2)
```

AddSolverOnDemandModel

Use: This method looks for a local component of the name passed in, and to this component it adds an SOD model definition using the information passed in the VARIANT. It returns the name of the SOD model added.

Parameters: BSTR component name.

Parameters: VARIANT which is the SOD model data.

Return Value: Returns the name of the model added.

ClearSolutionCache [component manager]

Use: Clear the solution cache for dynamic link component.

Command: Each of the following commands will clear the solution cache:

- **Dynamic Link Item RCM > Clear Solution Cache**
- **Dynamic Link Component in schematic RCM > Clear Solution Cache**

Syntax: ClearSolutionCache <Component Name >

Return Value: None

Parameters: <component name> is the name of the dynamic link component

VB Example:

```
oComponentManager.ClearSolutionCache "TeeModel1"
```

Edit [component manager]

Modifies an existing component

Command: Tools > Edit Configured Libraries > Components > Edit Component

Syntax: Edit <ComponentName>,

```
    Array("NAME:<NewComponentName>",  
    "Info:=", <ComponentInfo>,  
    "RefBase:=", <string>, // reference designator  
    "NumParts:=", <int>, // parts per component  
    "OriginalComponent:=", <string>  
    "Terminal:=", <TerminalInfo>,  
    "Terminal:=", <TerminalInfo>, ...  
    // The remaining parameters are optional  
    Array("NAME:Parameters", // any combo of the following  
    "VariableProp:=", <VariableInfo>,  
    "CheckboxProp:=", <CheckBoxInfo>,  
    "ButtonProp:=", <ButtonInfo>,  
    "TextProp:=", <TextInfo>,
```

```
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>),
Array("NAME:Properties", // any combo of the following
"CheckboxProp:=", <CheckBoxInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>),
"VPointProp:=", <VPointInfo>,
"PointProp:=", <PointInfo>),
Array("Quantities",
"QuantityProp:=", <QuantityPropInfo>...),
Array("NAME:CosimDefinitions",
<CosimDefInfo>,
<CosimDefInfo>...)
```

Return Value: <string>

// [composite name](#) of the component.

// If the name requested conflicts with the name of an existing
// component, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.

Parameters: <ComponentName>:

<string> // [composite name](#) of the component to edit

<NewComponentName>:

<string> // new [simple name](#) for the component

<ComponentInfo>:

Array("Type:=", <TypeInfo>,

"NumTerminals:=", <int>,

"DataSource:=", <string>,

"ModifiedOn:=", <ModifiedOnInfo>,

"Manufacturer:=", "<string>,"

"Symbol:=", <string>,"

"Footprint:=", <string>,"

"Description:=", <string>,"

"InfoTopic:=", <string>,"

"InfoHelpFile:=", <string>,"

```
"IconFile:=", <string>,  
"LibraryName:=", <string>,  
"OriginalLocation:=", <string>, // Project Location  
"Author:=", <string>,  
"OriginalAuthor:=", <string>,  
"CreationDate:= ", <int>)
```

<TypeInfo>:

An integer that is the or-ing of bits for each product listed below. The default setting is 0xffffffff (4294967295) which indicates valid for all products. In the component editing dialog, checking different boxes in the "Specify products for which this component is valid" grid control sets specific flags that correspond to the following hex/decimal settings:

Nexxim -- 100 binary, 4 decimal, 0x4

SlwaveDeNovo -- 1000 binary, 8 decimal, 0x8

Simplorer -- 10000 binary, 16 decimal, 0x10

MaxwellCircuit -- 100000 binary, 32 decimal, 0x20

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<TerminalInfo>:

```
Array(<string>, // symbol pin
<string> // footprint pin
<string >, // gate name
<bool>, // shared
<int>, // equivalence number
<int>, // what to do if unconnected: flag as error:0, ignore:1
<string>, // description
<Nature>)
```

```
<Nature>:
<string> // content varies as follows
```

```
Nexxim/Circuit:
"Electrical" // the only choice
```

```
Simplorer:
// several choices
"Electrical", "Magnetic", "Fluidic", "Translational",
"Translational_V", "Rotational", "Rotational_V",
""Radiant", "Thermal", or <VHDLPackageName>
```

<VHDLPackageName>:

<string> // in the form <Library>.<Package>

<Library>:

<string> // name of the VHDL library

<Package>:

<string> // name of the VHDL package

<VariableInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back

<string>) // value: number, variable, or expression

<FlagLetters>:

<string> // "D" - has description parameter,

// "RD" - readonly & has description parameter,

// or "RHD" - readonly, hidden, & has description parameter

<CheckBoxInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back

<bool>) // value: true or false

<ButtonInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

<string>, // button title

<string>, // extra text

<ClientID>,

"ButtonPropClientData:= ", <ClientDataArray>)

<ClientID>:

<int> // specifies Button Prop Client

// 0 - unknown, "ButtonPropClientData

// array will be empty

// 1 - Netlist Prop Client

// 2 - not used

// 3 - File Name Prop Client

<ClientDataArray>:

varies with <ClientID>

<ClientID> is 0 or 1: empty array

Array()

<ClientID> is 3:

Array("InternalFormatText:=", "<prefix><RelativePath>")

<prefix>:

<string> // "<Project>", "<PersonalLib>", "<UserLib>", or "<SysLib>"

<RelativePath>:

<string> // relative path to file from <prefix>

<TextInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>) // value: a text string
```

<NumberInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<real>, // value: a number  
<string>) // units
```

<SeparatorInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>) // value: a text string
```

<ValueInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>) // value: a number, variable or expression
```

<MenuPropInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
<string>, // menu choices - separated by commas  
<int>) // 0 based index of current menu choice
```

<VPointInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>, // x value: number with length units  
<string>) // y value: number with length units
```

<PointInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

"CB:=", <string>, // optional - script for call back

<real>, // x value

<real>) // y value

<QuantityPropInfo>:

Array(<string>, // name

<FlagLetters>,

<string>, // description

<string>, // value

<TypeString>,

<TypeStringDependentInfo>)

<TypeString>:

<string> // "Across", "Through", or "Free"

<TypeStringDependentInfo>:

```
"Free" :  
<string>, // direction: "In", "Out", "InOut", or "DontCare"  
// Following <string> is not present if direction is "DontCare"  
<string> // when to calculate: "BeforeAnalogSolver",  
// "BeforeStateGraph", "AfterStateGraph", or "DontCareWhen"
```

"Across" or "Through"

```
<int>, // terminal 1
```

```
<int> // terminal 2
```

```
<CosimDefInfo>:
```

```
Array("NAME:CosimDefinition",
```

```
"CosimulatorType:=", <int>,
```

```
"CosimDefName:=", <string> // "HFSS3D", "Circuit",
```

```
// "Custom", or "Netlist"
```

```
"IsDefinition:=", <bool>,
```

```
final array member(s) vary with CosimDefName)
```

final array members for HFSS 3D Layout:

```
"CosimStackup:=", <string>,
```

"CosimDmbedRatio:=", <int>

final array members for Circuit:

"ExportAsNport:=", <int>,

"UsePjt:=", <int>

final array member for Custom:

"DefinitionCompName:=", <string>

final array member for Netlist:

"NetlistString:=", <string>

VB Example:

```
Dim name
```

```
name = oComponentManager.Edit ("Nexxim Circuit Elements\BJTs:Level01_NPN", _  
Array("NAME:Level01_NPN", _  
"Info:=", Array("Type:=", 4294901764,_  
"NumTerminals:=", 3, _  
"DataSource:=", "Ansys built-in component", _  
"ModifiedOn:=", 1152722112, _  
"Manufacturer:=", "", _  
"Symbol:=", "nexx_bjt_npn", _
```

```
"Footprint:=", "", _
"Description:=", "BJT, GP, NPN", _
"InfoTopic:=", "NXBJT1.htm", _
"InfoHelpFile:=", "nexximcomponents.chm", _
"IconFile:=", "bjtsn.bmp", _
"Library:=", "Nexxim Circuit Elements\BJTs", _
"OriginalLocation:=", "SysLibrary ", _
"Author:=", "", _
"OriginalAuthor:=", "", _
"CreationDate:=", 1152722102), _
"Refbase:=", "Q", _
"NumParts:=", 1, _
"Terminal:=", Array("collector", _
"collector", _
"A", _
false, _
6, _
0, _
"", _
"Electrical"), _
```

```
"Terminal:=", Array("base", _  
"base", _  
"A", _  
false, _  
7, _  
0, _  
"", _  
"Electrical"), _  
"Terminal:=", Array("emitter", _  
"emitter", _  
"A", _  
false, _  
8, _  
0, _  
"", _  
"Electrical"), _  
Array("NAME:Parameters", _  
"TextProp:=", Array("LabelID", _  
"HD", _  
"Property string for netlist ID", _  
"Q@ID"), _
```

```
"TextProp:=", Array("MOD", _  
"D", _  
"Name of model data reference", _  
"required"), _  
"VariableProp:=", Array("AREA", _  
"D", _  
"Emitter area multiplying factor, which affects currents, resistances, and capacitances", _ "1"),  
_  
"VariableProp=", Array("AREAB", _  
"D", _  
"Base AREA", _  
"1"), _  
"VariableProp:=", Array( "AREAC", _  
"D", _  
Collector AREA", _  
"1"), _  
"VariableProp:=", Array("DTEMP", _  
"D", _  
"The difference between element and circuit temperature deg Cel)", _  
"0"), _  
"VariableProp:=", Array("M", _
```



```

"D", _
"Multiplier factor to simulate multiple BJTs in paralel", _ "1"), _
"ButtonProp:=", Array("NexximNetlist", _
"HD", _
"", _
"Q@ID %0 %1 %2 *MOD(@MOD) *AREA (AREA=@AREA) " & _
" *AREAB (AREAB=@AREAB) *AREAC (AREAC=@ " & _
"AREAC) *DTEMP (DTEMP=@DTEMP) *M (M=@M) ", _
"Q@ID %0 %1 %2 *MOD(@MOD) " & _ "*AREA (AREA=@AREA) AREAB (AREAB=@AREAB) *AREAC (AREAC=@ " & _
"AREAC) *DTEMP (DTEMP=@DTEMP) *M (M=@M) ", _
1, _
"ButtonPropClientData:=", Array()), _
"TextProp:=", Array( "ModelName", _
"HD", _
"", _
"Q"))))

```

VB Example:

```

Dim name2
name2 = oComponentManager.Edit "MyComponent", _
(Array("NAME:MyOtherComponent", _
"Info:=", Array("Type:=", 4294901767, _

```

```
"NumTerminals:=", 2, _  
"DataSource:=", "", _  
"ModifiedOn:=", 1071096503, _  
"Manufacturer:=", "Ansys", _  
"Symbol:=", "bendo", _  
"Footprint:=", "BENDO", _  
"Description:=", "", _  
"InfoTopic:=", "", _  
"InfoHelpFile:=", "", _  
"IconFile:=", "", _  
"LibraryName:=", "", _  
"OriginalLocation:=", "Project", _  
"Author:=", "", _  
"OriginalAuthor:=", "", _  
"CreationDate:= ", 1147460679), _  
"Refbase:=", "U", _  
"NumParts:=", 1, _  
"OriginalComponent:=", "", _  
"Terminal:=", Array("n1", _  
"n1", _
```

```
"A", _  
false, _  
0, _  
0, _  
"", _  
"Electrical"), _  
"Terminal:=", Array("n2", _  
"n2", _  
"A", _  
false, _  
1, _  
0, _  
"", _  
Electrical"), _  
Array("NAME:Parameters", _  
"MenuProp:=", Array("CoSimulator", _  
"D", _  
"", _  
"Default, HFSS3D, Circuit, Custom, Netlist", _  
0), _  
"ButtonProp:=", Array("CosimDefinition", _
```

```
"D", _  
"  
"  
"  
"Edit", _  
0, _  
"ButtonPropClientData:=", Array()), _  
Array("NAME:CosimDefinitions", _  
Array("NAME:CosimDefinition", _  
"CosimulatorType:=", 0, _  
"CosimDefName:=", "HFSS3D", _  
"IsDefinition:=", true, _  
"CosimStackup:=", "Layout stackup", _  
"CosimDmbedRatio:=", 3), _  
Array("NAME:CosimDefinition", _  
"CosimulatorType:=", 1, _  
"CosimDefName:=", "Circuit", _  
"IsDefinition:=", true, _  
"ExportAsNport:=", 0, _  
"UsePjt:=", 0), _  
Array("NAME:CosimDefinition", _
```

```
"CosimulatorType:=", 2, _  
"CosimDefName:=", "Custom", _  
"IsDefinition:=", true, _  
"DefinitionCompName:=", ""), _  
Array("NAME:CosimDefinition", _  
"CosimulatorType:=", 3, _  
"CosimDefName:=", "Netlist", _  
"IsDefinition:=", true, _  
"NetlistString:=", "")))
```

Python Syntax	<pre>Edit <ComponentName>, ["NAME:<NewComponentName>", "Info:=", <ComponentInfo>, "RefBase:=", <string>, // reference designator "NumParts:=", <int>, // parts per component "OriginalComponent:=", <string> "Terminal:=", <TerminalInfo>, "Terminal:=", <TerminalInfo>, ... #The remaining parameters are optional ["NAME:Parameters", // any combo of the following "VariableProp:=", <VariableInfo>,</pre>

```
"CheckboxProp:=", <CheckBoxInfo>,
"ButtonProp:=", <ButtonInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>],
["NAME:Properties", # any combo of the following
"CheckboxProp:=", <CheckBoxInfo>,
"TextProp:=", <TextInfo>,
"NumberProp:=", <NumberInfo>,
"SeparatorProp:=", <SeparatorInfo>,
"ValueProp:=", <ValueInfo>,
"MenuProp:=", <MenuInfo>),
"VPointProp:=", <VPointInfo>,
"PointProp:=", <PointInfo>),
["Quantities",
"QuantityProp:=", <QuantityPropInfo>...],
["NAME:CosimDefinitions",
<CosimDefInfo>,
```

	<CosimDefInfo>...])
Python Example	<pre> name = oComponentManager.Edit ("Simplorer Circuit Elements\BJTs:Level01_NPN", _ ["NAME:Level01_NPN", "Info:=", ["Type:=", 4294901764,_ "NumTerminals:=", 3, "DataSource:=", "Ansoft built-in component",_ "ModifiedOn:=", 1152722112, "Manufacturer:=", "", _ "Symbol:=", "nexx_bjt_npn", "Footprint:=", "", _ "Description:=", "BJT, GP, NPN", "InfoTopic:=", "NXBJT1.htm", _ "InfoHelpFile:=", "nexximcomponents.chm", "IconFile:=", "bjtsn.bmp", _ "Library:=", "Nexxim Circuit Elements\BJTs",_ "OriginalLocation:=", "SysLibrary ", "Author:=", "", _ "OriginalAuthor:=", "", "CreationDate:=", 1152722102], _ "Refbase:=", "Q", "NumParts:=", 1, "Terminal:=", ["collector", _ "collector", "A", false, 6, 0, "", "Electrical"], _ "Terminal:=", ["base", "base", "A", false, _ 7, 0, "", "Electrical"], "Terminal:=", ["emitter", _ "emitter", "A", false, 8, 0, "", "Electrical"], _ ["NAME:Parameters", "TextProp:=", ["LabelID", _ "HD", "Property string for netlist ID", _ "Q@ID"], "TextProp:=", ["MOD", "D", _ "Name of model data reference", "required"], _ </pre>

	<pre> "VariableProp:=", ["AREA", "D", _ "Emitter area multiplying factor, which affects currents, resistances, and capacitances", "1"], _ "VariableProp:=", ["AREAB", "D", "Base AREA", _ "1"], "VariableProp:=", ["AREAC", "D", "Collector AREA", _ "1"], "VariableProp:=", ["DTEMP", "D", _ "The difference between element and circuit temperature (deg Cel)", _ "0"], "VariableProp:=", ["M", "D", _ "Multiplier factor to simulate multiple BJTs in parallel", _ "1"], "ButtonProp:=", ["NexximNetlist", "HD", "", _ "Q@ID %0 %1 %2 *MOD(@MOD) *AREA (AREA=@AREA) "& _ " *AREAB (AREAB=@AREAB) *AREAC (AREAC=@" & _ "AREAC) *DTEMP (DTEMP=@DTEMP) *M (M=@M) ", _ "Q@ID %0 %1 %2 *MOD(@MOD) " & "*AREA (AREA=@AREA) *AREAB (AREAB=@AREAB) *AREAC (AREAC=@" & _ "AREAC) *DTEMP (DTEMP=@DTEMP) *M (M=@M) ", 1, _ "ButtonPropClientData:=", [], "TextProp:=", ["ModelName", "HD", "", "Q"]]]) </pre>
Python	<pre> name2 = oComponentManager.Edit ("MyComponent",_ </pre>

**Example
2**

```
(["NAME:MyOtherComponent","Info:=", ["Type:=", 4294901767, _  
  
"NumTerminals:=", 2, "DataSource:=", "", _  
  
"ModifiedOn:=", 1071096503, "Manufacturer:=", "Ansoft", _  
  
"Symbol:=", "bendo", "Footprint:=", "BENDO", _  
  
"Description:=", "", "InfoTopic:=", "", _  
  
"InfoHelpFile:=", "", "IconFile:=", "", _  
  
"LibraryName:=", "", "OriginalLocation:=", "Project", _  
  
"Author:=", "", "OriginalAuthor:=", "", _  
  
"CreationDate:= ", 1147460679], "Refbase:=", "U", _
```

```
"NumParts:=", 1, "OriginalComponent:=", "", _  
  
"Terminal:=", ["n1", "n1", "A", false, 0, 0, "", _  
  
"Electrical"], "Terminal:=", ["n2", "n2", "A", _  
  
false, 1, 0, "", Electrical"], ["NAME:Parameters", _  
  
"MenuProp:=", ["CoSimulator", "D", "", _  
  
"Default,Custom,Netlist", 0], "ButtonProp:=", ["CosimDefinition", _  
  
"D", "", "", "Edit", 0, "ButtonPropClientData:=", []], _  
  
["NAME:CosimDefinitions", ["NAME:CosimDefinition", _  
  
"CosimulatorType:=", 0, "CosimDefName:=", "HFSS3D", _
```

```
"IsDefinition:=", true, "CosimStackup:=", "Layout stackup", _  
  
"CosimDmbedRatio:=", 3], ["NAME:CosimDefinition", _  
  
"CosimulatorType:=", 1, "CosimDefName:=", "", _  
  
"IsDefinition:=", true, "ExportAsNport:=", 0, _  
  
"UsePjt:=", 0], ["NAME:CosimDefinition", _  
  
"CosimulatorType:=", 2, "CosimDefName:=", "Custom", _  
  
"IsDefinition:=", true, "DefinitionCompName:=", ""], _  
  
["NAME:CosimDefinition", "CosimulatorType:=", 3, _  
  
"CosimDefName:=", "Netlist", "IsDefinition:=", true, _  
  
"NetlistString:=", ""]]])
```


EditSolverOnDemandModel

Use: This method looks for a local component of the name passed in, and in this component it looks for an SOD model using the name passed in the second BSTR. It modifies the SOD model using the data in the VARIANT. It returns the name of the SOD model edited.

Return Value: Returns the name of the model edited.

Parameters: BSTR component name.

Parameters: BSTR SOD model name.

Parameters: VARIANT which is the new SOD model data (can include changed name).

EditWithComps [component manager]

Edit an existing component.

Command: None

Syntax: EditWithComps <ComponentName>,
 Array("NAME:<NewComponentName>",
 "ModTime:=", <ModifiedTimeInfo>,
 "Library:=", <string>, // Library name
 "LibLocation:=", <string>, // Project Location
 <PinDefInfo>,
 <PinDefInfo>,... // optional, to define pins
 <GraphicsDataInfo>, // optional, to define graphics
 <PropDisplayMapInfo>), // optional, to define property displays

Array(<ListOfComponentNames>) // Component names

Return Value: <string>

// [composite name](#) of the component.

// If the name requested conflicts with the name of an existing

// component, the requested name is altered to be unique.

// The name returned reflects any change made to be unique.

Parameters: <ComponentName>:

<string> // [composite name](#) of the component being edited

<NewComponentName>:

<string> // new [simple name](#) for the component

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed

since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<PinDefInfo>:

Array("NAME:PinDef",

"Pin:=", Array (<string>, // pin name

```
<real>, // x location  
<real>, // y location  
<real>, // angle in radians  
<PinType>,  
<real>, // line width  
<real>, // line length  
<bool>, // mirrored  
<int>, // color  
<bool>, // true if visible, false if not  
<string>, // hidden net name  
<OptionalPinInfo>, // optional info  
<PropDisplayMapInfo>)) // optional
```

```
<PinType>:  
<string> // "N" : normal pin  
// "I" : input pin  
// "O" : output pin
```

```
<OptionalPinInfo>:  
// Specify both or neither
```

<bool>, // true if name is to be shown

<bool>, // true if number is to be shown

<PropDisplayMapInfo>:

Array("NAME:PropDisplayMap",

<PropDisplayInfo>,

<PropDisplayInfo>,...)

<PropDisplayInfo>:

<NameString>, Array(<DisplayTypeInfo>,

<DisplayLocationInfo>,

<int>, // optional, level number

<TextInfo>)

<NameString>:

<string> // PropertyName:=, where PropertyName is the name of

// the property to be displayed

<DisplayTypeInfo>:

<int> // 0 : No display

// 1 : Display name only

// 2 : Display value only
// 3 : Display both name and value
// 4: Display evaluated value only
// 5: Display both name and evaluated value

<DisplayLocationInfo>:

<int> // 0 : Left

// 1 : Top

// 2 : Right

// 3 : Bottom

// 4 : Center

// 5 : Custom placement

<GraphicsDataInfo>:

Array("NAME:Graphics",

// one or more of the following

<RectInfo>,

<CircleInfo>,

<ArcInfo>,

<LineInfo>,


```
<PolygonInfo>,  
<TextInfo>,  
<ImageInfo>)
```

```
<RectInfo>:  
"Rect:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // angle, in radians  
<real>, // x position of center  
<real>, // y position of center  
<real>, // width  
< real>) // height
```

```
<CircleInfo>:  
"Circle:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // x position of center  
<real>, // y position of center  
< real>) // radius
```

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>, // radius

<real>, // start angle, in radians

<end>) // end angle, in radians

<LineInfo>:

"Line:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<PointInfo>, // must specify at least 2 points

<PointInfo>...)

<PointInfo>:

<real>, // x position

<real> // y position

<PolygonInfo>:

"Polygon:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<PointInfo>, // must specify at least 3 points

<PointInfo>...)

<TextInfo>:

"Text:=", Array(<real>, // x position

<real>, // y position

<real>, // angle, in radians

<Justification>,

<bool>, // is plotter font

<string>, // font name

<int>, // color

<string>) // text string

<Justification>:

<int> // 0 : left top

// 1 : left base

// 2 : left bottom

// 3 : center top

// 4 : center base

// 5 : center bottom

// 6 : right top

// 7 : right base

// 8 : right bottom

<ImageInfo>:

"Image:=", Array(<RectInfo>,

<ImageData>,

<bool>) // is mirrored

<ImageData>:

<string>, // file path

<int>, // 0 : use the file path and link to it

// 1 : ignore file path and use next parameter

<string> // text data, only present if preceding int is 1

<ListOfComponentNames>:

<string>,<string> ...

// The list may be empty. When not empty, each string that is listed is a component

// that references the component to be edited. Prior to editing, a clone of the component is

// made, and the components that are listed are modified so that they now refer to

// the clone.

VB Example:

```
Dim nam
```

```
oModelManager.EditWithComps_
```

```
("Nexxim Circuit Elements\Distributed\Distributed:bendo", _
```

```
Array("NAME:bendo new name", _
```

```
"ModTime:=", 1152722165, _
```

```
"Library:=", "Nexxim Circuit Elements\Distributed\Distributed", _
```

```
"LibLocation:=", "SysLibrary", _
```

```
Array("NAME:PinDef", _
```

```
"Pin:=", Array( "n1", _
```

```
-0.00254, _
```

```
0.00254, _
```

```
0, _
```

```
"N", _
```

```
0, _
```

```
0, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _  
Array("NAME:PinDef", _  
"Pin:=", Array( "n2", _  
0.00254, _  
-0.00254, _  
1.5708, _  
"N", _  
0, _  
0, _  
false, _  
0, _  
true, _  
"", _  
false, _
```

```
false)), _
Array("NAME:Graphics", _
"Polygon:=", Array( 0, 0, 12566272, 0, 0.00381, 0, .00127, 0.00127, _
0.00127, 0.00127, 0, 0.00381, 0, 0.00381, _
0.002032, 0.00127, 0.00381), _
"Line:=", Array(0, 1, 12566272, -0.00254, 0.00254, 0, .00254), _
"Line:=", Array(0, 1, 12566272, 0.00254, -0.00254, .00254, 0)), _
Array("NAME:PropDisplayMap", _
"W:=", Array(3, _
5, _
0, _
"Text:=", Array( 2.1684E-019, _
0.00504119, _
0, _
1, _
5, _
false, _
"Arial", _
0, _
"W=***")))), _
Array("MY_COMP")
```

Export [component manager]

Export component(s) to a library

Command: Tools > Edit Configured Libraries > Components > Export to Library

Syntax: Export Array("NAME:<LibraryName>",

 <ComponentName>,

 <ComponentName>...),

 <LibraryLocation>

Return Value: None

Parameters: <LibraryName>:

 <string> // name of the library

 <ComponentName>:

 <string> // [composite name](#) of the component to export

 <LibraryLocation>:

 <string> // location of the library in <LibraryName>

 // One of "Project", "PersonalLib", or "UserLib"

VB Example:


```
oComponentManager.Export Array("NAME:mylib", "Nexxim Circuit Elements\BJTs:Level01_NPN"),"PersonalLib"
```

Python Syntax	Export(["NAME:<LibraryName>", <ComponentName>, <ComponentName>...], <LibraryLocation>)
Python Example	<pre>oComponentManager.Export (["NAME:mylib", "Simplorer Circuit Elements\BJTs:Level01_NPN"], "PersonalLib")</pre>

GetData [component manager]

Gets data that describes the definition.

Command: None

Syntax: GetData(<DefinitionName>)

Return Value: <DefinitionData> This is an array of data for the definition.

Parameters: <DefinitionName>:

<string> // [composite name](#) of the definition to edit

VB Example:

```
Dim compData
```

```
compData = oComponentManager.GetData ("Level01_NPN")
```

Note:

GetData allows the user to access definition information, make modifications, and then use the Edit or EditWithComps script commands to save the modified definition. Accordingly, for each type of definition, the array data returned to GetData should be the same array information that is supplied to the [Edit](#) or [EditWithComps](#) commands.

Python Syntax	GetData(<DefinitionName>)
Python Example	compData = oComponentManager.GetData("Level01_NPN")

GetNames [component manager]

Returns the names of the components (used and unused) in a design. The following script command, **IsUsed**, can then be used to separate used and unused components.

Command: None

Syntax: GetNames()

Return Value: An array of strings

Parameters: None

VB Example:

```
Dim componentNames
componentNames = oComponentManager.GetNames()
```

Python Syntax	GetNames()
Python Example	componentNames = oComponentManager.GetNames()

GetNPortData [component manager]

Returns NPort data for the component with the specified name.

Command: None

Return Value: Variant array, whose contents depend on the type of component. The array will be empty if the component does not have NPort data. See the syntax for AddDynamicNPortData and AddNPortData for descriptions of the array contents for components with those types of NPort data.

Parameters: <ComponentName>:

<string>

Example: This script displays each item in the returned array for each component in the design.

```
Sub DisplayVariant(x)
```

```
Dim index
```

```
index = 0
```

```
For Each info In x
```

```
    curr = x(index)
```

```
    If TypeName(curr) <> "Variant()" Then
```

```
        If TypeName(curr) <> "Empty" And TypeName(curr) <> "Null" Then
```

```
            str = CStr(curr)
```

```
            If str = "" Then
```

```
        str = ChrW(34) & ChrW(34)

    End If

    MsgBox str

Else

    str = "Empty/Null item."

    MsgBox str

End If

Else

    DisplayVariant curr

End If

    index = index + 1

Next

End Sub
```

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim dnpInfo
```

```
Dim index
```

```
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
```

```
Set oDesktop = oAnsoftApp.GetAppDesktop()
```

```
oDesktop.RestoreWindow
```

```
Set oProject = oDesktop.GetActiveProject()
```

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

```
Set oComponentManager = oDefinitionManager.GetManager("Component")
```

```
Dim componentNames
```

```
componentNames = oComponentManager.GetNames()
```

```
index = 0
```

```
For Each name In componentNames
```

```
    name = componentNames(index)
```

```
    message = "NPort data for component " + name
```

```
    MsgBox message
```

```
    dnpInfo = oComponentManager.GetNPortData(name)
```

```
    DisplayVariant dnpInfo
```

```
index = index + 1
```

Next

GetSolverOnDemandData

Use: This method looks for a local component of the name passed in, and in this component it looks for an SOD model of the name passed in and returns the SOD data pertaining to that model.

Parameters: BSTR component name.

Parameters: BSTR SOD model name

Return Value: VARIANT which is the SOD data.

GetSolverOnDemandModelList

Use: This method looks for a local component of the name passed in, and returns a list of SOD model names defined in the component.

Parameters: BSTR component name.

Return Value: VARIANT which is a list of SOD model names.

IsUsed [component manager]

Used to determine if a component is used in the design.

Command: None

Syntax: IsUsed(<ComponentName>)

Return Value: <Boolean> // true if the specified component is used in the design

Parameters: <ComponentName>:

<string>

VB Example:

```
Dim isUsed  
isUsed = oComponentManager.IsUsed("MyComponent")
```

Python Syntax	IsUsed(<ComponentName>)
Python Example	IsUsed = oComponentManager.IsUsed("MyComponent")

Remove [component manager]

Remove a component from a library

Command: Tools > Edit Configured Libraries > Components > Remove Component

Syntax: Remove <ComponentName>,

 <IsProjectComponent>,

 <LibraryName>,

 <LibraryLocation>

Return Value: None

Parameters: <ComponentName>:

 <string> // [composite name](#) of the component to remove

 <IsProjectComponent>:

 <bool>

 <LibraryName>:

```
<string> // name of the library

<LibraryLocation>:
<string> // location of the library in <LibraryName>
// One of "Project", "PersonalLib", or "UserLib"
```

VB Example:

```
oComponentManager.Remove "Nexxim Circuit Elements\BJTs:Level01_NPN", _ true, "Project"
```

Python Syntax	Remove (<ComponentName>, <IsProjectComponent>, <LibraryName>, <LibraryLocation>)
Python Example	oComponentManager.Remove ("Simplorer Circuit Elements\BJTs:Level01_NPN", _ true, "Project")

RemoveSolverOnDemandModel

Use: This method looks for a local component of the name passed in, and in this component it looks for an SOD model of the name passed in and deletes the SOD model definition from the component.

Parameters: BSTR component name.

Parameters: BSTR SOD model name

Return Value: None.

RemoveUnused [component manager]

Removes components that are not used in the design.

Command: **Project->Remove Unused Definitions** is similar but operates slightly different and does not record script commands.

Syntax: RemoveUnused()

Return Value: <bool> True if one or more components are removed.

Parameters: None

VB Example:

```
Dim removedDefs  
removedDefs = oComponentManager.RemoveUnused()
```

Note:

The order of calls to RemoveUnused is significant. As a result, removing definitions in an unordered fashion may cause other components in dependent definitions to be rendered unusable.

Also, the symbol and footprint of an unused component are not unusable until after the component itself is removed using the Component Manager Remove script.

Python Syntax	RemoveUnused()
Python Example	removedDefs = oComponentManager.RemoveUnused()

Update Dynamic Link [component manager]

Use: Reads data from the linked design and updates the dynamic link component. This will update the following: properties, solutions, ports, geometry.

Command: Each of the following commands will record a non-undoable script command:

- Dynamic Link Item RCM > Refresh Dynamic Link**
- Dynamic Link Component in schematic RCM > Refresh Dynamic Link**
- Click the **Refresh Dynamic Link** button in the Footprint tab of the **Properties Window** for selected Dynamic Link components in the Layout Editor.

Syntax: UpdateDynamicLink(<component name>)

Return Value: None

Parameters: <component name> is the name of the dynamic link component

VB Example:

```
oComponentManager.UpdateDynamicLink("TeeModel_L1")
```

Material Manager Script Commands

The material manager provides access to materials in a project. The manager object is accessed via the definition manager.

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

```
Set oMaterialManager = oDefinitionManager.GetManager("Material")
```

The topics for this section include:

[GetData](#)

[GetNames](#)

[GetProperties](#)

[IsUsed](#)

[RemoveUnused](#)

GetData [material manager]

Use: Get material data

Command: None

Syntax: GetData <material_name>

Return Value: Array of material data

Parameters: <material_name>

Type: string

Value: Name of the project material

VB Example:

```
materialData = oMaterialMgr.GetData("MyMaterial2")
```

Python Syntax	GetData(<SimpleName>)
Python Example	<pre>dataArray = oMaterialManager.GetData ("mymaterial")</pre>

GetNames [material manager]

Use: Get the names of the materials in a project

Command: None

Syntax: GetNames

Return Value: Names of the materials in a project

Parameters: None

Example:

```
materialNames = oMaterialMgr.GetNames()
```

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.NewProject
oProject.InsertDesign "Nexxim Circuit", "Nexxim1", "", ""
Set oMaterialMgr = oProject.GetDefinitionManager().GetManager("Material")
oMaterialMgr.Add( Array("NAME:MyMaterial1", "CoordinateSystemType=", "Cartesian", "per-
mittivity=", "0.123") )
oMaterialMgr.Add( Array("NAME:MyMaterial2", "CoordinateSystemType=", "Cartesian", "per-
mittivity=", "0.456") )
Dim materialNames
materialNames = oMaterialMgr.GetNames()
```

Python Syntax	GetNames()

Python Example	<code>materialnames = oMaterialManager.GetNames()</code>

GetProperties [material manager]

Get material properties. Differs from GetData in that only material properties available to the user are returned by GetProperties.

UI Access	NA		
Parameters	Name	Type	Description
	<material_name>	string Boolean	Name of the project material True or False. If you use False or only a single argument, then GetProperties returns only the properties that are different from the defaults.
Return Value	Array of material data		

Python Syntax	<code>GetProperties (<materialname>)</code>
Python Example	<pre>oMaterialManager.GetProperties("Gold", True) oMaterialManager.GetProperties("vacuum")</pre>

VB Syntax	<code>GetProperties <material_name></code>
VB Example	<code>materialProps = oMaterialMgr.GetProperties("MyMaterial2")</code>

IsUsed [material manager]

Use: Checks if a project material is in use

Command: None

Syntax: IsUsed <material_name>

Return Value: Returns 'True' if the material is in use.

Parameters: <material_name>

Type: string

Value: Name of the project material to check.

VB Example:

```
used = oMaterialMgr.IsUsed( "MyMaterial2" )
```

Python Syntax	IsUsed(<ComboName>)
Python Example	isused = oMaterialManager.IsUsed("mylib:mymaterial")

RemoveUnused [material manager]

Use: Remove all unused materials from the project.

Command: None

Syntax: RemoveUnused

Return Value: None

Parameters: None

VB Example:

```
oMaterialMgr.RemoveUnused()
```

Python Syntax	RemoveUnused()

Python Example

```
thereWereExtras = oMaterialManager.RemoveUnused()
```

Model Manager Script Commands

The model manager provides access to models in a project. The manager object is accessed via the definition manager.

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

```
Set oModelManager = oDefinitionManager.GetManager("Model")
```

The model manager script commands are listed below:

[Add](#)

[ConvertToDynamic](#)

[ConvertToParametric](#)

[Edit](#)

[EditWithComps](#)

[Export](#)

[GetData](#)

[GetNames](#)

[IsUsed](#)

[Remove](#)

[RemoveUnused](#)

Add [model manager]

Use: Add a model

Command: Tools > Edit Configured Libraries > Models > Add Model

Syntax: Add Array("NAME:<modelName>",

```
"ModTime:=", <ModifiedTimeInfo>,  
"Library:=", "", // Library name  
"LibLocation:=", "Project", // Project Location  
<PinDefInfo>,  
<PinDefInfo>,... // optional, to define pins  
<GraphicsDataInfo>, // optional, to define graphics  
<PropDisplayMapInfo>)) // optional, to define property displays
```

Return Value: <string>

```
// composite name of the model.  
// If the name requested conflicts with the name of an existing  
// model, the requested name is altered to be unique.  
// The name returned reflects any change made to be unique.
```

Parameters: <modelName>:

```
<string> // simple name of the model being added
```

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.


```
<PinDefInfo>:  
Array("NAME:PinDef",  
"Pin:=", Array (<string>, // pin name  
<real>, // x location  
<real>, // y location  
<real>, // angle in radians  
<PinType>,  
<real>, // line width  
<real>, // line length  
<bool>, // mirrored  
<int>, // color  
<bool>, // true if visible, false if not  
<string>, // hidden net name  
<OptionalPinInfo>, // optional info  
<PropDisplayMapInfo>)) // optional
```

```
<PinType>:  
<string> // "N" : normal pin  
// "I" : input pin  
// "O" : output pin
```

<OptionalPinInfo>:

// Specify both or neither

<bool>, // true if name is to be shown

<bool>, // true if number is to be shown

<PropDisplayMapInfo>:

Array("NAME:PropDisplayMap",

<PropDisplayInfo>,

<PropDisplayInfo>,...)

<PropDisplayInfo>:

<NameString>, Array(<DisplayTypeInfo>,

<DisplayLocationInfo>,

<int>, // optional, level number

<TextInfo>)

<NameString>:

<string> // PropertyName:=, where PropertyName is the name of

// the property to be displayed

<DisplayTypeInfo>:

<int> // 0 : No display

// 1 : Display name only

// 2 : Display value only

// 3 : Display both name and value

// 4: Display evaluated value only

// 5: Display both name and evaluated value

<DisplayLocationInfo>:

<int> // 0 : Left

// 1 : Top

// 2 : Right

// 3 : Bottom

// 4 : Center

// 5 : Custom placement

<GraphicsDataInfo>:

Array("NAME:Graphics",

// one or more of the following

<RectInfo>,

```
<CircleInfo>,  
<ArcInfo>,  
<LineInfo>,  
<PolygonInfo>,  
<TextInfo>,  
<ImageInfo>)
```

```
<RectInfo>:
```

```
"Rect:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // angle, in radians  
<real>, // x position of center  
<real>, // y position of center  
<real>, // width  
< real>) // height
```

```
<CircleInfo>:
```

```
"Circle:=", Array(<real>, // line width  
<int>, // fill pattern
```

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>) // radius

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>, // radius

<real>, // start angle, in radians

<end>) // end angle, in radians

<LineInfo>:

"Line:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<PointInfo>, // must specify at least 2 points

<PointInfo>...)

<PointInfo>:

<real>, // x position

<real> // y position

<PolygonInfo>:

"Polygon:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<PointInfo>, // must specify at least 3 points

<PointInfo>...)

<TextInfo>:

"Text:=", Array(<real>, // x position

<real>, // y position

<real>, // angle, in radians

<Justification>,

<bool>, // is plotter font

<string>, // font name

<int>, // color

<string>) // text string

<Justification>:

<int> // 0 : left top

// 1 : left base

// 2 : left bottom

// 3 : center top

// 4 : center base

// 5 : center bottom

// 6 : right top

// 7 : right base

// 8 : right bottom

<ImageInfo>:

"Image:=", Array(<RectInfo>,

<ImageData>,

<bool>) // is mirrored

<ImageData>:

<string>, // file path

<int>, // 0 : use the file path and link to it

// 1 : ignore file path and use next parameter

<string> // text data, only present if preceding int is 1

VB Example:

```
oModelManager.Add Array("NAME:MyModel", _  
"ModTime:=", 1070989137, _  
"Library:=", "", _  
"LibLocation:=", "Project", _  
Array("NAME:PinDef", _  
"Pin:=", Array ("newpin0", _  
0.00254, _  
0, _  
0, _  
"N", _  
0, _  
0.00254, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _
```



```
Array("NAME:PinDef", _  
  "Pin:=", Array ("newpin1", _  
    -0.00254, _  
    0, _  
    3.14159265358979, _  
    "N", _  
    0, _  
    0.00254, _  
    false, _  
    0, _  
    true, _  
    "", _  
    false, _  
    false)),  
Array("NAME:Graphics", _  
  "Rect:=", Array(0, _  
    0, _  
    12566272, _  
    0, _  
    4.33680868994202e-019, _  
    -0.000635, _
```

```
0.00508, _  
0.002794), _  
"Circle:=", Array(0, _  
0, _  
12566272, _  
0.000127, _  
-0.000635, _  
0.000635)))
```

ConvertToDynamic

Use: Build a new dynamic model based on an existing parametric model.

Command: Right-click on a model under Definitions/Models in the Project Tree and choose ConvertToDynamic.

Syntax: ConvertToDynamic(defName, newname)

Return Value: <newname> // Name of the new model added

Parameters: <defName> // Model that is the base for the new conversion

VB Example:

```
Dim newname  
  
newname = oModelManager.ConvertToDynamic([in] BSTR defName, [out, retval] BSTR* newName);
```

ConvertToParametric

Use: Build a new parametric model based on an existing dynamic model.

Command: Right-click on a model under Definitions/Models in the Project Tree and choose ConvertToParametric.

Syntax: ConvertToParametric(defName, newname)

Return Value: <newname> // Name of the new model added

Parameters: <defName> // Model that is the base for the new conversion

VB Example: Dim newname

```
newname = oModelManager.ConvertToParametric([in] BSTR defName, [out, retval] BSTR* newName);
```

Edit [deprecated]

Deprecated command — please use [EditWithComps](#).

EditWithComps [model manager]

Use: Edit an existing model.

Command: None

Syntax: EditWithComps <ModelName> ,

```
Array("NAME:<NewModelName>",
```

```
"ModTime:=", <ModifiedTimeInfo> ,
```

```
"Library:=", <string> , // Library name
```

```
"LibLocation:=", <string> , // Project Location
```

```
<PinDefInfo> ,
```

```
<PinDefInfo> ,... // optional, to define pins
```

```
<GraphicsDataInfo> , // optional, to define graphics
```

```
<PropDisplayMapInfo> , // optional, to define property displays
```

```
Array(<ListOfComponentNames>) // Component names
```

Return Value: <string>

// [composite name](#) of the model.
// If the name requested conflicts with the name of an existing
// model, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.

Parameters: <ModelName>:

<string> // [composite name](#) of the model being edited

<NewModelName>:

<string> // new [simple name](#) for the model

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed
since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<PinDefInfo>:

Array("NAME:PinDef",
"Pin:=", Array (<string>, // pin name
<real>, // x location

<real>, // y location
<real>, // angle in radians
<PinType>,
<real>, // line width
<real>, // line length
<bool>, // mirrored
<int>, // color
<bool>, // true if visible, false if not
<string>, // hidden net name
<OptionalPinInfo>, // optional info
<PropDisplayMapInfo>)) // optional

<PinType>:
<string> // "N" : normal pin
// "I" : input pin
// "O" : output pin

<OptionalPinInfo>:
// Specify both or neither
<bool>, // true if name is to be shown
<bool>, // true if number is to be shown

```
<PropDisplayMapInfo>:  
Array("NAME:PropDisplayMap",  
<PropDisplayInfo>,  
<PropDisplayInfo>,...)
```

```
<PropDisplayInfo>:  
<NameString>, Array(<DisplayTypeInfo>,  
<DisplayLocationInfo>,  
<int>, // optional, level number  
<TextInfo>)
```

```
<NameString>:  
<string> // PropertyName:=, where PropertyName is the name of  
// the property to be displayed
```

```
<DisplayTypeInfo>:  
<int> // 0 : No display  
// 1 : Display name only  
// 2 : Display value only
```

// 3 : Display both name and value
// 4: Display evaluated value only
// 5: Display both name and evaluated value

<DisplayLocationInfo>:

<int> // 0 : Left

// 1 : Top

// 2 : Right

// 3 : Bottom

// 4 : Center

// 5 : Custom placement

<GraphicsDataInfo>:

Array("NAME:Graphics",

// one or more of the following

<RectInfo>,

<CircleInfo>,

<ArcInfo>,

<LineInfo>,

<PolygonInfo>,

<TextInfo>,

<ImageInfo>)

<RectInfo>:

"Rect:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<real>, // angle, in radians

<real>, // x position of center

<real>, // y position of center

<real>, // width

< real>) // height

<CircleInfo>:

"Circle:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>) // radius

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>, // radius

<real>, // start angle, in radians

<end>) // end angle, in radians

<LineInfo>:

"Line:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<PointInfo>, // must specify at least 2 points

<PointInfo>...)

<PointInfo>:

<real>, // x position

<real> // y position

<PolygonInfo>:

```
"Polygon:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<PointInfo>, // must specify at least 3 points  
<PointInfo>...)
```

<TextInfo>:

```
"Text:=", Array(<real>, // x position  
<real>, // y position  
<real>, // angle, in radians  
<Justification>,  
<bool>, // is plotter font  
<string>, // font name  
<int>, // color  
<string>) // text string
```

<Justification>:

```
<int> // 0 : left top  
// 1 : left base  
// 2 : left bottom
```

```
// 3 : center top
// 4 : center base
// 5 : center bottom
// 6 : right top
// 7 : right base
// 8 : right bottom
```

<ImageInfo>:

```
"Image:=", Array(<RectInfo>,
<ImageData>,
<bool>) // is mirrored
```

<ImageData>:

```
<string>, // file path
<int>, // 0 : use the file path and link to it
// 1 : ignore file path and use next parameter
<string> // text data, only present if preceding int is 1
```

<ListOfComponentNames>:

```
<string>, <string> ...
```

```
// The list may be empty. When not empty, each string that is listed is a component
```

// that references the model to be edited. Prior to editing, a clone of the model is
// made, and the components that are listed are modified so that they now refer to
// the clone.

VB Example:

```
Dim nam
oModelManager.EditWithComps_
("Nexxim Circuit Elements\Distributed\Distributed:bendo", _
Array("NAME:bendo new name", _
"ModTime:=", 1152722165, _
"Library:=", "Nexxim Circuit Elements\Distributed\Distributed", _
"LibLocation:=", "SysLibrary", _
Array("NAME:PinDef", _
"Pin:=", Array( "n1", _
-0.00254, _
0.00254, _
0, _
"N", _
0, _
0, _
```

```
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _  
Array("NAME:PinDef", _  
"Pin:=", Array( "n2", _  
0.00254, _  
-0.00254, _  
1.5708, _  
"N", _  
0, _  
0, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _  
Array("NAME:Graphics", _
```

```
"Polygon:=", Array( 0, 0, 12566272, 0, 0.00381, 0, .00127, 0.00127, _
0.00127, 0.00127, 0, 0.00381, 0, 0.00381, _
0.002032, 0.00127, 0.00381), _
"Line:=", Array(0, 1, 12566272, -0.00254, 0.00254, 0, .00254), _
"Line:=", Array(0, 1, 12566272, 0.00254, -0.00254, .00254, 0)), _
Array("NAME:PropDisplayMap", _
"W:=", Array(3, _
5, _
0, _
"Text:=", Array( 2.1684E-019, _
0.00504119, _
0, _
1, _
5, _
false, _
"Arial", _
0, _
"W=***")))), _
rray("MY_COMP")
```

Export [model manager]

Use: Exports model(s) to a library

Command: Tools > Edit Configured Libraries > Models > Export to Library

Syntax: Export Array("NAME:<LibraryName>",

 <ModelName>,

 <ModelName>...),

 <LibraryLocation>

Return Value: None

Parameters: <LibraryName>:

 <string> // name of the library

 <ModelName>:

 <string> // [composite name](#) of model to export

 <LibraryLocation>:

 <string> // location of the library in <LibraryName>

 // One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oModelManager.Export _  
Array("NAME Nexxim Circuit Elements\Distributed\Distributed:bendo:mylib", _ "myModel"), "Per-  
sonalLib"
```

Python Syntax	Export (["NAME:<LibraryName>", <ComboName>, <ComboName>...], <LibraryLocation>)
Python Example	<code>oModelManager.Export (["NAME:mylib", "model1", "model2"])</code>

GetData [model manager]

Use: Gets data that describes the definition.

Command: None

Syntax: GetData(<DefinitionName>)

Return Value: <DefinitionData> This is an array of data for the definition.

Parameters: <DefinitionName>:

<string> // [composite name](#) of the definition to edit

VB Example:

```
Dim ModelData
```

```
ModelData = oModelManager.GetData("Nexxim Circuit Elements\Hspice:nexx_bjt_npn")
```

Note:

GetData allows the user to access definition information, make modifications, and then use the Edit or EditWithComps script commands to save the modified definition. Accordingly, for each type of definition, the array data returned to GetData should be the same array information that is supplied to the [Edit](#) or [EditWithComps](#) commands.

Python Syntax	GetData(<ComboName>)
Python Example	<pre>dataArray = oModelManager.GetData ("mylib:mymodel")</pre>

GetNames [model manager]

Use: Returns the names of the models (used and unused) in a design. The following script command, **IsUsed**, can then be used to separate used and unused models.

Command: None

Syntax: GetNames()

Return Value: An array of strings

Parameters: None

VB Example:

```
Dim modelNames
```

```
modelNames = oModelManager.GetNames()
```

Python Syntax	GetNames()
Python Example	<pre>modelnames = oModelManager.GetNames()</pre>

IsUsed [model manager]

Use: Used to determine if a model is used in the design.

Command: None

Syntax: IsUsed(<ModelName>)

Return Value: <Boolean> // true if the specified model is used in the design

Parameters: <ModelName>:

<string>

VB Example:

```
Dim isUsed
isUsed = oModelManager.IsUsed("MyModel")
```

Python Syntax	IsUsed(<ComboName>)
Python Example	isused = oModelManager.IsUsed ("mylib:mymodel")

Remove [model manager]

Use: Removes a model from a library
Command: Tools > Edit Configured Libraries > Models > Remove Model
Syntax: Remove <ModelName>,

<IsProjectModel>,
<LibraryName>,
<LibraryLocation>

Return Value: None

Parameters: <ModelName>:

<string> // [composite name](#) of the model to remove

<IsProjectModel>:

<bool>

<LibraryName>:

<string> // name of the library

<LibraryLocation>:

<string> // location of the library in <LibraryName>

// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oModelManager.Remove "Nexxim Circuit Elements\Distributed\Distributed:bendo", true, "Project"
```

Python Syntax	Remove (<ModelName>, <IsLocal>, <LibraryName>, <LibraryLocation>)
Python Example	<pre>oModelManager.Export (["NAME:mylib", "model1", "model2"])</pre>

RemoveUnused [model manager]

Use: Removes models that are not used in the design.

Command: **Project->Remove Unused Definitions** is similar but operates slightly different and does not record script commands.

Syntax: RemoveUnused()

Return Value: <bool> True if one or more models are removed.

Parameters: None

VB Example:

```
Dim removedDefs
removedDefs = oModelManager.RemoveUnused()
```

Note:

The order of calls to RemoveUnused is significant. As a result, removing definitions in an unordered fashion may cause other models in dependent definitions to be rendered unusable.

Also, the model and footprint of an unused component are not unusable until after the component itself is removed using the Component Manager Remove script.

Python Syntax	RemoveUnused()
Python Example	thereWereExtras = oModelManager.RemoveUnused()

Network Data Explorer Manager Script Commands

The network data Explorer (NDE) Manager provides access to certain NDE data.

```
Set oDefinitionManager = oProject.GetDefinitionManager()
Set oNdExplorerManager = oDefinitionManager.GetManager("NdExplorer")
```

For NDE scripts accessed via the ndExplorer tool, see: [Network Data Explorer Script Commands](#).

The topics for this section include:

[ExportFullWaveSpice](#)

[ExportNetworkData](#)

[ExportNMFData](#)

ExportFullWaveSpice

Use: Export FullWaveSpice data in a format of your choice.

Command: File > Export MacroModel > Broadband (SYZ, FWS....)

Syntax: ExportFullWaveSpice

```
"DesignName", // Design name. Can be left blank, if loading solution from a file.  
true/false, // true - solution loaded from file, false- loaded from design  
"Name", // If loading from design this is the solution name, else this is the  
           // full path of the file from which the solution is loaded  
"variation", // Pick a particular variation. Leave blank if no variation.  
Array("NAME:Frequencies"), // Optional; if none defined all frequencies are used  
Array("NAME:SpiceData", // Spice export options object  
"SpiceType:=", "SSS", // SpiceType can be "PSpice", "HSpice", "Spectre", "SSS",  
           // "Simplorer", "TouchStone1.0", "TouchStone2.0"  
"EnforcePassivity:=", false, // Enforce Passivity true/false  
"EnforceCausality:=", false, // Enforce Causality true/false  
"UseCommonGround:=", false, // Use common ground true/false  
"FittingError:=", 0.5, // Fitting error  
"MaxPoles:=", 400, // Maximum Order
```

```

"PassivityType:=", "ConvexOptimization", // Passivity Type can be "ConvexOptimization",
                                     // "PassivityByPerturbation", or "IteratedFittingOfPV"
"ColumnFittingType:=", "Column", // Column FittingType can be "Column", "Entry", "Matrix"
"SSFittingType:=", "TWA", // SS Fitting Type can be "TWA", "IterativeRational"
"RelativeErrorToleranc:=", false, // Relative error tolerance true/false
"TouchstoneFormat:=", "MA", // Touchstone Format "MA", "RI", "DB"
"TouchstoneUnits:=", "Hz", // Touchstone Units "Hz", "KHz", "MHz", "MHz"
"TouchStonePrecision:=", 8, // Touchstone precision
"ExportDirectory:=", "C:/Examples/LNA/", // Directory to export to
"ExportSpiceFileName:=", "Linckt_HBTest_2.sss", // Spice export file
"FullwaveSpiceFileName:=", "Linckt_HBTest.sss", // FWS file
"CreateNPortModel:=", true // Create a model based on the exported file true/false
)

```

ExportNetworkData

Exports matrix solution data to a file.

UI Access	N/A		
Parameters	Name	Type	Description
	<DesignVariationKey>	String	Design variation key. Pass empty string for the current nominal variation.
	<SolnSelectionArray>	Array	Array of selected solutions.

		Array(<SolnSelector>, <SolnSelector>, ...) If more than one array entry, this indicates a combined Interpolating sweep.
<SolnSelector>	String	Solution setup name and solution name, separated by a colon.
<FileFormat>	Integer	File format value. 2 : Tab delimited spreadsheet format (.tab) 3 : Touchstone (.sNp) 4 : CitiFile (.cit) 7 : Matlab (.m) 8 : Terminal Z0 spreadsheet
<OutFile>	String	Full path to the file to write out.
<FreqsArray>	Array	The frequencies to export. The <FreqsArray> argument contains a vector (e.g. "1GHz", "2GHz", ...) to use, or "all". To export all frequencies, use Array("all"). If no frequencies are specified, all frequencies are used.
<DoRenorm>	Boolean	Specifies whether to renormalize the data before export.
<RenormImped>	Double	Real impedance value in ohms, for renormalization. Required in syntax, but ignored if DoRenorm is false.
<DataType>	String	Optional. Type: "S", "Y", or "Z". The matrix to export.
<pass>	Integer	Optional. The pass to export. This is ignored if the sourceName is a frequency sweep. Leaving out this value or specifying -1 gets all passes.
<ComplexFormat>	Integer	Optional. Type: "0", "1", or "2" The format to use for the exported data. 0 = Magnitude/Phase. 1 = Real/Imaginary. 2 = db/Phase.

	<i><Precision></i>	Integer	Optional. Touchstone number of digits precision. Default if not specified is 15.
	<i><UseExportFreqs></i>	Boolean	Specifies whether to use export frequencies.
	<i><IncludeGammaComments></i>	Boolean	Specifies whether to include Gamma and Impedance comments.
	<i><SupportNonStdExport></i>	Boolean	Specifies whether to support non-standard Touchstone extensions for mixed reference impedance.
Return Value	None.		

Python Syntax	ExportNetworkData(<i><DesignVariationKey></i> , <i><SolnSelectionArray></i> , <i><SolnSelector></i> , <i><FileFormat></i> , <i><OutFile></i> , <i><FreqsArray></i> , <i><DoRenorm></i> , <i><RenormImped></i> , [Optional <i><DataType></i>], [Optional <i><pass></i>], [Optional <i><ComplexFormat></i>], [Optional <i><Precision></i>], [Optional <i><UseExportFreqs></i>], [Optional <i><IncludeGammaComments></i>], [Optional <i><SupportNonStdExport></i>])
Python Example	

VB Syntax	ExportNetworkData <i><DesignVariationKey></i> , <i><SolnSelectionArray></i> , <i><SolnSelector></i> , <i><FileFormat></i> , <i><OutFile></i> , <i><FreqsArray></i> , <i><DoRenorm></i> , <i><RenormImped></i> , [Optional <i><DataType></i>], [Optional <i><pass></i>], [Optional <i><ComplexFormat></i>], [Optional <i><Precision></i>], [Optional <i><UseExportFreqs></i>], [Optional <i><IncludeGammaComments></i>], [Optional <i><SupportNonStdExport></i>]
VB Example	<pre>oModule.ExportNetworkData "width='2in'", _ Array("Setup1:Sweep1"), 2, "c:\mydir\out.tab", _ Array("all"), false, 0 oModule.ExportNetworkData "width='2in'", _ Array("Setup1:Sweep1", "Setup1:Sweep2"), 3, _</pre>

	<pre>"c:\mydir\out.s2p", Array(1.0e9, 1.5e9, 2.0e9), _ true, 50.0</pre>

ExportNMFData

Symbol Manager Script Commands

The symbol manager provides access to symbols in a project. The manager object is accessed via the definition manager.

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

```
Set oSymbolManager = oDefinitionManager.GetManager("Symbol")
```

The symbol manager script commands are listed below.

[Add](#)

[BringToFront](#)

[Edit](#)

[EditWithComps](#)

[Export](#)

[GetData](#)

[GetNames](#)

[IsUsed](#)

[Remove](#)

[RemoveUnused](#)

Add [symbol manager]

Use: Add a symbol

Command: Tools > Edit Configured Libraries > Symbols > Add Symbol

Syntax: Add Array("NAME:<SymbolName>",
"ModTime:=", <ModifiedTimeInfo>,
"Library:=", "", // Library name
"LibLocation:=", "Project", // Project Location
<PinDefInfo>,
<PinDefInfo>,... // optional, to define pins
<GraphicsDataInfo>, // optional, to define graphics
<PropDisplayMapInfo>)) // optional, to define property displays

Return Value: <string>

// [composite name](#) of the symbol.
// If the name requested conflicts with the name of an existing
// symbol, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.

Parameters: <SymbolName>:

<string> // [simple name](#) of the symbol being added

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed

since 00:00 hours, Jan 1, 1970 UTC from the system clock.

```
<PinDefInfo>:  
Array("NAME:PinDef",  
"Pin:=", Array (<string>, // pin name  
<real>, // x location  
<real>, // y location  
<real>, // angle in radians  
<PinType>,  
<real>, // line width  
<real>, // line length  
<bool>, // mirrored  
<int>, // color  
<bool>, // true if visible, false if not  
<string>, // hidden net name  
<OptionalPinInfo>, // optional info  
<PropDisplayMapInfo>)) // optional  
  
<PinType>:  
<string> // "N" : normal pin  
// "I" : input pin
```

// "O" : output pin

<OptionalPinInfo>:

// Specify both or neither

<bool>, // true if name is to be shown

<bool>, // true if number is to be shown

<PropDisplayMapInfo>:

Array("NAME:PropDisplayMap",

<PropDisplayInfo>,

<PropDisplayInfo>,...)

<PropDisplayInfo>:

<NameString>, Array(<DisplayTypeInfo>,

<DisplayLocationInfo>,

<int>, // optional, level number

<TextInfo>)

<NameString>:

<string> // PropertyName:=, where PropertyName is the name of

// the property to be displayed

<DisplayTypeInfo>:

<int> // 0 : No display

// 1 : Display name only

// 2 : Display value only

// 3 : Display both name and value

// 4 : Display evaluated value only

// 5 : Display both name and evaluated value

<DisplayLocationInfo>:

<int> // 0 : Left

// 1 : Top

// 2 : Right

// 3 : Bottom

// 4 : Center

// 5 : Custom placement

<GraphicsDataInfo>:

Array("NAME:Graphics",

// one or more of the following

```
<RectInfo>,  
<CircleInfo>,  
<ArcInfo>,  
<LineInfo>,  
<PolygonInfo>,  
<TextInfo>,  
<ImageInfo>)
```

<RectInfo>:

```
"Rect:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // angle, in radians  
<real>, // x position of center  
<real>, // y position of center  
<real>, // width  
< real>) // height
```

<CircleInfo>:

```
"Circle:=", Array(<real>, // line width
```

<int>, // fill pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>) // radius

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>, // radius

<real>, // start angle, in radians

<end>) // end angle, in radians

<LineInfo>:

"Line:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<PointInfo>, // must specify at least 2 points

<PointInfo>...)

<PointInfo>:

<real>, // x position

<real> // y position

<PolygonInfo>:

"Polygon:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<PointInfo>, // must specify at least 3 points

<PointInfo>...)

<TextInfo>:

"Text:=", Array(<real>, // x position

<real>, // y position

<real>, // angle, in radians

<Justification>,

<bool>, // is plotter font

<string>, // font name

<int>, // color

<string>) // text string

<Justification>:

<int> // 0 : left top

// 1 : left base

// 2 : left bottom

// 3 : center top

// 4 : center base

// 5 : center bottom

// 6 : right top

// 7 : right base

// 8 : right bottom

<ImageInfo>:

"Image:=", Array(<RectInfo>,

<ImageData>,

<bool>) // is mirrored

<ImageData>:

<string>, // file path

<int>, // 0 : use the file path and link to it

// 1 : ignore file path and use next parameter

<string> // text data, only present if preceding int is 1

VB Example:

```
oSymbolManager.Add Array("NAME:MySymbol", _  
"ModTime:=", 1070989137, _  
"Library:=", "", _  
"LibLocation:=", "Project", _  
Array("NAME:PinDef", _  
"Pin:=", Array ("newpin0", _  
0.00254, _ 0, _  
0, _  
"N", _  
0, _  
0.00254, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _
```

```
Array("NAME:PinDef", _  
  "Pin:=", Array ("newpin1", _  
    -0.00254, _  
    0, _  
    3.14159265358979, _  
    "N", _  
    0, _  
    0.00254, _  
    false, _  
    0, _  
    true, _  
    "", _  
    false, _  
    false)),  
Array("NAME:Graphics", _  
  "Rect:=", Array(0, _  
    0, _  
    12566272, _  
    0, _  
    4.33680868994202e-019, _  
    -0.000635, _
```

```
0.00508, _  
0.002794), _  
"Circle:=", Array(0, _  
0, _  
12566272, _  
0.000127, _  
0.000635, _  
0.000635)))
```

BringToFront [symbol manager]

Use: Changes the drawing for the symbol so that the specified objects are drawn on top of other overlapping objects.

Command: Draw > Bring To Front

Syntax: BringToFront Array("NAME:Selections", "Selections:=", Array (<Object>, <Object>, ...))

Return Value: None

Parameters: <Object>

<string> // object to bring to the front

VB Example:

```
oDefinitionEditor.BringToFront Array("NAME:Selections", "Selections:=", Array( "SchObj@10"))
```

Edit [deprecated]

Deprecated command — please use [EditWithComps](#).

EditWithComps [symbol manager]

Use: Edit an existing symbol.

Command: None

Syntax: EditWithComps <SymbolName>,
 Array("NAME:<NewSymbolName>",
 "ModTime:=", <ModifiedTimeInfo>,
 "Library:=", <string>, // Library name
 "LibLocation:=", <string>, // Project Location
 <PinDefInfo>,
 <PinDefInfo>,... // optional, to define pins
 <GraphicsDataInfo>, // optional, to define graphics
 <PropDisplayMapInfo>), // optional, to define property displays
 Array(<ListOfComponentNames>) // Component names

Return Value: <string>

 // [composite name](#) of the symbol.
 // If the name requested conflicts with the name of an existing
 // symbol, the requested name is altered to be unique.
 // The name returned reflects any change made to be unique.

Parameters: <SymbolName>:

<string> // [composite name](#) of the symbol being edited

<NewSymbolName>:

<string> // new [simple name](#) for the symbol

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<PinDefInfo>:

Array("NAME:PinDef",
"Pin:=", Array (<string>, // pin name
<real>, // x location
<real>, // y location
<real>, // angle in radians
<PinType>,
<real>, // line width
<real>, // line length
<bool>, // mirrored

```
<int>, // color  
<bool>, // true if visible, false if not  
<string>, // hidden net name  
<OptionalPinInfo>, // optional info  
<PropDisplayMapInfo>)) // optional
```

```
<PinType>:  
<string> // "N" : normal pin  
// "I" : input pin  
// "O" : output pin
```

```
<OptionalPinInfo>:  
// Specify both or neither  
<bool>, // true if name is to be shown  
<bool>, // true if number is to be shown
```

```
<PropDisplayMapInfo>:  
Array("NAME:PropDisplayMap",  
<PropDisplayInfo>,  
<PropDisplayInfo>,...)
```

<PropDisplayInfo>:

<NameString>, Array(<DisplayTypeInfo>,
<DisplayLocationInfo>,
<int>, // optional, level number
<TextInfo>)

<NameString>:

<string> // PropertyName:=, where PropertyName is the name of
// the property to be displayed

<DisplayTypeInfo>:

<int> // 0 : No display
// 1 : Display name only
// 2 : Display value only
// 3 : Display both name and value
// 4: Display evaluated value only
// 5: Display both name and evaluated value

<DisplayLocationInfo>:

<int> // 0 : Left


```
// 1 : Top  
// 2 : Right  
// 3 : Bottom  
// 4 : Center  
// 5 : Custom placement
```

```
<GraphicsDataInfo>:  
Array("NAME:Graphics",  
// one or more of the following  
<RectInfo>,  
<CircleInfo>,  
<ArcInfo>,  
<LineInfo>,  
<PolygonInfo>,  
<TextInfo>,  
<ImageInfo>)
```

```
<RectInfo>:  
"Rect:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color
```

<real>, // angle, in radians

<real>, // x position of center

<real>, // y position of center

<real>, // width

< real>) // height

<CircleInfo>:

"Circle:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>) // radius

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>, // radius

<real>, // start angle, in radians

<end>) // end angle, in radians

<LineInfo>:

"Line:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<PointInfo>, // must specify at least 2 points

<PointInfo>...)

<PointInfo>:

<real>, // x position

<real> // y position

<PolygonInfo>:

"Polygon:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<PointInfo>, // must specify at least 3 points

<PointInfo>...)

<TextInfo>:

"Text:=", Array(<real>, // x position

<real>, // y position

<real>, // angle, in radians

<Justification>,

<bool>, // is plotter font

<string>, // font name

<int>, // color

<string>) // text string

<Justification>:

<int> // 0 : left top

// 1 : left base

// 2 : left bottom

// 3 : center top

// 4 : center base

// 5 : center bottom

// 6 : right top

// 7 : right base

// 8 : right bottom

<ImageInfo>:

"Image:=", Array(<RectInfo>,

<ImageData>,

<bool>) // is mirrored

<ImageData>:

<string>, // file path

<int>, // 0 : use the file path and link to it

// 1 : ignore file path and use next parameter

<string> // text data, only present if preceding int is 1

<ListOfComponentNames>:

<string>, <string> ...

// The list may be empty. When not empty, each string that is listed is a component

// that references the symbol to be edited. Prior to editing, a clone of the symbol is

// made, and the components that are listed are modified so that they now refer to

// the clone.

VB Example:

Dim nam

```
oSymbolManager.EditWithComps _  
("Nexxim Circuit Elements\Distributed\Distributed:bendo", _  
Array("NAME:bendo new name", _  
"ModTime:=", 1152722165, _  
"Library:=", "Nexxim Circuit Elements\Distributed\Distributed", _  
"LibLocation:=", "SysLibrary", _  
Array("NAME:PinDef", _  
"Pin:=", Array( "n1", _  
-0.00254, _  
0.00254, _  
0, _  
"N", _  
0, _  
0, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _
```

```
Array("NAME:PinDef", _  
  "Pin:=", Array( "n2", _  
    0.00254, _  
    -0.00254, _  
    1.5708, _  
    "N", _  
    0, _  
    0, _  
    false, _  
    0, _  
    true, _  
    "", _  
    false, _  
    false)), _  
Array("NAME:Graphics", _  
  "Polygon:=", Array( 0, 0, 12566272, 0, 0.00381, 0, .00127, 0.00127, _  
    0.00127, 0.00127, 0, 0.00381, 0, 0.00381, _  
    0.002032, 0.00127, 0.00381)), _  
  "Line:=", Array(0, 1, 12566272, -0.00254, 0.00254, 0, .00254), _  
  "Line:=", Array(0, 1, 12566272, 0.00254, -0.00254, .00254, 0)), _  
Array("NAME:PropDisplayMap", _
```

```
"W:=", Array(3, _  
5, _  
0, _  
"Text:=", Array( 2.1684E-019, _  
0.00504119, _  
0, _  
1, _  
5, _  
false, _  
"Arial", _  
0, _  
"W=***")))), _  
Array("MY_COMP")
```

Export [symbol manager]

Use: Exports symbol(s) to a library

Command: Tools > Edit Configured Libraries > Symbols > Export to Library

Syntax: Export Array("NAME:<LibraryName>",

<SymbolName>,

<SymbolName>...),

<LibraryLocation>

Return Value: None

Parameters: <LibraryName>:

<string> // name of the library

<SymbolName>:

<string> // [composite name](#) of symbol to export

<LibraryLocation>:

<string> // location of the library in <LibraryName>

// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oSymbolManager.Export _  
Array("NAME Nexxim Circuit Elements\Distributed\Distributed:bendo:mylib", _ "mySymbol"), "Per-  
sonalLib"
```

GetData [symbol manager]

Use: Gets data that describes the definition.

Command: None

Syntax: GetData(<DefinitionName>)

Return Value: <DefinitionData> This is an array of data for the definition.

Parameters: <DefinitionName>:

<string> // [composite name](#) of the definition to edit

VB Example:

```
Dim symbolData  
symbolData = oSymbolManager.GetData("Nexxim Circuit Elements\Hspice:nexx_bjt_npn")
```

Note:

GetData allows the user to access definition information, make modifications, and then use the Edit or EditWithComps script commands to save the modified definition. Accordingly, for each type of definition, the array data returned to GetData should be the same array information that is supplied to the [Edit](#) or [EditWithComps](#) commands.

GetNames [symbol manager]

Use: Returns the names of the symbols (used and unused) in a design. The following script command, **IsUsed**, can then be used to separate used and unused symbols.

Command: None

Syntax: GetNames()

Return Value: An array of strings

Parameters: None

VB Example:

```
Dim symbolNames  
symbolNames = oSymbolManager.GetNames()
```

IsUsed [symbol manager]

Use: Used to determine if a symbol is used in the design.

Command: None

Syntax: IsUsed(<SymbolName>)

Return Value: <Boolean> // true if the specified symbol is used in the design

Parameters: <SymbolName>:

<string>

VB Example: Dim isUsed

isUsed = oSymbolManager.IsUsed("MySymbol")

Remove [symbol manager]

Use: Removes a symbol from a library

Command: Tools > Edit Configured Libraries > Symbols > Remove Symbol

Syntax: Remove <SymbolName>,

<IsProjectSymbol>,

<LibraryName>,

<LibraryLocation>

Return Value: None

Parameters: <SymbolName>:

<string> // [composite name](#) of the symbol to remove

<IsProjectSymbol>:

<bool>

<LibraryName>:

<string> // name of the library

<LibraryLocation>:

<string> // location of the library in <LibraryName>

// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oSymbolManager.Remove "Nexxim Circuit Elements\Distributed\Distributed:bendo", true, "Project"
```

RemoveUnused [symbol manager]

Use: Removes symbols that are not used in the design.

Command: **Project->Remove Unused Definitions** is similar but operates slightly different and does not record script commands.

Syntax: RemoveUnused()

Return Value: <bool> True if one or more symbols are removed.

Parameters: None

VB Example:

```
Dim removedDefs  
removedDefs = oSymbolManager.RemoveUnused()
```

Note:

The order of calls to RemoveUnused is significant. As a result, removing definitions in an unordered fashion may cause other symbols in dependent definitions to be rendered unusable.

Also, the symbol and footprint of an unused component are not unusable until after the component itself is removed using the Component Manager Remove script.

Add [footprint manager]

Use: Add a footprint

Command: Tools > Edit Configured Libraries > Footprints > Add Footprint

Syntax: Add Array("NAME:<FootprintName>,"
 "ModTime:=", <ModifiedOnInfo>,"
 "Library:=", "",
 "LibLocation:=", "Project",
 "OkayToMirror:=", <bool>,"
 "DefUnits:=", <UnitType>,"
 Array(NAME:Lyrs",
 "Layer:=", <LayerArray>,"
 "Layer:=", <LayerArray>...,
 "SLayer:=", <StackupLayerArray>,"
 "SLayer:=", <StackupLayerArray>...),

```
"ActLyr:=", <string>, // name of active layer
"Tol:=", <ToleranceArray> // optional
<PrimitivesInfo>, // optional
<PinsInfo>, // optional
<ViasInfo>, // optional
<EdgeportsInfo>, // optional
<ComponentPropertyInfo>,
<ScriptInfo>) // optional, specified for scripted footprints
```

Return Value: <string>

```
// composite name of the footprint.
// If the name requested conflicts with the name of an existing
// footprint, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.
```

Parameters: <FootprintName>:

```
<string> // simple name of footprint to create
```

```
<ModifiedOnInfo>:
```

An integer that corresponds to the number of seconds that have elapsed
since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<UnitType>:

<string> // default length units to use if units are not specified in other
// parameters

<LayerArray>:

Array("N:=", <string>, // layer name
"ID:=", <int> ,
"T:=", <LayerTypeInfo>, // layer type
"TB:=", <TopBottomInfo>,
"Col:=", <int>, // optional - color
"Pat:=", <int>, // optional - fill pattern
"Vis:=", <bool>, // optional - are objects on layer visible
"Sel:=", <bool>, // optional - are objects on layer selectable
"L:=", <bool>) // optional
// are objects on layer locked (can't be edited)

<LayerTypeInfo>:

<string> // one of: signal, dielectric, metalized signal, assembly, silkscreen, soldermask, solderpaste, glue, or user

<TopBottomInfo>:

<string> // one of: top, neither, bottom, or template

<StackupLayerArray>:

```
Array(<LayerArray>,  
"Elev:=", <ElevationInfo>,  
"SubL:=", Array("Th:=", <Dimension>,  
"LElev:=", <Dimension>,  
"R:=", <Dimension>,  
"M:=", <MaterialInfo>))
```

<ElevationInfo>:

```
<string> // "top" - snap to top  
// "mid" - snap to middle  
// "bot" - snap to bottom  
// "edit" - manual edit  
// "none"
```

<Dimension>:

<string> // real number, may include units

<MaterialInfo>:

<string> // name of the layer material

<ToleranceArray>:

Array(<real>, // distance tolerance

<real>, // angle tolerance (radians)

<real>) // dimensionless tolerance

<PrimitivesInfo>:

Array("NAME:Prims",

// one or more of the following

<RectInfo>,

<CircleInfo>,

<ArcInfo>,

<LineInfo>,

<PolygonInfo>,

<TextInfo>,

<ImageInfo>)

<RectInfo>:

"Rect:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<real>, // angle, in radians

<real>, // x position of center

<real>, // y position of center

<real>, // width

< real>) // height

<CircleInfo>:

"Circle:=", Array(<real>, // line width

<int>, // fill pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center

< real>) // radius

<ArcInfo>:

"Arc:=", Array(<real>, // line width

<int>, // line pattern

<int>, // color

<real>, // x position of center

<real>, // y position of center
< real>, // radius
<real>, // start angle, in radians
<end>) // end angle, in radians

<LineInfo>:
"Line:=", Array(<real>, // line width
<int>, // line pattern
<int>, // color
<PointInfo>, // must specify at least 2 points
<PointInfo>...)
<PointInfo>:
<real>, // x position
<real> // y position

<PolygonInfo>:
"Polygon:=", Array(<real>, // line width
<int>, // fill pattern
<int>, // color
<PointInfo>, // must specify at least 3 points
<PointInfo>...)

```
<TextInfo>:  
"Text:=", Array(<real>, // x position  
<real>, // y position  
<real>, // angle, in radians  
<Justification>,  
<bool>, // is plotter font  
<string>, // font name  
<int>, // color  
<string>) // text string  
<Justification>: <int>  
// 0 : left top  
// 1 : left base  
// 2 : left bottom  
// 3 : center top  
// 4 : center base  
// 5 : center bottom  
// 6 : right top  
// 7 : right base  
// 8 : right bottom
```

<ImageInfo>:

"Image:=", Array(<RectInfo>,

<ImageData>,

<bool>) // is mirrored

<ImageData>:

<string>, // file path

<int>, // 0 : use the file path and link to it

// 1 : ignore file path and use next parameter

<string> // text data, only present if preceding int is 1

<PinsInfo>:

Array("NAME:Pins",

"P:=", <PinArray>,

"P:=", <PinArray>,...)

<PinArray>:

Array("Port:=", Array("Id:=", <int>,

"Clr:=", <real>, // optional - clearance

"N:=", <string>), // pin name

```
"Pos:=", Array("x:=", <Location>, // padstack (x,y) position  
"y:=", <Location>),  
"VRt:=", <Angle>, // optional - rotation  
"HD:=", <Size>, // optional - hole diameter  
<PadstackImplementationInfo>)
```

<Location>:

<string> specifying real number and units (may use variables)

<Angle>:

<string> specifying angle with a real number and units

<Size>:

<string> specifying size with a real number and units

<PadstackImplementationInfo>:

If another with the same implementation has already been specified:

```
"Ref:=", <int> // id of the other
```

If not:

```
"Ref:=", <string>, // name
```

```
"Frm:=", <int>, // id of highest layer
"To:=", <int>, // id of lowest layer
<LayerPlacementInfo>,
"Man:=", <int>, // optional, 1 if manually placed, 0 if not (default)
"Use:=", Array(<PadUseInfo>, <PadUseInfo>...) // array may be empty
```

```
<LayerPlacementInfo>:
"Lyr:=", Array("Mrg:=", <int>, // optional,
// 1 if all layers have been merged (default)
// 0 if layer are not merged
"Flp:=", <int>, // optional,
// 1 if placed bottom up
// 0 if not (default)
"Map:=", <ParentToLocalLayerInfo>)
```

```
<ParentToLocalLayerInfo>:
Array("U:=", <DirectionOfUniqueness>,
"F:=", Array(<int>, <int>, ...), // forward mapping
// -1 is not mapped
"B:=", Array(<int>, <int>, ...)) // backward mapping
// -1 is not mapped
```

<PadUseInfo>:

```
"Pad:=", Array("Lid:=", <int>, // layer id
"T:=", <string>, // type : "connected", "thermal",
// "no_pad, "not_connected",
// or "not_connected_thermal"
"Man:=", <int>) // optional, 1 if manually placed
// 0 if not (default)
```

<DirectionOfUniqueness>:

```
<string> // one of "forward", "backward", or "two ways"
```

<ViasInfo>:

```
Array("NAME:Vias", <VialInfo>, <VialInfo>...)
```

<VialInfo>:

```
"V:=", Array("Id:=", <int>,
"N:=", <string>, // name
"Pos:=", Array("x:=", <Location>, // via (x,y) position
"y:=", <Location>),
```


"VRt:=", <Angle>, // optional - rotation

<ImplementationInfo>

<EdgeportsInfo>:

Array("NAME:EPorts", <EdgePortArray>, <EdgePortArray>...)

<EdgePortArray>:

Array("NAME:EP",

"LP:=", Array("Id:=", <int>, // port id

"N:=", <string>), // port name

"Eo:=", Array(<edge description>, <edge description>,...))

<edge description> for primitive edges

"et:=", "pe", "pr:=", <id>, "ei:=", <edge#>

<id>: integer that is the primitive id

<edge#>: integer that is the edge number on the primitive

<edge description> for via edges

```
"et:=", "pse", "layer:=", <layer id>, "se:=", <via id>,  
"sx:=", <start X location>, "sy:=", <start Y location>, "ex:=", <end X location>,  
"ey:=", <end Y location>, "h:=", <arc height>, "rad:=", <radians>
```

<via id>: an integer that is the id of the via to use

<layer id>: an integer that is the id of the layer of the pad of the via to use

<start X location>

<start Y Location>:

doubles that are the X, Y location of the start point of the edge arc <end X location>

<end Y Location>:

doubles that are the X, Y location of the end point of the edge arc

<arc height>: double giving the height of the edge arc (0 for a straight edge)

<radians>: double giving the arc size in radians (0 for a straight edge)

<ComponentPropertyInfo>:

Array("NAME:CProps",

"VariableProp:=", <VariableInfo>,

```
"VariableProp:=", <VariableInfo>,  
...)
```

<VariableInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>) // value: number, variable, or expression
```

<ScriptInfo>:

```
Array("NAME:script",  
"language:=", <string>, // one of "javascript" or "vbscript"  
"UsesScript:=", true,  
"script:=", <string>) // contents of script
```

VB Example:

```
oFootprintManager.Add (Array("NAME:BCL", _  
"ModTime:=", 1023388445, _  
"Library:=", "", _  
"LibLocation:=", "Project", _
```

```
"OkayToMirror:=", false, _  
"DefUnits:=", "mm", _  
Array("NAME:Lyrs", _  
"Layer:=", Array("N:=", "Measures", _  
"ID:=", 8, _  
"T:=", "measures", _  
"Col:=", 4144959, _  
"Pat:=", 1), _  
"Layer:=", Array("N:=", "Rats", _  
"ID:=", 1, _  
"T:=", "rat", _  
"Col:=", 16711680, _  
"Pat:=", 1), _  
"Layer:=", Array("N:=", "Errors", _  
"ID:=", 2, _  
"T:=", "error", _  
"Col:=", 255, _  
"Pat:=", 1, _  
"L:=", true), _  
"Layer:=", Array("N:=", "Symbols", _
```

```
"ID:=", 3, _  
"T:=", "symbol", _  
"Col:=", 8323199, _  
"Pat:=", 4), _  
"Layer:=", Array("N:=", "Assembly", _  
"ID:=", 4, _  
"T:=", "assembly", _  
"TB:=", "top", _  
"Col:=", 16711680, _  
Pat:=", 3), _  
"Layer:=", Array("N:=", "Silkscreen", _  
"ID:=", 5, _  
"T:=", "silkscreen", _  
TB:=", "top", _  
"Col:=", 8454143, _  
"Pat:=", 6), _  
SLayer:=", Array("Layer:=", Array("N:=", "Cover", _  
"ID:=", 12, _  
"T:=", "metalizedsignal", _  
"Col:=", 32639, _  
Pat:=", 7), _
```

```
"Elev:=", "mid", _  
"SubL:=", Array("Th:=", "0mm", _  
"LElev:=", "118.110236220472mil", _  
"R:=", "0mm", _  
"Mat:=", "")), _  
"SLayer:=", Array("Layer:=", Array("N:=", "Dielec3", _  
"ID:=", 11, _  
"T:=", "dielectric", _  
"Col:=", 8323199, _  
"Pat:=", 6), _  
"Elev:=", "none", _  
"SubL:=", Array("Th:=", "1mm", _  
"LElev:=", "78.740157480315mil", _  
"R:=", "0mm", _  
"Mat:=", "")), _  
"SLayer:=", Array("Layer:=", Array("N:=", "Trace2", _  
"ID:=", 10, _  
"T:=", "signal", _  
"Col:=", 8355584, _  
"Pat:=", 5), _
```

```
"Elev:=", "mid", _
"SubL:=", Array("Th:=", "0mm", _
"LElev:=", "78.740157480315mil", _
"R:=", "0mm", _
"Mat:=", "")), _
"SLayer:=", Array("Layer:=", Array("N:=", "Dielec2", _
"ID:=", 6, _
"T:=", "dielectric", _
"Col:=", 8323072, _
"Pat:=", 4), _
"Elev:=", "none", _
"SubL:=", Array("Th:=", "1mm", _
"LElev:=", "39.3700787401575mil", _
"R:=", "0mm", _
"Mat:=", "")), _
"SLayer:=", Array("Layer:=", Array("N:=", "Tracel", _
"ID:=", 0, _
"T:=", "signal", _
"Col:=", 32512, _
"Pat:=", 3), _
"Elev:=", "mid", _
```

```
"SubL:=", Array("Th:=", "0mm", _  
"LElev:=", "39.3700787401575mil", _  
"R:=", "0mm", _  
"Mat:=", "")), _  
"SLayer:=", Array("Layer:=", Array("N:=", "Dielec1", _  
"ID:=", 7, _  
"T:=", "dielectric", _  
"Col:=", 127, _  
"Pat:=", 7), _  
"Elev:=", "none", _  
"SubL:=", Array("Th:=", "1mm", _  
"LElev:=", "0mil", _  
"R:=", "0mil", _  
"Mat:=", "")), _  
"SLayer:=", Array("Layer:=", Array("N:=", "Ground", _  
"ID:=", 9, _  
"T:=", "metalizedsignal", _  
"Col:=", 4144959, _  
"Pat:=", 6), _  
"Elev:=", "mid", _
```



```
"SubL:=", Array("Th:=", "0mil", _  
"LElev:=", "0mil", _  
"R:=", "0mil", _  
"Mat:=", ""))), _  
"ActLyr:=", "Trace2", _  
Array("NAME:Prims",  
"rect:=", Array("Id:=", 10003, _  
"Lyr:=", 10, _  
"N:=", "rect101", _  
"x:=", "0mm", _  
"y:=", "0mm", _  
"w:=", "P", _  
"h:=", "W", _  
"Vds:=", Array()), _  
"rect:=", Array("Id:=", 10103, _  
"Lyr:=", 0, _  
"N:=", "rect103", _  
"x:=", "0mm", _  
"y:=", "0mm", _  
"w:=", "P", _  
"h:=", "W", _
```

```
"Vds:=", Array()), _  
Array("NAME:Pins",  
"P:=", Array("Port:=", Array("Id:=", 3, "N:=", "n1"), _  
"Pos:=", Array("x:=", "-P/2", "y:=", "0mm"), _  
"VRt:=", "180deg", _  
"Ref:=", "NoPad SMT East", _  
"Frm:=", 12, _  
"To:=", 9, _  
"Lyr:=", Array("Map:=", Array("U:=", "forward", _  
"F:=", Array(-1, -1, -1, -1, _  
-1, -1, -1, -1, _  
-1, -1, 0, -1, -1), _  
"B:=", Array(10))), _  
"Use:=", Array()), _  
"P:=", Array("Port:=", Array("Id:=", 4, "N:=", "n4"), _  
"Pos:=", Array("x:=", "P/2", "y:=", "0mm"), _  
"HD:=", "1mm", _  
"Ref:=", 3), _  
"P:=", Array("Port:=", Array("Id:=", 5, "N:=", "n2"), _  
"Pos:=", Array("x:=", "-P/2", "y:=", "0mm"), _
```

```
"VRt:=", "180deg", _  
"Ref:=", "NoPad SMT East", _  
"Frm:=", 12, _  
"To:=", 9, _  
"Lyr:=", Array("Map:=", Array("U:=", "forward", _  
"F:=", Array(0, -1, -1, -1, -1, _  
-1, -1, -1, -1, _  
-1, -1, -1, -1), _  
"B:=", Array(0))), _  
"Use:=", Array()), _  
"P:=", Array("Port:=", Array("Id:=", 6, "N:=", "n3"), _  
"Pos:=", Array("x:=", "P/2", "y:=", "0mm"), _  
"HD:=", "1mm", _  
"Ref:=", 5)), _  
Array("NAME:Nets"), _  
Array("NAME:CProps",  
"VariableProp:=", Array("W", _  
"UD", _  
"", _  
"1.5mm"), _  
"VariableProp:=", Array("P", _
```

```
"UD", _  
" ", _  
10mm" ) ) )
```

Edit [footprint manager]

Deprecated command — please use [EditWithComps](#).

EditWithComps [footprint manager]

Use: Edit an existing footprint.

Command: None

Syntax: EditWithComps <FootprintName>,
 Array("NAME:<NewFootprintName>,"
 "ModTime:=", <ModifiedOnInfo>,"
 "Library:=", "",
 "LibLocation:=", "Project",
 "OkayToMirror:=", <bool>,"
 "DefUnits:=", <UnitType>,"
 Array(NAME:Lyrs",
 "Layer:=", <LayerArray>,"
 "Layer:=", <LayerArray>...,
 "SLayer:=", <StackupLayerArray>,"

```
"SLayer:=", <StackupLayerArray>...),  
"ActLyr:=", <string>, // name of active layer  
"Tol:=", <ToleranceArray> // optional  
<PrimitivesInfo>, // optional  
<PinsInfo>, // optional  
<ViasInfo>, // optional  
<EdgeportsInfo>, // optional  
<ComponentPropertyInfo>,  
<ScriptInfo>, // optional, specified for scripted footprints  
Array(<ListOfComponentNames>) // Component names
```

Return Value: <string>

```
// composite name of the footprint.  
// If the name requested conflicts with the name of an existing  
// footprint, the requested name is altered to be unique.  
// The name returned reflects any change made to be unique.
```

Parameters: <FootprintName>:

```
<string> // composite name of the footprint being edited
```

```
<NewFootprintName>:
```

```
<string> // new simple name for the footprint
```

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<UnitType>:

<string> // default length units to use if units are not specified in other
// parameters

<LayerArray>:

Array("N:=", <string>, // layer name
"ID:=", <int> ,
"T:=", <LayerTypeInfo>, // layer type
"TB:=", <TopBottomInfo>,
"Col:=", <int>, // optional - color
"Pat:=", <int>, // optional - fill pattern
"Vis:=", <bool>, // optional - are objects on layer visible
"Sel:=", <bool>, // optional - are objects on layer selectable
"L:=", <bool>) // optional
// are objects on layer locked (can't be edited)

<LayerTypeInfo>:

<string> // one of: signal, dielectric, metalized signal, assembly, silkscreen, soldermask, solderpaste, glue, or user

<TopBottomInfo>:

<string> // one of: top, neither, bottom, or template

<StackupLayerArray>:

```
Array(<LayerArray>,  
"Elev:=", <ElevationInfo>,  
"SubL:=", Array("Th:=", <Dimension>,  
"LElev:=", <Dimension>,  
"R:=", <Dimension>,  
"M:=", <MaterialInfo>))
```

<ElevationInfo>:

```
<string> // "top" - snap to top  
// "mid" - snap to middle  
// "bot" - snap to bottom  
// "edit" - manual edit  
// "none"
```

<Dimension>:

<string> // real number, may include units

<MaterialInfo>:

<string> // name of the layer material

<ToleranceArray>:

Array(<real>, // distance tolerance

<real>, // angle tolerance (radians)

<real>) // dimensionless tolerance

<PrimitivesInfo>:

Array("NAME:Prims",

// one or more of the following

<RectInfo>,

<CircleInfo>,

<ArcInfo>,

<LineInfo>,

<PolygonInfo>,


```
<TextInfo>,  
<ImageInfo>)
```

<RectInfo>:

```
"Rect:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // angle, in radians  
<real>, // x position of center  
<real>, // y position of center  
<real>, // width  
< real>) // height
```

<CircleInfo>:

```
"Circle:=", Array(<real>, // line width  
<int>, // fill pattern  
<int>, // color  
<real>, // x position of center  
<real>, // y position of center  
< real>) // radius
```

<ArcInfo>:

```
"Arc:=", Array(<real>, // line width
<int>, // line pattern
<int>, // color
<real>, // x position of center
<real>, // y position of center
< real>, // radius
<real>, // start angle, in radians
<end>) // end angle, in radians
```

<LineInfo>:

```
"Line:=", Array(<real>, // line width
<int>, // line pattern
<int>, // color
<PointInfo>, // must specify at least 2 points
<PointInfo>...)
<PointInfo>:
<real>, // x position
<real> // y position
```

<PolygonInfo>:

```
"Polygon:=", Array(<real>, // line width
<int>, // fill pattern
<int>, // color
<PointInfo>, // must specify at least 3 points
<PointInfo>...)
```

<TextInfo>:

```
"Text:=", Array(<real>, // x position
<real>, // y position
<real>, // angle, in radians
<Justification>,
<bool>, // is plotter font
<string>, // font name
<int>, // color
<string>) // text string
<Justification>: <int>
// 0 : left top
// 1 : left base
// 2 : left bottom
// 3 : center top
```

```
// 4 : center base  
// 5 : center bottom  
// 6 : right top  
// 7 : right base  
// 8 : right bottom
```

<ImageInfo>:

```
"Image:=", Array(<RectInfo>,  
<ImageData>,  
<bool>) // is mirrored
```

<ImageData>:

```
<string>, // file path  
<int>, // 0 : use the file path and link to it  
// 1 : ignore file path and use next parameter  
<string> // text data, only present if preceding int is 1
```

<PinsInfo>:

```
Array("NAME:Pins",  
"P:=", <PinArray>,
```

"P:=", <PinArray>,...)

<PinArray>:

Array("Port:=", Array("Id:=", <int>,
"Clr:=", <real>, // optional - clearance
"N:=", <string>), // pin name
"Pos:=", Array("x:=", <Location>, // padstack (x,y) position
"y:=", <Location>),
"VRt:=", <Angle>, // optional - rotation
"HD:=", <Size>, // optional - hole diameter
<PadstackImplementationInfo>)

<Location>:

<string> specifying real number and units (may use variables)

<Angle>:

<string> specifying angle with a real number and units

<Size>:

<string> specifying size with a real number and units

<PadstackImplementationInfo>:

If another with the same implementation has already been specified:

"Ref:=", <int> // id of the other

If not:

"Ref:=", <string>, // name

"Frm:=", <int>, // id of highest layer

"To:=", <int>, // id of lowest layer

<LayerPlacementInfo>,

"Man:=", <int>, // optional, 1 if manually placed, 0 if not (default)

"Use:=", Array(<PadUseInfo>, <PadUseInfo>...) // array may be empty

<LayerPlacementInfo>:

"Lyr:=", Array("Mrg:=", <int>, // optional,

// 1 if all layers have been merged (default)

// 0 if layer are not merged

"Flp:=", <int>, // optional,

// 1 if placed bottom up

// 0 if not (default)

"Map:=", <ParentToLocalLayerInfo>)

<ParentToLocalLayerInfo>:

```
Array("U:=", <DirectionOfUniqueness>,  
"F:=", Array(<int>, <int>, ...), // forward mapping  
// -1 is not mapped  
"B:=", Array(<int>, <int>, ...)) // backward mapping  
// -1 is not mapped
```

<PadUseInfo>:

```
"Pad:=", Array("Lid:=", <int>, // layer id  
"T:=", <string>, // type : "connected", "thermal",  
// "no_pad", "not_connected",  
// or "not_connected_thermal"  
"Man:=", <int>) // optional, 1 if manually placed  
// 0 if not (default)
```

<DirectionOfUniqueness>:

```
<string> // one of "forward", "backward", or "two ways"
```

<ViasInfo>:

```
Array("NAME:Vias", <VialInfo>, <VialInfo>...)
```

<VialInfo>:

```
"V:=", Array("Id:=", <int>,
"N:=", <string>, // name
"Pos:=", Array("x:=", <Location>, // via (x,y) position
"y:=", <Location>),
"VRt:=", <Angle>, // optional - rotation
<ImplementationInfo>
```

<EdgeportsInfo>:

```
Array("NAME:EPorts", <EdgePortArray>, <EdgePortArray>...)
```

<EdgePortArray>

```
Array("NAME:EP",
"LP:=", Array("Id:=", <int>, // port id
"N:=", <string>), // port name
"Eo:=", Array(<edge description>, <edge description>,...))
```

<edgedescription> for primitive edges

```
"et:=", "pe", "pr:=", <id>, "ei:=", <edge#>
```


<id>: integer that is the primitive id

<edge#>: integer that is the edge number on the primitive

<edge description> for via edges

"et:=", "pse", "sel:=", <"via">, "layer:=", <layer id>,

"sx:=", <start X location>, "sy:=", <start Y location>, "ex:=", <end X location>, "ey:=", <end Y location>, "h:=", <arc height>,
"rad:=", <radians>

<"via">: text that is the name of the via to use

<layer id>: an integer that is the id of the layer of the pad of the via to use

<start X location>, <start Y Location>:

doubles that are the X, Y location of the start point of the edge arc

<end X location>, <end Y Location>:

doubles that are the X, Y location of the end point of the edge arc

<arc height>: double giving the height of the edge arc (0 for a straight edge)

<radians>: double giving the arc size in radians (0 for a straight edge)

<ComponentPropertyInfo>:

```
Array("NAME:CProps",  
"VariableProp:=", <VariableInfo>,  
"VariableProp:=", <VariableInfo>,  
...)
```

<VariableInfo>:

```
Array(<string>, // name  
<FlagLetters>,  
<string>, // description  
"CB:=", <string>, // optional - script for call back  
<string>) // value: number, variable, or expression
```

<ScriptInfo>:

```
Array("NAME:script",  
"language:=", <string>, // one of "javascript" or "vbscript"  
"UsesScript:=", true,  
"script:=", <string>) // contents of script
```

<ListOfComponentNames>:

<string>,<string> ...

// The list may be empty. When not empty, each string that is listed is a component
 // that references the footprint to be edited. Prior to editing, a clone of the footprint is
 // made, and the components that are listed are modified so that they now refer to
 // the clone.

VB Example:

```
Dim nam
oFootprintManager.EditWithComps _
("Nexxim Circuit Elements\Distributed\Distributed:bendo", _
Array("NAME:bendo new name", _
"ModTime:=", 1152722165, _
"Library:=", "Nexxim Circuit Elements\Distributed\Distributed", _
"LibLocation:=", "SysLibrary", _
Array("NAME:PinDef", _
"Pin:=", Array( "n1", _
-0.00254, _
0.00254, _
0, _
"N", _
0, _
```

```
0, _  
false, _  
0, _  
true, _  
"", _  
false, _  
false)), _  
Array("NAME:PinDef", _  
"Pin:=", Array( "n2", _  
0.00254, _  
-0.00254, _  
1.5708, _  
"N", _  
0, _  
0, _  
false, _  
0, _  
true, _  
"", _  
false, _
```

```

false)), _
Array("NAME:Graphics", _
"Polygon=", Array( 0, 0, 12566272, 0, 0.00381, 0, .00127, 0.00127, _
0.00127, 0.00127, 0, 0.00381, 0, 0.00381, _
0.002032, 0.00127, 0.00381), _
"Line=", Array(0, 1, 12566272, -0.00254, 0.00254, 0, .00254), _
"Line=", Array(0, 1, 12566272, 0.00254, -0.00254, .00254, 0)), _
Array("NAME:PropDisplayMap", _
"W=", Array(3, _
5, _
0, _
"Text=", Array( 2.1684E-019, _
0.00504119, _
0, _
1, _
5, _
false, _
"Arial", _
0, _
"W=***")))), _
Array("MY_COMP")

```

Export [footprint manager]

Use: Export a footprint to a library

Command: Tools > Edit Configured Libraries > Footprints > Export to Library

Syntax: Export Array("NAME:<LibraryName>",
 <FootprintName>,
 <FootprintName>...),
 <LibraryLocation>

Return Value: None

Parameters: <LibraryName>:

<string> // name of the library

<FootprintName>:

<string> // [composite name](#) of footprint to export

<LibraryLocation>:

<string> // location of the library in <LibraryName>

// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oFootprintManager.Export Array("NAME:mylib", "Distributed Footprints:BPAD"), "PersonalLib"
```

GetData [footprint manager]

FootprintGetData

Use: Gets data that describes the definition.

Command: None

Syntax: GetData(<DefinitionName>)

Return Value: <DefinitionData> This is an array of data for the definition.

Parameters: <DefinitionName>:

<string> // [composite name](#) of the definition to edit

VB Example:

```
Dim footprintData
```

```
footprintData = oFootprintManager.GetData("Nexxim Circuit Elements\Distributed\Nexxim_Footprints:MCPL13_Nexx")
```

Note:

GetData allows the user to access definition information, make modifications, and then use the Edit or EditWithComps script commands to save the modified definition. Accordingly, for each type of definition, the array data returned to GetData should be the same array information that is supplied to the [Edit](#) or [EditWithComps](#) commands.

GetNames [footprint manager]

Use: Returns the names of the footprints (used and unused) in a design. The following script command, **IsUsed**, can then be used to separate used and unused footprints.

Command: None

Syntax: GetNames()

Return Value: An array of strings

Parameters: None

VB Example:

```
Dim footprintNames  
footprintNames = oFootprintManager.GetNames()
```

IsUsed [footprint manager]

Use: Used to determine if a footprint is used in the design.

Command: None

Syntax: IsUsed(<FootprintName>)

Return Value: <Boolean> // true if the specified footprint is used in the design

Parameters: <FootprintName>:

<string>

VB Example:

```
Dim isUsed  
isUsed = oFootprintManager.IsUsed("MyFootprint")
```

Remove [footprint manager]

Use: Removes a footprint from a library

Command: Tools > Edit Configured Libraries > Footprints > Remove Footprint

Syntax: Remove <FootprintName>,

<IsProjectFootprint>,

<LibraryName>,
<LibraryLocation>

Return Value: None

Parameters: <FootprintName>:

<string> // [composite name](#) of the footprint to remove

<IsProjectFootprint>:
<bool>

<LibraryName>:
<string> // name of the library

<LibraryLocation>:
<string> // location of the library in <LibraryName>
// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oFootprintManager.Remove "BPAD", true, "Distributed Footprints", "Project"  
oFootprintManager.Remove "BPAD", false, "MyLib", "PersonalLib"
```

RemoveUnused [footprint manager]

Use: Removes footprints that are not used in the design.

Command: **Project->Remove Unused Definitions** is similar but operates slightly different and does not record script commands.

Syntax: RemoveUnused()

Return Value: <bool> True if one or more footprints are removed.

Parameters: None

VB Example:

```
Dim removedDefs  
removedDefs = oFootprintManager.RemoveUnused()
```

Note:

The order of calls to RemoveUnused is significant. As a result, removing definitions in an unordered fashion may cause other footprints in dependent definitions to be rendered unusable.

Also, the symbol and footprint of an unused component are not unusable until after the component itself is removed using the Component ManagerRemove script.

Add [padstack manager]

Use: Add a padstack

Command: Tools > Edit Configured Libraries > Padstacks > Add Padstack

Syntax: Add Array("NAME:<PadstackName>",
"ModTime:=", <ModifiedOnInfo>,
"Library:=", "", // name of the library

```
"LibLocation:=", "Project", // location of the named library
```

```
Array("NAME:psd",
```

```
"nam:=", <PadstackName>,
```

```
"lib:=", "", // name of the library
```

```
"mat:=", "", // hole plating material
```

```
"plt:=", "0", // percent of hole's radius filled by plating
```

```
Array("NAME:pds",
```

```
<LayerGeometryArray>,
```

```
<LayerGeometryArray....>),
```

```
"hle:=", <PadInfo>
```

```
"hRg:=", <HoleRange>,
```

```
"sbsh:=", <SolderballShape>,
```

```
"sbpl:=", <SolderballPlacement>,
```

```
"sbr:=", <string>, // solderball diameter, real with units
```

```
"sb2:=", <string>, // solderball mid diameter, real with units
```

```
"sbn:=", <string>), // name of solderball material
```

```
"ppl:=", <PadPortLayerArray>)
```

Return Value: [simple name](#) of the added padstack

// If the name requested conflicts with the name of an existing

// padstack, the requested name is altered to be unique.

// The name returned reflects any change made to be unique.

Parameters: <PadstackName>:

<string> // [simple name](#) of padstack to create

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<LayerGeometryArray>:

Array("Name:lgm",

"lay:=", <string>, // definition layer name

"id:=", <int>, // definition layer id

"pad:=", <PadInfo>, // pad

"ant:=", <PadInfo>, // antipad

"thm:=", <PadInfo>, // thermal pad

"X:=", <string>, // pad x connection, real with units

"Y:=", <string>, // pad y connection, real with units

"dir:=", <DirectionString>) // pad connection direction

<PadInfo>:

```
Array("shp:=", <PadShape>,  
"Szs:=", <DimensionArray>,  
"X:=", <string>, // x offset, real with units  
"Y:=", <string>, // y offset, real with units  
"R:=", <string>) // rotation, real with units
```

<PadShape>:

<string> one of these choices

"No" // no pad

"Cir" // Circle

"Sq" // Square

"Rct" // Rectangle

"Ov" // Oval

"Blt" // Bullet

"Ply" // Polygons

"R45" // Round 45 thermal

"R90" // Round 90 thermal

"S45" // Square 45 thermal

"S90" // Square 90 thermal

<DimensionArray>:

Array(<string>, ...) // each string is a real with units for one of the dimensions of the shape

<DirectionString>:

<string> one of these choices

"No" // no direction

"Any" // any direction

"0" // 0 degrees

"45" // 45 degrees

"90" // 90 degrees

"135" // 135 degrees

"180" // 180 degrees

"225" // 225 degrees

"270" // 270 degrees

"315" // 315 degrees

<HoleRange>:

<string> one of these choices

"Thr" // through all layout layers

"Beg" // from upper pad layer to lowest layout layer

"End" // from upper layout layer to lowest pad layer

"UTL" // from upper pad layer to lowest pad layer

<SolderballShape>:

<string> one of these choices

"None" // no solderball

"Cyl" // cylinder solderball

"Sph" // spheroid solderball

<SolderballPlacement>:

<string> one of these choices

"abv" // above padstack

"blw" // below padstack

<PadPortLayerArray>:

Array(<int>, <int>,....) where each int is a layer id

VB Example:

```
oPadstackManager.Add Array("NAME:Circle - through3", "ModTime:=", 1235765635, "Library:=", _
"", "LibLocation:=", "Project", Array("NAME:psd", "nam:=", "Circle - through3", "lib:=", _
"", "mat:=", "", "plt:=", "0", Array("NAME:pds", Array("NAME:lgm", "lay:=", "Top Signal",
"id:=", _
0, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2.5mm"), "X:=", "0mm", "Y:=", _
"0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array("3.5mm"), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _
```

```
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"SignalA", "id:=", 1, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array(_  
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array(),  
"X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"SignalB", "id:=", 2, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array(_  
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array("shp:=", "No", "Szs:=", Array(  
( ), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"Ground", "id:=", 3, "pad:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "R90", "Szs:=", Array(_  
"3mm", "0.75mm", "1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _  
"0mm", "dir:=", "Any"), Array("NAME:lgm", "lay:=", "Bottom signal", "id:=", 5, "pad:=", Array  
("shp:=", _  
"Cir", "Szs:=", Array("1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=",  
_
```



```
"Cir", "Szs:=", Array("2mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array
("shp:=", _
"No", "Szs:=", Array(), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _
"0mm", "dir:=", "Any")), "hle:=", Array("shp:=", "Cir", "Szs:=", Array("1.5mm"), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0deg"), "hRg:=", "End", "sbsh:=", "Sph", "sbpl:=", _
"abv", "sbr:=", "750um", "sb2:=", "1200um", "1200um", "sbn:=", "solder"), "ppl:=", Array( _
0, 1, 2, 3, 5))
```

Edit [padstack manager]

Use: Edit an existing padstack.

Command: Tools > Edit Configured Libraries > Padstacks > Edit Padstack

Syntax: Edit <PadstackName>,

```
Array("NAME:<NewPadstackName>",
"ModTime:=", <ModifiedOnInfo>,
"Library:=", "", // name of the library
"LibLocation:=", "Project", // location of the named library
Array("NAME:psd",
"nam:= ", <PadstackName>,
"lib:=", "", // name of the library
"mat:=", "", // hole plating material
"plt:=", "0", // percent of hole's radius filled by plating
Array("NAME:pds",
<LayerGeometryArray>,
```

```
<LayerGeometryArray....),  
"hle:=", <PadInfo>  
"hRg:=", <HoleRange>,  
"sbsh:=", <SolderballShape>,  
"sbpl:=", <SolderballPlacement>,  
"sbr:=", <string>, // solderball diameter, real with units  
"sb2:=", <string>, // solderball mid diameter, real with units  
"sbn:=", <string>), // name of solderball material  
"ppl:=", <PadPortLayerArray>)
```

Return Value: <string> // [composite name](#) of the padstack

// If the name requested conflicts with the name of an existing
// padstack, the requested name is altered to be unique.
// The name returned reflects any change made to be unique.

Parameters: <PadstackName>:

<string> // [composite name](#) of padstack to edit

<NewPadstackName>:

<string> // new [simple name](#) for padstack

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<LayerGeometryArray>:

```
Array("Name:lgm",  
"lay:=", <string>, // definition layer name  
"id:=", <int>, // definition layer id  
"pad:=", <PadInfo>, // pad  
"ant:=", <PadInfo>, // antipad  
"thm:=", <PadInfo>, // thermal pad  
"X:=", <string>, // pad x connection, real with units  
"Y:=", <string>, // pad y connection, real with units  
"dir:=", <DirectionString>) // pad connection direction
```

<PadInfo>:

```
Array("shp:=", <PadShape>,  
"Szs:=", <DimensionArray>,  
"X:=", <string>, // x offset, real with units  
"Y:=", <string>, // y offset, real with units  
"R:=", <string>) // rotation, real with units
```

<PadShape>:

<string> one of these choices

"No" // no pad

"Cir" // Circle

"Sq" // Square

"Rct" // Rectangle

"Ov" // Oval

"Blt" // Bullet

"Ply" // Polygons

"R45" // Round 45 thermal

"R90" // Round 90 thermal

"S45" // Square 45 thermal

"S90" // Square 90 thermal

<DimensionArray>:

Array(<string>, ...) // each string is a real with units for one of the

// dimensions of the shape

<DirectionString>:

<string> one of these choices

"No" // no direction

"Any" // any direction

"0" // 0 degrees

"45" // 45 degrees

"90" // 90 degrees

"135" // 135 degrees

"180" // 180 degrees

"225" // 225 degrees

"270" // 270 degrees

"315" // 315 degrees

<HoleRange>:

<string> one of these choices

"Thr" // through all layout layers

"Beg" // from upper pad layer to lowest layout layer

"End" // from upper layout layer to lowest pad layer

"UTL" // from upper pad layer to lowest pad layer

<SolderballShape>:

<string> one of these choices

"None" // no solderball

"Cyl" // cylinder solderball

"Sph" // spheroid solderball

<SolderballPlacement>:

<string> one of these choices

"abv" // above padstack

"blw" // below padstack

<PadPortLayerArray>:

Array(<int>, <int>,....) where each int is a layer id

VB Example:

```
oPadstackManager.Edit "Circle - through1", Array("NAME:Circle - through1", "ModTime:=", _  
1235765635, "Library:=", "", "LibLocation:=", "Project", Array("NAME:psd", "nam:=", _  
"Circle - through1", "lib:=", "", "mat:=", "", "plt:=", "0", Array("NAME:pds", Array("NAME:lgm",  
"lay:=", _  
"Top Signal", "id:=", 0, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2.5mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array( _  
"3.5mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array  
( ), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _
```

```

"SignalA", "id:=", 1, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array( _
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array(),
"X:=", _
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",
"lay:=", _
"SignalB", "id:=", 2, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array( _
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array("shp:=", "No", "Szs:=", Array
(), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",
"lay:=", _
"Ground", "id:=", 3, "pad:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _
"0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "R90", "Szs:=", Array( _
"3mm", "0.75mm", "1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _
"0mm", "dir:=", "Any"), Array("NAME:lgm", "lay:=", "Bottom signal", "id:=", 5, "pad:=", Array
("shp:=", _
"Cir", "Szs:=", Array("1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array
("shp:=", _
"Cir", "Szs:=", Array("2mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array
("shp:=", _
"No", "Szs:=", Array(), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _
"0mm", "dir:=", "Any")), "hle:=", Array("shp:=", "Cir", "Szs:=", Array("1.5mm"), "X:=", _

```

```
"0mm", "Y:=", "0mm", "R:=", "0deg"), "hRg:=", "End", "sbsh:=", "Sph", "sbpl:=", _  
"abv", "sbr:=", "750um", "sb2:=", "1200um", "1200um", "sbn:=", "solder"), "ppl:=", Array( _  
0, 1, 2, 3, 5))
```

EditWithComps [padstack manager]

Use: Edit an existing padstack.

Command: None

Syntax: EditWithComps <PadstackName>,
 Array("NAME:<NewPadstackName>",
 "ModTime:=", <ModifiedOnInfo>,
 "Library:=", "", // name of the library
 "LibLocation:=", "Project", // location of the named library
 Array("NAME:psd",
 "nam:= ", <PadstackName>,
 "lib:=", "", // name of the library
 "mat:=", "", // hole plating material
 "plt:=", "0", // percent of hole's radius filled by plating
 Array("NAME:pds",
 <LayerGeometryArray>,
 <LayerGeometryArray....>),
 "hle:=", <PadInfo>


```
"hRg:=", <HoleRange>,  
"sbsh:=", <SolderballShape>,  
"sbpl:=", <SolderballPlacement>,  
"sbr:=", <string>, // solderball diameter, real with units  
"sb2:=", <string>, // solderball mid diameter, real with units  
"sbn:=", <string>), // name of solderball material  
"ppl:=", <PadPortLayerArray>,  
Array(<ListOfComponentNames>) // Component names
```

Return Value: <string>

```
// composite name of the padstack.  
// If the name requested conflicts with the name of an existing  
// padstack, the requested name is altered to be unique.  
// The name returned reflects any change made to be unique.
```

Parameters: <PadstackName>:

```
<string> // composite name of the padstack being edited
```

<NewPadstackName>:

```
<string> // new simple name for the padstack
```

<ModifiedOnInfo>:

An integer that corresponds to the number of seconds that have elapsed since 00:00 hours, Jan 1, 1970 UTC from the system clock.

<LayerGeometryArray>:

```
Array("Name:lgm",  
"lay:=", <string>, // definition layer name  
"id:=", <int>, // definition layer id  
"pad:=", <PadInfo>, // pad  
"ant:=", <PadInfo>, // antipad  
"thm:=", <PadInfo>, // thermal pad  
"X:=", <string>, // pad x connection, real with units  
"Y:=", <string>, // pad y connection, real with units  
"dir:=", <DirectionString>) // pad connection direction
```

<PadInfo>:

```
Array("shp:=", <PadShape>,  
"Szs:=", <DimensionArray>,  
"X:=", <string>, // x offset, real with units  
"Y:=", <string>, // y offset, real with units
```

"R:=", <string>) // rotation, real with units

<PadShape>:

<string> one of these choices

"No" // no pad

"Cir" // Circle

"Sq" // Square

"Rct" // Rectangle

"Ov" // Oval

"Blt" // Bullet

"Ply" // Polygons

"R45" // Round 45 thermal

"R90" // Round 90 thermal

"S45" // Square 45 thermal

"S90" // Square 90 thermal

<DimensionArray>:

Array(<string>, ...) // each string is a real with units for one of the

// dimensions of the shape

<DirectionString>:

<string> one of these choices

"No" // no direction

"Any" // any direction

"0" // 0 degrees

"45" // 45 degrees

"90" // 90 degrees

"135" // 135 degrees

"180" // 180 degrees

"225" // 225 degrees

"270" // 270 degrees

"315" // 315 degrees

<HoleRange>:

<string> one of these choices

"Thr" // through all layout layers

"Beg" // from upper pad layer to lowest layout layer

"End" // from upper layout layer to lowest pad layer

"UTL" // from upper pad layer to lowest pad layer

<SolderballShape>:

<string> one of these choices

"None" // no solderball

"Cyl" // cylinder solderball

"Sph" // spheroid solderball

<SolderballPlacement>:

<string> one of these choices

"abv" // above padstack

"blw" // below padstack

<PadPortLayerArray>:

Array(<int>, <int>,....) where each int is a layer id

<ListOfComponentNames>:

<string>, <string> ...

// The list may be empty. When not empty, each string that is listed is a component

// that references the padstack to be edited. Prior to editing, a clone of the padstack is

// made, and the components that are listed are modified so that they now refer to

// the clone.

VB Example:

```
oPadstackManager.EditWithComps "Circle - through1", Array("NAME:Circle - through1", "ModTime:=",  
_  
1235765635, "Library:=", "", "LibLocation:=", "Project", Array("NAME:psd", "nam:=", _
```

```
"Circle - through1", "lib:=", "", "mat:=", "", "plt:=", "0", Array("NAME:pds", Array("NAME:lgm",  
"lay:=", _  
"Top Signal", "id:=", 0, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2.5mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array(_  
"3.5mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array(  
( ), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"SignalA", "id:=", 1, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array(_  
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "No", "Szs:=", Array(),  
"X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"SignalB", "id:=", 2, "pad:=", Array("shp:=", "Cir", "Szs:=", Array("2mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "Cir", "Szs:=", Array(_  
"3mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array("shp:=", "No", "Szs:=", Array(  
( ), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", "0mm", "dir:=", "Any"), Array("NAME:lgm",  
"lay:=", _  
"Ground", "id:=", 3, "pad:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array("shp:=", "No", "Szs:=", Array(), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0"), "thm:=", Array("shp:=", "R90", "Szs:=", Array(_
```

```
"3mm", "0.75mm", "1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _  
"0mm", "dir:=", "Any"), Array("NAME:lgm", "lay:=", "Bottom signal", "id:=", 5, "pad:=", Array  
("shp:=", _  
"Cir", "Szs:=", Array("1mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "ant:=", Array  
("shp:=", _  
"Cir", "Szs:=", Array("2mm"), "X:=", "0mm", "Y:=", "0mm", "R:=", "0deg"), "thm:=", Array  
("shp:=", _  
"No", "Szs:=", Array(), "X:=", "0mm", "Y:=", "0mm", "R:=", "0"), "X:=", "0mm", "Y:=", _  
"0mm", "dir:=", "Any")), "hle:=", Array("shp:=", "Cir", "Szs:=", Array("1.5mm"), "X:=", _  
"0mm", "Y:=", "0mm", "R:=", "0deg"), "hRg:=", "End", "sbsh:=", "Sph", "sbpl:=", _  
"abv", "sbr:=", "750um", "sb2:=", "1200um", "1200um", "sbn:=", "solder"), "ppl:=", Array( _  
0, 1, 2, 3, 5), Array(""))
```

Export [padstack manager]

Use: Export a padstack to a library

Command: Tools > Edit Configured Libraries > Padstacks > Export to Library

Syntax: Export Array("NAME:<LibraryName>",

<PadstackName>,
<PadstackName>...),
<LibraryLocation>

Return Value: None

Parameters: <LibraryName>:

<string> // name of the library

<PadstackName>:

<string> // [simple name](#) of padstack to export

<LibraryLocation>:

<string> // location of the library in <LibraryName>

// One of "Project", "PersonalLib", or "UserLib"

VB Example:

```
oPadstackManager.Export Array("NAME:mylib", "myPadstack"), "PersonalLib"
```

GetData [padstack manager]

Use: Gets data that describes the definition.

Command: None

Syntax: GetData(<DefinitionName>)

Return Value: <DefinitionData> This is an array of data for the definition.

Parameters: <DefinitionName>:

<string> // [composite name](#) of the definition to edit

VB Example:

```
Dim padstackData
```

```
padstackData = oPadstackManager.GetData("NoPad SMT East")
```


Note:

GetData allows the user to access definition information, make modifications, and then use the Edit or EditWithComps script commands to save the modified definition. Accordingly, for each type of definition, the array data returned to GetData should be the same array information that is supplied to the [Edit](#) or [EditWithComps](#) commands.

GetNames [padstack manager]

Use: Returns the names of the padstack (used and unused) in a design. The following script command, **IsUsed**, can then be used to separate used and unused padstacks.

Command: None

Syntax: GetNames()

Return Value: An array of strings

Parameters: None

VB Example:

```
Dim padstackNames  
padstackNames = oPadstackManager.GetNames()
```

IsUsed [padstack manager]

Use: Used to determine if a component is used in the design.

Command: None

Syntax: IsUsed(<PadstackName>)

Return Value: <Boolean> // true if the specified padstack is used in the design

Parameters: <PadstackName>:

<string>

VB Example:

```
Dim isUsed  
  
isUsed = oPadstackManager.IsUsed("MyPadstack")
```

Remove [padstack manager]

Use: Removes a padstack from a library

Command: Tools > Edit Configured Libraries > Padstacks > Remove Padstacks

Syntax: Remove <PadstackName>,

 <IsProjectPadstack>,

 <LibraryName>,

 <LibraryLocation>

Return Value: None

Parameters: <PadstackName>:

 <string> // [simple name](#) of the padstack to remove

 <IsProjectPadstack>:

 <bool>

 <LibraryName>:

 <string> // name of the library

 <LibraryLocation>:

```
<string> // location of the library in <LibraryName>  
// One of "Project", "PersonalLib", or "UserLib"
```

VB Example:

```
oPadstackManager.Remove "Polygon SMT", true, "Padstacks", "Project"  
oPadstackManager.Remove "Polygon SMT", false, "MyLib", "PersonalLib"
```

RemoveUnused [padstack manager]

Use: Removes padstacks that are not used in the design.

Command: **Project->Remove Unused Definitions** is similar but operates slightly different and does not record script commands.

Syntax: RemoveUnused()

Return Value: <bool> True if one or more padstacks are removed.

Parameters: None

VB Example:

```
Dim removedDefs  
removedDefs = oPadstackManager.RemoveUnused()
```

Note:

The order of calls to RemoveUnused is significant. As a result, removing definitions in an unordered fashion may cause other padstacks in dependent definitions to be rendered unusable.

Script and Library Scripts

The definition manager provides access to materials in a project. The manager object is accessed via the definition manager.

```
Set oDefinitionManager = oProject.GetDefinitionManager()
```

The script and library script commands are listed below.

[AddScript](#)

[EditScript](#)

[ExportScript](#)

[RemoveScript](#)

[ModifyLibraries](#)

AddScript

Adds a script to the definition manager.

UI Access	N/A		
Parameters	Name	Type	Description
	<AddScriptArray>	Array	Structured array. Array("NAME:<string script name>", "ScriptLang:=", <string "vbscript" or "javascript"> "ScriptText:=, <string text of script>)
Return Value	None.		

Python Syntax	AddScript(<AddScriptArray>)
Python Example	<pre>oDefinitionManager.AddScript(["NAME:MyScript", "ScriptLang:=", "vbscript", "ScriptText:=", "MsgBox(\"HelloWorld\") "])</pre>

VB Syntax	AddScript <AddScriptArray>
VB Example	<pre>oDefinitionManager.AddScript Array("NAME:MyScript", "ScriptLang:=", "vbscript", "ScriptText:=", "MsgBox(""HelloWorld"")")</pre>

EditScript

Edits a script in the definition manager.

UI Access	N/A		
Parameters	Name	Type	Description
	<OriginalName>	String	Name of the script to be edited.
	<EditScriptArray>	Array	Structured array. <pre>Array("NAME:<string script name>", "ScriptLang:=", <string "vbscript" or "javascript"> "ScriptText:=, <string text of script>)</pre>
Return Value	None.		

Python Syntax	EditScript(<OriginalName>, <EditScriptArray>)
Python Example	<pre>oDefinitionManager.EditScript("MyScript", ["NAME:MyNewScript", "ScriptLang:=", "vbscript", "ScriptText:=", "MsgBox(\"HelloAgain\") "])</pre>

VB Syntax	EditScript <OriginalName> <EditScriptArray>
VB Example	<pre>oDefinitionManager.EditScript "MyScript" Array("NAME:MyNewScript", "ScriptLang:=", "vbscript", "ScriptText:=", "MsgBox("HelloAgain")")</pre>

ExportScript

Use: Export to Library in the script definition manager

Command: None

Syntax: ExportScript <ExportData>,<Library location>

Return Value: None

Parameters: <ExportData>

Array("NAME:<LibraryName>",<ScriptName>,<ScriptName>,...)

VB Example:

```
oProject.ExportComponent Array("NAME:mylib", "myscript"), "PersonalLib"
```

Python Syntax	ExportScript(<ExportData>,<Library location>)
Python Example	<pre>oProject.ExportComponent (["NAME:mylib", "myscript"], "PersonalLib")</pre>

RemoveScript

Use: Remove Script in the script definition manager

Command: None

Syntax: RemoveScript <ScriptName>,<IsProjectScript>, <LibraryName>,<LibraryLocation>

Return Value: None

Parameters: <ScriptName>

Type: <string>

<IsProjectScript>

Type: <bool>

<LibraryName>

Type: <string>

<LibraryLocation>

Type: <string>

VB Example:

```
oDefinitionManager.RemoveScript "myscript", true, "Local", "Project"
```

Python Syntax	RemoveScript (<ScriptName>,<IsProjectScript>, <LibraryName>,<LibraryLocation>)
Python Example	<pre>oDefinitionManager.RemoveScript("myscript", true, "Local", "Project")</pre>

ModifyLibraries

Use: Configure Libraries on the Tools menu

Command: None

Syntax: ModifyLibraries <DesignName>,Array(<ConfigLibArray>)

Return Value: None

```
Parameters:  <DesignName>  
Type: <string>  
<ConfigLibArray>  
Array("NAME:<LibraryType>,<ConfiguredLib>,<ConfiguredLib>,...) , ...  
<ConfiguredLib> // blank to leave unchanged  
<DefinitionType>  
Array("<libraryname >","<libraryname>","...)  
VB Example:  
oDefinitionManager.ModifyLibraries "MyCircuit", _  
Array("NAME:PersonalLib"), _  
Array("NAME:UserLib"), _  
Array("NAME:SystemLib", _
```



```
"Symbols:=", Array( "Circuit Elements", "Symbols", _  
"ParamExtraElements\PE_Symbols", _  
"Vendor Elements\Nonlinear"))
```

Python Syntax	ModifyLibraries (<DesignName>,[<ConfigLibArray>])
Python Example	<pre>oDefinitionManager.ModifyLibraries("MyCircuit", _ ["NAME:PersonalLib"], _ ["NAME:UserLib"], _ ["NAME:SystemLib", _ "Symbols:=", ["Circuit Elements", "Symbols", _ "ParamExtraElements\PE_Symbols", _ "Vendor Elements\Nonlinear"]])</pre>

This page intentionally
left blank.

8 - Property Script Commands

Property script commands allow you to navigate through all objects and properties in a project. You can get and set all properties for all objects in the Project tree with simple data types.

Property Object is the base class defined for all script objects that support the properties Get and Set.

`GetName ()`

- Returns the name of the object.

`GetChildTypes ()`

- An object may have different types of children. For example, a design may have variables, modules, and editors.
- Returns an array of text strings; may be empty if the children are not divided into different types.

`GetChildNames (<type>)`

- <type> – Child type name. By default, returns all children names for all types.
- Returns an array of immediate children names, belonging to a type if specified.

`GetChildObject (<objPath>)`

- <objPath> – A child object path; can contain multiple generations (for example, designObject/moduleObject/SetupObject).
- Returns a child property object if the object is found.

`GetPropNames (<bIncludeReadOnly>)`

- <bIncludeReadOnly> – Optional; defaults to true. True includes read-only properties; False excludes read-only properties.
- Returns an array of the object's property names.

`GetPropValue (<propertyPath>)`

- <propertyPath> – The property's path; may be a child object's path appended with a property name (for example, TeeModel/Offset/SIValue).

- Returns the property value if found. Otherwise causes script error.

`SetPropValue(<propertyPath>, <data>)`

- <propertyPath> – The property's path; may be a child object's path appended with a property name (for example, TeeModel/Offset/SIValue).
- <data> – New data; type depends on property type.
- Returns True if updated successfully; False if new data is invalid.

For a detailed summary of how Property script commands are used in a range of contexts, including Variable objects, see: [Object Script Property Function Summary](#). Additional examples for these commands are listed under [Project Objects](#), [Design Objects](#), [3D Modeler](#), [Optimetrics](#), Radiation Module and [Reporter](#).

Note:

Older property commands should be executed by the oProject object.

```
Set oProject = oDesktop.SetActiveProject("Project1")
```

```
oProject.CommandName <args>
```

Some of the topics covered in this chapter are as follows:

[Object Script Property Function Summary](#)

[Conventions Used in this Chapter](#)

[ChangeProperty](#)

[GetArrayVariables](#)

[GetProperties](#)

[GetPropertyValue](#)

[GetVariables](#)[GetVariableValue](#)[SetPropertyValue](#)[SetVariableValue](#)[Example Use of Record Script and Edit Properties](#)[Additional Property Scripting Example](#)

Object Script Property Function Summary

Object Path

The Object path can be used to navigate through objects and properties in an Ansys EM project.

- An Object path consisted of one or multiple Object-ID-Nodes separated by "/" .
- Object-ID-Node; may exist in the following forms:
 - A simple object name or property name.
 - Type[Name] for object; Tab[name] for property.
 - Name[attr1="v1", attr2 = "v2", ...]. When more than one child object have the same name, use attributes to specify the difference.
 - ArrayName[index]. For example, in an Optimetric setup with multiple calculations, "Calculation[0]" could be used to identify the first calculation.
 - Name beginning with '@' character denoted as a property name, when an object has a child and property with the same name.

Property Object

The Property Object is the base class defined for all script object that support property Get & Set.

- GetName()
 - Returns the name of the object.

- GetChildTypes()
 - An object may have different type of children. For example, a design may have variables, modules, and editors.
 - Returns array of text strings; may be empty if the children are *not* divided to different types.
- GetChildNames(<type>)
 - <type> – children type name; default returns all children names for all types.
 - Returns an array of immediate children names, belonging to the type if specified.
- GetChildObject(<objPath>)
 - <objPath> – A child object path. The path may include multiple generations, such as (designObject/moduleObj/SetupObject).
 - Returns a child property object if the object found.
- GetPropEvaluatedValue(<propName>)
 - Return the Evaluated-Value for Value-Property and Variable.
 - Return the Property-value as text string for other property types.
- GetPropSIValue(<propName>)
 - Return the SI-Value for Value-Property and Variable.
 - Return NAN for other property type if its value is cannot be converted to a double-floating point value.
- GetPropNames(<blIncludeReadOnly>);
 - <blIncludeReadOnly> – optional, default to true; True will include read-only properties, False will exclude read-only properties.
 - Returns an array of the object's property names.
- GetPropValue(<propertyPath>)
 - <propertyPath> – the property's full path. A property name or child object's path appended with a property name, like "TeeModel/Offset/SIValue"
 - Returns the property value if the property is found; otherwise causes script error.

- `SetPropValue(<propertyPath>, <data>)`
 - `<propertyPath>` – the property's full path. A property name or child object's path appended with a property name, like "TeeModel/Offset/Value"
 - `<data>` – new data, type is dependent on property type.
 - Returns True if property data is updated successfully; False if the new data is invalid.

Project Object

Project Object inherited all functions defined in the Property Object. But it doesn't have property, `GetPropValue()` & `SetPropValue()` function can be used to set its child object's property.

- `GetChildTypes()` always return ["Design", "Variable"].
- `GetChildNames(type)`
`GetChildNames()` & `GetChildName("Design")` will return all Design names of the project.
`GetChildNames("Variable")` return all project variable names.
- `GetChildObject(objPath)`

`oDesign = oProject("TeeModel")`

`oVariable = oProject.GetChildObject("VariableName")`

`oReport = oProject.GetChildObject("TeeModel/Results/S Parameter Plot 1")`
- `GetPropNames(bIncludeReadOnly)` always return empty array since the project has no property.
- `GetPropValue(propertyPath)`

`oProject.GetPropValue("TeeModel/offset")` //get the offset variable value in the TeeModel Design
`oProject.GetPropValue("TeeModel/Results/S Parameter Plot 1/Display Type")` // Get the report display type.
- `SetPropValue(propertyPath, newValue)`

`oProject.SetPropValue("TeeModel/offset", "2mm")` //Set the offset variable value to "2mm" in the TeeModel Design
`oProject.SetPropValue("TeeModel/Results/S Parameter Plot 1/Display Type", "Data Table")` // Set the report display type to data table.

Design Object

Design Object inherited all functions defined in the Property Object. But it doesn't have property, GetPropValue() & SetPropValue() function can be used to set its child object's property..

- GetChildTypes() always return ['Module', 'Editor', 'Variable'].
- GetChildNames(type)
GetChildNames() will return modules & editor child names.
GetChildNames("Variable") will return all variable names
GetChildNames("Module") will return all module names that support property-object-script like ['Optimetrics', 'RadField', 'Results']
GetChildNames("Editor") will return a 3D editor name for all 3D Designs
- GetChildObject()
oVariable = oDesign.GetChildObject("VariableName")
oReport = oDesign.GetChildObject("Results/S Parameter Plot 1")
oRptModule = oDesign.GetChildObject("ReportSetup")
- GetPropNames(bIncludeReadOnly) always return empty array since the design has no property.
- GetPropValue()
oDesign.GetPropValue("offset/SIValue") //get the offset variable SI value in the Design
oDesign.GetPropValue("offset/SIValue") //get the offset variable SI value in the Design
oDesign.GetPropValue("Results/S Parameter Plot 1/Display Type") // Get the report display type
- SetPropValue()
oDesign.SetPropValue("offset", "2mm") //Set the offset variable value to "2mm" in the Design
oDesign.SetPropValue("Results/S Parameter Plot 1/Display Type", "Data Table") // Set the report display type to data table.

3D Modeler Object

GetChild commands returns the appropriate properties for modeler objects. For 3D Components and UDMs, these commands do not return parts, coordinate systems, plans, as top-level modeler children.


```
oModeler = oDesktop.GetActiveProject().GetActiveDesign().GetChildObject("3D Modeler")
oModeler.GetChildNames()
oModeler.GetChildNames("ModelParts")
oModeler.GetChildNames("AllParts")
oModeler.GetChildNames("NonModelParts")
oModeler.GetChildNames("Planes")
oModeler.GetChildNames("CoordinateSystems")
```

Variable Object

Is a Property Object that has no child. It also provides quick function call to get/set its properties by adding functions with property name appended to Get_ & Set_ prefix. To find what functions it provided enter dir(oVar) the command window. It can be accessed by the project or design object's GetChildObject(VariableName) function.

```
oProjVar = oProject.GetChildObject("$VarName")
oVar = oProject.GetChildObject("DesignName/VarName")
oVar = oDesign.GetChildObject("variableName")
oProject..GetChildNames("Variable") will return all project variable names.
oDesign.GetChildNames(Variable) will return all Design Variable names.
```

- GetChildTypes() always return empty array.
- GetChildNames() always return empty array, since variable has no child.
- GetChildObject(objPath) it has no child.
- GetPropNames(bIncludeReadOnly) ['EvaluatedValue', 'SIValue'] are read-only properties
 - Independent variable :['Value', 'EvaluatedValue', 'SIValue', 'Description', 'ReadOnly', 'Hidden', 'Sweep', 'Optimization/Included', 'Optimization/Min', 'Optimization/Max', 'Sensitivity/Included', 'Sensitivity/Min', 'Sensitivity/Max', 'Sensitivity/IDisp', 'Statistical', 'Statistical/Included', 'Tuning/Included', 'Tuning/Step', 'Tuning/Min', 'Tuning/Max'].
 - Dependent variable ['Value', 'EvaluatedValue', 'SIValue', 'Description', 'ReadOnly', 'Hidden', 'Sweep']
- GetPropValue(propName)
 - oVar.GetPropValue() return the variable value as text string.
 - oVar.GetPropValue("Value") return the variable value as text string.
 - oVar.GetPropValue("SIValue") return the SI-value of variable as number.
 - oVar.Get_SIValue() also return the SI value.

- SetPropValue(propName, newValue)
oVar.SetPropValue("Value", 888)
oVar.SetPropValue("Sensitivity/Included", True)
oVar.SetPropValue("Sensitivity/Max", '1.8pF')
oVar.Set_Sensitivity_Max('1.8pF') also works as last call.
oVar.SetPropValue("Sensitivity", ['Min:=' , '0.8pF', 'Max:=' , '1.8pF'])
//set multiple attributes at one call:
oVar.SetPropValue("@", ['Value:=' , 288, 'Sensitivity', ['Included', True, 'Min', '0.0']])
oVar.SetPropValue("", ['Value:=' , 288, 'Sensitivity', ['Included', True, 'Min', '0.0']])

Optimetrics Module Object:

Optimetrics Module Object inherited all functions defined in the Property Object. But it doesn't have property, GetPropValue() & SetPropValue() function can be used to set its child object's property..

- GetChildTypes() there are six type of children, they are ['OptiParametric', 'OptiOptimization', 'OptiSensitivity', 'OptiStatistical', 'OptiDesignExplorer', 'OptiDXDOE']. But the return array only included those that have setup defined, so it may be an empty array if no optimetrics setup is defined. The GetChildNames(type) function also recognized the type name without the prefix "Opti".
- GetChildNames(type)
GetChildNames() will return all setup for all types.
GetChildNames("OptiOptimization") & GetChildNames("Optimization") will return all Optimization setup.
- GetChildObject()
oParamSetup = oOptModule.GetChildObject('ParametricSetup1') get the
oOptSetup = oOptModule.GetChildObject('OptimizationSetup1')
- GetPropNames(bIncludeReadOnly) always return empty array since the it has no property.
- GetPropValue(propPath) may be used to get its child's property value
oOptModule.GetPropValue("OptimizationSetup1\Optimizer") get the optimizer name for OptimizationSetup1
- SetPropValue(propPath, newValue) may be used to set its child's property value
oOptModule.SetPropValue(ParametricSetup1\Enabled", False) //disable ParametricSetup1

Optimetrics Setup Object

This is a new Object inherited all functions defined in the Property Object. But it doesn't have child. It is accessible through its parents.

```
oOptSetup = oOptModule.GetChildObject('OptimizationSetup1')
```

```
oOptSetup = oDesign.GetChildObject('Optimetrics\OptimizationSetup1')
```

```
oOptSetup = oProject.GetChildObject('TeeModel\Optimetrics\OptimizationSetup1')
```

- GetChildTypes() always return empty array.
- GetChildNames(type) always return empty array
- GetChildObject()
- GetPropNames(bIncludeReadOnly) will return the property names listed in the property window when the setup is selected.
- GetPropValue(propName)
oOptSetup.GetPropValue("Optimizer") return the selected optimizer name.
oOptSetup.GetPropValue("Optimizer/Choices") return all optimizer names.
- SetPropValue(propName, newValue)
oOptSetup.SetPropValue("Optimizer", "NotAnOptimizerName"; will return false.
oOptSetup.SetPropValue("Optimizer", "Quasi Newton"; return true, since "Quasi Newton" is one of the optimizer name returned as the Optimizer Choices.
- HasResult() return true if the setup is solved. Otherwise return false.
- Validate() return true if the setup is valid for analyze. Otherwise return false. Calling the SetPropValue() function to change the property may invalid the setup.

ReportSetup(Results) Module Object:

ReportSetup module Object inherited all functions defined in the Property Object. But it doesn't have property, GetPropValue() & SetPropValue() function can be used to get/set its child object's property..

- GetChildTypes() always empty array.
- GetChildNames(type)
GetChildNames() return all report names

- `GetChildObject(objPath)`
`oRpt = oRptModule.GetChildObject("S Parameter Plot 1")` return the report property object
`oTrace = oRptModule.GetChildObject("S Parameter Plot 1/dB(S(Port1,Port1))")` return the trace property object
`oAxisX = oRptModule.GetChildObject("S Parameter Plot 1/AxisX")` return the axis X property object
- `GetPropNames(bIncludeReadOnly)` always return empty array since the it \has no property.
- `GetPropValue()`
`oRptModule.GetPropValue("S Parameter Plot 1/Display Type")`
- `SetPropValue()`
`oRptModule.SetPropValue("S Parameter Plot 1/Display Type", "DataTable")`

ReportSetup(Results) Module Child Objects:

These are Property Objects. Its first level of child object is report. Report has trace, axis, header, Legend, and more children. Trace has curve as child etc.

Those child objects can be accessed by calling all levels of parent object's `GetChildObject(path)` function.

```
oRpt = oRptModule.GetChildObject(reportName)
```

```
oRpt = oDesign.GetChildObject("Results/reportName")
```

```
oTrace = oRpt.GetChildObject(traceName)
```

```
oTrace = oRptModule.GetChildObject(ReportName/TraceName)
```

- `GetChildTypes()` always return empty array.
- `GetChildNames()` get the object's child names. What will be returned will depended on the object instance.
- `GetChildObject(objPath)`
- `GetPropNames(bIncludeReadOnly)` will return the property names listed in the property window when the object is selected.
- `GetPropValue(propName)`
`oRpt.GetPropValue("Display Type")` return the report's display type.
`oOptSetup.GetPropValue("Display Type/Choices")` return all optimizer names.
`oTrace.GetPropValue("X Component")`

- `SetPropValue(propName, newValue)`
`oTrace.SetPropValue("Primary sweep", "Freq")`

Radiation Module Object:

This inherited all functions defined in the Property Object. But it doesn't have property, `GetPropValue()` & `SetPropValue()` function can be used to set its child object's property.

- `GetChildTypes()` always return empty array, now its children
- `GetChildNames(type)`
`GetChildNames()` return all setup names.
- `GetChildObject(setupName)` return the setup object as Property object.
`oOverlay= oRadModule.GetChildObject('Antenna Parameter Overlay1')`
`oSphere = oRadModule.GetChildObject('Infinite Sphere1')`
- `GetPropNames()` return empty array; it has no property.
- `GetPropValue()`
`oRadModule.GetPropValue('Line1/Num Points')` //Get the the Line1 setups' "Num Points" property value.
- `SetPropValue()`
`oRadModule.SetPropValue('Line1/Num Points', 100)` ; Set the Line1 setups' "Num Points" property to 100.

Radiation Module Child Objects:

These are Property Objects. It also provides quick function call to get/set its properties by adding functions with property name appended to `Get_` & `Set_` prefix. To find what functions it provides enter `dir(oVar)` the command window.

Those child objects can be access by call all levels of parent object's `GetChildObject(path)` function.

`oRadSetup = oRadModule.GetChildObject(setupName)`

`oRadSetup = oDesign.GetChildObject(RadField/setupName)`

- `GetChildTypes()` always return empty array.
- `GetChildNames()` always return empty array , since Radiation setup has no child.
- `GetChildObject(objPath)` it has no child.

- GetPropNames(bIncludeReadOnly) will return the property names listed in the property window when the setup is selected.
- GetPropValue(propName)
 - oRadSetup.GetPropValue("Num Points") return the line setup's "Num Points" property value.
 - oRadSetup.Get_NumPoints() will also get the same value.
- SetPropValue(propName, newValue)
 - oRadSetup.SetPropValue('Num Points', 888)
 - oRadSetup.Set_NumPoints(888)

Conventions Used in this Chapter

General Definitions:

Property	A single item that can be modified in the Properties window or in the modal Properties pop-up window.
<PropServer>	The item whose properties are being modified. This is usually a compound name, giving all information needed by the editor, design, or project in order to locate the item.
<PropTab>	Corresponds to one tab in the Properties window, the one under which properties are being edited.
<PropName>	The name of a single property.

The following tables list specific <PropServer> and <PropTab> values for different property types.

For Project Variables:

<PropServer>	"ProjectVariables"
<PropTab>	"ProjectVariableTab"

For Local Variables:

<PropServer>	"LocalVariables"
<PropTab>	"LocalVariableTab"

For Passed Parameters:

<PropServer>	"Instance:<Name of Circuit Instance>"
<PropTab>	"PassedParameter Tab"

For Definition Parameters:

<PropServer>	"DefinitionParameters"
<PropTab>	"DefinitionParameters"

For Modules and Editors:

<PropServer>	<ModuleName>:<ItemName> where <ItemName> is the boundary name, solution setup name, etc. For example, "BoundarySetup:PerfE1"
<PropTab>	Boundary Module: "HfssTab" Mesh Operations Module: "MeshSetupTab" Analysis Module: "HfssTab" Optimetrics Module: "OptimetricsTab" Solutions Module: <i>Does not support properties.</i> Field Overlays Module: "FieldsPostProcessorTab" Radiation Module: "RadFieldSetupTab" Circuit Module: "CCircuitTab" System Module: "SystemTab" HFSS 3D Layout Module: "HFSS 3D LayoutTab" Nexxim Module: "NexximTab" Layout elements: "BaseElementTab" Schematic elements: "ComponentTab" Optimetrics Module: "OptimetricsTab"

For 3D Model Editor objects:

<PropServer>	Name of the object. For example, "Box1".
<PropTab>	"Geometry3DAttributeTab"

For 3D Model Editor operations:

<PropServer>	<ObjName>:<OperationName>:<int> where <int> is the operation's history index. For example, "Box2:CreateBox:2" refers to the second "CreateBox" operation in Box2's history.
<PropTab>	"Geometry3DCmdTab"

For Reporter operations on Report properties:

<PropServer>	<ReportSetup>
<ChangeProperty>	Array. For example, to set the company name in a plot header to "My Company": <pre>Set oModule = oDesign.GetModule("ReportSetup") oModule.ChangeProperty Array("NAME:AllTabs",_ Array("NAME:Header",_ Array ("NAME:PropServers",_ "XY Plot1:Header"), Array("NAME:ChangedProps",_ Array("NAME:Company Name", "Value:=", "My Company"))))</pre>

For Maxwell Modules:

<PropServer>	<ModuleName>:<ItemName> where <ItemName> is the solution setup name.
<PropTab>	Boundary module: "MaxwellTab" Mesh Operations module: "MeshSetupTab" Analysis module: "MaxwellTab" Optimetrics module: "OptimetricsTab"

	Solutions module: <i>Does not support properties.</i> Field Overlays module: "FieldsPostProcessorTab" Radiation module: "RadFieldSetupTab"

Note:

For scripted property changes in the various modules and editors, refer to the chapters on the System, HFSS 3D Layout, and Nexxim tools, as well as the Layout and Schematic editors.

ChangeProperty (HFSS and Maxwell)

Use: Changes to properties are scripted using the ChangeProperty command. This command can be executed by the oEditor to change editor properties, by the oDesign to change design level properties, and by the oProject to change project level properties. The command can be used to create, edit, and/or remove properties. In HFSS and Maxwell, only Variable and Separator properties can be deleted.

Use the script recording feature and edit a property, and then view the resulting script entry or use GetPropertyValue for the desired property to see the expected format.

Command: None

Syntax: ChangeProperty Array("Name:AllTabs", <PropTabArray>, <PropTabArray>, ...)

ChangeProperty(<modulename>:<setup name>:<sweep name>)

Return Value: None

Parameters: <PropTabArray>

```
Array("Name:<PropTab>",
    <PropServersArray>,
    <NewPropArray>,
    <ChangedPropsArray>,
```

```
<DeletedPropsArray>)
```

```
<PropServersArray>
```

```
Array("Name:PropServers", <PropServer>,  
<PropServer>, ...)
```

```
<NewPropArray>
```

```
Array("Name:NewProps", <PropDataArray>,  
<PropDataArray>, ...)
```

```
<ChangedPropsArray>
```

```
Array("Name:ChangedProps", <PropDataArray>,  
<PropDataArray>, ...)
```

```
<DeletedPropsArray>
```

```
Array("Name:DeletedProps", <PropName>,  
<PropName>, ...)
```

```
<PropDataArray>
```

```
Array("NAME:<PropName>",
```

```
"PropType:=", <PropType>,  
"NewName:=", <string>,  
"Description:=", <string>,  
"NewRowPosition:=", <int>,  
"ReadOnly:=", <bool>,  
"Hidden:=", <bool>,  
<PropTypeSpecificArgs>
```

<PropType>

Type: string

Identifies the type of property when a new property is added. In HFSS and Maxwell, only separator properties and variable properties can be added.

```
"SeparatorProp"  
"VariableProp"  
"TextProp"  
"NumberProp"  
"ValueProp"  
"CheckboxProp"  
"MenuProp"  
"PointProp"  
"VPointProp"  
"V3DPointProp"
```

"ButtonProp"

NewName

Specify the new name of a property if the property's name is being edited. In HFSS and Maxwell, the name can only be changed for separators and variables.

Description

Specify a description of the property. In HFSS and Maxwell, the description can only be changed for separators and variables.

NewRowPosition

Used to reorder rows in the **Property** dialog box.

In HFSS, this only applies to the **Project>Project Variables** panel and the **Hfss>DesignProperties** panel.

In Maxwell, this only applies to the **Project>Project Variables** panel and the **Maxwell3D>DesignProperties**, **Maxwell2D>Design Properties**, or **RMxpert>Design Properties** panels.

Specify the new zero-based row index of the variable or separator.

ReadOnly

Used to mark a property as "read only" so it can not be modified. In HFSS and Maxwell, this flag can only be set for variables and separators.

Hidden

Used to hide a property so it can not be viewed outside of the **Property** dialog box. In HFSS and Maxwell, this flag can only be set for variables and separators.

<PropTypeSpecificArgs>

SeparatorProp: no arguments

TextProp: "Value:=", <string>

NumberProp: "Value:=", <double>

ValueProp: "Value:=", <value>

CheckboxProp: "Value:=", <bool>

MenuProp: "Value:=", <string>

PointProp "X:=", <double>, "Y:=", <double>

VPointProp: "X:=", <value>, "Y:=", <value>

V3DPointProp: "X:=", <value>, "Y:=", <value>, "Z:=", <value>

Material Button: "Material:=", <string>

Color Button: "R:=", <int>, "G:=", <int>, "B:=", <int>

Transparency Button: "Value:=", <double>

<PropTypeSpecificArgs> for VariableProps

Syntax:

"Value:=", <value>, <OptimizationFlagsArray>,

<TuningFlagsArray>, <SensitivityFlagsArray>,

<StatisticsFlagsArray>

Parameters:

<OptimizationFlagsArray>

Array("NAME:Optimization",

"Included:=", <bool>,

"Min:=", <value>,

"Max:=", <value>)

<Tuning flagsArray>

Array("NAME:Tuning",

"Included:=", <bool>,

"Step:=", <value>,

"Min:=", <value>,

"Max:=", <value>)

<SensitivityFlagsArray>

Array("NAME:Sensitivity",

"Included:=", <bool>,

"Min:=", <value>,

"Max:=", <value>)

"IDisp:=", <value>)

<StatisticsFlagsArray>

Array("NAME:Statistical",

"Included:=", <bool>,

"Dist:=", <Distribution>,

"StdD:=", <value>,

"Min:=", <value>,

"Max:=", <value>,

"Tol:=", <string>)

<Distribution>

Type: string

Value should be "Gaussian" or "Uniform"

StdD

Standard deviation.

Min

Low cut-off for the distribution.

Max

High cut-off for the distribution.

Tol

Tolerance for uniform distributions. Format is "<int>%".

Example: "20%".

VB Example: Adding a new project level variable "\$width":

```
oProject.ChangeProperty Array("NAME:AllTabs",_
    Array("NAME:ProjectVariableTab",_
        Array("NAME:PropServers", "ProjectVariables"),_
        Array("NAME:NewProps",_
            Array("NAME:$width",_
                "PropType:=", "VariableProp",_
                "Value:=", "3mm",_
                "Description:=", "my new variable"))))
```

VB Example: Deleting the design level variable "height":


```
oDesign.ChangeProperty Array("NAME:AllTabs", _  
    Array("NAME:LocalVariableTab", _  
        Array("NAME:PropServers", "DefinitionParameters"), _  
        Array("NAME:DeletedProps", "height"))
```

VB Example: Changing a property's value. If the following command were executed, then the value of the property "XSize" of the PropServer "Box1:CreateBox:1" on the "Geometry3DCmdTab" tab would be changed. (oEditor is the Geometry3D editor in HFSS.)

```
oEditor.ChangeProperty Array("NAME:AllTabs", _  
    Array("NAME:Geometry3DCmdTab", _  
        Array("NAME:PropServers", "Box1:CreateBox:1"), _  
        Array("NAME:ChangedProps", _  
            Array("NAME:XSize", "Value:=", "1.4mil"))))
```

VB Example: Changing the Company Name, Design Name, the background color, and the Axis scaling in a Report.

```
Set oProject = oDesktop.SetActiveProject("wgcombiner")  
Set oDesign = oProject.SetActiveDesign("HFSSDesign2")  
Set oModule = oDesign.GetModule("ReportSetup")  
oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Header", _
```

```
Array("NAME:PropServers", "XY Plot1:Header"), _  
Array("NAME:ChangedProps", Array("NAME:Company Name", _  
    "Value:=", "My Company"))))  
oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Header", _  
    Array("NAME:PropServers", "XY Plot1:Header"), _  
    Array("NAME:ChangedProps", Array("NAME:Design Name", _  
        "Value:=", "WG Combiner"))))  
oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:General", _  
    Array("NAME:PropServers", "XY Plot1:General"), _  
    Array("NAME:ChangedProps", Array("NAME:Back Color", _  
        "R:=", 128, "G:=", 255, "B:=", 255))))  
oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Axis", _  
    Array("NAME:PropServers", "XY Plot1:AxisX"), _  
    Array("NAME:ChangedProps", Array("NAME:Axis Scaling", _  
        "Value:=", "Log"))))
```

GetArrayVariables

Returns a list of array variables. To get a list of indexed project variables, execute with oProject. To get a list of indexed local variables, use oDesign.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing names of variables.

Python Syntax	<code>GetArrayVariables()</code>
Python Example	<code>oProject.GetArrayVariables()</code> <code>oDesign.GetArrayVariables()</code>

VB Syntax	<code>GetArrayVariables</code>
VB Example	<code>oProject.GetArrayVariables</code> <code>oDesign.GetArrayVariables</code>

GetProperties

Gets a list of all the properties belonging to a specific <PropServer> and <PropTab>. This can be executed by the oProject, oDesign, or oEditor variables.

UI Access	N/A		
Parameters	Name	Type	Description
	<PropTab>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables")

			• ProjectVariableTab ("Project variables") • ConstantsTab ("Constants") • BaseElementTab ("Symbol" or "Footprint") • ComponentTab ("General") • Component("Component") • CustomTab ("Intrinsic Variables") • Quantities ("Quantities") • Signals ("Signals")
	<PropServer>	String	An object identifier, generally returned from another script method, such as CompInst@R;2;3
Return Value	Array of strings containing the names of the appropriate properties.		

Python Syntax	GetProperties(<PropTab>, <PropServer>)
Python Example	oEditor.GetProperties('PassedParameterTab', 'k')

VB Syntax	GetProperties <PropTab>, <PropServer>
VB Example	oEditor.GetProperties "PassedParameterTab", "k"

GetPropertyValue

Returns the value of a single property belonging to a specific <PropServer> and <PropTab>. This function is available with the Project, Design or Editor objects, including definition editors.

Tip: Use the script recording feature and edit a property, and then view the resulting script to see the format for that property.

UI Access	N/A		
Parameters	Name	Type	Description
	<PropTab>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables") - ProjectVariableTab ("Project variables") - ConstantsTab ("Constants") - BaseElementTab ("Symbol" or "Footprint") - ComponentTab ("General") - Component("Component") - CustomTab ("Intrinsic Variables") - Quantities ("Quantities") - Signals ("Signals")
	<PropServer>	String	An object identifier, generally returned from another script method, such as <code>CompInst@R;2;3</code>
	<PropName>	String	Name of the property.
Return Value	String value of the property.		

Python Syntax	GetPropertyValue (<PropTab>, <PropServer>, <PropName>)
Python Example	<code>selectionArray = oEditor.GetSelections()</code>

	<pre>for k in selectionArray: val = oEditor.GetPropertyValue("PassedParameterTab", k, "R") ...</pre>

VB Syntax	<code>GetPropertyValue (<PropTab>, <PropServer>, <PropName>)</code>
VB Example	<pre>selectionArray = oEditor.GetSelections for k in selectionArray: val = oEditor.GetPropertyValue("PassedParameterTab", k, "R") ...</pre>

GetVariables

Returns a list of all defined variables. To get a list of project variables, execute this command using `oProject`. To get a list of local variables, use `oDesign`.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing the variables.

Python Syntax	<code>GetVariables ()</code>

Python Example	<pre>oProject.GetVariables() oDesign.GetVariables()</pre>

VB Syntax	GetVariables
VB Example	<pre>oProject.GetVariables oDesign.GetVariables</pre>

GetVariableValue

Gets the value of a single specified variable. To get the value of project variables, execute this command using `oProject`. To get the value of local variables, use `oDesign`.

UI Access	N/A		
Parameters	Name	Type	Description
	<VarName>	String	Name of the variable to access.
Return Value	String represents the value of the variable.		

Python Syntax	GetVariableValue(<VarName>)
Python Example	<pre>oProject.GetVariableValue("var_name")</pre>

VB Syntax	GetVariableValue <VarName>

VB Example	<code>oProject.GetVariableValue "var_name"</code>

SetPropertyValue

Sets the value of a single property belonging to a specific PropServer and PropTab. This function is available with the Project, Design or Editor objects, including definition editors. This is not supported for properties of the following types: ButtonProp, PointProp, V3DPointProp, and VPointProp. Only the ChangeProperty command can be used to modify these properties.

Use the script recording feature and edit a property, and then view the resulting script entry or use GetPropertyValue for the desired property to see the expected format.

UI Access	N/A		
Parameters	Name	Type	Description
	<propTab>	String	One of the following, where tab titles are shown in parentheses: - PassedParameterTab ("Parameter Values") - DefinitionParameterTab (Parameter Defaults") - LocalVariableTab ("Variables" or "Local Variables") - ProjectVariableTab ("Project variables") - ConstantsTab ("Constants") - BaseElementTab ("Symbol" or "Footprint") - ComponentTab ("General") - Component("Component") - CustomTab ("Intrinsic Variables") - Quantities ("Quantities") - Signals ("Signals")
	<propServer>	String	An object identifier, generally returned from another script method, such as <code>CompInst@R;2;3</code>

	<propName>	String	Name of the property.
	<propValue>	String	The value for the property
Return Value	None.		

Python Syntax	SetPropertyValue(<propTab>, <propServer>, <propName>, <propValue>)		
Python Example	oEditor.SetPropertyValue("PassedParameterTab", "k", "R", "2200")		

VB Syntax	SetPropertyValue <propTab>, <propServer>, <propName>, <propValue>		
VB Example	oEditor.SetPropertyValue "PassedParameterTab", "k", "R", "2200"		

SetVariableValue

Sets the value of a variable. To set the value of a project variable, execute this command using `oProject`. To set the value of a local variable, use `oDesign`.

UI Access	N/A		
Parameters	Name	Type	Description
	<VarName>	String	Variable name.
	<VarValue>	Value	New value for the variable.
Return Value	None.		

Python Syntax	SetVariableValue (<VarName>, <VarValue>)
Python Example	oProject.SetVariableValue('\$Var1', '3mm')

VB Syntax	SetVariableValue <VarName>, <VarValue>
VB Example	oProject.SetVariableValue "\$Var1", "3mm"

9 - Design Object Script Commands

Design object commands should be executed by the oDesign object.

```
oDesign.CommandName <args>
```

For example:

```
Set oDesign = oProject.SetActiveDesign("MaxwellDesign1")
```

Conventions Used in this Chapter

<ModuleName> is a placeholder for any one of the following modules:

- [Analysis Module](#) – "AnalysisSetup"
- [Boundary Module](#) – "BoundarySetup"
- [Field Overlays Module](#) – "FieldsReporter"
- [Mesh Module](#) – "MeshSetup"
- [Optimetrics Module](#) – "Optimetrics"
- Radiation Module – "RadField"
- Reduce Matrix Module – "ReduceMatrix"
- [Reporter Module](#) – "ReportSetup"
- Simulation Setup Module – "SimSetup"
- [Solutions Module](#) – "Solutions"

[AnalyzeAll \(Maxwell\)](#)

[AnalyzeDistributed](#)

[ApplyMeshOps](#)

[ClearLinkedData](#)

[ConfigureFluentConductivityCoupling](#)

[ConstructVariationString](#)

[Create3DDesign](#)

[DeleteFieldVariation](#)

[DeleteFullVariation](#)

[DeleteLinkedDataVariation](#)

[EditNotes](#)

[EnableHarmonicForceCalculation](#)

[ExportArrayMetadata](#)

[ExportConvergence](#)

[ExportElementBasedHarmonicForce](#)

[ExportHarmonicTransientForce](#)

["ExportLayoutLayerGridField"](#) on page 17-2

["ExportLayoutViaCurrent"](#) on page 17-5

["ExportLayoutViaCurrentDensity"](#) on page 17-6

[ExportMatrixData](#)

[ExportMeshStats](#)

[ExportProfile](#)

[ExportReport](#)

[GenerateMesh](#)

[GetActiveEditor](#)

[GetAllPorts](#)

[GetBoundingBox](#)

[GetChildNames \[Design\]](#)

[GetChildObject \[Design\]](#)

[GetChildTypes \[Design\]](#)

[GetConfigurableData](#)

[GetData](#)

[GetDesignName](#)

[GetDesignType](#)

[GetDesignValidationInfo](#)

[GetGeometryMode](#)

[GetManagedFilePath](#)

[GetModule](#)

[GetName](#)

[GetNominalVariation](#)

[GetNoteText](#)

[GetPostProcessingVariables](#)

[GetPropNames \[Design\]](#)

[GetPropValue \[Design\]](#)

[GetSelections](#)

[GetSolutionType](#)

[InitializeSystemCoupling](#)

[Is2D](#)

[Is3D](#)

[PasteDesign](#)

[Redo](#)

[RenameDesignInstance](#)

[RenameSource](#)

[RunToolkit](#)

[SetActiveEditor](#)

[SetConductivityThreshold](#)

[SetDesignSettings \(Maxwell\)](#)

[SetFastTransformationForLayoutComponent](#)

[SetObjectDeformation](#)

[SetObjectTemperature](#)

[SetPropValue \[Design\]](#)

[SetPropertyValue](#)

[SetShowLayoutForLayoutComponent](#)

[SetSolutionType \(Maxwell\)](#)

[Solve](#)

[StartAnalysis](#)

[StopSimLink](#)[Undo](#)

ValidateCircuit

[ValidateDesign](#)[ValidateLink](#)

ApplyMeshOps

If any mesh operations were defined and not yet performed in the current variation for the specified solution setups, they will be applied to the current mesh. If necessary, an initial mesh will be computed first. No further analysis will be performed.

Important:

This is a legacy script. Use [GenerateMesh](#) instead.

UI Access	Maxwell > Analysis Setup > Generate Mesh.		
Parameters	Name	Type	Description
	<SetupNameArray>	Array	Array containing the string names of setups.
Return Value	Integer value: • 0 – Success • Else – Error		

Python Syntax	ApplyMeshOps(<SetupNameArray>)
Python Example	<code>oDesign.ApplyMeshOps(['Setup1','Setup2'])</code>

VB Syntax	<code>ApplyMeshOps(<SetupNameArray>)</code>
VB Example	<code>status = oDesign.ApplyMeshOps Array("Setup1", "Setup2")</code>

Analyze

Solves a single solution setup and all of its frequency sweeps.

UI Access	Right-click a solution setup in the project tree, and select Analyze .		
Parameters	Name	Type	Description
	<code><setup></code>	String	Setup name.
	<code>isBlocking</code>	Boolean	An optional arg that defaults to <code>true</code> . If it's <code>false</code> , the command immediately returns while the analysis or analyses run in the background. The status of the analyses can be checked with the <code>AreThereSimulationsRunning</code> command.
Return Value	Integer value: • 0 – Success • Else – Error		

Python Syntax	<code>Analyze (<setup>)</code>
Python Example	<code>oDesign.Analyze("Setup1", isBlocking)</code>

VB Syntax	<code>Analyze <setup></code>

VB Example	<code>oDesign.Analyze "Setup1"</code>

AnalyzeAll (Maxwell menu)

Analyzes the nominal problems and all the Optimetrics problems for all the setups under the selected design.

UI Access	Click Maxwell > Analyze All , or right-click a design instance in the project tree and select Analyze All .		
Parameters	Name	Type	Description
	isBlocking	Boolean	An optional arg that defaults to <code>true</code> . If it's <code>false</code> , the command immediately returns while the analysis or analyses run in the background. The status of the analyses can be checked with the <code>AreThereSimulationsRunning</code> command.
Return Value	None		

Python Syntax	<code>AnalyzeAll ()</code>
Python Example	<code>oDesign.AnalyzeAll(isBlocking)</code>

VB Syntax	<code>AnalyzeAll</code>
VB Example	<code>oDesign.AnalyzeAll</code>

AnalyzeDistributed

Performs a distributed analysis.

UI Access	N/A

Parameters	Name	Type	Description
	<SetupName>	String	String name of the setup to be analyzed.
Return Value	Integer value: • 0 – Success • Else – Error		

Python Syntax	AnalyzeDistributed(<SetupName>)
Python Example	<code>oDesign.AnalyzeDistributed('Setup1')</code>

VB Syntax	AnalyzeDistributed <SetupName>
VB Example	<code>oDesign.AnalyzeDistributed "Setup1"</code>

ClearLinkedData (Design)

Clear the linked data of all solution setups in the design. (This command is similar to the ClearLinkedData command for the module level, which clears link data for one or more specific solution setups.)

UI Access	Maxwell > Analysis Setup > Clear Linked Data (with no solution setup selected in the Project Manager)
Parameters	None.
Return Value	None.

Python Syntax	ClearLinkedData()
Python Example	<code>oDesign.ClearLinkedData()</code>

VB Syntax	ClearLinkedData
VB Example	<code>oDesign.ClearLinkedData</code>

ConfigureFluentConductivityCoupling

Use: Configures the directory for the conductivity coupling between a Maxwell 3D magnetostatic design and Ansys Fluent.

Command: Right-click on the solution setup under Analysis in the project tree and choose **Configure Fluent Conductivity Coupling**.

Syntax: ConfigureFluentConductivityCoupling <SolveSetupName>, <Pathofcouplingdirectory>

Return Value: None

Parameters: <SolveSetupName>

Type: <string>

Name of the solution setup.

<PathOfCouplingDirectory>

Type: string

Path of the coupling directory.

Example:

```
Set oProject = oDesktop.SetActiveProject("Project18")
```

```
Set oDesign = oProject.SetActiveDesign("Maxwell13DDesign1")  
  
Set oModule = oDesign.GetModule("AnalysisSetup")  
  
oModule.ConfigureFluentConductivityCoupling "Setup1",  
  
"C:\Users\test\Documents\Ansys\Maxwell\"
```

ConstructVariationString

Lists and orders the variables and values associated with a design variation.

UI Access	N/A		
Parameters	Name	Type	Description
	<ArrayOfVariableNames>	Array of Strings	List of variable names.
	<ArrayOfVariableValuesIncludingUnits>	Array of Strings	List of variable values, including units, in the same order as the list of names.
Return Value	Returns variation string with the variables ordered to correspond to the order of variables in design variations. The values for the variables are inserted into the variation string. For an example of how ConstructVariationString can be used, see the Python example.		

Python Syntax	ConstructVariationString(<ArrayOfVariableNames>, <ArrayOfVariableValuesIncludingUnits>)
Python Example	<pre>varStr = oDesign.ConstructVariationString(["xx", "yy"], ["2mm", "1mm"]) oDesign.ExportProfile("Setup1", varStr, "C:\profile.prof")</pre>

VB Syntax	ConstructVariationString <ArrayOfVariableNames>, <ArrayOfVariableValuesIncludingUnits>

VB Example	<code>oDesign.ConstructVariationString Array("x_size","y_size"), Array("2mm", "1mm")</code>

CreateParametricCircuit

Note:

This command is for internal Ansys use only.

Python Syntax	<code>CreateParametricCircuit()</code>
Python Example	<code>oDesign.CreateParametricCircuit()</code>

Create3DDesign [Maxwell]

Use: Create a Maxwell 3D design from a Maxwell 2D design.

Command: Click **Maxwell2D>Create 3D Design**, or right-click a Maxwell 2D design instance in the project tree and select **Create 3D Design**.

Syntax: Create3DDesign <DesignSettingsArray>

Return Value: None

Parameters: <DesignSettingsArray>

For a 2D design in "Cartesian, XY" geometry mode:

`Array("NAME:Parameters",`

`"ZLength:=", <value>,`

`"PararmetrizeZLength:=", <bool>,`

For a 2D design in "Cylindrical about Z" geometry mode:

```
Array("NAME:Parameters",
"SweepAngle:=", <value>,
"PararmetrizeSweepAngle:=", <bool>,
"DraftAngle:=", <value>,
"DraftType:=", <string>, (one of: "Extended", "Round", "Natural")
"NumberOfSegments:=", <integer>)
```

Example:

For a 2D design in "Cartesian, XY" geometry mode:

```
oDesign.Create3DDesign Array("NAME:Parameters", "ZLength:=", "12mm", "PararmetrizeZLength:=",
true)
```

For a 2D design in "Cylindrical about Z" geometry mode:

```
oDesign.Create3DDesign Array("NAME:Parameters", "SweepAngle:=", "120deg", "PararmetrizeSweepAngle:=", true, "DraftAngle:=", "30deg", "DraftType:=", "Round", "NumberOfSegments:=", "4")
```

DeleteFieldVariation

Deletes field variations, fields and meshes, or fields and meshes for specified variations.

UI Access	[Solver] > Results > Clean Up Solutions > [Fields Only / Fields and Meshes].		
Parameters	Name	Type	Description
	<variationKeys>	Array	Array containing either "All" string to select all variations, or strings of variations to include.
	<deleteMesh>	Boolean	When true, deletes mesh data.
	<deleteLinkedData>	Boolean	When true, deletes linked data.

Return Value	None.

Python Syntax	DeleteFieldVariation(<variationKeys>, <deleteMesh>, <deleteLinkedData>
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDesign = oProject.GetActiveDesign() Design.DeleteFieldVariation(['All'], True, False)</pre>

VB Syntax	DeleteFieldVariation <variationKeys>, <deleteMesh>, <deleteLinkedData>
VB Example	<pre>Set oProject = oDesktop.SetActiveProject "OptimTee" Set oDesign = oProject.SetActiveDesign "TeeModel" oDesign.DeleteFieldVariation Array("offset=" & Chr(39) & "0.0947046688081817in" & Chr(39) & ""), true, false</pre>

DeleteFullVariation

Deletes either all solution data, or selected variation data.

UI Access	[Maxwell] > Results > Clean Up Solutions.		
Parameters	Name	Type	Description
	<variationKeys>	Array	Array containing either "All" string to select all variations, or strings of variations to include.
	<deleteLinkedData>	Boolean	When true, deletes linked data as well.
Return Value	None.		

Python Syntax	DeleteFullVariation(<variationKeys>, <deleteLinkedData>)
Python Example	<code>oDesign.DeleteFullVariation(['All'], false)</code>

VB Syntax	DeleteFullVariation <variationKeys>, <deleteLinkedData>
VB Example	<code>oDesign.DeleteFullVariation Array('All'), false</code>

DeleteLinkedDataVariation

Deletes the linked data of specified variations.

UI Access	[Maxwell] > Results > Clean Up Solutions.		
Parameters	Name	Type	Description
	<DesignVariationKeys>	Array	Array containing strings of the variations keys whose linked data are going to be deleted.
Return Value	None.		

Python Syntax	DeleteLinkedDataVariation(<DesignVariationKeys>)
Python Example	<code>oDesign.DeleteLinkedDataVariation(["current=\'0.9mA\'", "current=\'1.0mA\'"])</code>

VB Syntax	DeleteLinkedDataVariation <DesignVariationKeys>
VB Example	<code>oDesign.DeleteLinkedDataVariation Array("current=\'0.9mA\'", "current=\'1.0mA\'")</code>

DeleteVariation

Deletes a variation.

Important:

This script is obsolete. Use [DeleteFullVariation](#), [DeleteFieldVariation](#), or [DeleteLinkedDataVariation](#).

EditNotes

Updates the notes for the design.

UI Access	Maxwell > Edit Notes.		
Parameters	Name	Type	Description
	<DesignNotes>	String	New design notes that are applied, replacing any current notes.
Return Value	None.		

Python Syntax	EditNotes(<DesignNotes>)
Python Example	oDesign.EditNotes("My design notes.")

VB Syntax	EditNotes <DesignNotes>
VB Example	oDesign.EditNotes "My design notes."

EnableHarmonicForceCalculation

This command sets the objects for harmonic force calculation.

UI Access	Maxwell 2D or 3D > Enable Harmonic Force Calculation, or right-click a design instance in the project tree and select Enable Harmonic Force Calculation		
Parameters	Name	Type	Description
	<ObjectArray>	String	List of object names for force calculations; for example, Array("EnabledObjects:=", Array("Magnet", "Steel"))
	<ForceType>	Integer	0, ObjectBased; 1, Element Based (Surface); 2, Element Based (Volumetric)
	<NumberOfRepeatedWindows>	Integer	The number of repeated windows; it can also be a variable that evaluates to an integer.
	<UseNumberOfLastCycles>	Boolean	Use number of last cycles for force calculations
	<NumberOfLastCycles>	Integer	Number of last cycles
	<StartTime>	Double	The start time of data collection
	<UseNumberOfCyclesForStopTime>	Boolean	Use number of cycles or time to define the stop time. True if user wants to use the number of cycles for the stop time; otherwise, False.
	<NumberOfCyclesForStopTime>	String	Number of cycles after the start time to make the stop time.
	<StopTime>	Double	Sets specific stop time. Should be greater than the <StartTime>
	<CouplingSteps>	String	Specify "All", "Steps from end", or "Range of steps"
	<ThirdParty>	String	Third-party output file type. Specify "LMS", "Romax", or "UNV"
	<OutputDir>	String	Specify the path to the output directory for third-party files
	<WindowFunctionType>	String	Specify "Rectangular", "Tri", "Van Hann", "Hamming", "Blackman", "Lanczos", or "Welch"
	<OutputFreqRangeType>	String	Specify "Use Range", "Use Number", or "Use All"
	<OutputFreqRangeStop>	Double	Specify the output range stop frequency (with unit)
	<OutputFreqRangeStart>	Double	Specify the output range start frequency (with unit)
	<OutputFreqRangeNum>	Integer	Number count of frequencies. The number count starts from the start frequency from FFT results.
	<CalculateForceType>	String	"Harmonic Force", "Transient Force", or "Harmonic and Tran-

			sient Force"
	<EnableInverterFeedback>	Boolean	Specify switching and maximum frequency to mimic inverter-fed machine
	<SwitchingFrequency>	String	Input, must be greater than zero
	<MaximumFrequency>	String	Input, must be less that switching frequency
Return Value	None		

Python Syntax	EnableHarmonicForceCalculation (<ObjectArray>, <ForceType>, <NumberOfRepeatedWindows>, <UseNumberOfLastCycles>, <NumberOfLastCycles>, <StartTime>, <UseNumberOfCyclesForStopTime>, <NumberOfCyclesForStopTime>, <StopTime>, <CouplingSteps>, <ThirdParty>, <OutputDir>, <WindowFunctionType>, <OutputFreqRangeType>, <OutputFreqRangeStop>, <OutputFreqRangeStart>, <OutputFreqRangeNum>, <CalculateForceType>, <EnableInverterFeedback>, <MaximumFrequency>)
Python Example	Example 1: Setting WindowFunctionType options for a Transient design <pre>oDesign.EnableHarmonicForceCalculation(["EnabledObjects:=", ["Cylinder1","Cylinder2","Cylinder3"], "ForceType:=", 0, "NumberOfRepeatedSamples:=", "1", "UseNumberOfLastCycles:=", False, "NumberOfLastCycles:=", 1, "StartTime:=", "0s", "UseNumberOfCyclesForStopTime:=", False, "NumberOfCyclesForStopTime:=", 1,</pre>

```
"StopTime:=", "0.01s",  
"WindowFunctionType:=", "Rectangular",  
])
```

Example 2: Using inverter feedback

```
oDesign.EnableHarmonicForceCalculation(  
[  
"EnabledObjects:=", ["Cylinder1"],  
"ForceType:=", 0,  
"WindowFunctionType:=", "Rectangular",  
"NumberOfRepeatedSamples:=", "1",  
"UseNumberOfLastCycles:=", True,  
"NumberOfLastCycles:=", 1,  
"StartTime:=", "0s",  
"UseNumberOfCyclesForStopTime:=", True,  
"NumberOfCyclesForStopTime:=", 1,  
"StopTime:=", "0.01s",  
"OutputFreqRangeType:=", "Use All",  
"CalculateForceType:=", "Harmonic Force",  
"EnableInverterFeedback:=", True,  
"SwitchingFrequency:=", "4000Hz",
```

```
"MaximumFrequency:="      , "8000Hz"
```

```
])
```

Example 3: Setting OutputFreqRangeType to "Use Range"

```
oDesign.EnableHarmonicForceCalculation(
[
  "EnabledObjects:="      , ["Cylinder1","Cylinder2"],
  "ForceType:="           , 0,
  "WindowFunctionType:="  , "Rectangular",
  "NumberOfRepeatedSamples:=", "1",
  "UseNumberOfLastCycles:=", False,
  "NumberOfLastCycles:="  , 1,
  "StartTime:="           , "0s",
  "UseNumberOfCyclesForStopTime:=", False,
  "NumberOfCyclesForStopTime:=", 1,
  "StopTime:="            , "0.01s",
  "OutputFreqRangeStart:="      , "1Hz",
  "OutputFreqRangeStop:="      , "3Hz",
  "OutputFreqRangeType:="      , "Use Range"
])
```

Example 4: Setting OutputFreqRangeType to "Use Number"

```
oDesign.EnableHarmonicForceCalculation(
```

```
[
  "EnabledObjects:="      , ["Cylinder1","Cylinder2"],
  "ForceType:="           , 0,
  "WindowFunctionType:="  , "Rectangular",
  "NumberOfRepeatedSamples:=" , "1",
  "UseNumberOfLastCycles:=" , False,
  "NumberOfLastCycles:="   , 1,
  "StartTime:="            , "0s",
  "UseNumberOfCyclesForStopTime:=" , False,
  "NumberOfCyclesForStopTime:=" , 1,
  "StopTime:="             , "0.01s",
  "OutputFreqRangeStart:=" , "1Hz",
  "OutputFreqRangeNum:="   , "43",
  "OutputFreqRangeType:="  , "Use Number"
])
```

Example 5: Setting OutputFreqRangeType to "Use All"

```
oDesign.EnableHarmonicForceCalculation(
```

```
[
  "EnabledObjects:="      , ["Cylinder1","Cylinder2"],
  "ForceType:="           , 0,
```

```

"WindowFunctionType:=" , "Rectangular",
"NumberOfRepeatedSamples:=", "1",
"UseNumberOfLastCycles:=", False,
"NumberOfLastCycles:=" , 1,
"StartTime:=" , "0s",
"UseNumberOfCyclesForStopTime:=", False,
"NumberOfCyclesForStopTime:=", 1,
"StopTime:=" , "0.01s",
" OutputFreqRangeType:=" , "Use All"
])

```

Example 6: Setting layout force nets and layers sets

```

oDesign.EnableHarmonicForceCalculation(
[
"ForceType:=" , 0,
"WindowFunctionType:=" , "Rectangular",
"NumberOfRepeatedSamples:=", "1",
"UseNumberOfLastCycles:=", True,
"NumberOfLastCycles:=" , 1,
"StartTime:=" , "0s",
"UseNumberOfCyclesForStopTime:=", True,
"NumberOfCyclesForStopTime:=", 1,

```

```
"StopTime:="      , "0.01s",
"OutputFreqRangeType:=" , "Use All",
"CaculateForceType:=" , "Harmonic Force",
[
  "NAME:NetsAndLayersChoices",
  [
    "NAME:LC1_1",
    [
      "NAME:NetLayerSetMap",
      [
        "NAME:net1",
        "LayerSet:="      , ["Top"]
      ],
      [
        "NAME:net2",
        "LayerSet:="      , ["Top"]
      ]
    ]
  ]
]
```



```
])
```

```
...
```

Example 7: Calculating Harmonic AND Transient force

```
oDesign.EnableHarmonicForceCalculation (
[
    "EnabledObjects:=" , ["Cylinder1"],
    "ForceType:=" , 1,
    "WindowFunctionType:=" , "Rectangular",
    "NumberOfRepeatedSamples:=" , "1",
    "UseNumberOfLastCycles:=" , True,
    "NumberOfLastCycles:=" , 1,
    "StartTime:=" , "0s",
    "UseNumberOfCyclesForStopTime:=" , True,
    "NumberOfCyclesForStopTime:=" , 1,
    "StopTime:=" , "0.01s",
    "OutputFreqRangeType:=" , "Use All",
    "CalculateForceType:=" , "Harmonic and Transient Force",
    "EnableInverterFeedback:=" , False
])
```

VB	EnableHarmonicForceCalculation Array(<ObjectArray>, <ForceType>, <NumberOfRepeatedWindows>, <UseNum-

Syntax	berOfLastCycles>, <NumberOfLastCycles>, <StartTime>, <UseNumberOfCyclesForStopTime>, <NumberOfCyclesForStopTime>, <StopTime>, <CouplingSteps>, <ThirdParty>, <OutputDir>, <WindowFunctionType>, <OutputFreqRangeType>, <OutputFreqRangeStop>, <OutputFreqRangeStart>, <OutputFreqRangeNum>, <CalculateForceType>, <EnableInverterFeedback>, <MaximumFrequency>)
VB Example	Example 1: Setting EnableHarmonicForceCalculation options for a 2D Transient design with element-based force type <pre> oDesign.EnableHarmonicForceCalculation Array("EnabledObjects=", Array("Rectangle1", "Rectangle2"), "ForceType=", 2, "NumberOfRepeatedWindows=", 3, "UseNumberOfLastCycles=", true, "NumberOfLastCycles=", 1, "StartTime=", "0s", "UseNumberOfCyclesForStopTime=", true, "NumberOfCyclesForStopTime=", 1, "StopTime=", "0.01s", "WindowFunctionType=", "Rectangular") </pre> Example 2: Setting EnableHarmonicForceCalculation options <pre> SetScriptVersion (Version="16.0") system1 = GetSystem (Name="Maxwell2DDesign3") Ansoft.EditSystem (System=system1) string1 = Ansoft.ExecuteGenericDesktopCommand(System=system1, </pre>

```

CommandClass="WB_ACTIVATE_GIVEN_SYSTEMID",
Argument="Maxwell",
ExecuteOnlyIfSystemIsAlreadyOpenInDesktop=True)

system1.SendAnsoftCommand(PyCommand="""oDesktop.SetActiveProject(\"test_project_from_
CADFEM\").SetActiveDesign(\"Maxwell2DDesign3\").EnableHarmonicForceCalculation(
[
\"EnabledObjects:=\ ", [\"IntegLine_0\", \"IntegLine_0_1\"],
\"UseNumberOfLastCycles:=\", False,
\"NumberOfLastCycles:=\ ", 2,
\"StartTime:=\ ", \"0.002s\",
\"UseNumberOfCyclesForStopTime:=\", False,
\"NumberOfCyclesForStopTime:=\", 2,
\"StopTime:=\ ", \"0.003s\"
\"WindowFunctionType:=\ ", \"Rectangular\"
]) """)

```

ExportConvergence

For a given variation, exports convergence data (max mag delta S, E, freq) to *.conv file.

UI Access	Results > Solution Data. Select Convergence tab and click Export .		
Parameters	Name	Type	Description
	<Setup>	String	Setup name.

	<VariationKeys>	String	The variation. Pass empty string for the current nominal variation.
	<FilePath>	String	Full path to desired *.conv file location.
Return Value	None.		

Python Syntax	ExportConvergence(<Setup>, <VariationKeys>, <FilePath> <OverwriteIfExists>)
Python Example	oDesign.ExportConvergence('Setup1', 'x_size = 2mm', 'c:\convergence.conv')

VB Syntax	ExportConvergence <Setup>, <VariationKeys>, <FilePath> <OverwriteIfExists>
VB Example	oDesign.ExportConvergence "Setup1", "x_size = 2mm", "c:\convergence.conv"

ExportElementBasedHarmonicForce

Exports a element-based harmonic force data to a .csv file.

UI Access	Maxwell 3D > Export Transient / Harmonic Force		
Parameters	Name	Type	Description
	<outputDirectory>	String	The path for the output directory.
	<solveSetupName>	String	The name of the solution setup.
	<number1> ,<number2>		Possible values for <i>number1</i> and <i>number2</i> : - When <i>frequencyOption</i> is 1, <i>number1</i> is –1, <i>number2</i> is -1 - When <i>frequencyOption</i> is 2, <i>number1</i> is <i>startFrequency</i><double>, <i>number2</i> is <i>endFrequency</i><double>

			When <i>frequencyOption</i> is 3, <i>number1</i> is number of frequencies<int>, <i>number2</i> is -1
Return Value	None.		

Python Syntax	<code>ExportElementBasedHarmonicForce(<outputDirectory>, <solveSetupName>, <frequencyOption>, <number1>, <number2>)</code>		
Python Example	<pre>ExportElementBasedHarmonicForce("E:\\tmp", "Setup1", 1, -1, -1) ExportElementBasedHarmonicForce("E:\\tmp", "Setup1", 2, 2.3, 10.5) ExportElementBasedHarmonicForce("E:\\tmp", "Setup1", 3, 4, -1)</pre>		

VB Syntax	<code>ExportElementBasedHarmonicForce <outputDirectory>, <solveSetupName>, <frequencyOption>, <number1>, <number2></code>		
VB Example	<pre>ExportElementBasedHarmonicForce "E:\\tmp", "Setup1", 1, -1, -1 ExportElementBasedHarmonicForce "E:\\tmp", "Setup1", 2, 2.3, 10.5 ExportElementBasedHarmonicForce "E:\\tmp", "Setup1", 3, 4, -1</pre>		

ExportHarmonicTransientForce

Exports harmonic force solution data from transient designs for use by Ansys Motion and third-party software.

UI Access	Maxwell 2D (or 3D)>Export Harmonic/Transient Force...		
Parameters	Name	Type	Description
	<ExportType>	string	One of:

			• UNV • Romax • LMS • Ansys Motion • Ansys Mechanical
	<ParametricSetupName>	string	The name of the parametric setup whose solution data is to be exported
	<ExportDirectory>	string	The path for the exported files
	<SolverSetupName>	string	The name of the design setup whose solution data is to be exported. If no solution setup is specified, all solve setups under the specified parametric setup will be exported.
	<ExportTimeOption>	integer	Export time steps: 0 for All steps, 1 for Steps from end, 2 for Range of steps.
	<ExportBeginStep>	integer	Integer value for begin step, only valid for <i>ExportTimeOption</i> : Range of steps
	<ExportEndStep>	integer	Integer value for end step, only valid for <i>ExportTimeOption</i> : Range of steps
Return Value	None		

Python Syntax	ExportHarmonicTransientForce (<ExportType>, <ParametricSetupName>, <ExportDirectory>, <SolverSetupName>, <ExportTimeOption>, <ExportBeginStep>, <ExportEndStep>)
Python Example	oDesign.ExportHarmonicTransientForce("Ansys Motion", " ParametricSetup1", "\$PROJECTDIR", "Setup1", 0, -1, -1)

VB Syntax	ExportHarmonicTransientForce <ExportType>, <ParametricSetupName>, <ExportDirectory>, <SolverSetupName>, <ExportTimeOption>, <ExportBeginStep>, <ExportEndStep>

VB Example	<pre>oDesign.ExportHarmonicTransientForce "Ansys Motion", "ParametricSetup1", "\$PROJECTDIR", "Setup1", 0, -1, -1</pre>

ExportMeshStats

Exports mesh statistics to a file.

UI Access	N/A		
Parameters	Name	Type	Description
	<setupName>	String	Name of the solution setup.
	<variationString>	String	Name of the variation. An empty string is interpreted as the current nominal variation.
	<filePath>	String	Full file path, including extension *.ms.
	<overwriteIfExists>	Boolean	(Optional). Defaults to True. If True, script overwrites any existing file of the same name. If False, no overwrite.
Return Value	None.		

Python Syntax	<code>ExportMeshStats(<setupName>, <variationString>, <filePath>, <overwriteIfExists>)</code>
Python Example	<pre>oDesign.ExportMeshStats ("Setup1", "radius=3mm", "C:\mydir\meshstats.ms", True)</pre>

VB Syntax	<code>ExportMeshStats <setupName>, <variationString>, <filePath>, <overwriteIfExists></code>
VB Example	<pre>oDesign.ExportMeshStats "Setup1", "radius=3mm", "C:\mydir\meshstats.ms", true</pre>

ExportProfile

Exports a solution profile to file.

UI Access	N/A		
Parameters	Name	Type	Description
	<SetupName>	String	Text of the design's notes.
	<VariationString>	String	Variation name. Pass empty string for the current nominal variation.
	<filePath>	String	Full file path, including extension *.prof.
Return Value	None.		

Python Syntax	ExportProfile(<SetupName>, <VariationString>, <filePath>, <overwriteIfExists>)
Python Example	oDesign.ExportProfile('Setup1', 'radius=3mm', 'C:\Files\Profile.prof', True)

VB Syntax	ExportProfile <SetupName>, <VariationString>, <filePath>, <overwriteIfExists>
VB Example	oDesign.ExportProfile "Setup1", "radius=3mm", "C:\Files\Profile.prof", True

GetConfigurableData

Obtains configurable data of a specified type

Note:
This command is for internal Ansys use only.

UI Access	N/A		
Parameters	Name	Type	Description
	<type>	String	Specified type.
Return Value	None.		

Python Syntax	GetConfigurableData(<type>)		
Python Example	oDesign.GetConfigurableData ("model")		

VB Syntax	GetConfigurableData <type>		
VB Example	oDesign.GetConfigurableData "model"		

GetData

Note:

This command is for internal Ansys use only.

Python Syntax	GetData()		
Python Example	oDesign.GetData()		

GetDesignValidationInfo

Copies a validation cache file that is generated by oDesign.ValidateDesign() or doing design validation from UI to a file. This file contains more information besides messages. It also has validation results of some validation check items.

If there is no validation cache file, a file won't be generated. This happens if GetDesignValidationInfo is executed and no ValidateDesign was performed.

UI Access	NA		
Parameters	Name	Type	Description
	None		
Return Value	Integer 0 if invalid 1 if valid		

Python Syntax	GetDesignValidationInfo()
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDesign = oProject.GetActiveDesign() oDesign.GetDesignValidationInfo()</pre>

GetGeometryMode

Returns the geometry mode for the design. For Maxwell 2D only.

UI Access	N/A		
Parameters	Name	Type	Description
	None		
Return Value	String "XY" if design is "Cartesian, XY" "about Z" if design is "Cylindrical about Z"		

Python Syntax	GetGeometryMode()
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDesign = oProject.GetActiveDesign() oDesign.GetGeometryMode()</pre>

GetManagedFilePath

Get the path to the project's results folder.

UI Access	N/A
Parameters	None.
Return Value	String containing path where project results are located.

Python Syntax	<code>oDesign.GetManagedFilePath()</code>
Python Example	<code>oDesign.GetManagedFilePath()</code>

VB Syntax	<code>oDesign.GetManagedFilePath</code>
VB Example	<code>oDesign.GetManagedFilePath</code>

GetModule

Returns the IDispatch for the specified module.

UI Access	N/A		
Parameters	Name	Type	Description
	<code><modulename></code>	String	One of the following: • Analysis Module – "AnalysisSetup" • Boundary Module – "BoundarySetup" • Field Overlays Module – "FieldsReporter" • Mesh Module – "MeshSetup" • Optimetrics Module – "Optimetrics" • Radiation Module – "RadField" • Reduce Matrix Module – "ReduceMatrix" • Reporter Module – "ReportSetup" • Simulation Setup Module – "SimSetup"

	• Solutions Module – "Solutions"
Return Value	Module IDispatch

Python Syntax	GetModule (<modulename>)
Python Example	<code>oModule = oDesign.GetModule("SimSetup")</code>

VB Syntax	GetModule <modulename>
VB Example	<code>set oModule = oDesign.GetModule "SimSetup"</code>

GetName

Returns the design name of the active design, in that order separated by a semicolon.

UI Access	N/A
Parameters	None.
Return Value	String indicating the name of the active design.

Python Syntax	GetName()
Python Example	<code>design_name = oDesign.GetName()</code>

VB Syntax	GetName
VB Example	<code>design_name = oDesign.GetName</code>

GetNominalVariation

Returns the current nominal variation.

UI Access	N/A
Parameters	None.
Return Value	String containing current nominal variation.

Python Syntax	GetNominalVariation()
Python Example	<code>oDesign.GetNominalVariation()</code>

VB Syntax	GetNominalVariation
VB Example	<code>oDesign.GetNominalVariation</code>

GetNoteText

Returns the text of the note attached to a design.

UI Access	N/A
Parameters	None.
Return Value	String: text of the design note.

Python Syntax	<code>GetNoteText()</code>
Python Example	<code>oDesign.GetNoteText()</code>

VB Syntax	<code>GetNoteText</code>
VB Example	<code>oDesign.GetNoteText</code>

GetPostProcessingVariables

Returns the list of post-processing variables.

UI Access	N/A
Parameters	None.
Return Value	Returns array containing variables.

Python Syntax	<code>GetPostProcessingVariables()</code>
Python Example	<code>oDesign.GetPostProcessingVariables()</code>

VB Syntax	GetPostProcessingVariables
VB Example	<code>oDesign.GetPostProcessingVariables</code>

GetSelections [Design]

This script serves no function at the Design level. See: GetSelections (Layout Editor), [GetSelections \(Model Editor\)](#), or [Get Selections \(Schematic Editor\)](#).

GetSolutionType

Returns solution type of the design.

UI Access	N/A
Parameters	None.
Return Value	String containing the solution type. Possible values are: "SBR+", "HFSS [Modal Terminal] [Network Composite]","Transient [Network Composite]","Eigenmode", or "Characteristic".

Python Syntax	GetSolutionType()
Python Example	<code>oDesign.GetSolutionType()</code>

VB Syntax	GetSolutionType
VB Example	<code>oDesign.GetSolutionType</code>

GetVariationVariableValue

Returns the value for a specified variation's variable.

UI Access	N/A		
Parameters	Name	Type	Description
	<VariationString>	String	The name of the design variation.
	<VariableName>	String	The name of the variable.
Return Value	Returns a double precision value in SI units, interpreted to mean the value of the variable contained in the variation string.		

Python Syntax	GetVariationVariableValue(<VariationString>, <VariableName>)
Python Example	<pre>oDesign.GetVariationVariableValue('x_size = 2mm y_size = 1mm', 'y_size')</pre>

VB Syntax	GetVariationVariableValue <VariationString>, <VariableName>
VB Example	<pre>varval = oDesign.GetVariationVariableValue "x_size = 2mm y_size = 1mm", "y_size"</pre>

Is2D

Use: Determines whether the current design is 2D.

Command: None

Syntax: Is2D

Return Value: Boolean

True if the design is 2D.

Parameters: None

Example: `oDesign.Is2D`

Is3D

Use: Determines whether the current design is 3D.

Command: None

Syntax: Is3D

Return Value: Boolean

True if the design is 3D.

Parameters: None

Example: `oDesign.Is3D`

Redo [Design]

Reapplies the last design-level command.

UI Access	Edit > Redo.
Parameters	None.
Return Value	None.

Python Syntax	Redo()

Python Example	<code>oDesign.Redo()</code>

VB Syntax	Redo
VB Example	<code>oDesign.Redo</code>

RenameDesignInstance

Renames a design instance.

UI Access	Right-click a design instance in the project tree, and then click Rename on the shortcut menu.		
Parameters	Name	Type	Description
	<OldName>	String	The current name of the design, which must be the design on which this command is invoked.
	<NewName>	String	The new name for the design.
Return Value	None.		

Python Syntax	<code>RenameDesignInstance (<OldName>, <NewName>)</code>
Python Example	<code>oDesign.RenameDesignInstance ("Design1", "Design2")</code>

VB Syntax	<code>RenameDesignInstance <OldName>, <NewName></code>
VB Example	<code>oDesign.RenameDesignInstance "Design1", "Design2"</code>

ResetToTimeZero

Resets a simulation to time zero.

UI Access	CleanStop when running Electronics Desktop in Batchmode.		
Parameters	Name	Type	Description
	<setupName>	String	Name of the simulation setup to be reset.
Return Value	None.		

Python Syntax	ResetToTimeZero(<setupName>)
Python Example	<code>oModule.ResetToTimeZero("Setup1")</code>

VB Syntax	ResetToTimeZero <setupName>
VB Example	<code>oModule.ResetToTimeZero "Setup1"</code>

SetActiveEditor

Sets the active editor.

UI Access	N/A.		
Parameters	Name	Type	Description
	<EditorName>	String	Text of the design's notes.
Return Value	Editor object.		

Python Syntax	SetActiveEditor(<EditorName>)
Python Example	oDesign.SetActiveEditor('3D Modeler')

VB Syntax	SetActiveEditor <EditorName>
VB Example	oDesign.SetActiveEditor "3D Modeler"

SetConductivityThreshold

Use: Sets the thresholds for a perfect conductor and an insulator/conductor.

Command: **Maxwell>Set Material Thresholds** or right click on design instance in project tree and choose **"Set Material Thresholds"**

Syntax: SetConductivityThreshold <PerfectConductorThreshold> <Insulator/ConductorThreshold>

Return Value: None

Parameters: <PerfectConductorThreshold>

Type: <double>

Perfect conductor threshold

<Insulator/ConductorThreshold>

Type: <double>

Insulator/conductor threshold

Example: oDesign.SetConductivityThreshold 1E+030, 1.00

SetDesignSettings [Maxwell]

Use: Sets design settings for the active design. Settings vary with solution type.

Command: Click **Maxwell 3D > Design Settings**, **Maxwell 2D > Design Settings**, or right-click a design instance in the project tree and select **Design Settings**.

Syntax: SetDesignSettings <DesignSettingsArray>

Return Value: None.

Parameters: <DesignSettingsArray>

For 3D Transient solution type:

```
Array("NAME:Design Settings Data",  
      "Allow Material Override:=", <bool>,  
      "Perform Minimal validation:=", <bool>,  
      "PreservTranSolnAfterDatasetEdit:=", <bool>,  
      "ComputeTransientInductance:=", <bool>,  
      "ComputeIncrementalMatrix:=", <bool>,  
      "PerfectConductorThreshold:=", <real>,  
      "InsulatorThreshold:=", <real>,  
      "UseSkewModel:", <bool>,  
      "EnableTranTranLinkWithSimplorer:=", <bool>,  
      "SolveFraction:=" , <bool>, //"True" enables solve using circumferential fraction. Set to "False" by default.  
      "FractionNumber:=" , <int>, //Fraction number to generate the partial model. (Used when "Solve Fraction" is "True".) Set  
      to 1 by default.  
      "HalfPeriodicField:=" , <bool>, //"True" enables half-periodic field. (Used when "Solve Fraction" is "True".) Set to  
      "False" by default.
```

"HalfAxialGeometry:=" , <bool>, //"True" cuts the original geometry in half in the axial direction. (Used when "Solve Fraction" is "True".) Set to "False" by default.

"Multiplier:=" , <int>), //Symmetry multiplier value used when "Solve Fraction" is "False".

"SkipMeshChecks:=" , <bool>

Array("NAME:Model Validation Settings",

"EntityCheckLevel:=" , "Strict", //One of "Strict", "None", "WarningOnly", "Basic"

"IgnoreUnclassifiedObjects:=" , <bool>,

"SkipIntersectionChecks:=" , <bool>

Example:

```
oDesign.SetDesignSettings(  
[  
    "NAME:Design Settings Data",  
    "Allow Material Override:=", True,  
    "Perform Minimal validation:=", False,  
    "EnabledObjects:=" , [],  
    "PreserveTranSolnAfterDatasetEdit:=", True,  
    "ComputeTransientInductance:=", True,  
    "ComputeIncrementalMatrix:=", False,  
    "PerfectConductorThreshold:=", 1E+30,  
    "InsulatorThreshold:=" , 2500000,  
    "UseSkewModel:=" , False,
```

```
"EnableTranTranLinkWithSimplorer:=", False,  
"SolveFraction:=", True,  
"FractionNumber:=", "4",  
"HalfPeriodicField:=", True,  
"HalfAxialGeometry:=", True  
"SkipMeshChecks:=", False  
],  
[  
  "NAME:Model Validation Settings",  
  "EntityCheckLevel:=", "Strict",  
  "IgnoreUnclassifiedObjects:=", True,  
  "SkipIntersectionChecks:=", False  
])
```

For 3D Magnetostatic solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"ComputeIncrementalMatrix:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"SkipMeshChecks:=", <bool>)
```


For 3D Eddy Current solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"SymmetryMultiplier=", <real>,  
"SkipMeshChecks:=" , <bool>)
```

For 3D Electrostatic solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"Multiplier:=", <int>,  
"SkipMeshChecks:=" , <bool>)
```

For 3D DC Conduction solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"SkipMeshChecks:=" , <bool>)
```

For 3D Electric Transient solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"PreservTranSolnAfterDatasetEdit:=", <bool>,  
"SkipMeshChecks:=" , <bool>)
```

For 3D AC Conduction solution type:

```
Array("NAME:Design Settings Data",  
"Allow Material Override:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"SkipMeshChecks:=" , <bool>)
```

For 2D Transient solution type:

```
Array("NAME:Design Settings Data",  
"PreserveTranSolnAfterDatasetEdit:=", <bool>,  
"ComputeTransientInductance:=", <bool>,  
"PerfectConductorThreshold:=", <real>,  
"InsulatorThreshold:=", <real>,  
"ModelDepth:=", <real>,  
"UseSkewModel:=", <bool>,  
"SkewModelType:=", <string>  
//possible values are: "Continuous", "Step", "V-Shape", "User Defined"
```

```
"SkewPart:=", <string>
//possible values are: "Stator", "Rotor"
"SkewAngle:=", <real>,
//for "Continuous", "Step", "V-Shape" model types only
"SkewAngleUnit:=", <string>,
//for "User Defined" model type only; possible values are: "rad", "degsec", "degmin", "deg"
"NumberOfSlices:=", <int>,
//SkewSliceTable array below is for skew model type "User Defined" only.
Array("NAME:SkewSliceTable",
    Array("NAME:OneSliceInfo", "SkewAngle:=", <real>, "SliceLength:=", <real>,
        ...
        Array("NAME:OneSliceInfo", "SkewAngle:=", <real>, "SliceLength:=", <real>,,))

"EnableTranTranLinkWithSimplorer:=", <bool>,
"BackgroundMaterialName:=", <string>
"Multiplier:=", <int>)
```

For 2D Magnetostatic solution type:

```
Array("NAME:Design Settings Data",
    "PerfectConductorThreshold:=", <real>,
    "InsulatorThreshold:=", <real>,
    "ModelDepth:=", <real>,
```

```
"ComputeIncrementalMatrix:=", <bool>,
"BackgroundMaterialName:=", <string>)
```

For 2D Eddy Current, Electrostatic, DC Conduction, and AC Conduction solution types:

```
Array("NAME:Design Settings Data",
"PerfectConductorThreshold:=", <real>,
"InsulatorThreshold:=", <real>,
"SymmetryMultiplier:=", <int>,
"ModelDepth:=", <real>,
"BackgroundMaterialName:=", <string>)
```

For all solution types:

```
Array("NAME:Validation Options",
"EntityCheckLevel:=", "Strict",
//possible values are: Strict, Basic, Warning Only, None
"IgnoreUnclassifiedObjects:=", <bool>,
"SkipIntersectionChecks:=", <bool>),
"Design Validation Settings:=", "Perform full validations"
//possible values are: Perform full validations, Perform minimal validations
```

Example: oDesign.SetDesignSettings(
[

```
"NAME:Design Settings Data",
```

```
"Perform Minimal validation:=", False,  
"EnabledObjects:=" , [],  
"PreserveTranSolnAfterDatasetEdit:=", False,  
"ComputeTransientInductance:=", False,  
"ComputeIncrementalMatrix:=", False,  
"PerfectConductorThreshold:=", 1E+30,  
"InsulatorThreshold:=" , 1,  
"ModelDepth:=" , "1mm",  
"UseSkewModel:=" , True,  
"SkewModelType:=" , 0,  
"SkewAngle:=" , "6deg",  
"NumberOfSlices:=" , "5",  
"EnableTranTranLinkWithSimplorer:=", False,  
"BackgroundMaterialName:=", "vacuum",  
"Multiplier:=" , "fractions"  
],  
[  
"NAME:Model Validation Settings",  
"EntityCheckLevel:=" , "Strict",  
"IgnoreUnclassifiedObjects:=", True,  
"SkipIntersectionChecks:=", False
```

```
] )
```

Python Examples: 2D Transient solution type

```
oDesign.SetDesignSettings(  
    [  
        "NAME:Design Settings Data",  
        "Perform Minimal validation:=", False,  
        "EnabledObjects:=" , [],  
        "PreserveTranSolnAfterDatasetEdit:=", False,  
        "ComputeTransientInductance:=", False,  
        "ComputeIncrementalMatrix:=", False,  
        "PerfectConductorThreshold:=", 1E+30,  
        "InsulatorThreshold:=" , 1,  
        "ModelDepth:=" , "1meter",  
        "UseSkewModel:=" , True,  
        "SkewType:=" , "Continuous",  
        "SkewPart:=" , "Stator",  
        "SkewAngle:=" , "5deg",  
        "NumberOfSlices:=" , "3",  
        "EnableTranTranLinkWithSimplorer:=", False,
```

```
    "BackgroundMaterialName:=", "vacuum",  
    "SolveFraction:=" , False,  
    "Multiplier:=" , "1"  
  ]  
oDesign.SetDesignSettings(  
  [  
    ...  
    "SkewType:=" , "Step",  
    "SkewPart:=" , "Stator",  
    "SkewAngle:=" , "0.6rad",  
    "NumberOfSlices:=" , "2",  
    ...  
  ]  
oDesign.SetDesignSettings(  
  [  
    ...  
    "SkewType:=" , "V-Shape",  
    "SkewPart:=" , "Rotor",  
    "SkewAngle:=" , "5deg",  
    "NumberOfSlices:=" , "7",  
    ...  
  ]  
)
```

```
]
oDesign.SetDesignSettings(
[
  ..
  "SkewType:=" , "User Defined",
  "SkewPart:=" , "Rotor",
  "SkewAngleUnit:=" , "deg",
  "NumberOfSlices:=" , "4",
  [
    "NAME:SkewSliceTable",
    [
      "NAME:OneSliceInfo",
      "SkewAngle:=" , 0.6,
      "SliceLength:=" , 0.1
    ],
    [
      "NAME:OneSliceInfo",
      "SkewAngle:=" , -0.9,
      "SliceLength:=" , 0.2
    ],
  ],
],
```



```
[
  "NAME:OneSliceInfo",
  "SkewAngle:=" , 4,
  "SliceLength:=" , 0.3
],
[
  "NAME:OneSliceInfo",
  "SkewAngle:=" , 18,
  "SliceLength:=" , 0.4
]
],
...
]
```

VBExamples: 2D Transient solution type

```
oDesign.SetDesignSettings
Array("NAME:Design Settings Data",
"Perform Minimal validation:=", false,
"EnabledObjects:=", Array(),
"PreserveTranSolnAfterDatasetEdit:=", false, "ComputeTransientInductance:=", false, "Com-
puteIncrementalMatrix:=", false,
"PerfectConductorThreshold:=", 1E+30,
"InsulatorThreshold:=", 1,
"ModelDepth:=", "1meter",
"UseSkewModel:=", true,
"SkewType:=", "Continuous",
"SkewPart:=", "Rotor",
```

```
"SkewAngle:=", "3rad",  
"NumberOfSlices:=", "3",  
"EnableTranTranLinkWithSimplorer:=", false, "BackgroundMaterialName:=", "vacuum",  
"SolveFraction:=", false,  
"Multiplier:=", "1"),  
Array("NAME:Model Validation Settings",  
"EntityCheckLevel:=", "Strict",  
"IgnoreUnclassifiedObjects:=", false,  
"SkipIntersectionChecks:=", false)
```

```
oDesign.SetDesignSettings  
Array(  
...  
"ModelDepth:=", "1meter",  
"UseSkewModel:=", true,  
"SkewType:=", "Step",  
"SkewPart:=", "Stator",  
"SkewAngle:=", "3rad",  
"NumberOfSlices:=", "4",  
...  
)
```

```
oDesign.SetDesignSettings  
Array(  
...  
"ModelDepth:=", "1meter",  
"UseSkewModel:=", true,  
"SkewType:=", "V-Shape",  
"SkewPart:=", "Stator",  
"SkewAngle:=", "3rad",
```

```
"NumberOfSlices:=", "5",
...
)

oDesign.SetDesignSettings
Array(
...
"ModelDepth:=", "1meter", "UseSkewModel:=", true,
"SkewType:=", "User Defined",
"SkewPart:=", "Rotor",
"SkewAngleUnit:=", "deg",
"NumberOfSlices:=", "5",
Array("NAME:SkewSliceTable",
Array("NAME:OneSliceInfo", "SkewAngle:=", "10", "SliceLength:=", "0.3"),
Array("NAME:OneSliceInfo", "SkewAngle:=", "-2.7", "SliceLength:=", "0.2"),
Array("NAME:OneSliceInfo", "SkewAngle:=", "-1.8", "SliceLength:=", "0.25"),
Array("NAME:OneSliceInfo", "SkewAngle:=", "-0.3", "SliceLength:=", "0.15"),
Array("NAME:OneSliceInfo", "SkewAngle:=", "3", "SliceLength:=", "0.1"))
...
)
```

SetFastTransformationForLayoutComponent

Toggles whether layout components in Maxwell show outlines of objects across layers during view changes. This improves the visualization performance.

UI Access	Fast Transformation		
Parameters	Name	Type	Description
	<Layout ComponentName>	String	Name of the Layout Component.
	Boolean	String	True or False

Return Value	None.

Python Syntax	SetFastTransformationForLayoutComponent ("<LayoutComponentName", <boolean>)
Python Example	<code>oDesign.SetFastTransformationForLayoutComponent ("LC1_1", False)</code>

SetObjectAttributesForLayoutComponent

Sets visualization attributes for layout components in Maxwell for a named component for layers, nets, and objects.

UI Access	Object Attributes		
Parameters	Name	Type	Description
	<Layout ComponentName>	String	Name of the Layout Component.
Return Value	None.		

Python Syntax	SetObjectAttributesForLayoutComponent (["<LayoutComponentName", [<ObjectAttributesinNetMode>][<ObjectAttributesinLayerMode>], [<ObjectAttributesinObjectMode>])
Python Example	<pre>oDesign.SetObjectAttributesForLayoutComponent (["Name:=", "ShowDielectric:=", , "LC1_1", , False,</pre>

```
"DisplayMode:=" , 2,
[
  "NAME:ObjectAttributesInLayerMode",
  "GND_1_L2:=" , [True,True,0],
  "GND_2_L4:=" , [True,True,0],
  "GND_3_L6:=" , [True,True,0],
  "GND_4_L9:=" , [True,True,0],
  "GND_5_L11:=" , [True,True,0],
  "Inner_Layer_3_L10:=" , [True,True,0],
  "Top_L1:=" , [True,True,100],
  "VCC1_L7:=" , [True,True,0],
  "VCC2_L8:=" , [True,True,0]
],
[
  "NAME:ObjectAttributesInNetMode",
  "GND:=" , [True,True,29],
  "VCC:=" , [True,True,0],
  "neg:=" , [True,True,67],
  "pos:=" , [True,True,67]
],
[
```

```
"NAME:ObjectAttributesInObjectMode",  
  "line_1:="          , [True,True,0],  
  "line_2:="          , [True,True,0],  
  "line_3:="          , [True,True,100],  
  "line_4:="          , [True,True,100],  
  "rect_6:="          , [True,True,0],  
  "rect_17:="         , [True,True,0],  
  "rect_18:="         , [True,True,0],  
  "rect_19:="         , [True,True,0],  
  "rect_20:="         , [True,True,0],  
  "rect_21:="         , [True,True,0],  
  "rect_22:="         , [True,True,0],  
  "line_24:="         , [True,True,100],  
  "via_0:="           , [True,True,100],  
  "via_1:="           , [True,True,100],  
  "via_2:="           , [True,True,100],  
  "via_3:="           , [True,True,100]  
]  
])
```

SetObjectDeformation

Sets the deformation of objects when calling Ansys Electronics Desktop from Ansys Workbench.

For Maxwell magnetostatic, transient, electrostatic, and eddy current designs, sets the deformation of objects when calling Ansys Electronics Desktop from Ansys Workbench. (Can also be used for setting deformation of objects in Maxwell 2D/3D magnetostatic and transient standalone designs.)

UI Access	N/A		
Parameters	Name	Type	Description
	<ObjectArray>	Array	Array containing the following parameters: "EnabledStressObjectsName:= "The names of objects that are enabled for deformation. - Optionally, for Maxwell 2D magnetostatic and transient designs: " Enabled2DXYStress<double array>:=" Every 3 numbers are a group of initial stress data (real with unit) for one object in the EnabledStressObjectsName list. (Applies only to 2D magnetostatic and transient XY designs.) " Enabled2DRZStress<double array>:=" Every 3 numbers are a group of initial stress data (real with unit) for one object in the EnabledStressObjectsName list. (Applies only to 2D magnetostatic and transient RZ designs.) - Optionally, for Maxwell 3D magnetostatic and transient designs: "Enabled3DStress<double array>:=" Every 6 real numbers with unit are a group of initial stress data for one object in the EnabledStressObjectsName list.

Return Value	None.

Python Syntax	SetObjectDeformation(<ObjectArray>)
Python Example	<pre>oDesign.SetObjectDeformation(["EnabledStressObjectsName:=" , ["Cylinder1", "Sphere1"]])</pre>

For Maxwell 3D Magnetostatic and Transient designs with initial stress

Python Syntax	SetObjectDeformation(<ObjectArray>)
Python Example	<pre>oDesign.SetObjectDeformation(["EnabledStressObjectsName:=" , ["Cylinder1", "Sphere1"], "Enabled3DStressXXObjects:=", ["12.35psi","30.64psi"], "Enabled3DStressYYObjects:=", ["5.72psi","16.93psi"], "Enabled3DStressZZObjects:=", ["14.75psi","12.22psi"], "Enabled3DStressXYObjects:=", ["64.3psi","28.13psi"], "Enabled3DStressYZObjects:=", ["19.43psi","58.67psi"], "Enabled3DStressXZObjects:=", ["27.63psi","40.31psi"])</pre>

For Maxwell 2D Magnetostatic and Transient XY designs with initial stress

Python Syntax	SetObjectDeformation(<ObjectArray>)
Python Example	<pre>oDesign.SetObjectDeformation(["EnabledStressObjectsName:=", ["Cylinder1", "Sphere1"], "Enabled2DXYStressXXObjects:=", ["12.35psi", "30.64psi"], "Enabled2DXYStressYYObjects:=", ["5.72psi", "16.93psi"], "Enabled2DXYStressXYObjects:=", ["64.3psi", "28.13psi"],])</pre>

For Maxwell 2D Magnetostatic and Transient RZ designs with initial stress

Python Syntax	SetObjectDeformation(<ObjectArray>)
Python Example	<pre>oDesign.SetObjectDeformation(["EnabledStressObjectsName:=", ["Cylinder1", "Sphere1"], "Enabled2DRZStressRRObjects:=", ["12.35psi", "30.64psi"], "Enabled2DRZStressZZObjects:=", ["5.72psi", "16.93psi"], "Enabled2DRZStressRZObjects:=", ["64.3psi", "28.13psi"],])</pre>

VB Syntax	SetObjectDeformation <ObjectArray>
VB Example	<pre>oDesign.SetObjectDeformation Array("EnabledStressObjectsName:=", Array("Cylinder1", "Sphere1"))</pre>

SetObjectTemperature

Sets the temperature of objects.

UI Access	Maxwell > Set Object Temperature		
Parameters	Name	Type	Description
	<TemperatureSettings>	Array	["NAME": <SettingName>, "IncludeTemperatureDependence:=", <boolean, True to include temperature dependence. False to not include.>, "EnableFeedback:=", <boolean, True to enable feedback. False to not enable feedback.>, "Temperatures:=", <Temperatures>]
	<Temperatures>	Array	Objects and temperatures.
Return Value	None.		

Python Syntax	oDesign.SetObjectTemperature(<TemperatureSettings>)
Python Example	<pre>oDesign.SetObjectTemperature(["NAME:TemperatureSettings", "IncludeTemperatureDependence:=", True, "EnableFeedback:=" , False, "Temperatures:=" , ["RegularPolyhedron1","22cel","Polygon1","22cel"])</pre>

VB Syntax	<code>oDesign.SetObjectTemperature <TemperatureSettings></code>
VB Example	<pre>oDesign.SetObjectTemperature Array("NAME:TemperatureSettings", "IncludeTemperatureDependence:=", _ true, "EnableFeedback:=", true, "Temperatures:=", Array("RegularPolyhedron1", _ "22cel", "Polygon1", "22cel"))</pre>

SetShowAllLayoutComponents

This command will show/hide all layout components in a Maxwell design.

UI Access	Edit Layout... View > Visibility > Show All or Hide All		
Parameters	Name	Type	Description
	Boolean	String	True or False
Return Value	None		

Python Syntax	<code>SetShowAllLayoutComponents (<boolean>)</code>
Python Example	<pre>oDesign = oProject.GetActiveDesign() oDesign.SetShowAllComponentLayouts(True)</pre>

SetShowLayoutForLayoutComponent

Layout visualization, rather than bounding box, for a Layout Component in a Maxwell design.

UI Access	Edit Layout...		
Parameters	Name	Type	Description
	<Layout ComponentName>	String	Name of the Layout Component.
	Boolean	String	True or False
Return Value	None.		

Python Syntax	SetShowLayoutForLayoutComponent ("<LayoutComponentName", <boolean>)
Python Example	oDesign.SetShowLayoutForLayoutComponent ("LC1_1", True)

SetSolutionType (Maxwell)

Sets the solution type for the design.

UI Access	Maxwell3D or Maxwell2D > Solution Type , or right-click a design instance in the Project Manager tree and select Solution Type .		
Parameters	Name	Type	Description
	<SolutionType>	String	Possible values for 3D designs are "Magnetostatic", "EddyCurrent", "EddyCurrentAPhiFormulation", "Transient", "TransientAPhiFormulation", "Electrostatic", "DCConduction", "ElectroDCConduction", "ElectricTransient", "ACConduction" Possible values for 2D designs are "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "DCConduction", "ACConduction"
	<GeomMode>	String	Possible values for 2D geometry mode: "XY", "about Z"
Return Value	None		

Python Syntax	SetSolutionType(<SolutionType>, <Geomode>)
Python Example	oDesign.SetSolutionType("EddyCurrentAPhiFormulation")

VB Syntax	SetSolutionType <SolutionType>, <AutoOpenMode>, <ModelExteriorAsIE>
VB Example	oDesign.SetSolutionType "ElectroDCConduction" oDesign.SetSolutionType "Electrostatic", "XY"

Solve

Performs one or more simulation. The next script command will not be executed until the simulation(s) are complete.

UI Access	Select solution setup(s). Right-click and select Analyze .		
Parameters	Name	Type	Description
	<SimulationNames>	Array	Array containing string simulation names.
Return Value	Integer: • 0 – Simulation(s) completed. • 1 – Simulation error. • -1 – Command execution error.		

Python Syntax	Solve <SimulationNames>
Python Example	oDesign.Solve(['Setup1', 'Setup2', 'Setup3'])

VB Syntax	Solve <SimulationNames>
VB Example	oDesign.Solve Array("Setup1", "Setup2", "Setup3")

StopSimLink

Stops linked simulation.

UI Access	N/A		
Parameters	Name	Type	Description
	<simId>	Integer	ID of specified simulation.
	<abort>	Boolean	Whether to stop the running simulation: • True - abort the running simulation. • False - do not abort running simulation.
Return Value	None.		

Python Syntax	StopSimLink(<simId>, <abort>)
Python Example	oDesign.StopSimLink(18, True)

VB Syntax	StopSimLink <simId>, <abort>
VB Example	oDesign.StopSimLink 18, true

RunToolkit

Runs a Python toolkit script, applying it to the active design. The toolkit script itself may have prerequisites, such as sweeps defined, or specific model characteristics, such as port definitions.

Important:

Full scripting for cable modeling is not supported and no arguments are allowed in the script. The **RunToolkit** command will not automatically create a cable bundle, but will simply open the **Cable Modeling** dialog box. You will have to manually input your parameters.

UI Access	[Maxwell] > Toolkit > [Script Name].		
Parameters	Name	Type	Description
	<LibraryType>	String	Name of the library in which the script is located.
	<ToolkitName>	String	Name of the Python script.
	<ToolkitArg>	Array	Structured array containing arguments required by the toolkit script.
Return Value	User-defined solution(s) and/or output(s).		

Python Syntax	RunToolkit(<LibraryType>, <ToolkitName>, <ToolkitArg>)		
Python Example	<pre>oDesign.RunToolkit("SysLib", "Cable Modeling/Automotive Cable Bundle", []) oDesign.RunToolkit("SysLib", "Cable Modeling/Oil-Gas Cable Bundle", [])</pre>		

VB Syntax	RunToolkit <LibraryType>, <ToolkitName>, <ToolkitArg>		
VB Example	<pre>oDesign.RunToolkit "SysLib", "Cable Modeling/Automotive Cable Bundle", Array()</pre>		

	<code>oDesign.RunToolkit "SysLib", "Cable Modeling/Oil-Gas Cable Bundle", Array()</code>

Undo [Design]

Cancels the last design-level command.

UI Access	Edit > Undo
Parameters	None.
Return Value	None.

Python Syntax	<code>Undo()</code>
Python Example	<code>oDesign.Undo()</code>

VB Syntax	<code>Undo</code>
VB Example	<code>oDesign.Undo</code>

ValidateDesign

Returns whether a design is valid.

UI Access	Maxwell > Validation Check.
Parameters	None.

Return Value	Integer: • 1 – Validation passed. • 0 – Validation failed.
Python Syntax	ValidateDesign()
Python Example	<code>oDesign.ValidateDesign()</code>
VB Syntax	ValidateDesign
VB Example	<code>oDesign.ValidateDesign</code>

ValidateLink

Note:

This command is for internal Ansys use only.

Python Syntax	ValidateLink()
Python Example	<code>oDesign.ValidateLink()</code>

This page intentionally
left blank.

10 - 3D Modeler Editor Script Commands

3D Modeler commands should be executed by the "3D Modeler" editor:

```
Set oEditor = oDesign.SetActiveEditor("3D Modeler")
```

```
oEditor.<CommandName>
```

Conventions Used in this Chapter:

<AttributesArray>

<AttributesArray> takes the following structure:

```
Array("NAME:Attributes",  
      "Name:=", <string>,  
      "Flags:=", <string>,  
      "Color:=", <string>,  
      "Transparency:=", <value>,  
      "PartCoordinateSystem:=", <string>,  
      "UDMId:=", <string>,  
      "MaterialValue:=", <string>,  
      "SurfaceMaterialValue:=", <string>,  
      "Solveinside:=", <boolean>,  
      "ShellElement:=", <boolean>,  
      "ShellElementThickness:=", <string>,  
      "IsMaterialEditable:=", <boolean>,
```

```
"UseMaterialAppearance:=", <boolean>,  
"IsLightweight:=", <boolean>)
```

Where:

- **Flags** – Takes a string containing "NonModel" and/or "Wireframe", separated by the # character. For example, "NonModel#Wireframe".
- **Color** – Takes a string containing an RGB triple, formatted as "(*<RGB>*)". For example, "(255 255 255)".
- **Transparency** – Takes a value between 0 and 1.
- **PartCoordinateSystem** – Orientation of the primitive. The name of one of the defined coordinate systems should be specified.
- **UDMId** – Takes a string containing an ID.
- **MaterialValue** – Takes a string of the material name.
- **SurfaceMaterialValue** – Takes a string of the surface material name.
- **Solveinside** – Takes a boolean value.
- **ShellElement** – Takes a boolean value specifying whether or not a shell element is present.
- **ShellElementThickness** – Takes a string containing the shell element thickness. If element is not present, pass empty string.
- **IsMaterialEditable** – Takes a boolean value.
- **IsLightweight** – Takes a boolean value.

<SelectionsArray>

<SelectionsArray> typically takes the following structure:

```
Array("NAME:Selections",  
"Selections:=", <string>)
```

Where:

- **Selections** – Takes a comma-separated list of parts on which to perform the action. For example, "Rect1, Rect2, Rect3".

In some cases, <SelectionsArray> takes additional parameters:

```
Array("NAME:Selections",
    "AllowRegionDependentPartSelectionForPMLCreation:=", <boolean>,
    "AllowRegionSelectionForPMLCreation:=", <boolean>,
    "Selections:=", <string>,
    "NewPartsModelFlag:=" , <string>,
    "UseCurrentCS:=", <boolean>)
```

Where:

- **AllowRegionDependentPartSelectionForPMLCreation** – Takes a boolean value. See individual script for whether this parameter is required.
- **AllowRegionSelectionForPMLCreation** – Takes a boolean value. See individual script for whether this parameter is required.
- **Selections** – Takes a comma-separated list of parts on which to perform the action. For example, "Rect1, Rect2, Rect3".
- **NewPartsModelFlag** – Takes either string "Model" or string "Nonmodel". See individual script for whether this parameter is required.
- **UseCurrentCS** – Takes a boolean value. See individual script for whether this parameter is required. Use [GetActiveCoordinateSystem](#) to determine the current CS.

Note:

Selections is the only parameter required in *all* 3D Modeler Editor scripts. See individual scripts for additional required parameters.

Organization

3D Modeler editor scripts are organized into the following categories:

[Draw Menu Commands](#)

[Edit Menu Commands](#)

[Modeler Menu Commands](#)

[Cable Modeling Commands](#)

[Other oEditor Commands](#)

Draw Menu Commands

[Create3DComponent](#)

[CreateBondwire](#)

[CreateBox](#)

[CreateCircle](#)

[CreateCone](#)

[CreateCutplane](#)

[CreateCylinder](#)

[CreateEllipse](#)

[CreateEquationCurve](#)

[CreateEquationSurface](#)

[CreateHelix](#)

[CreatePoint](#)

[CreatePolyline](#)

[CreateRectangle](#)

[CreateRegion](#)

[CreateRegularPolygon](#)

[CreateRegularPolyhedron](#)

[CreateSphere](#)

[CreateSpiral](#)

[CreateSubregion](#)

[CreateTorus](#)

[CreateUserDefinedModel](#)

[CreateUserDefinedPart](#)

[Edit3DComponent](#)

[EditPolyline](#)

[Get3DComponentDefinitionNames](#)

[Get3DComponentInstanceNames](#)

[Get3DComponentMaterialNames](#)

[Get3DComponentMaterialProperties](#)

[Get3DComponentParameters](#)

[Insert3DComponent](#)

[InsertPolylineSegment](#)

[SweepAlongPath](#)

[SweepAlongVector](#)

[SweepAroundAxis](#)

[SweepFacesAlongNormal](#)

[SweepFacesAlongNormalWithAttributes](#)

[UpdateComponentDefinition](#)

Create3DComponent

Create a 3D component.

UI Access	Draw>3D Component Library>Create 3D Component		
Parameters	Name	Type	Description
	<NAME>	string	CreateData
	<ComponentName>	string	The name of the component
	<Company>	string	Company name
	<Company URL>	string	Company website address
	<Model Number>	string	Component model designation
	<Help URL>	string	Help link address
	<Version>	string	Component version
	<Notes>	string	Notes about the component
	<IconType>	string	
	<Owner>	string	Owner name
	<Email>	string	Email address
	<Date>	string	Component creation date
	<HasLabel>	bool	
	<IsEncrypted>	bool	Is the component encrypted?
	<AllowEdit>	bool	Permit component to be edited?
	<SecurityMessage>	string	Security information
	<Password>	string	Enter password
	<EditPassword>	string	
	<PasswordType>	string	
	<HideContents>	bool	Hide selected contents?

	<ReplaceNames>	bool	Allow replacement of object and material names?
	<ComponentOutline>	string	"None" or "Bounding Box"
	<IncludedParts>	list	Array of included parts
	<HiddenParts>	list	Array of hidden parts
	<IncludedCS>	list	Array of included coordinate system(s)
	<ReferenceCS>	string	Reference coordinate system
	<IncludedParameters>	list	Array of included parameters
	<ParameterDescription>	list	Array of parameter descriptions
	<IsLicensed>	bool	Is the component licensed?
	<LicensingDllName>	string	License name
	<VendorComponentIdentifier>	string	Vendor component Identifier
	<PublicKeyFile>	string	Public Key filename - including path to the key
	<NAME>	string	
	<Excitations>	list	Array of excitations
	<Filename>	string	Path and name of component being saved.
	<NAME>	string	
	<ImageFile>	string	
Return Value	None		

Python Syntax	Create3DComponent (<i>ComponentParameters</i>)
Python Example	<pre>oEditor.Create3DComponent (["NAME:CreateData", "ComponentName:=", "CoilWindingsNew", "Company:=", "", "Company URL:=", ""</pre>

```
"Model Number:=" , "",
"Help URL:=" , "",
"Version:=" , "1.0",
"Notes:=" , "",
"IconType:=" , "",
"Owner:=" , "J Doe",
"Email:=" , "jdoe@",
"Date:=" , "4:40:58 PM Oct 14, 2019",
"HasLabel:=" , False,
"IsEncrypted:=" , False,
"AllowEdit:=" , False,
"SecurityMessage:=" , "",
"Password:=" , "",
"EditPassword:=" , "",
"PasswordType:=" , "UnknownPassword",
"HideContents:=" , True,
"ReplaceNames:=" , True,
"ComponentOutline:=" , "None",
"IncludedParts:=" , ["Coil_1st_Section1",
    "Coil_2nd_Section1"],
"HiddenParts:=" , [],
"IncludedCS:=" , [],
"ReferenceCS:=" , "Global",
"IncludedParameters:=" , ["nCond","wCurrent","wPhase"],
"ParameterDescription:=" , ["nCond:=", "", "wCurrent:=", "",
    "wPhase:=", ""],
"IsLicensed:=" , False,
"LicensingDllName:=" , "",
"VendorComponentIdentifier:=" , "",
"PublicKeyFile:=" , ""
],
[
"NAME:DesignData",
"Excitations:=" , ["CoilTerminal1","CoilTerminal2",
```


VB Syntax	Create3DComponent (<i>ComponentParameters</i>)
VB Example	<pre>oEditor.Create3DComponent Array("NAME:CreateData", "ComponentName:=", "CoilWindings", "Company:=", _ "", "Company URL:=", "", "Model Number:=", "", "Help URL:=", "", "Version:=", _ "1.0", "Notes:=", "", "IconType:=", "", "Owner:=", "J Doe", "Email:=", "jdoe@", "Date:=", _ "4:33:50 PM Oct 14, 2019", "HasLabel:=", false, "IsEncrypted:=", false, "AllowEdit:=", _ false, "SecurityMessage:=", "", "Password:=", "", "EditPassword:=", "", "PasswordType:=", _ "UnknownPassword", "HideContents:=", true, "ReplaceNames:=", true, "ComponentOutline:=", _ "None", "IncludedParts:=", Array("Coil_1st_Section1", "Coil_2nd_Section1"), "Hid- denParts:=", Array(), "IncludedCS:=", Array(), "ReferenceCS:=", _ "Global", "IncludedParameters:=", Array("nCond", "wCurrent", "wPhase"), "Para- meterDescription:=", Array("nCond:=", _ "", "wCurrent:=", "", "wPhase:=", ""), "IsLicensed:=", false, "LicensingDllName:=", _ "", "VendorComponentIdentifier:=", "", "PublicKeyFile:=", ""), Array("NAME:DesignData", "Excitations:=", Array(_ "CoilTerminal1", "CoilTerminal2", "Winding1", "Winding2")), _ "C:/work/3Dcomponents/CoilWindings.a3dcomp,Array("NAME:ImageFile", "ImageFile:=", ""))</pre>

CreateBondwire

Creates a bondwire.

UI Access	Draw > Bondwire.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:BondwireParameters", "WireType:=", <string("JEDEC_4Points", "JEDEC_5Points", or "LOW")>, "WireDiameter:=", <string>, "NumSides:=", <value>, "XPadPos:=", <value>, "YPadPos:=", <value>, "ZPadPos:=", <value>, "XDir:=", <value>, "YDir:=", <value>, "ZDir:=", <value>, "Distance:=", <value>, "h1:=", <value>, "h2:=", <value>, "alpha:=", <value>,

			"beta:=", <value>, "WhichAxis:=", <string("X","Y", or "Z")>, "ReverseDirection:=", <boolean>
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateBondwire(<Parameters>, <Attributes>)
Python Example	<pre> oEditor.CreateBondwire(["NAME:BondwireParameters", "WireType:=" , "JEDEC_4Points", "WireDiameter:=" , "0.025mm", "NumSides:=" , "6", "XPadPos:=" , "1.6mm", "YPadPos:=" , "-0.2mm", "ZPadPos:=" , "0mm", "XDir:=" , "-2.2mm", "YDir:=" , "-1.4mm", "ZDir:=" , "0mm", "Distance:=" , "2.60768096208106mm", "h1:=" , "0.2mm", "h2:=" , "0mm", </pre>

```

"alpha:="                , "80deg",
"beta:="                  , "0",
"WhichAxis:="            , "Z",
"ReverseDirection:="     , True
],
["NAME:Attributes",
  "Name:="                , "Bondwire1",
  "Flags:="                , "",
  "Color:="                , "(143 175 143)",
  "Transparency:="        , 0,
  "PartCoordinateSystem:=", "Global",
  "UDMId:="                , "",
  "MaterialValue:="        , "\"vacuum\"",
  "SurfaceMaterialValue:=", "\"\"",
  "SolveInside:="          , True,
  "ShellElement:="         , False,
  "ShellElementThickness:=", "0mm",
  "IsMaterialEditable:="   , True,
  "UseMaterialAppearance:=", False,
  "IsLightweight:="        , False

```

	l)

VB Syntax	CreateBondwire <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateBondwire Array("NAME:BondwireParameters", "WireType:=", "LOW", "WireDia- meter:=", _ "0.025mm", "NumSides:=", "6", "XPadPos:=", "6mm", "YPadPos:=", "-2.5mm", "ZPadPos:=", _ "0mm", "XDir:=", "-10mm", "YDir:=", "4mm", "ZDir:=", "0mm", "Distance:=", _ "10.770329614269mm", "h1:=", "0.2mm", "h2:=", "0mm", "alpha:=", "80deg", "beta:=", _ "80deg", "WhichAxis:=", "Z", "ReverseDirection:=", false), Array("NAME:Attributes", "Name:=", _ "Bondwire2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateBox

Creates a box.

UI Access	Draw > Box.

Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array ("NAME:BoxParameters", "XPosition:=", <string>, "YPosition:=", <string>, "ZPosition:=", <string>, "XSize:=" , <string>, "YSize:=" , <string>, "ZSize:=" , <string>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateBox(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateBox(["NAME:BoxParameters", "XPosition:=" , "0.5mm", "YPosition:=" , "-6.5mm", "ZPosition:=" , "0mm", "XSize:=" , "2mm", "YSize:=" , "1.5mm",</pre>


```
"ZSize:=" , "1.5mm"
],
["NAME:Attributes",
  "Name:=" , "Box3",
  "Flags:=" , "",
  "Color:=" , "(143 175 143)",
  "Transparency:=" , 0,
  "PartCoordinateSystem:=" , "Global",
  "UDMId:=" , "",
  "MaterialValue:=" , "\"copper\"",
  "SurfaceMaterialValue:=" , "\"\"",
  "SolveInside:=" , False,
  "ShellElement:=" , False,
  "ShellElementThickness:=" , "0mm",
  "IsMaterialEditable:=" , True,
  "UseMaterialAppearance:=" , False,
  "IsLightweight:=" , False
])
```

**VB Syn-
tax****CreateBox <Parameters>, <Attributes>**

VB Example	<pre>oEditor.CreateBox Array("NAME:BoxParameters", "XPosition:=", "0mm", "YPosition:=", _ "4mm", "ZPosition:=", "0mm", "XSize:=", "-4mm", "YSize:=", "1mm", "ZSize:=", _ "2mm"), Array("NAME:Attributes", "Name:=", "Box4", "Flags:=", "", "Color:=", _ "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", "Global", "UDMId:=", _ "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "Sur- faceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, _ "UseMaterialAppearance:=", false, "IsLightweight:=", false)</pre>

CreateCircle

Creates a circle.

UI Access	Draw > Circle.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:CircleParameters", "IsCovered:=", <boolean>, "XCenter:=", <value>, "YCenter:=", <value>, "ZCenter:=", <value>,

			"Radius:=", <value>, "WhichAxis:=", <string> "NumSegments:=", <string containing integer>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateCircle(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateCircle(["NAME:CircleParameters", "IsCovered:=" , True, "XCenter:=" , "5.5mm", "YCenter:=" , "-3mm", "ZCenter:=" , "0mm", "Radius:=" , "0.707106781186548mm", "WhichAxis:=" , "Z", "NumSegments:=" , "0"], ["NAME:Attributes", "Name:=" , "Circle1", "Flags:=" , "", "Color:=" , "(143 175 143)",</pre>

	<pre>"Transparency:=" , 0, "PartCoordinateSystem:=", "Global", "UDMId:=" , "", "MaterialValue:=" , "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False l)</pre>

VB Syntax	CreateCircle <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateCircle Array("NAME:CircleParameters", "IsCovered:=", true, "XCenter:=", _ "7mm", "YCenter:=", "-6mm", "ZCenter:=", "0mm", "Radius:=", "0.5mm", "WhichAxis:=", _ "Z", "NumSegments:=", "0"), Array("NAME:Attributes", "Name:=", "Circle2", "Flags:=", _ "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "Sur- faceMaterialValue:=", _</pre>

```
"" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _
false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa-
terialAppearance:=", _
false, "IsLightweight:=", false)
```

CreateCone

Creates a cone.

UI Access	Draw > Cone.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:ConeParameters", "XCenter:=", <string>, "YCenter:=", <string>, "ZCenter:=", <string>, "WhichAxis:=", <string>, "Height:=", <string>, "BottomRadius:=", <string>, "TopRadius:=", <string>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateCone(<Parameters>, <Attributes>)

Python Example

```
oEditor.CreateCone(  
  ["NAME:ConeParameters",  
    "XCenter:=" , "3mm",  
    "YCenter:=" , "-4.5mm",  
    "ZCenter:=" , "0mm",  
    "WhichAxis:=" , "Z",  
    "Height:=" , "2.5mm",  
    "BottomRadius:=" , "2.82842712474619mm",  
    "TopRadius:=" , "2.23606797749979mm"  
  ],  
  ["NAME:Attributes",  
    "Name:=" , "Cone1",  
    "Flags:=" , "",  
    "Color:=" , "(143 175 143)",  
    "Transparency:=" , 0,  
    "PartCoordinateSystem:=" , "Global",  
    "UDMId:=" , "",  
    "MaterialValue:=" , "\"copper\"",  
    "SurfaceMaterialValue:=" , "\"\"",  
    "SolveInside:=" , False,
```

	<pre>"ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False])</pre>

VB Syntax	CreateCone <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateCone Array("NAME:ConeParameters", "XCenter:=", "3mm", "YCenter:=", _ "-3mm", "ZCenter:=", "0mm", "WhichAxis:=", "Z", "Height:=", "4mm", "BottomRadius:=", _ "1.11803398874989mm", "TopRadius:=", "2.06155281280883mm"), Array("NAME:Attributes", "Name:=", _ "Cone1", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordin- ateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateCutplane

Creates a cutplane.

UI Access	Draw > Plane.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:PlaneParameters", "PlaneBaseX:=", <string>, "PlaneBaseY:=", <string>, "PlaneBaseZ:=", <string>, "PlaneNormalX:=", <string>, "PlaneNormalY:=", <string>), "PlaneNormalZ:=", <string>)
	<AttributesArray>	Array	See: AttributesArray . CreateCutplane only takes the Name and Color attributes.
Return Value	None.		

Python Syntax	CreateCutplane(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateCutplane(["NAME:PlaneParameters", "PlaneBaseX:=" , "-0.6mm", "PlaneBaseY:=" , "-0.8mm", "PlaneBaseZ:=" , "0mm", "PlaneNormalX:=" , "1.2mm",</pre>

	<pre> "PlaneNormalY:=" , "0.2mm", "PlaneNormalZ:=" , "0mm"], ["NAME:Attributes", "Name:=" , "Plane1", "Color:=" , "(143 175 143)"]) </pre>

VB Syntax	CreateCutplane <Parameters>, <Attributes>
VB Example	<pre> oEditor.CreateCutplane Array("NAME:PlaneParameters", "PlaneBaseX:=", "0.6mm", "PlaneBaseY:=", _ "1.2mm", "PlaneBaseZ:=", "0mm", "PlaneNormalX:=", "0.4mm", "PlaneNormalY:=", _ "0.6mm", "PlaneNormalZ:=", "0mm"), Array("NAME:Attributes", "Name:=", "Plane2", "Col- or:=", _ "(143 175 143)") </pre>

CreateCylinder

Creates a cylinder.

UI Access	Draw > Cylinder.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:CylinderParameters",

			"XCenter:=", <string>, "YCenter:=", <string>, "ZCenter:=", <string>, "Radius:=", <string>, "Height:=", <string>, "WhichAxis:=", <string>, "NumSides:=", <string containing integer>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateCylinder(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateCylinder(["NAME:CylinderParameters", "XCenter:=" , "6mm", "YCenter:=" , "-4.5mm", "ZCenter:=" , "0mm", "Radius:=" , "0.5mm", "Height:=" , "4.5mm", "WhichAxis:=" , "Z", "NumSides:=" , "0"]</pre>

```

],
["NAME:Attributes",
    "Name:="                , "Cylinder1",
    "Flags:="                , "",
    "Color:="                , "(143 175 143)",
    "Transparency:="        , 0,
    "PartCoordinateSystem:=" , "Global",
    "UDMId:="                , "",
    "MaterialValue:="        , "\"copper\"",
    "SurfaceMaterialValue:=" , "\"\"",
    "SolveInside:="          , False,
    "ShellElement:="         , False,
    "ShellElementThickness:=" , "0mm",
    "IsMaterialEditable:="    , True,
    "UseMaterialAppearance:=" , False,
    "IsLightweight:="         , False
])

```

VB Syntax	CreateCylinder <Parameters>, <Attributes>
VB Exempl-	oEditor.CreateCylinder Array("NAME:CylinderParameters", "XCenter:=", "1.5mm", "YCenter:=", _

e	<pre>"6.5mm", "ZCenter:=", "0mm", "Radius:=", "1mm", "Height:=", "1mm", "WhichAxis:=", _ "Z", "NumSides:=", "0"), Array("NAME:Attributes", "Name:=", "Cylinder2", "Flags:=", _ "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateEllipse

Creates an ellipse.

UI Access	Draw > Ellipse.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:EllipseParameters", "IsCovered:=", <string>, "XCenter:=", <string>, "YCenter:=", <string>, "ZCenter:=", <string>, "MajRadius:=", <string>,

			"Ratio:=", <string>, "WhichAxis:=", <string>, "NumSegments:=", <string>
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateEllipse(<Parameters>, <Attributes>)		
Python Example	<pre> oEditor.CreateEllipse(["NAME:EllipseParameters", "IsCovered:=" , True, "XCenter:=" , "0.6mm", "YCenter:=" , "-0.6mm", "ZCenter:=" , "0mm", "MajRadius:=" , "0.2mm", "Ratio:=" , "7", "WhichAxis:=" , "Z", "NumSegments:=" , "0"], ["NAME:Attributes", "Name:=" , "Ellipse1", "Flags:=" , "", </pre>		

	<pre>"Color:=" , "(143 175 143)", "Transparency:=" , 0, "PartCoordinateSystem:=", "Global", "UDMId:=" , "", "MaterialValue:=" , "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False])</pre>

VB Syntax	CreateEllipse <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateEllipse Array("NAME:EllipseParameters", "IsCovered:=", true, "XCenter:=", - "-0.4mm", "YCenter:=", "-3.2mm", "ZCenter:=", "0mm", "MajRadius:=", "0.4mm", "Ratio:=", - "0.5", "WhichAxis:=", "Z", "NumSegments:=", "0"), Array("NAME:Attributes", "Name:=", _</pre>

	<pre>"Ellipse2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMaterialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateEquationCurve

Creates an equation-based curve.

UI Access	Draw > Equation-Based Curve.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:EquationBasedCurveParameters", "XtFunction:=", <string>, "YtFunction:=", <string>, "ZtFunction:=", <string>, "tStart:=", <string>, "tEnd:=", <string>, "NumOfPointsOnCurve:=", <string>, "Version:=", <integer>), <polylineArray(optional>))</pre>

	<table><tr><td><AttributesArray></td><td>Array</td><td>Structured array. See: AttributesArray.</td></tr></table>			<AttributesArray>	Array	Structured array. See: AttributesArray .
<AttributesArray>	Array	Structured array. See: AttributesArray .				
Return Value	None.					

Python Syntax	CreateEquationCurve(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateEquationCurve(["NAME:EquationBasedCurveParameters", "XtFunction:=" , "1", "YtFunction:=" , "3", "ZtFunction:=" , "32", "tStart:=" , "1", "tEnd:=" , "3", "NumOfPointsOnCurve:=" , "0", "Version:=" , 1, ["NAME:PolylineXSection", "XSectionType:=" , "None", "XSectionOrient:=" , "Auto", "XSectionWidth:=" , "0", "XSectionTopWidth:=" , "0", "XSectionHeight:=" , "0", "XSectionNumSegments:=" , "0",</pre>


```
        "XSectionBendType:="      , "Corner"
    ]
],
["NAME:Attributes",
    "Name:="                      , "EquationCurve1",
    "Flags:="                     , "",
    "Color:="                     , "(143 175 143)",
    "Transparency:="              , 0,
    "PartCoordinateSystem:="      , "Global",
    "UDMId:="                     , "",
    "MaterialValue:="             , "\"copper\"",
    "SurfaceMaterialValue:="      , "\"\"",
    "SolveInside:="               , False,
    "ShellElement:="              , False,
    "ShellElementThickness:="     , "0mm",
    "IsMaterialEditable:="        , True,
    "UseMaterialAppearance:="     , False,
    "IsLightweight:="             , False
])
```

**VB
Syntax**

CreateEquationCurve <Parameters>, <Attributes>

VB Example	<pre>oEditor.CreateEquationCurve Array("NAME:EquationBasedCurveParameters", "XtFunction:=", _ "1", "YtFunction:=", "3", "ZtFunction:=", "32", "tStart:=", "1", "tEnd:=", "3", "NumOfPointsOnCurve:=", _ "0", "Version:=", 1, Array("NAME:PolylineXSection", "XSectionType:=", "None", "XSec- tionOrient:=", _ "Auto", "XSectionWidth:=", "0", "XSectionTopWidth:=", "0", "XSectionHeight:=", _ "0", "XSectionNumSegments:=", "0", "XSectionBendType:=", "Corner")), Array("NAME:At- tributes", "Name:=", _ "EquationCurve2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", _ 0, "PartCoordinateSystem:=", "Global", "UDMId:=", "", "MaterialValue:=", _ "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", _ false, "ShellElement:=", false, "ShellElementThickness:=", "0mm", "IsMa- terialEditable:=", _ true, "UseMaterialAppearance:=", false, "IsLightweight:=", false)</pre>

CreateEquationSurface

Creates an equation-based surface.

UI Access	Draw > Equation-Based Surface.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:EquationBasedSurfaceParameters",

			"XuvFunction:=" , <string equation containing Function, Operators and/or quantities _u, _v, or PI>, "YuvFunction:=" , <string equation containing Function, Operators and/or quantities _u, _v, or PI>, "ZuvFunction:=" , <string equation containing Function, Operators and/or quantities _u, _v, or PI>, "uStart:=" , <string>, "uEnd:=" , <string>, "vStart:=" , <string>, "vEnd:=" , <string>, "Version:=" , <integer>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateEquationSurface(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateEquationSurface(["NAME:EquationBasedSurfaceParameters", "XuvFunction:=" , "_u", "YuvFunction:=" , "_v", "ZuvFunction:=" , "sin(_u)+cos(_v)", "uStart:=" , "1", "uEnd:=" , "10",</pre>

```
"vStart:="          , "1",
"vEnd:="            , "10",
"Version:="         , 1
],
["NAME:Attributes",
  "Name:="          , "EquationSurface1",
  "Flags:="         , "",
  "Color:="         , "(143 175 143)",
  "Transparency:="  , 0,
  "PartCoordinateSystem:=", "Global",
  "UDMId:="         , "",
  "MaterialValue:=" , "\"copper\"",
  "SurfaceMaterialValue:=", "\"\"",
  "SolveInside:="   , False,
  "ShellElement:="  , False,
  "ShellElementThickness:=", "0mm",
  "IsMaterialEditable:=" , True,
  "UseMaterialAppearance:=", False,
  "IsLightweight:="   , False
])
```

VB Syntax	CreateEquationSurface <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateEquationSurface Array("NAME:EquationBasedSurfaceParameters", "XuvFunction:=", _ "_u", "YuvFunction:=", "_v", "ZuvFunction:=", "sin(_u)+cos(_v)", "uStart:=", "1", "uEnd:=", _ "10", "vStart:=", "1", "vEnd:=", "10", "Version:=", 1), Array("NAME:Attributes", "Name:=", _ "EquationSurface2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", _ 0, "PartCoordinateSystem:=", "Global", "UDMId:=", "", "MaterialValue:=", _ "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", _ false, "ShellElement:=", false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", _ true, "UseMaterialAppearance:=", false, "IsLightweight:=", false)</pre>

CreateHelix

Creates a helix based on a sweep of specified objects.

UI Access	Draw > Helix.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:HelixParameters", "XCenter:=" , <string>,

			<pre>"YCenter:=" , <string>, "ZCenter:=" , <string>, "XStartDir:=" , <string>, "YStartDir:=" , <string>, "ZStartDir:=" , <string>, "NumThread:=" , <string>, "RightHand:=" , <boolean>, "RadiusIncrement:=" , <string>, "Thread:=" , <string>)</pre>
Return Value	None.		

Python Syntax	CreateHelix(<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.CreateHelix(["NAME:Selections", "Selections:=" , "EquationSurface2", "NewPartsModelFlag:=" , "Model"], ["NAME:HelixParameters", "XCenter:=" , "10000mm", "YCenter:=" , "40000mm",</pre>

	<pre> "ZCenter:=" , "0mm", "XStartDir:=" , "0mm", "YStartDir:=" , "10000mm", "ZStartDir:=" , "0mm", "NumThread:=" , "1", "RightHand:=" , True, "RadiusIncrement:=" , "0mm", "Thread:=" , "1mm"]) </pre>

VB Syntax	CreateHelix <SelectionsArray>, <Parameters>
VB Example	<pre> oEditor.CreateHelix Array("NAME:Selections", "Selections:=", "EquationSurface1", "NewPartsModelFlag:=", _ "Model"), Array("NAME:HelixParameters", "XCenter:=", "1000mm", "YCenter:=", "10000mm", "ZCenter:=", _ "2.39945573144418mm", "XStartDir:=", "-41000mm", "YStartDir:=", "-10000mm", "ZStartDir:=", _ "-2.39945573144418mm", "NumThread:=", "1", "RightHand:=", true, "RadiusIncrement:=", _ "0mm", "Thread:=", "1mm") </pre>

CreatePoint

Creates a point.

UI Access	Draw > Point.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:PointParameters", "PointX=", <value>, "PointY=", <value>, "PointZ=", <value>)
	<AttributesArray>	Array	Structured array. See: AttributesArray . CreatePoint takes only the Name and Color attributes.
Return Value	None.		

Python Syntax	CreatePoint(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreatePoint(["NAME:PointParameters", "PointX=" , "0.2mm", "PointY=" , "-0.2mm", "PointZ=" , "0mm"], ["NAME:Attributes",</pre>

	<pre> Name:=" , "Point1", Color:=" , "(143 175 143) "]) </pre>

VB Syntax	CreatePoint <Parameters>, <Attributes>
VB Example	<pre> oEditor.CreatePoint Array("NAME:PointParameters", "PointX:=" , "0.2mm", "PointY:=" , "-0.2mm", "PointZ:=" , "0mm") , Array("NAME:Attributes", "Name:=" , "Point1", "Color:=" , "(143 175 143) ") </pre>

CreateUserDefinedPart

Creates a user-defined part.

UI Access	Draw > User-Defined Primitive > [Part].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array.

			<pre>Array("NAME:UserDefinedPrimitiveParameters", "DllName:=" , <string>, "Version:=" , <string>, "NoOfParameters:=" , <integer>, "Library:=" , <string>, <paramVectorArray>)</pre>
	<paramVectorArray>	Array	Structured array containing arrays for each pair: <pre>Array("NAME:ParamVector", <pair>, <pair>, <pair>,...)</pre>
	<pair>	Array	Structured array: <pre>Array("NAME:Pair", "Name:=" , <string>, "Value:=" , <string>)</pre>
	<Attributes>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateUserDefinedPart(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateUserDefinedPart(["NAME:UserDefinedPrimitiveParameters", "DllName:=" , "RMxpert/LapCoil.dll",</pre>

```
"Version:="          , "16.0",
"NoOfParameters:="   , 22,
"Library:="          , "syslib",
  ["NAME:ParamVector",
    ["NAME:Pair",
      "Name:="        , "DiaGap",
      "Value:="       , "100mm"
    ],
    ["NAME:Pair",
      "Name:="        , "DiaYoke",
      "Value:="       , "20mm"
    ],
    ["NAME:Pair",
      "Name:="        , "Length",
      "Value:="       , "100mm"
    ],
    ["NAME:Pair",
      "Name:="        , "Skew",
      "Value:="       , "0deg"
    ],
    ["NAME:Pair",
```

```

"Name:="                , "Slots",
"Value:="                , "18"
],
["NAME:Pair",
"Name:="                , "SlotType",
"Value:="                , "1"
],
["NAME:Pair",
"Name:="                , "Hs0",
"Value:="                , "1mm"
],
["NAME:Pair",
"Name:="                , "Hs1",
"Value:="                , "1mm"
],
["NAME:Pair",
"Name:="                , "Hs2",
"Value:="                , "10mm"
],
["NAME:Pair",

```

```
"Name:=" , "Bs0",
"Value:=" , "2.5mm"
],
["NAME:Pair",
"Name:=" , "Bs1",
"Value:=" , "8mm"
],
["NAME:Pair",
"Name:=" , "Bs2",
"Value:=" , "5mm"
],
["NAME:Pair",
"Name:=" , "Rs",
"Value:=" , "0mm"
],
["NAME:Pair",
"Name:=" , "FilletType",
"Value:=" , "0"
],
["NAME:Pair",
"Name:=" , "Layers",
```

```

"Value:="                , "2"
],
["NAME:Pair",
"Name:="                  , "CoilPitch",
"Value:="                  , "4"
],
["NAME:Pair",
"Name:="                  , "EndExt",
"Value:="                  , "5mm"
],
["NAME:Pair",
"Name:="                  , "SpanExt",
"Value:="                  , "25mm"
],
["NAME:Pair",
"Name:="                  , "BendAngle",
"Value:="                  , "0deg"
],
["NAME:Pair",
"Name:="                  , "SegAngle",

```

```

    "Value:="                , "10deg"
  ],
  ["NAME:Pair",
    "Name:="                , "LenRegion",
    "Value:="                , "200mm"
  ],
  ["NAME:Pair",
    "Name:="                , "InfoCoil",
    "Value:="                , "0"
  ]
]
],
["NAME:Attributes",
  "Name:="                  , "LapCoil1",
  "Flags:="                 , "",
  "Color:="                 , "(143 175 143)",
  "Transparency:="          , 0,
  "PartCoordinateSystem:=" , "Global",
  "UDMId:="                 , "",
  "MaterialValue:="          , "\"copper\"",
  "SurfaceMaterialValue:=" , "\"\"",

```

	<pre> "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False] </pre>

VB Syntax	CreateUserDefinedPart <Parameters>, <Attributes>
VB Example	<pre> oEditor.CreateUserDefinedPart Array("NAME:UserDefinedPrimitiveParameters", "DllName:=", _ "SegmentedHelix/RectHelix.dll", "Version:=", "1.0", "NoOfParameters:=", 8, "Library:=", _ "syslib", Array("NAME:ParamVector", Array("NAME:Pair", "Name:=", "RectHeight", "Value:=", _ "1mm"), Array("NAME:Pair", "Name:=", "RectWidth", "Value:=", "2mm"), Array("NAME:Pair", "Name:=", _ "StartHelixRadius", "Value:=", "10mm"), Array("NAME:Pair", "Name:=", "RadiusChange", "Value:=", _ "0mm"), Array("NAME:Pair", "Name:=", "Pitch", "Value:=", "3mm"), Array("NAME:Pair", "Name:=", _ "Turns", "Value:=", "2"), Array("NAME:Pair", "Name:=", "SegmentsPerTurn", "Value:=", _ </pre>


```
"36"), Array("NAME:Pair", "Name:=", "RightHanded", "Value:=", "1")), Array("NAME:Attributes", "Name:=", _
"RectHelix1", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0,
"PartCoordinateSystem:=", _
"Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _
"" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _
false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMaterialAppearance:=", _
false, "IsLightweight:=", false)
```

CreatePolyline

Creates a polyline.

UI Access	Draw > Line.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:PolylineParameters", "IsPolylineCovered:=", <bool>, "IsPolylineClosed:=", <bool>, <PolylinePointsArray>, <PolylineSegmentsArray>)
	<PolylinePointsArray>	Array	Array("NAME:PolylinePoints", <OnePointArray>, <OnePointArray>, ...)
	<OnePointArray>	Array	Array("NAME:PLPoint",

			"X:=", <value>, "Y:=", <value>, "Z:=", <value>))
	<PolylineSegmentsArray>	Array	<PolylineSegmentsArray> Array("NAME:PolylineSegments", <OneSegmentArray>, <OneSegmentArray>, ...)
	<OneSegmentArray>	Array	Array("NAME:PLSegment", "SegmentType:=", <"Line", "Arc", "Spline", or "AngularArc">, "StartIndex:=", <value>, "NoOfPoints:=", <value>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreatePolyline(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreatePolyline(["NAME:PolylineParameters", "IsPolylineCovered:=" , True, "IsPolylineClosed:=" , False, ["NAME:PolylinePoints", ["NAME:PLPoint",</pre>

```
"X:="                , "20000mm",
"Y:="                , "-20000mm",
"Z:="                , "0mm"
],
["NAME:PLPoint",
"X:="                , "-90000mm",
"Y:="                , "20000mm",
"Z:="                , "0mm"
],
["NAME:PLPoint",
"X:="                , "10000mm",
"Y:="                , "-140000mm",
"Z:="                , "0mm"
]
],
["NAME:PolylineSegments",
  ["NAME:PLSegment",
    "SegmentType:="    , "Line",
    "StartIndex:="     , 0,
    "NoOfPoints:="     , 2
  ],
```

```
[ "NAME:PLSegment",  
  "SegmentType:="          , "Line",  
  "StartIndex:="           , 1,  
  "NoOfPoints:="           , 2  
]  
],  
[ "NAME:PolylineXSection",  
  "XSectionType:="         , "None",  
  "XSectionOrient:="        , "Auto",  
  "XSectionWidth:="         , "0mm",  
  "XSectionTopWidth:="      , "0mm",  
  "XSectionHeight:="        , "0mm",  
  "XSectionNumSegments:="   , "0",  
  "XSectionBendType:="      , "Corner"  
]],  
[ "NAME:Attributes",  
  "Name:="                  , "Polyline1",  
  "Flags:="                  , "",  
  "Color:="                  , "(143 175 143)",  
  "Transparency:="          , 0,
```

	<pre> "PartCoordinateSystem:=", "Global", "UDMId:=", " ", "MaterialValue:=", "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=", False, "ShellElement:=", False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", True, "UseMaterialAppearance:=", False, "IsLightweight:=", False] </pre>

VB Syntax	CreatePolyline <Parameters>, <Attributes>
VB Example	<pre> oEditor.CreatePolyline Array("NAME:PolylineParameters", "IsPolylineCovered:=", true, "IsPolylineClosed:=", _ false, Array("NAME:PolylinePoints", Array("NAME:PLPoint", "X:=", "40000mm", "Y:=", _ "50000mm", "Z:=", "0mm"), Array("NAME:PLPoint", "X:=", "-150000mm", "Y:=", "-50000mm", "Z:=", _ "0mm")), Array("NAME:PolylineSegments", Array("NAME:PLSegment", "SegmentType:=", "Line", "StartIndex:=", _ 0, "NoOfPoints:=", 2)), Array("NAME:PolylineXSection", "XSectionType:=", "None", "XSec- tionOrient:=", _ </pre>

	<pre>"Auto", "XSectionWidth:=", "0mm", "XSectionTopWidth:=", "0mm", "XSectionHeight:=", _ "0mm", "XSectionNumSegments:=", "0", "XSectionBendType:=", "Corner")), Array("NAME:At- tributes", "Name:=", _ "Polyline2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateRectangle

Creates a rectangle.

UI Access	Draw > Rectangle.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:RectangleParameters", "IsCovered=" , <boolean>, "XStart=" , <string>, "YStart=" , <string>, "ZStart=" , <string>,

			"Width:=" , <string>, "Height:=" , <string>, "WhichAxis:=" , <string "X", "Y", or "Z">)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateRectangle(<Parameters>, <Attributes>)
Python Example	<pre> oEditor.CreateRectangle(["NAME:RectangleParameters", "IsCovered:=" , True, "XStart:=" , "-80000mm", "YStart:=" , "-90000mm", "ZStart:=" , "0mm", "Width:=" , "20000mm", "Height:=" , "30000mm", "WhichAxis:=" , "Z"], ["NAME:Attributes", "Name:=" , "Rectangle1", "Flags:=" , "", "Color:=" , "(143 175 143)", </pre>

	<pre>"Transparency:=" , 0, "PartCoordinateSystem:=", "Global", "UDMId:=" , "", "MaterialValue:=" , "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False])</pre>

VB Syntax	CreateRectangle <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateRectangle Array("NAME:RectangleParameters", "IsCovered:=", true, "XStart:=", _ "10000mm", "YStart:=", "-40000mm", "ZStart:=", "0mm", "Width:=", "40000mm", "Height:=", _ "10000mm", "WhichAxis:=", "Z"), Array("NAME:Attributes", "Name:=", "Rectangle2", "Flags:=", _ "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _</pre>

	<pre>"Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMaterialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateRegion

Creates a region containing the design.

UI Access	Draw > Region.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:RegionParameters", "+XPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">, "+XPadding:=", <string X value>, "-XPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">, "-XPadding:=", <string -X value>, "+YPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">, "+YPadding:=", <string Y value>, "-YPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">,

			<code>"-YPadding:=", <string -Y value>, "+ZPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">, "+ZPadding:=", <string Z value>, "-ZPaddingType:=", <string "Percentage Offset", "Absolute Offset", or "Absolute Position">, "-ZPadding:=", <string -Z value>)</code>
	<code><AttributesArray></code>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	<code>CreateRegion(<Parameters>, <Attributes>)</code>		
Python Example	<pre>oEditor.CreateRegion(["NAME:RegionParameters", "+XPaddingType:=" , "Percentage Offset", "+XPadding:=" , "0", "-XPaddingType:=" , "Percentage Offset", "-XPadding:=" , "0", "+YPaddingType:=" , "Percentage Offset", "+YPadding:=" , "0", "-YPaddingType:=" , "Percentage Offset", "-YPadding:=" , "0",</pre>		

```

"+ZPaddingType:="      , "Percentage Offset",
"+ZPadding:="          , "0",
"-ZPaddingType:="      , "Percentage Offset",
"-ZPadding:="          , "0"
],
["NAME:Attributes",
  "Name:="              , "Region",
  "Flags:="              , "Wireframe#",
  "Color:="              , "(143 175 143)",
  "Transparency:="       , 0,
  "PartCoordinateSystem:=", "Global",
  "UDMId:="              , "",
  "MaterialValue:="      , "\"vacuum\"",
  "SurfaceMaterialValue:=", "\"\"",
  "SolveInside:="        , False,
  "ShellElement:="       , False,
  "ShellElementThickness:=", "nan ",
  "IsMaterialEditable:=" , True,
  "UseMaterialAppearance:=", False,
  "IsLightweight:="      , False
])

```

VB Syntax	CreateRegion <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateRegion Array("NAME:RegionParameters", "+XPaddingType:=", _ "Percentage Offset", "+XPadding:=", "0", "-XPaddingType:=", "Percentage Offset", "-XPadding:=", _ "0", "+YPaddingType:=", "Percentage Offset", "+YPadding:=", "0", "-YPaddingType:=", _ "Percentage Offset", "-YPadding:=", "0", "+ZPaddingType:=", "Percentage Offset", "+ZPadding:=", _ "0", "-ZPaddingType:=", "Percentage Offset", "-ZPadding:=", "0"), Array("NAME:Attributes", "Name:=", _ "Region", "Flags:=", "Wireframe#", "Color:=", "(143 175 143)", "Transparency:=", _ 0, "PartCoordinateSystem:=", "Global", "UDMId:=", "", "MaterialValue:=", _ "" & Chr(34) & "vacuum" & Chr(34) & "", "SurfaceMaterialValue:=", "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", _ false, "ShellElement:=", false, "ShellElementThickness:=", "nan ", "IsMaterialEditable:=", _ true, "UseMaterialAppearance:=", false, "IsLightweight:=", false)</pre>

CreateRegularPolyhedron

Creates a regular polyhedron.

UI Access	Draw > Regular Polyhedron.

Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:PolyhedronParameters", "XCenter:=" , <string>, "YCenter:=" , <string>, "ZCenter:=" , <string>, "XStart:=" , <string>, "YStart:=" , <string>, "ZStart:=" , <string>, "Height:=" , <string>, "NumSides:=" , <string containing number greater than 2>, "WhichAxis:=" , <string "X", "Y", or "Z">)</pre>
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateRegularPolyhedron(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateRegularPolyhedron(["NAME:PolyhedronParameters", "XCenter:=" , "40000mm", "YCenter:=" , "-80000mm", "ZCenter:=" , "0mm",</pre>

```

"XStart:="          , "50000mm",
"YStart:="          , "-70000mm",
"ZStart:="          , "0mm",
"Height:="          , "50000mm",
"NumSides:="        , "8",
"WhichAxis:="       , "Z"
],
["NAME:Attributes",
  "Name:="           , "RegularPolyhedron1",
  "Flags:="          , "",
  "Color:="          , "(143 175 143)",
  "Transparency:="   , 0,
  "PartCoordinateSystem:=", "Global",
  "UDMId:="          , "",
  "MaterialValue:="   , "\"copper\"",
  "SurfaceMaterialValue:=", "\"\"",
  "SolveInside:="     , False,
  "ShellElement:="    , False,
  "ShellElementThickness:=", "0mm",
  "IsMaterialEditable:=" , True,

```

	<pre>"UseMaterialAppearance:=", False, "IsLightweight:=" , False 1)</pre>

VB Syntax	CreateRegularPolyhedron <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateRegularPolyhedron Array("NAME:PolyhedronParameters", "XCenter:=", _ "-10000mm", "YCenter:=", "-50000mm", "ZCenter:=", "0mm", "XStart:=", "0mm", "YStart:=", _ "-50000mm", "ZStart:=", "0mm", "Height:=", "90000mm", "NumSides:=", "8", "WhichAxis:=", _ "Z"), Array("NAME:Attributes", "Name:=", "RegularPolyhedron2", "Flags:=", "", "Col- or:=", _ "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", "Global", "UDMId:=", _ "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "Sur- faceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateRegularPolygon

Creates a regular polygon.

UI Access	Draw > Regular Polygon.

Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:RegularPolygonParameters", "IsCovered:=" , <boolean>, "XCenter:=" , <string>, "YCenter:=" , <string>, "ZCenter:=" , <string>, "XStart:=" , <string>, "YStart:=" , <string>, "ZStart:=" , <string>, "NumSides:=" , <string containing number greater than 2>, "WhichAxis:=" , <string "X", "Y", or "Z">)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateRegularPolygon(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateRegularPolygon(["NAME:RegularPolygonParameters", "IsCovered:=" , True, "XCenter:=" , "-70000mm",</pre>


```

"YCenter:="          , "-100000mm",
"ZCenter:="          , "0mm",
"XStart:="           , "-50000mm",
"YStart:="           , "-80000mm",
"ZStart:="           , "0mm",
"NumSides:="         , "12",
"WhichAxis:="        , "Z"
],
["NAME:Attributes",
  "Name:="            , "Polygon1",
  "Flags:="           , "",
  "Color:="           , "(143 175 143)",
  "Transparency:="    , 0,
  "PartCoordinateSystem:=", "Global",
  "UDMId:="           , "",
  "MaterialValue:="   , "\"copper\"",
  "SurfaceMaterialValue:=", "\"\"",
  "SolveInside:="     , False,
  "ShellElement:="    , False,
  "ShellElementThickness:=", "0mm",
  "IsMaterialEditable:=" , True,

```

	<pre>"UseMaterialAppearance:=", False, "IsLightweight:=" , False l)</pre>

VB Syntax	CreateRegularPolygon <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateRegularPolygon Array("NAME:RegularPolygonParameters", "IsCovered:=", _ true, "XCenter:=", "-60000mm", "YCenter:=", "40000mm", "ZCenter:=", "0mm", "XStart:=", _ "-50000mm", "YStart:=", "50000mm", "ZStart:=", "0mm", "NumSides:=", "8", "WhichAxis:=", _ "Z"), Array("NAME:Attributes", "Name:=", "Polygon2", "Flags:=", "", "Color:=", _ "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", "Global", "UDMId:=", _ "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "Sur- faceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateSphere

Creates a sphere.

UI Access	Draw > Sphere.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:SphereParameters", "XCenter:=", <string>, "YCenter:=", <string>, "ZCenter:=", <string>, "Radius:=", <string>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateSphere(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateSphere(["NAME:SphereParameters", "XCenter:=" , "-40000mm", "YCenter:=" , "-130000mm", "ZCenter:=" , "0mm", "Radius:=" , "22360.6797749979mm"], ["NAME:Attributes", "Name:=" , "Sphere1", "Flags:=" , "",</pre>

	<pre>"Color:=" , "(143 175 143)", "Transparency:=" , 0, "PartCoordinateSystem:=", "Global", "UDMId:=" , "", "MaterialValue:=" , "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False])</pre>

VB Syntax	CreateSphere <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateSphere Array("NAME:SphereParameters", "XCenter:=", "-120000mm", "YCenter:=", _ "90000mm", "ZCenter:=", "0mm", "Radius:=", "10000mm"), Array("NAME:Attributes", "Name:=", _ "Sphere2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _</pre>

```
"Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _
"" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _
false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMaterialAppearance:=", _
false, "IsLightweight:=", false)
```

CreateSpiral

Creates a spiral by sweeping the specified object(s).

UI Access	Draw > Spiral.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:SpiralParameters", "XCenter:=" , <string>, "YCenter:=" , <string>, "ZCenter:=" , <string>, "YStartDir:=" , <string>, "ZStartDir:=" , <string>, "NumThread:=" , <string>, "RightHand:=" , <boolean>, "RadiusIncrement:=" , <string>)
Return Value	None.		

Python Syntax	CreateSpiral(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateSpiral(["NAME:Selections", "Selections:=" , "Polygon2", "NewPartsModelFlag:=" , "Model"], ["NAME:SpiralParameters", "XCenter:=" , "-70000mm", "YCenter:=" , "50000mm", "ZCenter:=" , "0mm", "XStartDir:=" , "-60000mm", "YStartDir:=" , "-10000mm", "ZStartDir:=" , "0mm", "NumThread:=" , "1", "RightHand:=" , True, "RadiusIncrement:=" , "1mm"])</pre>
VB Syntax	CreateSpiral <Parameters>, <Attributes>

VB Example	<pre>oEditor.CreateSpiral Array("NAME:Selections", "Selections:=", "Rectangle2", "NewPartsModelFlag:=", _ "Model"), Array("NAME:SpiralParameters", "XCenter:=", "30000mm", "YCenter:=", _ "-30000mm", "ZCenter:=", "0mm", "XStartDir:=", "-90000mm", "YStartDir:=", _ "-30000mm", "ZStartDir:=", "0mm", "NumThread:=", "1", "RightHand:=", false, "Radi- usIncrement:=", _ "1mm")</pre>

CreateTorus

Creates a torus.

UI Access	Draw > Torus.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:TorusParameters", "XCenter:=" , <string>, "YCenter:=" , <string>, "ZCenter:=" , <string>, "MajorRadius:=" , <string>, "MinorRadius:=" , <string>, "WhichAxis:=" , <string "X", "Y", or "Z">)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateTorus(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateTorus(["NAME:TorusParameters", "XCenter:=" , "0.6mm", "YCenter:=" , "-0.6mm", "ZCenter:=" , "0mm", "MajorRadius:=" , "0.365028153987289mm", "MinorRadius:=" , "0.0821854415126694mm", "WhichAxis:=" , "Z"], ["NAME:Attributes", "Name:=" , "Torus1", "Flags:=" , "", "Color:=" , "(143 175 143)", "Transparency:=" , 0, "PartCoordinateSystem:=" , "Global", "UDMId:=" , "", "MaterialValue:=" , "\"copper\"", "SurfaceMaterialValue:=" , "\"\"", "SolveInside:=" , False,])</pre>

	<pre>"ShellElement:=" , False, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=", False, "IsLightweight:=" , False])</pre>

VB Syntax	CreateTorus <Parameters>, <Attributes>
VB Example	<pre>oEditor.CreateTorus Array("NAME:TorusParameters", "XCenter:=", "0.6mm", "YCenter:=", _ "-2mm", "ZCenter:=", "0mm", "MajorRadius:=", "0.365028153987288mm", "MinorRadius:=", _ "0.0821854415126694mm", "WhichAxis:=", "Z"), Array("NAME:Attributes", "Name:=", _ "Torus2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordin- ateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false)</pre>

CreateUserDefinedModel

Creates a user-defined model.

UI Access	Draw > User-Defined Model > [Model].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:UserDefinedModelParameters", <definitionArray>, <optionsArray>, <geometryParamsArray>, "DllName:=", <string filepath>, "Library:=", <string>, "Version:=", <string> "ConnectionID:=", <string>)</pre>
	<definitionArray>	Array	Structured array containing string "NAME:Definition".
	<optionsArray>	Array	Structured array containing string "NAME:Options".
	<geometryParamsArray>	Array	Structured array containing arrays for individual parameters: <pre>Array("NAME:GeometryParameters", Array("NAME:UDMParam", "Name:=" , <string>, "Value:=" , <string>, "PropType2:=" , <integer>, "PropFlag2:=" , <integer>))</pre> Required UDM parameters depend on the UDM being created. To see which properties apply to a UDM, right-click the UDM in the Project

		Tree and select Properties. Then select the Parameters tab. PropType2 can be any of the following: • 0 – Property takes a string value. • 1 – Property is a menu option. • 2 – Property takes a number (integer or double). • 3 – Property takes a value (numbers, variables, or expressions). • 4 – Property is a file name. • 5 – Property corresponds to a check box. • 6 – Property specifies a 3D position. PropFlag2 can be any of the following: • 0 – No flags • 1 – Read-only • 2 – Must be integer • 4 – Must be real • 8 – Hidden PropFlag2 values can be combined. For example, a read-only property that must be an integer would take the value 3. A hidden property that must be real would take the value 12. These values are further described in the <code>UserDefinedPrimitiveStructures.h</code> file included with the installation under <code>AnsysEM[Version]\Win64\UserDefinedPrimitives\Examples\Headers\</code>
Return Value	None.	

Pytho-	CreateUserDefinedModel(<Parameters>)

n Syntax	
Python Example	<pre>oEditor.CreateUserDefinedModel(["NAME:UserDefinedModelParameters", ["NAME:Definition"], ["NAME:Options"], ["NAME:GeometryParams", ["NAME:UDMParam", "Name:=" , "ILD Thickness (ILD)", "Value:=" , "0.006mm", "PropType2:=" , 3, "PropFlag2:=" , 4], ["NAME:UDMParam", "Name:=" , "Line Spacing (LS)", "Value:=" , "0.004mm", "PropType2:=" , 3, "PropFlag2:=" , 4], ["NAME:UDMParam",</pre>

```

"Name:="                , "Line Thickness (LT) ",
"Value:="               , "0.005mm",
"PropType2:="          , 3,
"PropFlag2:="          , 4
],
["NAME:UDMParam",
"Name:="                , "Line Width (LW) ",
"Value:="               , "0.004mm",
"PropType2:="          , 3,
"PropFlag2:="          , 4
],
["NAME:UDMParam",
"Name:="                , "No. of Turns (N) ",
"Value:="               , "2",
"PropType2:="          , 3,
"PropFlag2:="          , 2
],
["NAME:UDMParam",
"Name:="                , "Outer Diameter (OD) ",
"Value:="               , "0.15mm",
"PropType2:="          , 3,

```

```

"PropFlag2:="          , 4
],
["NAME:UDMParam",
"Name:="                , "Substrate Thickness",
"Value:="               , "0.2mm",
"PropType2:="           , 3,
"PropFlag2:="           , 4
],
["NAME:UDMParam",
"Name:="                , "Inductor Type",
"Value:="               , "\"Square,Square,Octagonal,Circular,Square-Differential,Octagonal-Dif-
ferential,Circular-Differential\"",
"DataType:="            , "String",
"PropType2:="           , 1,
"PropFlag2:="           , 0
],
["NAME:UDMParam",
"Name:="                , "Underpass Thickness (UT)",
"Value:="               , "0.001mm",
"PropType2:="           , 3,

```

	<pre> "PropFlag2:=" , 4], ["NAME:UDMParam", "Name:=" , "Via Thickness (VT) ", "Value:=" , "0.001mm", "PropType2:=" , 3, "PropFlag2:=" , 4]], "DllName:=" , "Maxwell3D/OnDieSpiralInductor.py", "Library:=" , "syslib", "Version:=" , "2.0", "ConnectionID:=" , ""]) </pre>

VB Syntax	CreateUserDefinedModel <Parameters>
VB Example	<pre> oEditor.CreateUserDefinedModel Array("NAME:UserDefinedModelParameters", Array ("NAME:Definition"), Arra ("NAME:Options"), Array("NAME:GeometryParams", Array("NAME:UDMParam", "Name:=", _ "ILD Thickness (ILD)", "Value:=", "0.006mm", "PropType2:=", 3, "PropFlag2:=", _ 4), Array("NAME:UDMParam", "Name:=", "Line Spacing (LS)", "Value:=", "0.004mm", </pre>

```
"PropType2:=", _
3, "PropFlag2:=", 4), Array("NAME:UDMParam", "Name:=", "Line Thickness (LT)",
"Value:=", _
"0.005mm", "PropType2:=", 3, "PropFlag2:=", 4), Array("NAME:UDMParam", "Name:=", _
"Line Width (LW)", "Value:=", "0.004mm", "PropType2:=", 3, "PropFlag2:=", 4), Array
("NAME:UDMParam",
"Name:=", _
"No. of Turns (N)", "Value:=", "2", "PropType2:=", 3, "PropFlag2:=", 2), Array
("NAME:UDMParam", "Name:=",
_
"Outer Diameter (OD)", "Value:=", "0.15mm", "PropType2:=", 3, "PropFlag2:=", 4), Array
("NAME:UDMParam",
"Name:=", _
"Substrate Thickness", "Value:=", "0.2mm", "PropType2:=", 3, "PropFlag2:=", 4), Array
("NAME:UDMParam",
"Name:=", _
"Inductor Type", "Value:=", _
"" & Chr(34) & "Square,Square,Octagonal,Circular,Square-Differential,Octagonal-" & _
"Differential,Circular-Differential" & Chr(34) & "", "DataType:=", "String",
"PropType2:=", _
1, "PropFlag2:=", 0), Array("NAME:UDMParam", "Name:=", "Underpass Thickness (UT)",
"Value:=", _
"0.001mm", "PropType2:=", 3, "PropFlag2:=", 4), Array("NAME:UDMParam", "Name:=", _
```



```
"Via Thickness (VT)", "Value:=", "0.001mm", "PropType2:=", 3, "PropFlag2:=", 4)),
"DllName:=", _
"Maxwell3D/OnDieSpiralInductor.py", "Library:=", "syslib", "Version:=", "2.0", "Con-
nectionID:=", _
"" )
```

CreateUserDefinedPart

Creates a user-defined part.

UI Access	Draw > User-Defined Primitive > [Part].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:UserDefinedPrimitiveParameters", "DllName:=" , <string>, "Version:=" , <string>, "NoOfParameters:=" , <integer>, "Library:=" , <string>, <paramVectorArray>)
	<paramVectorArray>	Array	Structured array containing arrays for each pair: Array("NAME:ParamVector", <pair>, <pair>, <pair>, ...)
	<pair>	Array	Structured array: Array("NAME:Pair",

			"Name:=" , <string>, "Value:=" , <string>)
	<Attributes>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateUserDefinedPart(<Parameters>, <Attributes>)
Python Example	<pre>oEditor.CreateUserDefinedPart(["NAME:UserDefinedPrimitiveParameters", "DllName:=" , "RMxpert/LapCoil.dll", "Version:=" , "16.0", "NoOfParameters:=" , 22, "Library:=" , "syslib", ["NAME:ParamVector", ["NAME:Pair", "Name:=" , "DiaGap", "Value:=" , "100mm"], ["NAME:Pair", "Name:=" , "DiaYoke", "Value:=" , "20mm"</pre>

```
],  
["NAME:Pair",  
  "Name:=", "Length",  
  "Value:=", "100mm",  
],  
["NAME:Pair",  
  "Name:=", "Skew",  
  "Value:=", "0deg",  
],  
["NAME:Pair",  
  "Name:=", "Slots",  
  "Value:=", "18",  
],  
["NAME:Pair",  
  "Name:=", "SlotType",  
  "Value:=", "1",  
],  
["NAME:Pair",  
  "Name:=", "Hs0",  
  "Value:=", "1mm",  
],
```

```
[ "NAME:Pair",
  "Name:="                , "Hs1",
  "Value:="               , "1mm"
],
[ "NAME:Pair",
  "Name:="                , "Hs2",
  "Value:="               , "10mm"
],
[ "NAME:Pair",
  "Name:="                , "Bs0",
  "Value:="               , "2.5mm"
],
[ "NAME:Pair",
  "Name:="                , "Bs1",
  "Value:="               , "8mm"
],
[ "NAME:Pair",
  "Name:="                , "Bs2",
  "Value:="               , "5mm"
],
```

```
[ "NAME:Pair",  
  "Name:=", "Rs",  
  "Value:=", "0mm",  
],  
[ "NAME:Pair",  
  "Name:=", "FilletType",  
  "Value:=", "0",  
],  
[ "NAME:Pair",  
  "Name:=", "Layers",  
  "Value:=", "2",  
],  
[ "NAME:Pair",  
  "Name:=", "CoilPitch",  
  "Value:=", "4",  
],  
[ "NAME:Pair",  
  "Name:=", "EndExt",  
  "Value:=", "5mm",  
],  
[ "NAME:Pair",
```

```

    "Name:="                , "SpanExt",
    "Value:="               , "25mm"
  ],
  ["NAME:Pair",
    "Name:="                , "BendAngle",
    "Value:="               , "0deg"
  ],
  ["NAME:Pair",
    "Name:="                , "SegAngle",
    "Value:="               , "10deg"
  ],
  ["NAME:Pair",
    "Name:="                , "LenRegion",
    "Value:="               , "200mm"
  ],
  ["NAME:Pair",
    "Name:="                , "InfoCoil",
    "Value:="               , "0"
  ]
]

```

```

],
["NAME:Attributes",
    "Name:="                , "LapCoill",
    "Flags:="                , "",
    "Color:="                , "(143 175 143)",
    "Transparency:="        , 0,
    "PartCoordinateSystem:=" , "Global",
    "UDMId:="                , "",
    "MaterialValue:="        , "\"copper\"",
    "SurfaceMaterialValue:=" , "\"\"",
    "SolveInside:="          , False,
    "ShellElement:="         , False,
    "ShellElementThickness:=" , "0mm",
    "IsMaterialEditable:="    , True,
    "UseMaterialAppearance:=" , False,
    "IsLightweight:="        , False
])

```

VB Syntax	CreateUserDefinedPart < <i>Parameters</i> >, < <i>Attributes</i> >
VB Examl-	oEditor.CreateUserDefinedPart Array("NAME:UserDefinedPrimitiveParameters", "DllName:=", —

e	<pre> "SegmentedHelix/RectHelix.dll", "Version:=", "1.0", "NoOfParameters:=", 8, "Library:=", _ "syslib", Array("NAME:ParamVector", Array("NAME:Pair", "Name:=", "RectHeight", "Value:=", _ "1mm"), Array("NAME:Pair", "Name:=", "RectWidth", "Value:=", "2mm"), Array("NAME:Pair", "Name:=", _ "StartHelixRadius", "Value:=", "10mm"), Array("NAME:Pair", "Name:=", "RadiusChange", "Value:=", _ "0mm"), Array("NAME:Pair", "Name:=", "Pitch", "Value:=", "3mm"), Array("NAME:Pair", "Name:=", _ "Turns", "Value:=", "2"), Array("NAME:Pair", "Name:=", "SegmentsPerTurn", "Value:=", _ "36"), Array("NAME:Pair", "Name:=", "RightHanded", "Value:=", "1"))), Array("NAME:At- tributes", "Name:=", _ "RectHelix1", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMa- terialAppearance:=", _ false, "IsLightweight:=", false) </pre>

Edit3DComponent

Edits a specified 3D component.

UI Access	N/A		
Parameters	Name	Type	Description
	<compName>	String	Component name.
	<Parameters>	Array	Structured array. <pre>Array("NAME:EditComponentParametersData", "NewComponentName:=", <string>, "GeometryParameters:=", <string>, "MaterialParameters:=", <string>, "DesignParameters:=", <string>, <ComponentMeshing>, <Excitations>)</pre>
	<ComponentMeshing>	Array	Structured array. <pre>Array("NAME:Component Meshing", "MeshAssembly:=", <boolean>)</pre>
	<Excitations>	Array	Structured array containing array of suppressed excitations. <pre>Array("NAME:Excitations", "Suppressed:=", <array>)</pre>
Return Value	None.		

Python Syntax	Edit3DComponent (<compName>, <Parameters>)
Python Example	<pre>oEditor.Edit3DComponent("Connector1",</pre>

	<pre> ["NAME:EditComponentParametersData", "NewComponentName:=", "Connector2", "GeometryParameters:=", "", "MaterialParameters:=", "", "DesignParameters:=", "", ["NAME:Component Meshing", "MeshAssembly:=", False], ["NAME:Excitations", "Suppressed:=", []]]) </pre>

VB Syntax	Edit3DComponent <compName>, <Parameters>
VB Example	<pre> oEditor.Edit3DComponent "Connector1", Array("NAME:EditComponentParametersData", "NewComponentName:=", "Connector2", "GeometryParameters:=", "", "MaterialParameters:=", "", "DesignParameters:=", "", </pre>

	<pre>Array("NAME:Component Meshing", "MeshAssembly:=", false), Array("NAME:Excitations", "Suppressed:=", Array()))</pre>

[Beta] EditNativeComponentDefinition [Maxwell]

Edit a native component definition.

UI Access	In the Project Manager , expand 3D Components , right-click on the component, and select Edit Definition .		
Parameters	Name	Type	Description
	<LinkParameters>	Array	Data of the 3D component.
Return Value	None.		

Python Syntax	EditNativeComponentDefinition (<LinkParameters>)
Python Example	<pre>oEditor.EditNativeComponentDefinition(["NAME:EditNativeComponentDefinitionData", "DefinitionName:=", , "LC1", ["NAME:GeometryDefinitionParameters", [</pre>

```

        "NAME:VariableOrders"

    ]

],

[

    "NAME:DesignDefinitionParameters",

    [

        "NAME:VariableOrders"

    ]

],

[

    "NAME:MaterialDefinitionParameters",

    [

        "NAME:VariableOrders"

    ]

],

"NextUniqueID:="          , 0,

"MoveBackwards:="        , False,

"DatasetType:="          , "ComponentDatasetType",

[

    "NAME:DatasetDefinitions"

```

```
],  
[  
  "NAME:NativeComponentDefinitionProvider",  
  "Type:=" , "Layout Component",  
  "Unit:=" , "mm",  
  "Version:=" , 1.1,  
  "EDBDefinition:=" , "EdgeCircuitPort_Inductor1",  
  [  
    "NAME:VariableMap"  
  ],  
  "ReferenceCS:=" , "U0",  
  "CSToImport:=" , ["Global","U0"]  
],  
"ComponentName:=" , "LC1",  
"Company:=" , "",  
"Company URL:=" , "",  
"Model Number:=" , "",  
"Help URL:=" , "",  
"Version:=" , "1.0",  
"Notes:=" , "",  
"IconType:=" , "Layout Component"
```

	])

EditPolyline

Modifies a specified polyline. See: [CreatePolyline](#).

UI Access	N/A		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:PolylineParameters", "IsPolylineCovered:=", <bool>, "IsPolylineClosed:=", <bool>, <PolylinePointsArray>, <PolylineSegmentsArray>)
	<PolylinePointsArray>	Array	Array("NAME:PolylinePoints", <OnePointArray>, <OnePointArray>, ...)
	<OnePointArray>	Array	Array("NAME:PLPoint", "X:=", <value>, "Y:=", <value>, "Z:=", <value>))
	<PolylineSegmentsArray>	Array	<PolylineSegmentsArray> Array("NAME:PolylineSegments",

		<OneSegmentArray>, <OneSegmentArray>, ...)
	<OneSegmentArray> Array	Array("NAME:PLSegment", "SegmentType:=", <"Line", "Arc", "Spline", or "AngularArc">, "StartIndex:=", <value>, "NoOfPoints:=", <value>)
Return Value	None.	

Python Syntax	EditPolyline(<SelectionsArray>, <Parameters>)
Python Example	<pre> oEditor.EditPolyline(["NAME:Selections", "Selections:=", "Polyline1"] ["NAME:PolylineParameters", "IsPolylineCovered:=" , True, "IsPolylineClosed:=" , False, ["NAME:PolylinePoints", ["NAME:PLPoint", "X:=" , "20000mm", "Y:=" , "-20000mm", "Z:=" , "0mm"],], ["NAME:PLPoint", </pre>

```

        "X:="                , "-90000mm",
        "Y:="                , "20000mm",
        "Z:="                , "0mm"
    ],
    ["NAME:PLPoint",
        "X:="                , "10000mm",
        "Y:="                , "-140000mm",
        "Z:="                , "0mm"
    ]
],
["NAME:PolylineSegments",
    ["NAME:PLSegment",
        "SegmentType:="      , "Line",
        "StartIndex:="       , 0,
        "NoOfPoints:="       , 2
    ],
    ["NAME:PLSegment",
        "SegmentType:="      , "Line",
        "StartIndex:="       , 1,
        "NoOfPoints:="       , 2
    ]
]
```


	<pre>]], ["NAME:PolylineXSection", "XSectionType:=" , "None", "XSectionOrient:=" , "Auto", "XSectionWidth:=" , "0mm", "XSectionTopWidth:=" , "0mm", "XSectionHeight:=" , "0mm", "XSectionNumSegments:=" , "0", "XSectionBendType:=" , "Corner"]]) </pre>

VB Syntax	EditPolyline <SelectionsArray>, <Parameters>
VB Example	<pre> oEditor.EditPolyline Array("NAME:Selections", "Selections:=", "Polyline1") Array ("NAME:PolylineParameters", "IsPolylineCovered:=", true, "IsPolylineClosed:=", _ false, Array("NAME:PolylinePoints", Array("NAME:PLPoint", "X:=", "40000mm", "Y:=", _ "50000mm", "Z:=", "0mm"), Array("NAME:PLPoint", "X:=", "-150000mm", "Y:=", "-50000mm", "Z:=", _ "0mm")), Array("NAME:PolylineSegments", Array("NAME:PLSegment", "SegmentType:=", "Line", "StartIndex:=", _ </pre>

	<pre>0, "NoOfPoints:=", 2)), Array("NAME:PolylineXSection", "XSectionType:=", "None", "XSectionOrient:=", _ "Auto", "XSectionWidth:=", "0mm", "XSectionTopWidth:=", "0mm", "XSectionHeight:=", _ "0mm", "XSectionNumSegments:=", "0", "XSectionBendType:=", "Corner")), Array("NAME:Attributes", "Name:=", _ "Polyline2", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", _ "Global", "UDMId:=", "", "MaterialValue:=", "" & Chr(34) & "copper" & Chr(34) & "", "SurfaceMaterialValue:=", _ "" & Chr(34) & "" & Chr(34) & "", "SolveInside:=", false, "ShellElement:=", _ false, "ShellElementThickness:=", "0mm", "IsMaterialEditable:=", true, "UseMaterialAppearance:=", _ false, "IsLightweight:=", false)</pre>

Get3DComponentParameters

Returns parameters for a specified 3D component.

UI Access	N/A		
Parameters	Name	Type	Description
	<compName>	String	3D component name.
Return Value	Array containing component parameters.		

Python Syntax	Get3DComponentParameters(<compName>)

Python Example	<code>oEditor.Get3DComponentParameters('Connector')</code>

VB Syntax	<code>Get3DComponentParameters <compName></code>
VB Example	<code>oEditor.Get3DComponentParameters "Connector"</code>

Get3DComponentDefinitionNames

Gets names of 3D component definitions.

UI Access	N/A
Parameters	None.
Return Value	Array of strings containing component definition names.

Python Syntax	<code>Get3DComponentDefinitionNames()</code>
Python Example	<code>oEditor.Get3DComponentDefinitionNames()</code>

VB Syntax	<code>Get3DComponentDefinitionNames()</code>
VB Example	<code>Dim defNames defNames = oEditor.Get3DComponentDefinitionNames()</code>

Get3DComponentInstanceNames

Returns instance names of 3D component definitions.

UI Access	N/A		
Parameters	Name	Type	Description
	<DefinitionName>	String	Definition name.
Return Value	Array containing instance names.		

Python Syntax	Get3DComponentInstanceNames(<DefinitionName>)		
Python Example	oEditor.Get3DComponentInstanceNames ("Connector")		

VB Syntax	Get3DComponentInstanceNames <DefinitionName>		
VB Example	oEditor.Get3DComponentInstanceNames "Connector"		

Get3DComponentMaterialNames

Returns material names for a specified *.a3dcomp format 3D component.

UI Access	N/A		
Parameters	Name	Type	Description
	<InstanceName>	String	Component instance name.
Return Value	Array containing material names.		

Python Syntax	Get3DComponentMaterialNames(<InstanceName>)		
Python Example	oEditor.Get3DComponentMaterialNames ("Connector1.a3dcomp")		

VB Syntax	Get3DComponentMaterialNames <InstanceName>
VB Example	oEditor.Get3DComponentMaterialNames "Connector1.a3dcomp"

Get3DComponentMaterialProperties

Returns material properties for a specified 3D component.

UI Access	N/A		
Parameters	Name	Type	Description
	<MaterialName>	String	Material name.
Return Value	Array containing material properties.		

Python Syntax	Get3DComponentMaterialProperties(<MaterialName>)
Python Example	oEditor.Get3DComponentMaterialProperties('Connector1:Material01')

VB Syntax	Get3DComponentMaterialProperties <MaterialName>
VB Example	oEditor.Get3DComponentMaterialProperties "Connector1:Material01"

Insert3DComponent

Inserts a 3D component.

UI Access	Draw > 3D Component Library > Browse > [Component].

Parameters	Name	Type	Description
	<ComponentData>	Array	Structured array. Array("NAME:InsertComponentData", "Parameters:=", <string>, "TargetCS:=", <string>, "ComponentFile:=", <string filepath>)
Return Value	None.		

Python Syntax	Insert3DComponent(<ComponentData>)
Python Example	<pre>oEditor.Insert3DComponent(["NAME:InsertComponentData", "Parameters:=", "", "TargetCS:=", "Global", "ComponentFile:=", "C:\tmp\Connector.a3dcomp"])</pre>

VB Syntax	Insert3DComponent <ComponentData>
VB Example	<pre>oEditor.Insert3DComponent Array("NAME:InsertComponentData", "Parameters:=", "", "TargetCS:=", "Global",</pre>

```
"ComponentFile:=", "C:\tmp\Connector.a3dcomp")
```

InsertNativeComponent [Layout Component to Maxwell with CS]

Inserts a Layout Component to Maxwell 3D with custom coordinate system definitions.

UI Access	Create Layout Component ...		
Parameters	Name	Type	Description
	<LinkParameters>	Array	Structured array containing data of inserted layout component.
Return Value	None.		

Python Syntax	InsertNativeComponent (<LinkedParameters>)
Python Example	<pre>oEditor.InsertNativeComponent (["NAME:InsertNativeComponentData", "TargetCS:=" , "Global", "SubmodelDefinitionName:=", "LC1", ["NAME:ComponentPriorityLists"], "NextUniqueID:=" , 0, "MoveBackwards:=" , False, "DatasetType:=" , "ComponentDatasetType",</pre>

```
[
  "NAME:DatasetDefinitions"
],
[
  "NAME:BasicComponentInfo",
  "ComponentName:="      , "LC1",
  "Company:="            , "",
  "Company URL:="        , "",
  "Model Number:="       , "",
  "Help URL:="            , "",
  "Version:="             , "1.0",
  "Notes:="               , "",
  "IconType:="            , "Layout Component"
],
[
  "NAME:GeometryDefinitionParameters",
  [
    "NAME:VariableOrders"
  ]
],
```



```
[
  "NAME:DesignDefinitionParameters",
  [
    "NAME:VariableOrders"
  ]
],
[
  "NAME:MaterialDefinitionParameters",
  [
    "NAME:VariableOrders"
  ]
],
"DefReferenceCSID:=" , 1,
"MapInstanceParameters:=", "DesignVariable",
"UniqueDefinitionIdentifier:=", "27e7d8f5-da68-409b-8c4d-bf2bc6d95f69",
"OriginFilePath:=" , "",
"IsLocal:=" , False,
"ChecksumString:=" , "",
"ChecksumHistory:=" , [],
"VersionHistory:=" , [],
[
```

```
"NAME:NativeComponentDefinitionProvider",
"Type:="                , "Layout Component",
"Unit:="                , "mm",
"EDBDefinition:="      , "EdgeCircuitPort_Inductor1",
[
    "NAME:VariableMap"
],
"ReferenceCS:="        , "Global",
"CSToImport:="        , ["Global","U0"]
],
[
    "NAME:InstanceParameters",
    "GeometryParameters:=" , "",
    "MaterialParameters:=" , "",
    "DesignParameters:="   , ""
]
])
```

InsertPolylineSegment

Inserts a polyline segment before or after a specified existing segment.

UI Access	Draw > Line Segment > [Selection].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:Insert Polyline Segment", "Selections:=" , <string>, "Segment Indices:=" , <array containing integers>, "At Start:=" , <boolean>, "SegmentType:=" , <string "Line", "Arc", "Spline", or "AngularArc">, <PolylinePointsArray>
	<PolylinePointsArray>	Array	Structured array. See: CreatePolyline .
Return Value	None.		

Python Syntax	InsertPolylineSegment(<Parameters>)
Python Example	<pre>oEditor.InsertPolylineSegment(["NAME:Insert Polyline Segment", "Selections:=" , "Polyline1:CreatePolyline:1", "Segment Indices:=" , [0], "At Start:=" , True, "SegmentType:=" , "Line", ["NAME:PolylinePoints", ["NAME:PLPoint",</pre>

	<pre>"X:=", "1.1mm", "Y:=", "0.8mm", "Z:=", "0mm"], ["NAME:PLPoint", "X:=", "0.6mm", "Y:=", "-0.8mm", "Z:=", "0mm"]]])</pre>

VB Syntax	InsertPolylineSegment < <i>Parameters</i> >
VB Example	<pre>oEditor.InsertPolylineSegment Array("NAME:Insert Polyline Segment", "Selections:=", _ "Polyline1:CreatePolyline:1", "Segment Indices:=", Array(1), "At Start:=", _ false, "SegmentType:=", "Spline", Array("NAME:PolylinePoints", Array("NAME:PLPoint", "X:=", _ "-1.4mm", "Y:=", "0.4mm", "Z:=", "0mm"), Array("NAME:PLPoint", "X:=", "0.5mm", "Y:=", _ "1.1mm", "Z:=", "0mm"), Array("NAME:PLPoint", "X:=", "0.7mm", "Y:=", "-2.1mm", "Z:=", _</pre>

```
"0mm"), Array("NAME:PLPoint", "X:=", "0.4mm", "Y:=", "-1.1mm", "Z:=", "0mm")))
```

SweepAlongPath

Sweeps the specified 1D or 2D parts along a path. The last 1D object specified is the path for the sweep.

UI Access	Draw > Sweep > Along Path.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<PathSweepParametersArray>	Array	Array("NAME:PathSweepParameters", "DraftAngle:=", <value>, "DraftType:=", <string>, "CheckFaceFaceIntersection:=", <bool>, "TwistAngle:=", <value>) Possible values for DraftType are "Extended", "Round", and "Natural".
Return Value	None.		

Python Syntax	SweepAlongPath(<SelectionsArray>, <PathSweepParametersArray>)
Python Example	<pre>oEditor.SweepAlongPath(["NAME:Selections", "Selections:=" , "Rectangle1,Polyline1", "NewPartsModelFlag:=" , "Model"</pre>

	<pre>], ["NAME:PathSweepParameters", "DraftAngle:=" , "0deg", "DraftType:=" , "Round", "CheckFaceFaceIntersection:=", False, "TwistAngle:=" , "0deg"])</pre>

VB Syntax	SweepAlongPath <SelectionsArray>, <PathSweepParametersArray>
VB Example	<pre>oEditor.SweepAlongPath Array("NAME:Selections", "Selections:=", "Polygon1,Polyline1"),_ Array("NAME:PathSweepParameters", _ "DraftAngle:=", "0deg",_ "DraftType:=", "Round",_ "CheckFaceFaceIntersection:=", False,_ "TwistAngle:=", "30deg")</pre>

SweepAlongVector

Sweeps the specified 1D or 2D parts along a vector.

UI Access	Draw > Sweep > Along Vector.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<VecSweepParametersArray>	Array	Array("NAME:VectorSweepParameters", "DraftAngle:=", <value>, "DraftType:=", <string>, "CheckFaceFaceIntersection:=", <bool>, "SweepVectorX:=", <value> "SweepVectorY:=", <value> "SweepVectorZ:=", <value>) Possible values for DraftType are "Extended", "Round", and "Natural".
Return Value	None.		

Python Syntax	SweepAlongVector(<SelectionsArray>, <VecSweepParametersArray>)
Python Example	<pre>oEditor.SweepAlongVector(["NAME:Selections", "Selections:=" , "Rectangle1",</pre>

	<pre> "NewPartsModelFlag:=" , "Model"], ["NAME:VectorSweepParameters", "DraftAngle:=" , "0deg", "DraftType:=" , "Round", "CheckFaceFaceIntersection:=", False, "SweepVectorX:=" , "0mm" "SweepVectorY:=" , "0mm" "SweepVectorZ:=" , "12mm"]) </pre>

VB Syntax	SweepAlongVector <SelectionsArray>, <VecSweepParametersArray>
VB Example	<pre> oEditor.SweepAlongPath Array("NAME:Selections",_ "Selections:=", "Rectangle1",_ "NewPartsModelFlag:=" , "Model") Array("NAME:VectorSweepParameters", _ "DraftAngle:=", "0deg",_ </pre>

	<pre> "DraftType:=", "Round", _ "CheckFaceFaceIntersection:=", False, _ "SweepVectorX:=" , "0mm", _ "SweepVectorY:=" , "0mm", _ "SweepVectorZ:=" , "12mm") </pre>

SweepAroundAxis

Sweeps the specified 1D or 2D parts around an axis.

UI Access	Draw > Sweep > Around Axis.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<AxisSweepParametersArray>	Array	<pre> Array("NAME:AxisSweepParameters", "DraftAngle:=", <value>, "DraftType:=", <string>, "CheckFaceFaceIntersection:=", <bool>, "SweepAxis:=", <value> "SweepAngle:=", <value> "NumOfSegments:=", <value>) </pre> Possible values for DraftType are "Extended", "Round", and "Natural". Possible values for SweepAxis are "X", "Y", and "Z".
Return Value	None.		

Python Syntax	SweepAroundAxis(<SelectionsArray>, <AxisSweepParametersArray>)
Python Example	<pre>oEditor.SweepAroundAxis(["NAME:Selections", "Selections:=" , "Rectangle1", "NewPartsModelFlag:=" , "Model"], ["NAME:AxisSweepParameters", "DraftAngle:=" , "0deg", "DraftType:=" , "Round", "CheckFaceFaceIntersection:=", False, "SweepAxis:=" , "X" "SweepAngle:=" , "360deg" "NumOfSegments:=" , "12"])</pre>
VB Syntax	SweepAroundAxis <SelectionsArray>, <AxisSweepParametersArray>

VB Example	<pre>oEditor.SweepAroundAxis Array("NAME:Selections",_ "Selections:=", "Rectangle1",_ "NewPartsModelFlag:=" , "Model") Array("NAME:AxisSweepParameters", _ "DraftAngle:=", "0deg",_ "DraftType:=", "Round",_ "CheckFaceFaceIntersection:=", False,_ "SweepAxis:=" , "X",_ "SweepAngle:=" , "360deg",_ "NumOfSegments:=" , "12")</pre>

SweepFacesAlongNormal

Sweep the specified face(s) along normal.

UI Access	Modeler > Surface > Sweep Faces Along Normal		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	Structured array. Array("NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", <faceIDarray>, "LengthOfSweep:=", "<value><units>")

Return Value	None

Python Syntax	SweepFacesAlongNormal(<SelectionsArray> <parameters>)
Python Example	<pre>oEditor.SweepFacesAlongNormal(["NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"], ["NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", [183], "LengthOfSweep:=", "0.1mm"])</pre>

VB Syntax	SweepFacesAlongNormal <SelectionsArray> <parameters>
VB Example	<pre>oEditor.SweepFacesAlongNormal Array("NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"), Array("NAME:Parameters",</pre>

	<pre> "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", Array(57), "LengthOfSweep:=", "0.5mm") </pre>

SweepFacesAlongNormalWithAttributes

Sweep a face along normal, and specify attributes of the new object.

UI Access	Modeler > Surface > Sweep Faces Along Normal		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	<pre> Array("NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", <faceIDarray>, "LengthOfSweep:=", "<value><units>") </pre>
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None		

Python Syntax	SweepFacesAlongNormalWithAttributes(<SelectionsArray>, <parameters>, <AttributesArray>)
Python Example	<pre> oEditor.SweepFacesAlongNormalWithAttributes(["NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"], </pre>

```
["NAME:Parameters",  
    "NAME:SweepFaceAlongNormalToParameters",  
    "FacesToDetach:=", [183],  
    "LengthOfSweep:=", "0.1mm"],  
["NAME:Attributes",  
    "Name:=", "Box3",  
    "Flags:=", "",  
    "Color:=", "(143 175 143)",  
    "Transparency:=", 0,  
    "PartCoordinateSystem:=", "Global",  
    "UDMId:=", "",  
    "MaterialValue:=", "\"copper\"",  
    "SurfaceMaterialValue:=", "\"\"",  
    "SolveInside:=", False,  
    "ShellElement:=", False,  
    "ShellElementThickness:=", "0mm",  
    "IsMaterialEditable:=", True,  
    "UseMaterialAppearance:=", False,  
    "IsLightweight:=", False])
```

VB Syntax	SweepFacesAlongNormalWithAttributes <SelectionsArray>, <parameters>, <AttributesArray>
VB Example	<pre> oEditor.SweepFacesAlongNormalWithAttributes Array("NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"), Array("NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", Array(57), "LengthOfSweep:=", "0.5mm"), Array("NAME:Attributes", "Name:=", "Box3", "Flags:=", "", "Color:=", "(143 175 143)", "Transparency:=", 0, "PartCoordinateSystem:=", "Global", "UDMId:=", "", "MaterialValue:=", "\"copper\"", "SurfaceMaterialValue:=", "\"\"", "SolveInside:=", false, "ShellElement:=", false, </pre>

	<pre>"ShellElementThickness:=", "0mm", "IsMaterialEditable:=" , true, "UseMaterialAppearance:=", false, "IsLightweight:=" , false))</pre>

UpdateComponentDefinition

Updates a 3D component's definition.

UI Access	Draw > 3D Component Library > Definitions.		
Parameters	Name	Type	Description
	<data>	Array	Structured array. Array("NAME:UpdateDefinitionData", "DefinitionNames:=", <string>, "Passwords:=", <array of strings>)
Return Value	None.		

Python Syntax	UpdateComponentDefinition(<data>)
Python Example	<pre>oEditor.UpdateComponentDefinition(['NAME:UpdateDefinitionData', 'DefinitionNames:=', 'Connector, Magic_Tee',</pre>

	<pre>'Passwords:='', ['','']])</pre>

VB Syntax	UpdateComponentDefinition <data>
VB Example	<pre>oEditor.UpdateComponentDefinition Array("NAME:UpdateDefinitionData", "DefinitionNames:=", " Connector, Magic_Tee", "Passwords:=", Array("", ""))</pre>

Edit Menu Commands

[Copy](#)

[DeletePolylinePoint](#)

[DuplicateAlongLine](#)

[DuplicateAroundAxis](#)

[DuplicateMirror](#)

[Mirror](#)

[Move](#)

[OffsetFaces](#)

[Paste](#)

[Rotate](#)

[Scale](#)

Copy

Copies specified part(s) to the clipboard.

UI Access	N/A		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	Copy(<SelectionsArray>)		
Python Example	<pre>oEditor.Copy(["NAME:Selections", "Selections:=", "Box1"])</pre>		

VB Syntax	Copy <SelectionsArray>		
VB Example	<pre>oEditor.Copy Array("NAME:Selections", "Selections:=", "Box1")</pre>		

DeletePolylinePoint

Deletes either a start point or an end point from an existing polyline segment.

UI Access	Edit > Delete [Start/End] Point		
Parameters	Name	Type	Description
	<DeletePointArray>	Array	Structured array. <pre>Array("NAME:Delete Point", "Selections:=", <string "<PolylineName>:<PolylineAction>:<int>", "Segment Index:=", <integer>, "At Start:=", <bool True for start point; False for end point>)</pre>
Return Value	None.		

Python Syntax	DeletePolylinePoint (<DeletePointArray>)
Python Example	<pre>oEditor.DeletePolylinePoint(["NAME:Delete Point", "Selections:=", "Polyline1:CreatePolyline:1", "Segment Index:=", 1, "At Start:=", True])</pre>

VB Syntax	DeletePolylinePoint <DeletePointArray>
VB Example	<pre>oEditor.DeletePolylinePoint Array("NAME:Delete Point", "Selections:=", "Polyline1:CreatePolyline:1", "Segment Index:=", 1, "At Start:=", True)</pre>

DuplicateAlongLine

Duplicates specified parts along a line.

UI Access	Edit > Duplicate > Along Line.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. <code>Array("NAME:Selections", "Selections:=" , <string>, "NewPartsModelFlag:=" , <string>)</code>
	<ParametersArray>	Array	Structured array. <code>Array("NAME:DuplicateToAlongLineParameters", "CreateNewObjects:=" , <boolean>, "XComponent:=" , <string>, "YComponent:=" , <string>, "ZComponent:=" , <string>, "NumClones:=" , <string containing number greater than 1>)</code>
	<OptionsArray>	Array	Structured array. <code>Array("NAME:Options", "DuplicateAssignments:=" , <boolean>)</code>
	<CreateGroup>	Array	Optional. Structured array. <code>Array("CreateGroupsForNewObjects:=" , <boolean>)</code>

Return Value	None.

Python Syntax	DuplicateAlongLine (<SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>)
Python Example	<pre>oEditor.DuplicateAlongLine(["NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model"], ["NAME:DuplicateToAlongLineParameters", "CreateNewObjects:=" , False, "XComponent:=" , "1mm", "YComponent:=" , "-0.7mm", "ZComponent:=" , "0mm", "NumClones:=" , "2"], ["NAME:Options", "DuplicateAssignments:=", False], ["CreateGroupsForNewObjects:=", False])</pre>

VB Syntax	DuplicateAlongLine <SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>
VB Example	<pre>oEditor.DuplicateAlongLine Array("NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model") Array("NAME:DuplicateToAlongLineParameters", "CreateNewObjects:=", false, "XComponent:=", "1mm", "YComponent:=", "-0.7mm", "ZComponent:=", "0mm", "NumClones:=", "2") Array("NAME:Options", "DuplicateAssignments:=", false) Array("CreateGroupsForNewObjects:=", false)</pre>

DuplicateAroundAxis

Duplicates specified parts around an axis.

UI Access	Edit > Duplicate > Around Axis.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. Array("NAME:Selections",

			"Selections:=" , <string>, "NewPartsModelFlag:=" , <string>)
	<ParametersArray>	Array	Structured array. Array("NAME:DuplicateAroundAxisParameters", "CreateNewObjects:=" , <boolean>, "WhichAxis:=" , <string>, "AngleStr:=" , <string>, "NumClones:=" , <string containing number greater than 1>)
	<OptionsArray>	Array	Structured array. Array("NAME:Options", "DuplicateAssignments:=" , <boolean>)
	<CreateGroup>	Array	Optional. Structured array. Array("CreateGroupsForNewObjects:=" , <boolean>)
Return Value	None.		

Python Syntax	DuplicateAroundAxis (<SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>)
Python Example	<pre>oEditor.DuplicateAroundAxis(["NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model"],</pre>

	<pre>["NAME:DuplicateAroundAxisParameters", "CreateNewObjects:=" , True, "WhichAxis:=" , "Z", "AngleStr:=" , "90deg", "NumClones:=" , "2"], ["NAME:Options", "DuplicateAssignments:=", False], ["CreateGroupsForNewObjects:=", False])</pre>

VB Syntax	DuplicateAroundAxis <SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>
VB Example	<pre>oEditor.DuplicateAroundAxis Array("NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model") Array("NAME:DuplicateAroundAxisParameters", "CreateNewObjects:=", true,</pre>

	<pre> "WhichAxis:=", "Z", "AngleStr:=", "90deg", "NumClones:=", "2") Array("NAME:Options", "DuplicateAssignments:=", false) Array("CreateGroupsForNewObjects:=", false) </pre>

DuplicateMirror

Duplicates specified parts according to a mirror plane.

UI Access	Edit > Duplicate > Mirror.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. <pre> Array("NAME:Selections", "Selections:=" , <string>, "NewPartsModelFlag:=" , <string>) </pre>
	<ParametersArray>	Array	Structured array. <pre> Array("NAME:DuplicateToMirrorParameters", "DuplicateMirrorBaseX:=", <string>, "DuplicateMirrorBaseY:=", <string>, "DuplicateMirrorNormalX:=", <string>, "DuplicateMirrorNormalY:=", <string>, "DuplicateMirrorNormalZ:=", <string>) </pre>

			For Maxwell 2D XY designs, Z parameters should be set to "0". For Maxwell 2D RZ designs, Y parameters should be set to "0".
	<OptionsArray>	Array	Structured array. Array("NAME:Options", "DuplicateAssignments:=", <boolean>)
	<CreateGroup>	Array	Optional. Structured array. Array("CreateGroupsForNewObjects:=", <boolean>)
Return Value	None.		

Python Syntax	DuplicateMirror (<SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>)
Python Example	<pre>oEditor.DuplicateMirror(["NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model"], ["NAME:DuplicateToMirrorParameters", "DuplicateMirrorBaseX:=", "-0.4mm", "DuplicateMirrorBaseY:=", "-1.2mm", "DuplicateMirrorBaseZ:=", "0mm", "DuplicateMirrorNormalX:=", "0.124034734589208mm",</pre>

	<pre> "DuplicateMirrorNormalY:=", "0.992277876713668mm", "DuplicateMirrorNormalZ:=", "0mm"], ["NAME:Options", "DuplicateAssignments:=", False], ["CreateGroupsForNewObjects:=", False]) </pre>

VB Syntax	DuplicateMirror <SelectionsArray>, <ParametersArray>, <OptionsArray>, <CreateGroup>
VB Example	<pre> oEditor.DuplicateMirror Array("NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model") Array("NAME:DuplicateToMirrorParameters", "DuplicateMirrorBaseX:=", "-0.4mm", "DuplicateMirrorBaseY:=", "-1.2mm", "DuplicateMirrorBaseZ:=", "0mm", "DuplicateMirrorNormalX:=", "0.124034734589208mm", "DuplicateMirrorNormalY:=", "0.992277876713668mm", "DuplicateMirrorNormalZ:=", "0mm") </pre>

	<pre>Array("NAME:Options", "DuplicateAssignments:=", false) Array("CreateGroupsForNewObjects:=", false)</pre>

Mirror

Mirrors specified part(s).

UI Access	Edit > Arrange > Mirror.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<MirrorParameters>	Array	Structured array.
			<pre>Array("NAME:MirrorParameters", "MirrorBaseX:=" , <string>, "MirrorBaseY:=" , <string>, "MirrorBaseZ:=" , <string>, "MirrorNormalX:=" , <string>, "MirrorNormalY:=" , <string>, "MirrorNormalZ:=" , <string>)</pre>
Return Value	None.		

Python Syntax	Mirror(<SelectionsArray>, <MirrorParameters>)
Python Example	oEditor.Mirror(

	<pre>["NAME:Selections", "Selections:=" , "Box1_1", "NewPartsModelFlag:=" , "Model"], ["NAME:MirrorParameters", "MirrorBaseX:=" , "-0.2mm", "MirrorBaseY:=" , "-1.2mm", "MirrorBaseZ:=" , "0mm", "MirrorNormalX:=" , "-0.316227766016838mm", "MirrorNormalY:=" , "0.948683298050514mm", "MirrorNormalZ:=" , "0mm"])</pre>

VB Syntax	Mirror <SelectionsArray> <MirrorParameters>
VB Example	<pre>oEditor.Mirror Array("NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model") Array("NAME:MirrorParameters", "MirrorBaseX:=", "-0.2mm", "MirrorBaseY:=", "-1.2mm",</pre>

	<pre>"MirrorBaseZ:=", "0mm", "MirrorNormalX:=", "-0.316227766016838mm", "MirrorNormalY:=", "0.948683298050514mm", "MirrorNormalZ:=", "0mm")</pre>

Move

Moves specified part(s).

UI Access	Edit > Arrange > Move.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<TranslateParameters>	Array	Structured array. Array(["NAME:TranslateParameters", "TranslateVectorX:=" , <string>, "TranslateVectorY:=" , <string>, "TranslateVectorZ:=" , <string>) For Maxwell 2D XY designs, "TranslateVectorZ:=" should be set to "0". For Maxwell 2D RZ designs, "TranslateVectorY:=" should be set to "0".
Return Value	None.		
Python Syntax	Move(<SelectionsArray>, <TranslateParameters>)		

Python Example	<pre>oEditor.Move(["NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model"], ["NAME:TranslateParameters", "TranslateVectorX:=", "-0.5mm", "TranslateVectorY:=", "0.1mm", "TranslateVectorZ:=", "0mm"])</pre>

VB Syntax	Move <SelectionsArray> <TranslateParameters>
VB Example	<pre>oEditor.Move Array("NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model") Array("NAME:TranslateParameters", "TranslateVectorX:=", "-0.5mm", "TranslateVectorY:=", "0.1mm", "TranslateVectorZ:=", "0mm")</pre>

OffsetFaces

Offsets the faces of selected part(s).

UI Access	Edit > Arrange > Offset.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<OffsetParameters>	Array	Structured array. Array("NAME:OffsetParameters", "OffsetDistance:=" , <string>)
Return Value	None.		

Python Syntax	OffsetFaces (<SelectionsArray>, <OffsetParameters>)
Python Example	<pre>oEditor.OffsetFaces(["NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model"], ["NAME:OffsetParameters", "OffsetDistance:=", "16mm"])</pre>

VB Syntax	OffsetFaces <SelectionsArray> <OffsetParameters>
VB Example	<pre>oEditor.OffsetFaces Array("NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model") Array("NAME:OffsetParameters", "OffsetDistance:=", "16mm")</pre>

Paste (Model Editor)

Pastes previously copied object(s). See: [Copy](#).

UI Access	Edit > Paste.
Parameters	None.
Return Value	None.

Python Syntax	Paste()
Python Example	<pre>oEditor.Copy(["NAME:Selections", "Selections:=" , "Box1_2"]) oEditor.Paste()</pre>

VB Syntax	Paste
VB Example	<pre>oEditor.Copy Array("NAME:Selections", "Selections:=" , "Box1_2") oEditor.Paste</pre>

Rotate

Rotates specified object(s).

UI Access	Edit > Arrange > Rotate.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<RotateParameters>	Array	Structured array. <pre>Array("NAME:RotateParameters", "RotateAxis:=" , <string "X", "Y", or "Z">, "RotateAngle:=" , <string>)</pre> For Maxwell 2D XY Designs, "RotateAxis:=" should be set to "Z". For Maxwell 2D RZ Designs, "RotateAxis:=" should be set to "Y".
Return Value	None.		

Python Syntax	Rotate(<SelectionsArray>, <RotateParameters>)
Python Example	<pre>oEditor.Rotate(["NAME:Selections", "Selections:=", "Box1_1",</pre>

	<pre> "NewPartsModelFlag:=", "Model"], ["NAME:RotateParameters", "RotateAxis:=", "Z", "RotateAngle:=", "90deg"]) </pre>

VB Syntax	Rotate <SelectionsArray> <RotateParameters>
VB Example	<pre> oEditor.Rotate Array("NAME:Selections", "Selections:=", "Box1_1", "NewPartsModelFlag:=", "Model") Array("NAME:RotateParameters", "RotateAxis:=", "Z", "RotateAngle:=", "90deg") </pre>

Scale

Scales specified object(s).

UI Access	Edit > Scale.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<ScaleParameters>	Array	Structured array.

	Array("NAME:ScaleParameters", "ScaleX:=" , <string containing scale factor>, "ScaleY:=" , <string containing scale factor>, "ScaleZ:=" , <string containing scale factor>) For Maxwell 2D XY Designs, "ScaleZ:=" should be set to "0". For Maxwell 2D RZ Designs, "ScaleY:=" should be set to "0".
Return Value	None.

Python Syntax	Scale (<SelectionsArray>, <ScaleParameters>)
Python Example	<pre>oEditor.Scale(["NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model"], ["NAME:ScaleParameters", "ScaleX:=", "2", "ScaleY:=", "2", "ScaleZ:=", "2"])</pre>

VB Syntax	Scale <SelectionsArray> <ScaleParameters>
VB Example	<pre> oEditor.Scale Array("NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", "Model") Array("NAME:ScaleParameters", "ScaleX:=", "2", "ScaleY:=", "2", "ScaleZ:=", "2") </pre>

Modeler Menu Commands

[AssignMaterial](#)

[Chamfer](#)

[Connect](#)

[CoverLines](#)

[CoverSurfaces](#)

[CreateEntityList](#)

[CreateFaceCS](#)

[CreateGroup](#)

[CreateObjectCS](#)

[CreateObjectFromEdges](#)

[CreateObjectFromFaces](#)

[CreateRelativeCS](#)

[DeleteEmptyGroups](#)

[DeleteLastOperation](#)

[DetachFaces](#)

[EditEntityList](#)

[EditFaceCS](#)

[EditObjectCS](#)

[EditRelativeCS](#)

[Export](#)

[ExportModelImageToFile](#)

[ExportModelMeshToFile](#)

[Fillet](#)

[FlattenGroup](#)

[Generate History](#)

[GetActiveCoordinateSystem](#)

[GetCoordinateSystems](#)

[HealObject](#)

[Import](#)

[ImportDXF](#)

[ImportGDSII \[Modeler\]](#)

[Intersect](#)

[MoveCStoEnd](#)

[MoveEntityToGroup](#)

[MoveFaces](#)

[ProjectSheet](#)

[PurgeHistory](#)

[ReplaceWith3DComponent](#)

[Section](#)

[SeparateBody](#)

[SetModelUnits](#)

[SetWCS](#)

[ShowWindow](#)

[Split](#)

[Subtract](#)

[SweepFacesAlongNormal](#)

[ThickenSheet](#)

[UncoverFaces](#)

[Unite](#)

[Ungroup](#)

[WrapSheet](#)

AssignMaterial

Assigns a material to specified object(s).

UI Access	Modeler > Assign Material.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<AttributesArray>	Array	Structured array. See: AttributesArray . This script supports the following attributes: - MaterialValue - SolveInside - ShellElement - ShellElementThickness - IsMaterialEditable - UseMaterialAppearance - IsLightweight
Return Value	None.		

Python Syntax	AssignMaterial(<SelectionsArray>, <AttributesArray>)

Python Example	<pre>oEditor.AssignMaterial(["NAME:Selections", "AllowRegionDependentPartSelectionForPMLCreation:=", True, "AllowRegionSelectionForPMLCreation:=", True, "Selections:=" , "Box1"], ["NAME:Attributes", "MaterialValue:=" , "diamond", "SolveInside:=" , False, "ShellElement:=" , False, "ShellElementThickness:=" , "nan", "IsMaterialEditable:=" , True, "UseMaterialAppearance:=" , False, "IsLightweight:=" , False])</pre>
VB Syntax	AssignMaterial <SelectionsArray> <AttributesArray>

VB Example	<pre>oEditor.AssignMaterial Array("NAME:Selections", "AllowRegionDependentPartSelectionForPMLCreation:=", true, "AllowRegionSelectionForPMLCreation:=", true, "Selections:=" , "Box1") Array("NAME:Attributes", "MaterialValue:=" , "diamond", "SolveInside:=" , false, "ShellElement:=" , false, "ShellElementThickness:=" , "nan", "IsMaterialEditable:=" , true, "UseMaterialAppearance:=" , false, "IsLightweight:=" , false)</pre>

Chamfer

Creates a chamfer.

UI Access	Modeler > Chamfer.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array.

			<pre> Array("NAME:Parameters", Array("NAME:ChamferParameters", "Edges:=" , <array containing integer>, "Vertices:=" , <array>, "LeftDistance:=" , <string>, "RightDistance:=" , <string>, "ChamferType:=" , <string "Symmetric", "Left Distance-Angle", "Right Distance-Angle", or "Left Distance-Right Distance">)) </pre>
Return Value	None.		

Python Syntax	Chamfer(<SelectionsArray>, <Parameters>)
Python Example	<pre> oEditor.Chamfer(["NAME:Selections", "Selections:=" , "Box2", "NewPartsModelFlag:=" , "Model"], ["NAME:Parameters", ["NAME:ChamferParameters", "Edges:=" , [42], "Vertices:=" , [], </pre>

	<pre>"LeftDistance:=" , "0.1mm", "RightDistance:=" , "0.1mm", "ChamferType:=" , "Symmetric"]])</pre>

VB Syntax	Chamfer <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.Chamfer Array("NAME:Selections", "Selections:=" , "Box2", "NewPartsModelFlag:=" , "Model") Array("NAME:Parameters", Array("NAME:ChamferParameters", "Edges:=" , [42], "Vertices:=" , [], "LeftDistance:=" , "0.1mm", "RightDistance:=" , "0.1mm", "ChamferType:=" , "Symmetric"))</pre>

Connect

Connects two or more 1D polyline objects or 2D sheet objects.

UI Access	Modeler > Surface > Connect.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	Connect(<SelectionsArray>)		
Python Example	<pre>oEditor.Connect (["NAME:Selections", "Selections:=", "Polyline2,Polyline1"])</pre>		

VB Syntax	Connect <SelectionsArray>		
VB Example	<pre>oEditor.Connect Array("NAME:Selections", "Selections:=", "Polyline2,Polyline1")</pre>		

CoverLines

Covers two or more 1D objects to form a sheet.

UI Access	Modeler > Surface > Cover Lines.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	CoverLines(<SelectionsArray>)
Python Example	<pre>oEditor.CoverLines(["NAME:Selections", "Selections:=", "Polyline3,Polyline4", "NewPartsModelFlag:=", "Model"])</pre>

VB Syntax	CoverLines <SelectionsArray>
VB Example	<pre>oEditor.CoverLines Array("NAME:Selections", "Selections:=", "Polyline3,Polyline4", "NewPartsModelFlag:=", "Model")</pre>

CoverSurfaces

Covers two or more faces to form a solid object.

UI Access	Modeler > Surface > Cover Faces.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	CoverSurfaces (<SelectionsArray>)

Python Example	<pre>oEditor.CoverSurfaces (["NAME:Selections", "Selections:=", "Obj1_Face1,Obj2_Face2", "NewPartsModelFlag:=", "Model"]) </pre>

VB Syntax	CoverSurfaces <SelectionsArray>
VB Example	<pre>oEditor.CoverSurfaces Array("NAME:Selections", "Selections:=", "Obj1_Face1,Obj2_Face2", "NewPartsModelFlag:=", "Model") </pre>

CreateEntityList

Creates a list of entities containing objects or faces (not both).

UI Access	Modeler > List > Create > [Object / Face] List.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:GeometryEntityListParameters", "EntityType:=" , <string "Object" or "Face">, "EntityList:=" , <string of object names or IDs>) </pre> See GetObjectIDByName for returning object IDs.

	<i><AttributesArray></i>	Array	Structured array. See: AttributesArray . CreateEntityList takes only the "Name" parameter.
Return Value	None.		

Python Syntax	CreateEntityList(<i><Parameters></i> , <i><AttributesArray></i>)
Python Example	<pre>oEditor.CreateEntityList(["NAME:GeometryEntityListParameters", "EntityType:=", "Object", "EntityList:=", "Bondwire1,Bondwire2"], ["NAME:Attributes", "Name:=", "Objectlist1"])</pre>

VB Syntax	CreateEntityList <i><Parameters></i> <i><AttributesArray></i>
VB Example	<pre>oEditor.CreateEntityList Array("NAME:GeometryEntityListParameters", "EntityType:=", "Object", "EntityList:=", "Bondwire1,Bondwire2") Array("NAME:Attributes",</pre>

"Name:=", "Objectlist1")

CreateFaceCS

Creates a Face Coordinate System from a selected face.

UI Access	Modeler > Coordinate System > Create > FaceCS.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:FaceCSParameters", <OriginArray>, "MoveToEnd:=", <boolean>, "FaceID:=", <integer>, <AxisPosnArray>, "WhichAxis:=" , <string "X", "Y", or "Z">, "ZRotationAngle:=" , <string>, "XOffset:=" , <string>, "YOffset:=" , <string>, "AutoAxis:=" , <boolean>)</pre>
	<OriginArray>	Array	Structured array. <pre>Array("NAME:Origin", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=", <integer>,</pre>

		<pre>"TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "FaceCenter", "EdgeCenter", "OnVertex", "OnEdge", or "OnFace">, "UParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "VParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)</pre> IsAttachedToEntity specifies whether the point is anchored to a vertex, edge, or face. If True, provide UParam and VParam. If False, provide XPosition, YPosition, and ZPosition to provide fixed position. Pass "0" for unused parameters.
<AxisPosnArray>	Array	Structured array. <pre>Array("NAME:AxisPosn", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=", <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "FaceCenter", "EdgeCenter", "OnVertex", "OnEdge", or "OnFace">,</pre>

			"UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)
	<AttributesArray>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateFaceCS(<Parameters>,<AttributesArray>)		
Python Example	<pre> oEditor.CreateFaceCS (["NAME:FaceCSParameters", ["NAME:Origin", "IsAttachedToEntity:=" , True, "EntityID:=" , 46, "FacetedBodyTriangleIndex:=" , -1, "TriangleVertexIndex:=" , -1, "PositionType:=" , "FaceCenter", "UParam:=" , 0, "VParam:=" , 0, "XPosition:=" , "0", "YPosition:=" , "0", </pre>		

```

        "ZPosition:="          , "0"
    ],
    "MoveToEnd:="              , False,
    "FaceID:="                  , 46,
    ["NAME:AxisPosn",
        "IsAttachedToEntity:="  , True,
        "EntityID:="            , 46,
        "FacetedBodyTriangleIndex:=", -1,
        "TriangleVertexIndex:=" , -1,
        "PositionType:="        , "OnFace",
        "UParam:="               , 0.487129134674319,
        "VParam:="               , 0.308528523557527,
        "XPosition:="            , "1292.27748080459mm",
        "YPosition:="            , "-814.882885865484mm",
        "ZPosition:="            , "0mm"
    ],
    "WhichAxis:="               , "X",
    "ZRotationAngle:="          , "0deg",
    "XOffset:="                  , "0mm",
    "YOffset:="                  , "0mm",

```

	<pre> "AutoAxis:=" , False], ["NAME:Attributes", "Name:=" , "FaceCS1", "PartName:=" , "Rectangle1"]) </pre>

VB Syntax	CreateFaceCS <Parameters> <AttributesArray>
VB Example	<pre> oEditor.CreateFaceCS Array("NAME:FaceCSParameters", Array("NAME:Origin", "IsAt- tachedToEntity:=", _ true, "EntityID:=", 58, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", _ -1, "PositionType:=", "FaceCenter", "UParam:=", 0, "VParam:=", 0, "XPosition:=", _ "0", "YPosition:=", "0", "ZPosition:=", "0"), "MoveToEnd:=", false, "FaceID:=", _ 58, Array("NAME:AxisPosn", "IsAttachedToEntity:=", true, "EntityID:=", 58, "FacetedBodyTriangleIndex:=", _ -1, "TriangleVertexIndex:=", -1, "PositionType:=", "OnFace", "UParam:=", _ 0.0664066713146499, "VParam:=", 0.407014331135309, "XPosition:=", _ "1826.56266852586mm", "YPosition:=", "-1355.79140131881mm", "ZPosition:=", "0mm"), "WhichAxis:=", _ "X", "ZRotationAngle:=", "0deg", "XOffset:=", "0mm", "YOffset:=", "0mm", "AutoAxis:=", _ false), Array("NAME:Attributes", "Name:=", "FaceCS2", "PartName:=", "Rectangle2") </pre>

CreateGroup

Creates a group from objects specified in the history tree.

UI Access	Modeler > Group > Create.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:GroupParameter", "ParentGroupID:=" , <string>, "Parts:=" , <string>, "SubmodelInstances:=" , <string>, "Groups:=" , <string>)
Return Value	None.		

Python Syntax	CreateGroup(<Parameters>)
Python Example	<pre>oEditor.CreateGroup(["NAME:GroupParameter", "ParentGroupID:=" , "Model", "Parts:=" , "Box1,Box2,Box3", "SubmodelInstances:=" , "", "Groups:=" , ""</pre>

	1)

VB Syntax	CreateGroup <Parameters>
VB Example	oEditor.CreateGroup Array("NAME:GroupParameter", "ParentGroupID:=", "Model", _ "Parts:=", "Box1,Box2,Box3", "SubmodelInstances:=", "", "Groups:=", "")

CreateObjectCS

Creates an object coordinate system from a selected object.

UI Access	Modeler > Coordinate System > Create > Object > [Offset / Rotated / Both].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:ObjectCSParameters", <OriginArray> "MoveToEnd:=" , <boolean>, "ReverseXAxis:=" , <boolean>, "ReverseYAxis:=" , <boolean>, <xAxisArray / xAxisPosArray> <yAxisArray / yAxisPosArray>) Note: xAxisArray and xAxisPosArray differ. Use xAxisArray for absolute position and xAxisPosArray for relative position. Do the same for yAxisArray and yAxisPosArray.
	<OriginArray>	Array	Structured array.

		<pre>Array("NAME:Origin", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=", <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", "OnEdge", or "AbsolutePosition">, "UParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "VParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)</pre> IsAttachedToEntity specifies whether the point is anchored. If True, provide UParam and VParam. If False, provide XPosition, YPosition, and ZPosition to provide fixed position. Pass "0" for unused parameters.
<xAxisArray>	Array	Structured array for absolute position: <pre>Array("NAME:xAxis", "DirectionType:=" , "AbsoluteDirection",</pre>

			<pre> "EdgeID:=" , <integer>, "FaceID:=" , <integer>, "xDirection:=" , <string>, "yDirection:=" , <string>, "zDirection:=" , <string>, "UParam:=" , <float>, "VParam:=" , <float> </pre>
	<xAxisPosArray>	Array	Structured array for relative position: <pre> Array("NAME:xAxisPos", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=" , <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", or "OnEdge">, "UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>) </pre>
	<yAxisArray>	Array	Structured array for absolute position: <pre> Array("NAME:yAxis", </pre>

			<pre> "DirectionType:=" , "AbsoluteDirection", "EdgeID:=" , <integer>, "FaceID:=" , <integer>, "xDirection:=" , <string>, "yDirection:=" , <string>, "zDirection:=" , <string>, "UParam:=" , <float>, "VParam:=" , <float> </pre>
	<yAxisPosArray>	Array	Structured array for relative position: <pre> Array("NAME:yAxisPos", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=" , <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", or "OnEdge">, "UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>) </pre>

	<code><AttributesArray></code>	Array	Structured array. See: AttributesArray .
Return Value	None.		

Python Syntax	CreateObjectCS(<Parameters>,<AttributesArray>)
Python Example	<pre>oEditor.CreateObjectCS(["NAME:ObjectCSParameters", ["NAME:Origin", "IsAttachedToEntity:=" , True, "EntityID:=" , 59, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=" , -1, "PositionType:=" , "OnVertex", "UParam:=" , 0, "VParam:=" , 0, "XPosition:=" , "0", "YPosition:=" , "0", "ZPosition:=" , "0"], "MoveToEnd:=" , False, "ReverseXAxis:=" , False, "ReverseYAxis:=" , False,</pre>

```
[ "NAME:xAxis",
  "DirectionType:="      , "AbsoluteDirection",
  "EdgeID:="             , -1,
  "FaceID:="             , -1,
  "xDirection:="         , "1",
  "yDirection:="         , "0",
  "zDirection:="         , "0",
  "UParam:="             , 0,
  "VParam:="             , 0
],
[ "NAME:yAxis",
  "DirectionType:="      , "AbsoluteDirection",
  "EdgeID:="             , -1,
  "FaceID:="             , -1,
  "xDirection:="         , "0",
  "yDirection:="         , "1",
  "zDirection:="         , "0",
  "UParam:="             , 0,
  "VParam:="             , 0
]
```

```

],
["NAME:Attributes",
    "Name:="                , "ObjectCS1",
    "PartName:="            , "Box2"
])

```

VB Syntax	CreateObjectCS <Parameters> <AttributesArray>
VB Example	<pre> oEditor.CreateObjectCS Array("NAME:ObjectCSParameters", Array("NAME:Origin", "IsAt- tachedToEntity:=", _ false, "EntityID:=", -1, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", _ -1, "PositionType:=", "AbsolutePosition", "UParam:=", 0, "VParam:=", 0, "XPosition:=", _ "0mm", "YPosition:=", "0mm", "ZPosition:=", "0mm"), "MoveToEnd:=", false, "ReverseXAx- is:=", _ false, "ReverseYAxis:=", false, Array("NAME:xAxisPos", "IsAttachedToEntity:=", true, "EntityID:=", _ 80, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", -1, "PositionType:=", _ "EdgeCenter", "UParam:=", 0, "VParam:=", 0, "XPosition:=", "0", "YPosition:=", _ "0", "ZPosition:=", "0"), Array("NAME:yAxisPos", "IsAttachedToEntity:=", true, "EntityID:=", _ 69, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", -1, "PositionType:=", _ "EdgeCenter", "UParam:=", 0, "VParam:=", 0, "XPosition:=", "0", "YPosition:=", _ "0", "ZPosition:=", "0")), Array("NAME:Attributes", "Name:=", "ObjectCS2", </pre>

	<pre>"PartName:=", _ "Box3")</pre>

CreateObjectFromEdges

Creates an object from the specified object edge.

UI Access	Modeler > Edge > Create Object From Edge		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<ParametersArray>	Array	Structured array. Array("NAME:Parameters", Array("NAME:BodyFromEdgeToParameters", "Edges:=" , <array containing integer edges>))
	<CreateGroupsForNewObjects>	Array	Structured array. Array("CreateGroupsForNewObjects:=", <boolean True to create groups for new objects; else False>)
Return Value	None.		

Python Syntax	CreateObjectFromEdges(<SelectionsArray>, <ParametersArray>, <CreateGroupsForNewObjects>)

Python Example	<pre>oEditor.CreateObjectFromEdges (["NAME:Selections", "Selections:=", "Box2", "NewPartsModelFlag:=", "Model"], ["NAME:Parameters", ["NAME:BodyFromEdgeToParameters", "Edges:=", [41]]], ["CreateGroupsForNewObjects:=", False]) </pre>

VB Syntax	CreateObjectFromEdges <SelectionsArray> <ParametersArray> <CreateGroupsForNewObjects>
VB Example	<pre>oEditor.CreateObjectFromEdges Array("NAME:Selections", "Selections:=", "Box1", "NewPartsModelFlag:=", _ "Model"), Array("NAME:Parameters", Array("NAME:BodyFromEdgeToParameters", "Edges:=", Array(_ 13))), Array("CreateGroupsForNewObjects:=", false) </pre>

CreateObjectFromFaces

Creates 2D objects from specified face(s).

UI Access	Modeler > Surface > Create Object from Face

Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array ("NAME:Parameters", <BodyFromFaceToParameters>)
	<CreateGroupsForNewObjects>	Array	Structured array. Array ("CreateGroupsForNewObjects:=", <boolean>)
Return Value	None.		

Python Syntax	CreateObjectFromFaces (<SelectionsArray>,<Parameters>,<CreateGroupsForNewObjects>)
Python Example	<pre>oEditor.CreateObjectFromFaces (["NAME:Selections", "Selections:=" , "Box3", "NewPartsModelFlag:=" , "Model"], ["NAME:Parameters", ["NAME:BodyFromFaceToParameters", "FacesToDetach:=" , [68]]],</pre>

	["CreateGroupsForNewObjects:=", False])

VB Syntax	CreateObjectFromFaces <SelectionsArray> <Parameters> <CreateGroupsForNewObjects>
VB Example	<pre>oEditor.CreateObjectFromFaces Array("NAME:Selections", "Selections:=", "Box3", "NewPartsModelFlag:=" , "Model") Array("NAME:Parameters", Array("NAME:BodyFromFaceToParameters", "FacesToDetach:=", Array(68))) Array("CreateGroupsForNewObjects:=", False)</pre>

CreateRelativeCS

Creates a Relative Coordinate System.

UI Access	Modeler > Coordinate System > Create > Relative CS > [Offset / Rotated / Both].		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:RelativeCSParameters", "Mode:=" , "Axis/Position", "OriginX:=" , <string>, "OriginY:=" , <string>, "OriginZ:=" , <string>, "XAxisXvec:=" , <string>, "XAxisYvec:=" , <string>,</pre>

			"XAxisZvec:=" , <string>, "YAxisXvec:=" , <string>, "YAxisYvec:=" , <string>, "YAxisZvec:=" , <string>)
	<AttributesArray>	Array	Structured array. See: AttributesArray . CreateRelativeCS supports only the "Name" parameter.
Return Value	None.		

Python Syntax	CreateRelativeCS(<Parameters>,<AttributesArray>)
Python Example	<pre>oEditor.CreateRelativeCS(["NAME:RelativeCSParameters", "Mode:=" , "Axis/Position", "OriginX:=" , "0.62mm", "OriginY:=" , "-0.7mm", "OriginZ:=" , "0mm", "XAxisXvec:=" , "1mm", "XAxisYvec:=" , "0mm", "XAxisZvec:=" , "0mm", "YAxisXvec:=" , "0mm", "YAxisYvec:=" , "1mm",</pre>

	<pre> "YAxisZvec:=" , "0mm"], ["NAME:Attributes", "Name:=" , "RelativeCS1"]) </pre>

VB Syntax	CreateRelativeCS <Parameters> <AttributesArray>
VB Example	<pre> oEditor.CreateRelativeCS Array("NAME:RelativeCSParameters", "Mode:=", "Axis/Position", "OriginX:=", _ "0mm", "OriginY:=", "0mm", "OriginZ:=", "0mm", "XAxisXvec:=", "0.66mm", "XAxisYvec:=", _ "0.28mm", "XAxisZvec:=", "0mm", "YAxisXvec:=", "0.06mm", "YAxisYvec:=", "0.14mm", "YAxisZvec:=", _ "0mm"), Array("NAME:Attributes", "Name:=", "RelativeCS1") </pre>

DeleteEmptyGroups

Deletes group(s) from the history tree.

UI Access	Modeler > Group > Delete Empty.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("Groups:=" , <array of string group IDs>)
Return Value	None.		

Python Syntax	DeleteEmptyGroups(<Parameters>)
Python Example	<pre>oEditor.DeleteEmptyGroups (["Groups:=", ["Group1", "Group2", "Group3"]])</pre>

VB Syntax	DeleteEmptyGroups <Parameters>
VB Example	<pre>oEditor.DeleteEmptyGroups Array("Groups:=", Array("Group1", "Group2", "Group3"))</pre>

DeleteLastOperation

Deletes the last operation performed on the specified object(s).

UI Access	Modeler > Delete Last Operation.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	DeleteLastOperation(<SelectionsArray>)
Python Example	<pre>oEditor.DeleteLastOperation (["NAME:Selections", "Selections:=", "Box3",</pre>

	<code>"NewPartsModelFlag:=", "Model"</code> <code>l)</code>

VB Syntax	<code>DeleteLastOperation <SelectionsArray></code>
VB Example	<code>oEditor.DeleteLastOperation Array("NAME:Selections", "Selections:=", "Box3", "NewPartsModelFlag:=", "Model")</code>

DetachFaces

Detaches the specified face(s) from an object.

UI Access	Modeler > Surface > Detach Faces.		
Parameters	Name	Type	Description
	<code><SelectionsArray></code>	Array	Structured array. See: SelectionsArray .
	<code><Parameters></code>	Array	Structured array. <code>Array("NAME:Parameters", <DetachFacesArray>)</code>
	<code><DetachFacesArray></code>	Array	Structured array. <code>Array("NAME:DetachFacesToParameters", "FacesToDetach:=" , <array containing integer face IDs>)</code>
Return Value	None.		

Python Syntax	<code>DetachFaces(<SelectionsArray>, <Parameters>)</code>

Python Example	<pre>oEditor.DetachFaces(["NAME:Selections", "Selections:=", "Box3", "NewPartsModelFlag:=", "Model"], ["NAME:Parameters", ["NAME:DetachFacesToParameters", "FacesToDetach:=", [68,67]]])</pre>

VB Syntax	DetachFaces <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.DetachFaces Array("NAME:Selections", "Selections:=", "Box3", "NewPartsModelFlag:=", "Model") Array("NAME:Parameters", Array("NAME:DetachFacesToParameters", "FacesToDetach:=", [68,67]))</pre>

EditEntityList

Modifies an entity list.

UI Access	Modeler > List > Reassign.

Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. <pre>Array("NAME:GeometryEntityListParameters", "EntityType:=" , <string "Object" or "Face">, "EntityList:=" , <string list>)</pre>
Return Value	None.		

Python Syntax	<code>EditEntityList(<SelectionsArray>, <Parameters>)</code>
Python Example	<pre>oEditor.EditEntityList(["NAME:Selections", "Selections:=" , "Objectlist1"], ["NAME:GeometryEntityListParameters", "EntityType:=" , "Object", "EntityList:=" , "Box1, Box2, Box3"]) </pre>

VB Syntax	<code>EditEntityList <SelectionsArray> <Parameters></code>
VB Example	<code>oEditor.EditEntityList</code>

	<pre>Array("NAME:Selections", "Selections:=", "Objectlist1") Array("NAME:GeometryEntityListParameters", "EntityType:=", "Object", "EntityList:=", "Box1, Box2, Box3")</pre>

EditFaceCS

Recreates an existing face coordinate system. See: [CreateFaceCS](#).

UI Access	Modeler > Coordinate System > Edit.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:FaceCSParameters", <OriginArray>, "MoveToEnd:=", <boolean>, "FaceID:=", <integer>, <AxisPosnArray>, "WhichAxis:=", <string "X", "Y", or "Z">, "ZRotationAngle:=", <string>, "XOffset:=", <string>, "YOffset:=", <string>, "AutoAxis:=", <boolean>)</pre>

	<OriginArray>	Array	Structured array. <pre>Array("NAME:Origin", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=", <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "FaceCenter", "EdgeCenter", "OnVertex", "OnEdge", or "OnFace">, "UParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "VParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)</pre> IsAttachedToEntity specifies whether the point is anchored to a vertex, edge, or face. If True, provide UParam and VParam. If False, provide XPosition, YPosition, and ZPosition to provide fixed position. Pass "0" for unused parameters.
	<AxisPosnArray>	Array	Structured array. <pre>Array("NAME:AxisPosn", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>,</pre>

			<pre>"FacetedBodyTriangleIndex:=", <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "FaceCenter", "EdgeCenter", "OnVertex", "OnEdge", or "OnFace">, "UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)</pre>
	<AttributesArray>	Array	Structured array. See: AttributesArray . Use to select the coordinate system to edit.
Return Value	None.		

Python Syntax	EditFaceCS (<Parameters>, <AttributesArray>)
Python Example	<pre>oEditor.EditFaceCS (["NAME:FaceCSParameters", ["NAME:Origin", "IsAttachedToEntity:=" , True, "EntityID:=" , 12, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=" , -1,</pre>

```

    "PositionType:="          , "FaceCenter",
    "UParam:="                , 0,
    "VParam:="                , 0,
    "XPosition:="             , "0",
    "YPosition:="             , "0",
    "ZPosition:="             , "0"
],
"MoveToEnd:="                , False,
"FaceID:="                   , 12,
["NAME:AxisPosn",
    "IsAttachedToEntity:="    , True,
    "EntityID:="              , 12,
    "FacetedBodyTriangleIndex:=", -1,
    "TriangleVertexIndex:="   , -1,
    "PositionType:="          , "OnFace",
    "UParam:="                 , 0.62951717774066,
    "VParam:="                 , 0.226514925559344,
    "XPosition:="              , "1200mm",
    "YPosition:="              , "-354.697014888131mm",
    "ZPosition:="              , "125.903435548132mm"
],

```

	<pre>"WhichAxis:=" , "X", "ZRotationAngle:=" , "0deg", "XOffset:=" , "0mm", "YOffset:=" , "0mm", "AutoAxis:=" , False], ["NAME:Attributes", "Name:=" , "FaceCS1", "PartName:=" , "Box1"])</pre>

VB Syntax	EditFaceCS <Parameters> <AttributesArray>
VB Example	<pre>oEditor.EditFaceCS Array("NAME:FaceCSParameters", Array("NAME:Origin", "IsAttachedToEntity:=" , True, "EntityID:=" , 12, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=" , -1, "PositionType:=" , "FaceCenter",</pre>

```

        "UParam:="                , 0,
        "VParam:="                , 0,
        "XPosition:="             , "0",
        "YPosition:="             , "0",
        "ZPosition:="             , "0"),
    "MoveToEnd:="                 , False,
    "FaceID:="                    , 12,
    Array("NAME:AxisPosn",
        "IsAttachedToEntity:="    , True,
        "EntityID:="              , 12,
        "FacetedBodyTriangleIndex:=", -1,
        "TriangleVertexIndex:="   , -1,
        "PositionType:="          , "OnFace",
        "UParam:="                 , 0.62951717774066,
        "VParam:="                 , 0.226514925559344,
        "XPosition:="              , "1200mm",
        "YPosition:="              , "-354.697014888131mm",
        "ZPosition:="              , "125.903435548132mm"),
    "WhichAxis:="                 , "X",
    "ZRotationAngle:="            , "0deg",
    "XOffset:="                   , "0mm",

```

	<pre>"YOffset:=" , "0mm", "AutoAxis:=" , False) Array("NAME:Attributes", "Name:=" , "FaceCS1", "PartName:=" , "Box1")</pre>

EditObjectCS

Edits an existing object coordinate system. See: [CreateObjectCS](#).

UI Access	Modeler > Coordinate System > Edit.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:ObjectCSParameters", <OriginArray> "MoveToEnd:=" , <boolean>, "ReverseXAxis:=" , <boolean>, "ReverseYAxis:=" , <boolean>, <xAxisArray / xAxisPosArray> <yAxisArray / yAxisPosArray>) Note: xAxisArray and xAxisPosArray differ. Use xAxisArray for absolute position and xAxisPosArray for relative position. Do the same for yAxisArray and yAxisPosArray.
	<OriginArray>	Array	Structured array.

		<pre> Array("NAME:Origin", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=", <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", "OnEdge", or "AbsolutePosition">, "UParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "VParam:=" , <float between 0 and 1 representing the relative position of the point on the edge or face>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>) </pre> IsAttachedToEntity specifies whether the point is anchored. If True, provide UParam and VParam. If False, provide XPosition, YPosition, and ZPosition to provide fixed position. Pass "0" for unused parameters.
	<xAxisArray>	Array Structured array for absolute position: <pre> Array("NAME:xAxis", "DirectionType:=" , "AbsoluteDirection", "EdgeID:=" , <integer>, "FaceID:=" , <integer>, </pre>

			<code>"xDirection:=" , <string>, "yDirection:=" , <string>, "zDirection:=" , <string>, "UParam:=" , <float>, "VParam:=" , <float>)</code>
	<code><xAxisPosArray></code>	Array	Structured array for relative position: <code>Array("NAME:xAxisPos", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=" , <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", or "OnEdge">, "UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>)</code>
	<code><yAxisArray></code>	Array	Structured array for absolute position: <code>Array("NAME:yAxis", "DirectionType:=" , "AbsoluteDirection",</code>

		<pre> "EdgeID:=" , <integer>, "FaceID:=" , <integer>, "xDirection:=" , <string>, "yDirection:=" , <string>, "zDirection:=" , <string>, "UParam:=" , <float>, "VParam:=" , <float> </pre>
<yAxisPosArray>	Array	Structured array for relative position: <pre> Array("NAME:yAxisPos", "IsAttachedToEntity:=" , <boolean>, "EntityID:=" , <integer>, "FacetedBodyTriangleIndex:=" , <integer>, "TriangleVertexIndex:=" , <integer>, "PositionType:=" , <string "OnVertex", "EdgeCenter", "FaceCenter", or "OnEdge">, "UParam:=" , <float>, "VParam:=" , <float>, "XPosition:=" , <string>, "YPosition:=" , <string>, "ZPosition:=" , <string>) </pre>
<AttributesArray>	Array	Structured array. See: AttributesArray . Use to select the coordinate system to edit.

Return Value	None.

Python Syntax	EditObjectCS(<Parameters>,<AttributesArray>)
Python Example	<pre>oEditor.EditObjectCS(["NAME:ObjectCSParameters", ["NAME:Origin", "IsAttachedToEntity:=" , True, "EntityID:=" , 59, "FacetedBodyTriangleIndex:=" , -1, "TriangleVertexIndex:=" , -1, "PositionType:=" , "OnVertex", "UParam:=" , 0, "VParam:=" , 0, "XPosition:=" , "0", "YPosition:=" , "0", "ZPosition:=" , "0"], "MoveToEnd:=" , False, "ReverseXAxis:=" , False, "ReverseYAxis:=" , False,</pre>

```
[ "NAME:xAxis",  
  "DirectionType:="      , "AbsoluteDirection",  
  "EdgeID:="             , -1,  
  "FaceID:="             , -1,  
  "xDirection:="         , "1",  
  "yDirection:="         , "0",  
  "zDirection:="         , "0",  
  "UParam:="             , 0,  
  "VParam:="             , 0  
],  
[ "NAME:yAxis",  
  "DirectionType:="      , "AbsoluteDirection",  
  "EdgeID:="             , -1,  
  "FaceID:="             , -1,  
  "xDirection:="         , "0",  
  "yDirection:="         , "1",  
  "zDirection:="         , "0",  
  "UParam:="             , 0,  
  "VParam:="             , 0  
]  
],
```

```
["NAME:Attributes",  
    "Name:="          , "ObjectCS1",  
    "PartName:="      , "Box2"  
])
```

VB Syntax	EditObjectCS <Parameters> <AttributesArray>
VB Example	<pre>oEditor.EditObjectCS Array("NAME:ObjectCSParameters", Array("NAME:Origin", "IsAt- tachedToEntity:=", _ false, "EntityID:=", -1, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", _ -1, "PositionType:=", "AbsolutePosition", "UParam:=", 0, "VParam:=", 0, "XPosition:=", _ "0mm", "YPosition:=", "0mm", "ZPosition:=", "0mm"), "MoveToEnd:=", false, "ReverseXAx- is:=", _ false, "ReverseYAxis:=", false, Array("NAME:xAxisPos", "IsAttachedToEntity:=", true, "EntityID:=", _ 80, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", -1, "PositionType:=", _ "EdgeCenter", "UParam:=", 0, "VParam:=", 0, "XPosition:=", "0", "YPosition:=", _ "0", "ZPosition:=", "0"), Array("NAME:yAxisPos", "IsAttachedToEntity:=", true, "EntityID:=", _ 69, "FacetedBodyTriangleIndex:=", -1, "TriangleVertexIndex:=", -1, "PositionType:=", _ "EdgeCenter", "UParam:=", 0, "VParam:=", 0, "XPosition:=", "0", "YPosition:=", _</pre>

```
"0", "ZPosition:=", "0")), Array("NAME:Attributes", "Name:=", "ObjectCS2",
"PartName:=", _
"Box3")
```

EditRelativeCS

Edits an existing Relative Coordinate System. See: [CreateRelativeCS](#).

UI Access	Modeler > Coordinate System > Edit.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:RelativeCSParameters", "Mode:=" , "Axis/Position", "OriginX:=" , <string>, "OriginY:=" , <string>, "OriginZ:=" , <string>, "XAxisXvec:=" , <string>, "XAxisYvec:=" , <string>, "XAxisZvec:=" , <string>, "YAxisXvec:=" , <string>, "YAxisYvec:=" , <string>, "YAxisZvec:=" , <string>)</pre>
	<AttributesArray>	Array	Structured array. See: AttributesArray . Use to select the coordinate system to edit.
Return Value	None.		

Python Syntax	EditRelativeCS(<Parameters>,<AttributesArray>)
Python Example	<pre>oEditor.EditRelativeCS(["NAME:RelativeCSParameters", "Mode:=" , "Axis/Position", "OriginX:=" , "0.62mm", "OriginY:=" , "-0.7mm", "OriginZ:=" , "0mm", "XAxisXvec:=" , "1mm", "XAxisYvec:=" , "0mm", "XAxisZvec:=" , "0mm", "YAxisXvec:=" , "0mm", "YAxisYvec:=" , "1mm", "YAxisZvec:=" , "0mm"], ["NAME:Attributes", "Name:=" , "RelativeCS1"])</pre>

VB Syntax	EditRelativeCS <Parameters> <AttributesArray>

VB Example	<pre>oEditor.EditRelativeCS Array("NAME:RelativeCSParameters", "Mode:=", "Axis/Position", "OriginX:=", _ "0mm", "OriginY:=", "0mm", "OriginZ:=", "0mm", "XAxisXvec:=", "0.66mm", "XAxisYvec:=", _ "0.28mm", "XAxisZvec:=", "0mm", "YAxisXvec:=", "0.06mm", "YAxisYvec:=", "0.14mm", "YAxisZvec:=", _ "0mm"), Array("NAME:Attributes", "Name:=", "RelativeCS1")</pre>

Export

Exports the model to a file.

Note: This script does not export image file types or GDSII files. See: [ExportModelImageToFile](#) and ExportGDSII.

UI Access	Modeler > Export		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:ExportParameters", "AllowRegionDependentPartSelectionForPMLCreation:=", <boolean>, "AllowRegionSelectionForPMLCreation:=", <boolean>, "Selections:=" , <string list>, "File Name:=" , <string filepath>, "Major Version:=" , <integer (-1 if not applicable)>,</pre>

			"Minor Version:=" , <integer (-1 if not applicable)>)
Return Value	None.		

Python Syntax	Export(< <i>Parameters</i> >)
Python Example	<pre>oEditor.Export (["NAME:ExportParameters", "AllowRegionDependentPartSelectionForPMLCreation:=", True, "AllowRegionSelectionForPMLCreation:=", True, "Selections:=" , "Box1,Box2,Box3", "File Name:=" , "C:/Users/jdoe/Desktop/export.sab", "Major Version:=" , 25, "Minor Version:=" , 0])</pre>

VB Syntax	Export < <i>Parameters</i> >
VB Example	<pre>oEditor.Export Array("NAME:ExportParameters", "AllowRegionDependentPartSelectionForPMLCreation:=", True, "AllowRegionSelectionForPMLCreation:=", True,</pre>

	"Selections:="	, "Box1,Box2,Box3",
	"File Name:="	, "C:/Users/jdoe/Desktop/export.sab",
	"Major Version:="	, 25,
	"Minor Version:="	, 0)

ExportModelImageToFile

Exports the model as an image file (*.avz, *.bmp, *.gif, *.jpeg, *.tiff, *.wrl). In Release 23.1, this command is fully supports -ng (non-graphical) mode. To export to Ensign use *.avz. For export to Ensign in -ng mode, the corresponding version of Ensign must be installed. On Linux, it might need manual set environment variable AWP_ROOT212 to its installation path, e.g. AWP_ROOT212-2=/installations/ansys_inc/v212/ for AnsysEDT v21.2 and Ensign 21.2.

ExportModelImageToFile supports export overlay of polar plot 3D with existing transformation (scaling, rotation and translation) in -ng (non-graphical) mode.

UI Access	Modeler > Export.		
Parameters	Name	Type	Description
	<path>	String	Full file path including extension.
	<width>	Integer	Width in pixels (use 0 for default).
	<height>	Integer	Height in pixels (use 0 for default).
	<Parameters>	Array	Structured array. Array ("NAME:SaveImageParams", "ShowAxis:=" , <string containing boolean>, "ShowGrid:=" , <string containing boolean>, "ShowRuler:=" , <string containing boolean>,

		<pre>"ShowRegion:=" , <string>, "Selections:=" , <string>, "FieldPlotSelections:=", <string>' # Comma delimited string. #Use to set which field plot to show. "FitToSelections:=" , "",</pre> Note: "FitToSelections" specify geometry objects for the "Fit" operation. <pre>"FitToFieldPlotSelection:=" , ""</pre> Note: "FitToFieldPlotSelections" specifies field plots for the "Fit" operation. <pre>"AutoFit:=" , "True",</pre> Note: If FitToSlections or FitToFieldPlotSelections are used , then then AutoFit is true, it makes sure color key does not overlap field plot. It is true by default. If neither is used, then "AutoFit" will "Fit" to full model. <pre>"Orientation:=" , <string> "ShowOrientationGadget:=" , <False></pre>

Return Value	None.

Python Syntax	<code>ExportModelImageToFile(<path> <width> <height> <Parameters>)</code>
Python Example	<pre>oEditor.ExportModelImageToFile ("C:/Users/jdoe/Desktop/export.bmp", 1920, 1080, ["NAME:SaveImageParams", "ShowAxis:=" , "True", "ShowGrid:=" , "True", "ShowRuler:=" , "True", "ShowRegion:=" , "Default", "Selections:=" , "", "FieldPlotSelections:=" , "", "FitToSelections:=" , "", "FitToFieldPlotSelections:=" , "", "AutoFit:=" , "True", "Orientation:=" , ""]) </pre>

VB Syntax	ExportModelImageToFile <path> <width> <height> <Parameters>
VB Example	<pre>oEditor.ExportModelImageToFile "C:/Users/jdoe/Desktop/export.bmp", 1383, 512, Array("NAME:SaveImageParams", "ShowAxis:=", "True", "ShowGrid:=", "True", "ShowRuler:=", "True", "ShowRegion:=", "Default", "Selections:=", "", "FitToFieldPlotSelections:=", "", "AutoFit:=", "True", "Orientation:=", ""</pre>

Fillet

Performs a fillet on specified edge(s).

UI Access	Modeler > Fillet.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array.

			Array("NAME:Parameters", <FilletParametersArray>)
	<FilletParametersArray>	Array	Structured array. Array("NAME:FilletParameters", "Edges:=" , <array containing integer edge IDs>, "Vertices:=" , <empty array>, "Radius:=" , <string>, "Setback:=" , <string>)
Return Value	None.		

Python Syntax	Fillet(<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.Fillet(["NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model"], ["NAME:Parameters", ["NAME:FilletParameters", "Edges:=" , [13], "Vertices:=" , [], "Radius:=" , "1mm",</pre>

	<pre>"Setback:=" , "0mm"]])</pre>

VB Syntax	Fillet <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.Fillet Array("NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model") Array("NAME:Parameters", Array("NAME:FilletParameters", "Edges:=" , Array(13) , "Vertices:=" , Array() , "Radius:=" , "1mm" , "Setback:=" , "0mm"))</pre>

FlattenGroup

Flattens a specified history tree group.

UI Access	Modeler > Group > Flatten.		
Parameters			
	Name	Type	Description
	<GroupID>	Array	Structured array.

			Array("Groups:=", Array(<string list of group IDs>))
Return Value	None.		

Python Syntax	FlattenGroup (<GroupID>)		
Python Example	oEditor.FlattenGroup(["Groups:=", ["Group1"]])		

VB Syntax	FlattenGroup <GroupID>		
VB Example	oEditor.FlattenGroup Array("Groups:=", Array("Group1"))		

GenerateHistory

Generates the history for specified 1D object(s).

UI Access	Modeler > Generate History.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	GenerateHistory (<SelectionsArray>)		
Python Example	oEditor.GenerateHistory(["NAME:Selections",		

	<pre>"Selections:=" , "Polyline1", "NewPartsModelFlag:=" , "Model", "UseCurrentCS:=" , True 1)</pre>

VB Syntax	GenerateHistory <SelectionsArray>
VB Example	<pre>oEditor.GenerateHistory Array("NAME:Selections", "Selections:=" , "Polyline1", "NewPartsModelFlag:=" , "Model", "UseCurrentCS:=" , True)</pre>

HealObject

Heals an imported object.

UI Access	Modeler > Model Preparation > Heal.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:ObjectHealingParameters", "Version:=" , <integer>,

			<pre> "AutoHeal:=" , <boolean>, "TolerantStitch:=" , <boolean>, "SimplifyGeom:=" , <boolean>, "TightenGaps:=" , <boolean>, "HealToSolid:=" , <boolean>, "StopAfterFirstStitchError:=" , <boolean>, "MaxStitchTol:=" , <float>, "ExplodeAndStitch:=" , <boolean>, "GeomSimplificationTol:=" , <integer>, "MaximumGeneratedRadiusForSimplification:=" , <integer>, "SimplifyType:=" , <integer>, "TightenGapsWidth:=" , <float>, "RemoveSliverFaces:=" , <boolean>, "RemoveSmallEdges:=" , <boolean>, "RemoveSmallFaces:=" , <boolean>, "SliverFaceTol:=" , <integer>, "SmallEdgeTol:=" , <integer>, "SmallFaceAreaTol:=" , <integer>, "BoundingBoxScaleFactor:=" , <integer>, "RemoveHoles:=" , <boolean>, "RemoveChamfers:=" , <boolean>, </pre>

			<code>"RemoveBlends:=" , <boolean>, "HoleRadiusTol:=" , <integer>, "ChamferWidthTol:=" , <integer>, "BlendRadiusTol:=" , <integer>, "AllowableSurfaceAreaChange:=", <integer>, "AllowableVolumeChange:=", <integer>)</code>
Return Value	None.		

Python Syntax	HealObject(<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.HealObject(["NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model"], ["NAME:ObjectHealingParameters", "Version:=" , 1, "AutoHeal:=" , True, "TolerantStitch:=" , True, "SimplifyGeom:=" , True, "TightenGaps:=" , True,</pre>

```

"HealToSolid:="          , False,
"StopAfterFirstStitchError:=", False,
"MaxStitchTol:="         , 0.001,
"ExplodeAndStitch:="     , True,
"GeomSimplificationTol:=", -1,
"MaximumGeneratedRadiusForSimplification:=", -1,
"SimplifyType:="         , 2,
"TightenGapsWidth:="     , 1E-06,
"RemoveSliverFaces:="    , False,
"RemoveSmallEdges:="     , False,
"RemoveSmallFaces:="     , False,
"SliverFaceTol:="        , 0,
"SmallEdgeTol:="         , 0,
"SmallFaceAreaTol:="     , 0,
"BoundingBoxScaleFactor:=", 1250,
"RemoveHoles:="          , False,
"RemoveChamfers:="       , False,
"RemoveBlends:="         , False,
"HoleRadiusTol:="        , 0,
"ChamferWidthTol:="      , 0,
"BlendRadiusTol:="       , 0,

```

	<pre>"AllowableSurfaceAreaChange:=", 5, "AllowableVolumeChange:=", 5])</pre>

VB Syntax	HealObject <SelectionsArray> <Parameters>

VB Examp- le	<pre>oEditor.HealObject Array("NAME:Selections", "Selections:=", "Box2", "NewPartsModelFlag:=", _ "Model"), Array("NAME:ObjectHealingParameters", "Version:=", 1, "AutoHeal:=", false, "TolerantStitch:=", _ true, "SimplifyGeom:=", true, "TightenGaps:=", true, "HealToSolid:=", false, "StopAfter- FirstStitchError:=", _ false, "MaxStitchTol:=", 0.001, "ExplodeAndStitch:=", true, "GeomSimplificationTol:=", _ -1, "MaximumGeneratedRadiusForSimplification:=", -1, "SimplifyType:=", 2, "TightenGapsWidth:=", _ 1E-06, "RemoveSliverFaces:=", false, "RemoveSmallEdges:=", false, "RemoveSmallFaces:=", _ false, "SliverFaceTol:=", 0, "SmallEdgeTol:=", 0, "SmallFaceAreaTol:=", 0, "Bound- ingBoxScaleFactor:=", _ 1250, "RemoveHoles:=", false, "RemoveChamfers:=", false, "RemoveBlends:=", _ false, "HoleRadiusTol:=", 0, "ChamferWidthTol:=", 0, "BlendRadiusTol:=", 0, "Allow- ableSurfaceAreaChange:=", _ 5, "AllowableVolumeChange:=", 5)</pre>

GetActiveCoordinateSystem

Returns the active coordinate system.

UI Access	None.
Parameters	None.
Return Value	String name of active coordinate system.

Python Syntax	GetActiveCoordinateSystem()
Python Example	<code>oEditor.GetActiveCoordinateSystem()</code>

VB Syntax	GetActiveCoordinateSystem
VB Example	<code>dim csName csName = oEditor.GetActiveCoordinateSystem</code>

GetCoordinateSystems

Returns the names of coordinate systems in the design.

UI Access	None.
Parameters	None.
Return Value	Array containing string names of coordinate systems.

Python Syntax	GetCoordinateSystems()
Python Example	<code>oEditor.GetCoordinateSystems()</code>

VB Syntax	GetCoordinateSystems()
VB Example	<code>dim csNames csNames = oEditor.GetCoordinateSystems</code>

Import

Imports a 3D model file.

Note:

This script does not import DXF or GDSII models. See: [ImportDXF](#) and [ImportGDSII](#).

UI Access	Modeler > Import.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre> Array("NAME:NativeBodyParameters", "HealOption:=" , <integer>, "Options:=" , <string>, "FileType:=" , <string "UnRecognized">, "MaxStitchTol:=" , <integer>, "ImportFreeSurfaces:=" , <boolean>, "GroupByAssembly:=" , <boolean>, "CreateGroup:=" , <boolean>, "STLFileUnit:=" , <string>, "MergeFacesAngle:=" , <float>, "HealSTL:=" , <boolean>, "ReduceSTL:=" , <boolean>, "ReduceMaxError:=" , <integer>,</pre>

			<code>"ReducePercentage:=" , <integer>, "PointCoincidenceTol:=" , <float>, "CreateLightweightPart:=", <boolean>, "ImportMaterialNames:=" , <boolean>, "SeparateDisjointLumps:=", <boolean>, "SourceFile:=" , <string>)</code>
Return Value	None.		

Python Syntax	Import(< <i>Parameters</i> >)
Python Example	<pre>oEditor.Import(["NAME:NativeBodyParameters", "HealOption:=" , 0, "Options:=" , "-1", "FileType:=" , "UnRecognized", "MaxStitchTol:=" , -1, "ImportFreeSurfaces:=" , False, "GroupByAssembly:=" , False, "CreateGroup:=" , True, "STLFileUnit:=" , "Auto", "MergeFacesAngle:=" , 0.02,</pre>

	<pre> "HealSTL:=" , False, "ReduceSTL:=" , False, "ReduceMaxError:=" , 0, "ReducePercentage:=" , 100, "PointCoincidenceTol:=" , 1E-06, "CreateLightweightPart:=", False, "ImportMaterialNames:=" , False, "SeparateDisjointLumps:=", False, "SourceFile:=" , "C:\\Users\\jdoe\\Desktop\\export.model" l) </pre>

VB Syntax	Import <Parameters>
VB Example	<pre> oEditor.Import Array("NAME:NativeBodyParameters", "HealOption:=" , 0, "Options:=" , "-1", "FileType:=" , "UnRecognized", "MaxStitchTol:=" , -1, "ImportFreeSurfaces:=" , False, "GroupByAssembly:=" , False, "CreateGroup:=" , True, "STLFileUnit:=" , "Auto", </pre>

	<pre>"MergeFacesAngle:=" , 0.02, "HealSTL:=" , False, "ReduceSTL:=" , False, "ReduceMaxError:=" , 0, "ReducePercentage:=" , 100, "PointCoincidenceTol:=" , 1E-06, "CreateLightweightPart:=", False, "ImportMaterialNames:=" , False, "SeparateDisjointLumps:=", False, "SourceFile:=" , "C:\\Users\\jdoe\\Desktop\\export.model")</pre>

ImportDXF [Modeler]

Imports a DXF model file.

UI Access	Modeler > Import.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:options", "FileName:=" , <string>, "Scale:=" , <float>, "AutoDetectClosed:=" , <boolean>, "SelfStitch:=" , <boolean>,

			<pre> "DefeatureGeometry:=" , <boolean>, "DefeatureDistance:=" , <integer>, "RoundCoordinates:=" , <boolean>, "RoundNumDigits:=" , <integer>, "WritePolyWithWidthAsFilledPoly:=", <boolean>, "ImportMethod:=" , <integer>, "2DSheetBodies:=" , <boolean>, <layersArray> </pre>
	<code><layersArray></code>	Array	Structured array. <pre> Array("NAME:LayerInfo", <layer>, <layer>, <layer>, ...) </pre>
	<code><layer></code>	Array	Structured array. <pre> Array("NAME:<layerName>", "source:=" , <string>, "display_source:=" , <string>, "import:=" , <boolean>, "dest:=" , <string>, "dest_selected:=" , <boolean>, "layer_type:=" , <string>) </pre>
Return Value	None.		
Python Syntax	ImportDXF(<Parameters>)		

Python Example

```
oEditor.ImportDXF(  
    ["NAME:options",  
    "FileName:=", "C:/Users/jdoo/Desktop/export.dxf",  
    "Scale:="          , 0.001,  
    "AutoDetectClosed:=" , True,  
    "SelfStitch:="      , True,  
    "DefeatureGeometry:=" , False,  
    "DefeatureDistance:=" , 0,  
    "RoundCoordinates:=" , False,  
    "RoundNumDigits:="   , 4,  
    "WritePolyWithWidthAsFilledPoly:=", False,  
    "ImportMethod:="     , 1,  
    "2DSheetBodies:="    , False,  
    ["NAME:LayerInfo",  
    ["NAME:0",  
    "source:="          , "0",  
    "display_source:="  , "0",  
    "import:="          , True,  
    "dest:="            , "0",  
    "dest_selected:="   , False,
```

```
"layer_type:="          , "signal"
],
["NAME:LAYER_1",
"source:="              , "LAYER_1",
"display_source:="      , "LAYER_1",
"import:="              , True,
"dest:="                , "LAYER_1",
"dest_selected:="       , False,
"layer_type:="          , "signal"
],
["NAME:LAYER_2",
"source:="              , "LAYER_2",
"display_source:="      , "LAYER_2",
"import:="              , True,
"dest:="                , "LAYER_2",
"dest_selected:="       , False,
"layer_type:="          , "signal"
]
]
])
```

VB Syntax	ImportDXF <Parameters>
VB Example	<pre>oEditor.ImportDXF Array("NAME:options", "FileName:=", _ "C:/Users/jdoe/Desktop/export.dxf", "Scale:=", 0.001, "AutoDetectClosed:=", _ true, "SelfStitch:=", true, "DefeatureGeometry:=", false, "DefeatureDistance:=", _ 0, "RoundCoordinates:=", false, "RoundNumDigits:=", 4, "WritePolyWithWidthAsFilledPoly:=", _ false, "ImportMethod:=", 1, "2DSheetBodies:=", false, Array("NAME:LayerInfo", Array ("NAME:0", "source:=", _ "0", "display_source:=", "0", "import:=", true, "dest:=", "0", "dest_selected:=", _ false, "layer_type:=", "signal"), Array("NAME:LAYER_1", "source:=", "LAYER_1", "dis- play_source:=", _ "LAYER_1", "import:=", true, "dest:=", "LAYER_1", "dest_selected:=", false, "layer_ type:=", _ "signal"), Array("NAME:LAYER_2", "source:=", "LAYER_2", "display_source:=", "LAYER_2", "import:=", _ true, "dest:=", "LAYER_2", "dest_selected:=", false, "layer_type:=", "signal")))</pre>

ImportGDSII [Modeler]

Imports a GDSII model file.

UI Access	Modeler > Import.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array.

			<pre> Array("NAME:options", "FileName:=" , <string>, "FlattenHierarchy:=" , <boolean>, "ImportMethod:=" , <integer>, <layerMapArray>, <layerMapArray>, <layerMapArray>,... "OrderMap:=" , ["entry:=" , <entry>, <entry>, <entry>,...]]) </pre>
	<layerMapArray>	Array	Structured array. <pre> Array("NAME:LayerMapInfo", "LayerNum:=" , <integer>, "DestLayer:=" , <string>, "layer_type:=" , <string>) </pre>
	<entry>	Array	Structured array. <pre> Array("order:=" , <integer LayerNum>, "layer:=" , <string DestLayer>) </pre>
Return Value	None.		

Python Syntax	ImportGDSII(<Parameters>)
Python Example	oEditor.ImportGDSII(

```
[ "NAME:options",
  "FileName:=", "C:/Users/coil2.gds",
  "FlattenHierarchy:=", , True,
  "ImportMethod:=", , 1,
  [ "NAME:LayerMap",
    [ "NAME:LayerMapInfo",
      "LayerNum:=", , 12,
      "DestLayer:=", , "Signal12",
      "layer_type:=", , "signal"
    ],
    [ "NAME:LayerMapInfo",
      "LayerNum:=", , 13,
      "DestLayer:=", , "Signal13",
      "layer_type:=", , "signal"
    ],
    [ "NAME:LayerMapInfo",
      "LayerNum:=", , 14,
      "DestLayer:=", , "Signal14",
      "layer_type:=", , "signal"
    ]
  ]
]
```



```
],  
"OrderMap:=",  
[  
  "entry:=",  
    ["order:=", 0, "layer:=", "Signal12"],  
  "entry:=",  
    ["order:=", 1, "layer:=", "Signal13"],  
  "entry:=",  
    ["order:=", 2, "layer:=", "Signal14"]  
]  
])
```

VB Syntax	ImportGDSII <Parameters>
VB Example	<pre>oEditor.ImportGDSII Array("NAME:options", "FileName:=", "C:/export.gds", "FlattenHierarchy:=", True, "ImportMethod:=", 1, Array("NAME:LayerMap", Array("NAME:LayerMapInfo", "LayerNum:=", 1, "DestLayer:=", "Signal1",</pre>

	<pre> "layer_type:=", "signal"])) "OrderMap:=", Array("entry:=", Array("order:=", 0, "layer:=", "Signal1")))</pre>

Intersect

Intersects specified objects.

UI Access	Modeler > Boolean > Intersect.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:IntersectParameters", "KeepOriginals:=" , <boolean True to keep original objects; False to delete>)
Return Value	None.		

Python Syntax	Intersect(<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.Intersect(["NAME:Selections", "Selections:=", "Rectangle1,Rectangle2"])</pre>

	<pre>], ["NAME:IntersectParameters", "KeepOriginals:=", False]</pre>

VB Syntax	Intersect <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.Intersect Array("NAME:Selections", "Selections:=", "Rectangle1,Rectangle2") Array("NAME:IntersectParameters", "KeepOriginals:=", False)</pre>

MoveCStoEnd

Moves a specified Object Coordinate System to the end of the History tree.

UI Access	Modeler > Coordinate System > Move CS to End.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	MoveCStoEnd (<SelectionsArray>)
Python Example	<pre>oEditor.MoveCStoEnd(</pre>

	<pre>["NAME:Selections", "Selections:=" , "ObjectCS1"]</pre>

VB Syntax	<code>MoveCSToEnd <SelectionsArray></code>
VB Example	<code>oEditor.MoveCSToEnd Array("NAME:Selections", "Selections:=", "ObjectCS1")</code>

MoveEntityToGroup

Moves a specified entity or entities to a specified group.

UI Access	Drag item into the group in the history tree.		
Parameters	Name	Type	Description
	<Objects>	Array	Structured array. Selects the entity/entities to move. Array("Objects:=" , <array containing string object IDs>)
	<MoveEntityToGroup>	Array	Structured array. Array("ParentGroup:=" , <string group name>)
Return Value	None.		

Python Syntax	<code>MoveEntityToGroup (<Objects>, <MoveEntityToGroup>)</code>
Python Example	<code>oEditor.MoveEntityToGroup(</code>

	<pre>["Objects:=", ["Box3"]], ["ParentGroup:=", "Box_Group"])</pre>

VB Syntax	<code>MoveEntityToGroup <Objects> <MoveEntityToGroup></code>
VB Example	<pre>oEditor.MoveEntityToGroup Array("Objects:=", Array("Box3")) Array("ParentGroup:=", "Box_Group")</pre>

MoveFaces

Moves the specified faces along normal or along a vector.

UI Access	Modeler > Surface > Move Faces > Along [Normal/Vector].		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<MoveFacesParametersArray>	Array	Structured array. <pre>Array("NAME:Parameters", <FacesOfOneObjToMove>, <FacesOfOneObjToMove>, ...)</pre>
	<FacesOfOneObjToMove>	Array	Structured array. <pre>Array("Name:MoveFacesParameters", "MoveAlongNormalFlag:=", <boolean>, "OffsetDistance:=", <string>, "MoveVectorX:=", <string>,</pre>

			<pre>"MoveVectorY:=", <string>, "MoveVectorZ:=", <string>, "FacesToMove:=", <array>)</pre> MoveAlongNormalFlag specifies whether to move along the face normal or along a vector. If false, provide the MoveVectorX, MoveVectorY, and MoveVectorZ parameters. FacesToMove is an array of integers (the IDs of the faces to move).
Return Value	None		

Python Syntax	MoveFaces (<SelectionsArray>, <MoveFacesParametersArray>)
Python Example	<pre>oEditor.MoveFaces (["NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"], ["NAME:Parameters", ["NAME:MoveFacesParameters", "MoveAlongNormalFlag:=", True, "OffsetDistance:=", "1mm", "MoveVectorX:=", "0mm", "MoveVectorY:=", "0mm", "MoveVectorZ:=", "0mm",</pre>

	<pre> "FacesToMove:=", [183]] })</pre>

VB Syntax	MoveFaces <SelectionsArray>, <MoveFacesParametersArray>
VB Example	<pre>oEditor.MoveFaces _ Array("NAME:Selections", "Selections:=", _ "Box2,Box1"), _ Array("NAME:Parameters", _ Array("NAME:MoveFacesParameters", _ "MoveAlongNormalFlag:=", true, _ "OffsetDistance:=", "1mm", _ "FacesToMove:=", Array(218)), Array("NAME:MoveFacesParameters", _ "MoveAlongNormalFlag:=", false, _ "OffsetDistance:=", "1mm", _ "MoveVectorX:=", "1mm", _ "MoveVectorY:=", "0mm", _ "MoveVectorZ:=", "0mm", _ "FacesToMove:=", Array(185)))</pre>

ProjectSheet

Project a sheet object, typically for modeling thin conformal deposits. Typically followed by **Thicken Sheet**.

UI Access	Click Modeler > Surface > Project Sheet .		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Array containing string. Array("NAME:ProjectSheetParameters")
Return Value	None.		

Python Syntax	ProjectSheet (<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.ProjectSheet(['NAME:Selections', 'Selections:=' , 'Box1,Box2,Polyline1'], ['NAME:ProjectSheetParameters'])</pre>

VB Syntax	ProjectSheet <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.ProjectSheet Array("NAME:Selections",</pre>

	<pre>"Selections:=", "Box1,Box2,Polyline1"), Array("NAME:ProjectSheetParameters")</pre>

PurgeHistory

Purges the history of a specified object.

UI Access	Modeler > Purge History.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	PurgeHistory (<SelectionsArray>)
Python Example	<pre>oEditor.PurgeHistory(["NAME:Selections", "Selections:=", "Box2", "NewPartsModelFlag:=", "Model"])</pre>

VB Syntax	PurgeHistory <SelectionsArray>
VB Example	<pre>oEditor.PurgeHistory Array("NAME:Selections",</pre>

	<pre>"Selections:=", "Box2", "NewPartsModelFlag:=", "Model")</pre>

Section

Creates a 2D cross-section of the selection in the specified plane.

UI Access	Modeler > Surface > Section.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array ("NAME:SectionToParameters", "CreateNewObjects:=" , <boolean>, "SectionPlane:=" , <string "XY", "YZ", or "ZX">, "SectionCrossObject:=" , <boolean>)
Return Value	None.		

Python Syntax	Section (<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.Section(["NAME:Selections", "Selections:=" , "Cone1", "NewPartsModelFlag:=" , "Model"</pre>

```

],
["NAME:SectionToParameters",
    "CreateNewObjects:="      , True,
    "SectionPlane:="         , "XY",
    "SectionCrossObject:="    , False
])

```

VB Syntax	Section <SelectionsArray> <Parameters>
VB Example	<pre> oEditor.Section Array("NAME:Selections", "Selections:=" , "Cone1", "NewPartsModelFlag:=" , "Model") Array("NAME:SectionToParameters", "CreateNewObjects:=" , True, "SectionPlane:=" , "XY", "SectionCrossObject:=" , False) </pre>

SeparateBody

Separates the bodies of specified multi-lump objects.

UI Access	Modeler > Boolean > Separate Bodies.

Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	SeparateBody (<SelectionsArray>)
Python Example	<pre>oEditor.SeparateBody(["NAME:Selections", "Selections:=", "Rectangle1,Circle1", "NewPartsModelFlag:=", "Model"])</pre>

VB Syntax	SeparateBody <SelectionsArray>
VB Example	<pre>oEditor.SeparateBody Array("NAME:Selections", "Selections:=", "Rectangle1,Circle1", "NewPartsModelFlag:=", "Model")</pre>

SetModelUnits

Sets the model units.

UI Access	Modeler > Units.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. <pre>Array("NAME:Units Parameter", "Units:=", <string>, "Rescale:=", <boolean True to rescale model; else False>)</pre> To see valid unit strings, select Modeler > Units.
Return Value	None.		

Python Syntax	SetModelUnits (<Parameters>)
Python Example	<pre>oEditor.SetModelUnits(["NAME:Units Parameter", "Units:=", "km", "Rescale:=", False])</pre>

VB Syntax	SetModelUnits <Parameters>
VB Example	<pre>oEditor.SetModelUnits Array("NAME:Units Parameter",</pre>

	<pre>"Units:=", "km", "Rescale:=", False)</pre>

SetWCS

Sets the working coordinate system.

UI Access	Modeler > Coordinate System > Set Working CS.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. Array("NAME:SetWCS Parameter", "Working Coordinate System:=", <string CS name>, "RegionDepCSOk:=" , <boolean True if region-dependent; else False)
Return Value	None.		

Python Syntax	SetWCS (<Parameters>)
Python Example	<pre>oEditor.SetWCS(["NAME:SetWCS Parameter", "Working Coordinate System:=", "Global", "RegionDepCSOk:=", False])</pre>

VB Syntax	SetWCS <Parameters>
VB Example	<pre>oEditor.SetWCS Array("NAME:SetWCS Parameter", "Working Coordinate System:=", "Global", "RegionDepCSOk:=", False)</pre>

ShowWindow

Opens the active 3D Modeler window.

UI Access	None.
Parameters	None.
Return Value	None.

Python Syntax	ShowWindow
Python Example	<pre>oEditor.ShowWindow()</pre>

VB Syntax	ShowWindow
VB Example	<pre>oEditor.ShowWindow</pre>

Split

Splits the specified object(s) along a plane.

UI Access	Modeler > Boolean > Split.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. <pre>Array("NAME:SplitToParameters", "SplitPlane:=" , <string "XY", "YZ", "ZX", or "Dummy">, "WhichSide:=" , <string "PositiveOnly", "Neg- ativeOnly" or "Both">, "ToolType:=" , <string "PlaneTool" or "FaceTool">, "ToolEntityID:=" , <-1 if using SplitPlane; else FaceID>, "SplitCrossingObjectsOnly:=" , <boolean>, "DeleteInvalidObjects:=" , <boolean>)</pre> You can split an object either along an existing plane , or by creating a plane using an edge. For existing plane: • SplitPlane is "XY", "YZ", or "ZX" • ToolType is "PlaneTool" • ToolEntityID is -1 Possible values for 2D XY designs are "YZ", "ZX". Possible values for 2D RZ designs are "XY", "YZ".

			For an edge: • SplitPlane is "Dummy" • ToolType is "EdgeTool" • ToolEntityID is the face ID
Return Value	None.		

Python Syntax	Split(<SelectionsArray>, <Parameters>)
Python Example	<pre>oEditor.Split(["NAME:Selections", "Selections:=", "NewPartsModelFlag:=",], ["NAME:SplitToParameters", "SplitPlane:=", "WhichSide:=", "ToolType:=", "ToolEntityID:=", "SplitCrossingObjectsOnly:=", "DeleteInvalidObjects:=",])</pre>

VB Syntax	Split <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.Split Array("NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model") Array("NAME:SplitToParameters", "SplitPlane:=" , "XY", "WhichSide:=" , "PositiveOnly", "ToolType:=" , "PlaneTool", "ToolEntityID:=" , -1, "SplitCrossingObjectsOnly:=", False, "DeleteInvalidObjects:=", True)</pre>

Subtract

Subtracts the specified object(s).

UI Access	Modeler > Boolean > Subtract.		
Parameters	Name	Type	Description
	<Selections>	Array	Structured array. Array("NAME:Selections", "Blank Parts:=" , <string object name(s)>, "Tool Parts:=" , <string object name(s)>)

			The <code>Tool Parts</code> object is the object being removed. The <code>Blank Parts</code> object has any <code>Tool Parts</code> overlap removed. Either string can contain more than one object.
	<Parameters>	Array	Structured array. <pre>Array("NAME:SubtractParameters", "KeepOriginals:=" , <boolean>)</pre>
Return Value	None.		

Python Syntax	<code>Subtract(<Selections>, <Parameters>)</code>		
Python Example	<pre>oEditor.Subtract(["NAME:Selections", "Blank Parts:=", "Rectangle1", "Tool Parts:=", "Rectangle2"], ["NAME:SubtractParameters", "KeepOriginals:=", False])</pre>		

VB Syntax	<code>Subtract <Selections> <Parameters></code>		
VB Example	<pre>oEditor.Subtract Array("NAME:Selections",</pre>		

	<pre>"Blank Parts:=", "Rectangle1", "Tool Parts:=", "Rectangle2") Array("NAME:SubtractParameters", "KeepOriginals:=", false)</pre>

SweepFacesAlongNormal

Sweep the specified face(s) along normal.

UI Access	Modeler > Surface > Sweep Faces Along Normal		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	Structured array. Array("NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", <faceIDarray>, "LengthOfSweep:=", "<value><units>")
Return Value	None		

Python Syntax	SweepFacesAlongNormal(<SelectionsArray> <parameters>)
Python Example	<pre>oEditor.SweepFacesAlongNormal(["NAME:Selections",</pre>

	<pre>"Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"], ["NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", [183], "LengthOfSweep:=", "0.1mm"])</pre>

VB Syntax	SweepFacesAlongNormal <SelectionsArray> <parameters>
VB Example	<pre>oEditor.SweepFacesAlongNormal Array("NAME:Selections", "Selections:=", "Rectangle1", "NewPartsModelFlag:=", "Model"), Array("NAME:Parameters", "NAME:SweepFaceAlongNormalToParameters", "FacesToDetach:=", Array(57), "LengthOfSweep:=", "0.5mm"))</pre>

ThickenSheet

Thickens a sheet object to convert it to a 3D object.

UI Access	Modeler > Surface > Thicken Sheet		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	Structured array. Array ("NAME:SheetThickenParameters", "Thickness:=" , <string>, "BothSides:=" , <boolean>)
Return Value	None.		

Python Syntax	ThickenSheet (<SelectionsArray> <parameters>)
Python Example	<pre>oEditor.ThickenSheet(["NAME:Selections", "Selections:=" , "Rectangle3", "NewPartsModelFlag:=" , "Model"], ["NAME:SheetThickenParameters", "Thickness:=" , "0.01mm", "BothSides:=" , False])</pre>

VB Syntax	ThickenSheet <SelectionsArray> <parameters>
VB Example	<pre> oEditor.ThickenSheet Array("NAME:Selections", "Selections:=" , "Rectangle3", "NewPartsModelFlag:=" , "Model") Array("NAME:SheetThickenParameters", "Thickness:=" , "0.01mm", "BothSides:=" , false) </pre>

UncoverFaces

Uncovers the specified face(s).

UI Access	Modeler > Surface > Uncover Faces		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	Structured array. <pre> Array("NAME:Parameters", Array("NAME:UncoverFacesParameters", "FacesToUncover:=" , <array of face IDs>)) </pre>
Return Value	None.		

Python Syntax	UncoverFaces(<SelectionsArray> <parameters>)
Python Example	<pre>oEditor.UncoverFaces(["NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model"], ["NAME:Parameters", ["NAME:UncoverFacesParameters", "FacesToUncover:=" , [12,16,18]]])</pre>

VB Syntax	UncoverFaces <SelectionsArray> <parameters>
VB Example	<pre>oEditor.UncoverFaces Array("NAME:Selections", "Selections:=" , "Box1", "NewPartsModelFlag:=" , "Model") Array("NAME:Parameters", Array("NAME:UncoverFacesParameters",</pre>


```
"FacesToUncover:=" , Array(12,16,18)) )
```

Unite

Unites the specified objects.

UI Access	Modeler > Boolean > Unite		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<parameters>	Array	Structured array. <pre>Array("NAME:UniteParameters", "KeepOriginals:=" , <boolean>, "TurnOnNBodyBoolean:=" , <boolean>)</pre> Note: TurnOnNBodyBoolean is set to True by default. Enabled by Parasolid, this unites <i>n</i> bodies in a single step, rather than uniting them one by one.
Return Value	None.		

Python Syntax	Unite(<SelectionsArray>, <parameters>)
Python Example	<pre>oEditor.Unite(["NAME:Selections", "Selections:=", "Rectangle1,Rectangle2"]</pre>

	<pre>], ["NAME:UniteParameters", "KeepOriginals:=", False, "TurnOnNBodyBoolean:=", True])</pre>

VB Syntax	Unite <SelectionsArray> <parameters>
VB Example	<pre>oEditor.Unite Array("NAME:Selections", "Selections:=", "Rectangle1,Rectangle2") Array("NAME:UniteParameters", "KeepOriginals:=", false, "TurnOnNBodyBoolean:=", true)</pre>

Ungroup

Ungroups a specified history tree group.

UI Access	Modeler > Group > Ungroup		
Parameters	Name	Type	Description
	<Groups>	Array	Structured array.
			Array("Groups:=", <array of group names to ungroup>)

Return Value	None

Python Syntax	Ungroup(<Groups>)
Python Example	<code>oEditor.Ungroup(["Groups:=", ["Group1"]])</code>

VB Syntax	Ungroup <Groups>
VB Example	<code>oEditor.Ungroup Array("Groups:=", Array("Group1"))</code>

WrapSheet

Wraps a sheet object to another object.

UI Access	None.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
	<Parameters>	Array	Structured array. Array("NAME:WrapSheetParameters", "Imprinted:=", <boolean>)
Return Value	None.		

Python Syntax	WrapSheet (<SelectionsArray>, <Parameters>)
Python Example	<code>oEditor.WrapSheet (</code>

	<pre>["NAME:Selections", "Selections:=", "Rectangle1,Box1"], ["NAME:WrapSheetParameters", "Imprinted:=", True])</pre>

VB Syntax	WrapSheet <SelectionsArray> <Parameters>
VB Example	<pre>oEditor.WrapSheet Array("NAME:Selections", "Selections:=", "Rectangle1,Box1"), Array("NAME:WrapSheetParameters", "Imprinted:=", true)</pre>

Other oEditor Commands

- [AddDefinitionFromBlock](#)
- [AddDefinitionFromLibFile](#)
- [AddViewOrientation](#)
- [BreakUDMConnection](#)

[ChangeProperty](#)

[Delete](#)

[FitAll](#)

[GetBodyNamesByPosition](#)

[GetChildNames \[Modeler\]](#)

[GetChildObject \[Modeler\]](#)

[GetChildTypes \[Modeler\]](#)

[GetEdgeByPosition](#)

[GetEdgeIdsFromObject](#)

[GetEdgeIdsFromFace](#)

[GetEntityListIDByName](#)

[GetExtendedDefinitionObject](#)

[GetFaceArea](#)

[GetFaceByPosition](#)

[GetFaceCenter](#)

[GetFaceIds](#)

[GetGeometryModelerMode](#)

[GetMatchedObjectName](#)

[GetModelBoundingBox](#)

[GetModelUnits](#)

[GetNumObjects](#)

[GetObjectIDByName](#)

[GetObjectName](#)

[GetObjectNameByFaceID](#)

[GetObjectsByMaterial](#)

[GetObjectsInGroup](#)

[GetObjectVolume](#)

[GetPropertyValue](#)

[GetPropEvaluatedValue](#)

[GetPropNames](#)

[GetPropSIValue](#)

[GetPropValue](#)

[GetSelections](#)

[GetUserPosition](#)

[GetVertexIDsFromEdge](#)

[GetVertexIDsFromFace](#)

[GetVertexIDsFromObject](#)

[GetVertexPosition](#)

[OpenExternalEditor](#)

[PageSetup](#)

[RenamePart](#)

[SetPropValue \[Modeler\]](#)

AddDefinitionFromBlock

Adds a material definition from block text (same definition format as would be contained in the material library file) by library type (using definition folder name). This scripting command directly supports the .AMAT (or .ASURF) definition formats.

UI Access	N/A		
Parameters	Name	Type	Description
	<defBlock>	String	Text of the new material definition in block form.
	<defFolderName>	String	Library type (by definition folder name)
	<newTimeStamp>	String	New timestamp (time_t as integer number of seconds since 1/1/1970 12:00am, as string), default is current time
	<replaceExisting>	Boolean	True to replace existing, False to choose a new unique name if an existing definition is found
Return Value	A property scripting object for the definition.		

P- y- t- h- o- n S- y- n- t- a- x	AddDefinitionFromBlock(<defBlock>, <defFolderName>, <newTimeStamp>, <replaceExisting>)
P- y- t- h-	<pre>oProject = oDesktop.NewProject() oProject.InsertDesign("HFSS", "HFSSDesign1", "DrivenModal", "")</pre>

```
oDesign = oProject.SetActiveDesign("HFSSDesign1")
oEditor = oDesign.SetActiveEditor("3D Modeler")
oEditor.CreateBox(

    [

        "NAME:BoxParameters",

        "XPosition:="          , "-0.4mm",
        "YPosition:="          , "-1mm",
        "ZPosition:="          , "0mm",
        "XSize:="               , "1.4mm",
        "YSize:="               , "1.6mm",
        "ZSize:="               , "0.6mm"

    ],

    [

        "NAME:Attributes",
        "Name:="                , "Box1",

        "Flags:="               , "",
```



```
"Color:="                , "(143 175 143)",  
  
"Transparency:="        , 0,  
  
"PartCoordinateSystem:=", "Global",  
  
"UDMId:="                , "",  
  
"MaterialValue:="        , "\"vacuum\"",  
  
"SurfaceMaterialValue:=", "\"\"",  
  
"SolveInside:="          , True,  
  
    "ShellElement:="      , False,  
  
"ShellElementThickness:=", "0mm",  
  
"IsMaterialEditable:="    , True,  
  
"UseMaterialAppearance:=", False,  
  
"IsLightweight:="         , False  
  
])
```

```
oDefinitionManager = oProject.GetDefinitionManager()

defBlock = "$begin 'vacuum2' $begin 'AttachedData' $begin 'MatAppearanceData' property_data-
='appearance_data' Red=230 Green=230 Blue=230 Transparency=0.95 $end 'MatAppearanceData'
$end 'AttachedData' simple('permittivity', 1) ModTime=1499970477 $end 'vacuum2'"

added = oDefinitionManager.AddDefinitionFromBlock(defBlock, "Materials", "10101010", True)

addedName = ''

    if isinstance(added, basestring):
        addedName = added

    elif isinstance(added, list):
        addedName = added[0]
else:

    addedName = added.GetName().replace("Materials:", "")

AddInfoMessage(os.path.basename(__file__) + " result: " + addedName)

materialNameInQuotes = "\"" + addedName + "\""

oEditor.ChangeProperty(

    [

        "NAME:AllTabs",

    ]

)
```

```

"NAME:Geometry3DAttributeTab",

[

    "NAME:PropServers",

    "Box1"

],

[

    "NAME:ChangedProps",

    [

        "NAME:Material",

        "Value:=", materialNameInQuotes

    ]

    ]

] )

```

AddDefinitionFromLibFile

Adds a material definition from a library file (e.g. AMAT file), by name and library type (using definition folder name) . This scripting command directly supports the .AMAT (or .ASURF) definition formats.

UI Access	N/A		
Parameters	Name	Type	Description
	<FilePath>	String	Path of the library file (i.e. AMAT file or ASURF file)
	<defName>	String	Which definition to use, required because a lib file can have multiple definitions.
	<defFolderName>	String	Library type (by definition folder name).
	<newTimeStamp>	String	New timestamp string (time_t as integer, number of seconds since 1/1/1970 12:00am), default is current time
	<replaceExisting>	Boolean	True to replace existing, False to choose a new unique name if an existing definition is found, default is False
Return Value	Property scripting object for the definition.		

Python Syntax	AddDefinitionFromLibFile(<FilePath>, <defName>, <defFolderName>, <newTimeStamp>,<replaceExisting>)
Python Example	<pre>oProject = oDesktop.NewProject() oProject.InsertDesign("HFSS", "HFSSDesign1", "DrivenModal", "") oDesign = oProject.SetActiveDesign("HFSSDesign1") oEditor = oDesign.SetActiveEditor("3D Modeler") oEditor.CreateBox(</pre>

```
[  
  
  "NAME:BoxParameters",  
  
  "XPosition:="          , "-0.4mm",  
    "YPosition:="        , "-1mm",  
    "ZPosition:="        , "0mm",  
    "XSize:="            , "1.4mm",  
    "YSize:="            , "1.6mm",  
    "ZSize:="            , "0.6mm"  
  ],  
  
[  
  
  "NAME:Attributes",  
    "Name:="              , "Box1",  
  
  "Flags:="              , "",  
  
  "Color:="              , "(143 175 143)",  
  
  "Transparency:="       , 0,
```

```
"PartCoordinateSystem:=", "Global",  
  
"UDMId:="                , "",  
  
"MaterialValue:="        , "\"vacuum\"",  
  
"SurfaceMaterialValue:=", "\"\"",  
  
"SolveInside:="          , True,  
  
    "ShellElement:="      , False,  
  
"ShellElementThickness:=", "0mm",  
  
"IsMaterialEditable:="    , True,  
  
"UseMaterialAppearance:=", False,  
  
"IsLightweight:="         , False  
  
])  
  
oDefinitionManager = oProject.GetDefinitionManager()  
scriptDir = os.path.dirname(os.path.realpath(__file__))  
amatFilePath = scriptDir + '/material0.txt'
```

```
materialName = "material0"

materialNameInQuotes = "\"" + materialName + "\""

added = oDefinitionManager.AddDefinitionFromLibFile(amatFilePath, materialName, "Materials", "")

addedName = ''

    if isinstance(added, basestring):
addedName = added

    elif isinstance(added, list):
addedName = added[0]
else:

    addedName = added.GetName().replace("Materials:", "")
AddInfoMessage(os.path.basename(__file__) + " result: " + addedName)
oEditor.ChangeProperty(

    [
        "NAME:AllTabs",

        [
            "NAME:Geometry3DAttributeTab",

            [
                "NAME:PropServers",

                "Box1"

            ],

            [
```

```
        "NAME:ChangedProps",
        [
            "NAME:Material",
            "Value:=", materialNameInQuotes
        ]
    ]
]
])
```

AddViewOrientation

Creates a new orientation using a script. Input parameters can be *either* vector components *or* angles. Local view orientation specific to a design can be saved in project by calling Save() on the project object.

Note:

You cannot use the name of an existing view. Default views include: top, bottom, right, left, front, back, trimetric, dimetric, and isometric.

UI Access	View > Modify Attributes > Orientation List. Enter parameters and click Add.		
Parameters	Name	Type	Description
	<orientationName>	String	Name of the new orientation.
	<isGlobalOrientation>	Long	1 to save global; else 0.
	<OrientationVectorComponents>	Array	Structured array. Use <i>either</i> <OrientationVectorComponents> <i>or</i> <OrientationAnglesArray>.

		<pre> Array("NAME:OrientationVectorComponents", Array("NAME:Direction", <float>, <float>, <float>, Array("NAME:Up", <float>, <float>, <float>)) </pre>
<OrientationAnglesArray>	Array	Structured array. Use <u>either</u> <OrientationVectorComponents> <u>or</u> <OrientationAnglesArray>. <pre> Array("NAME:OrientationAngles", "Psi:=" , <float>, "Theta:=" , <float>, "Phi:=" , <float>) </pre>
<ProjectionParams>	Array	<i>Optional.</i> Structured array containing arrays of four optional parameters: Scaling, Translation, Projection Type, and Projection Limits. <pre> Array("NAME:ProjectionParams", Array("NAME:Scaling", <float>, <float>, <float> </pre>

			<pre>), Array("NAME:Translation", <float>, <float>, <float>), "ProjectionType:=" , <string "Orthographic" or "Perspective">, Array("NAME:ProjectionLimits", <float>, <float>, <float>, <float>, <float>, <float>))</pre>
Return Value	None.		

Python Syntax	AddViewOrientation(<orientationName>, <isGlobalOrientation>, <OrientationVectorComponents or Ori-

	<code>entationAnglesArray>, <ProjectionParams>)</code>
Python Example	Example Using Angles and Projection Parameters: <pre>oEditor.AddViewOrientation("orientation3", 0, ["NAME:OrientationAngles", "Psi:=" , -161.565002441406, "Theta:=" , 32.3115005493164, "Phi:=" , 0], ["NAME:ProjectionParams", ["NAME:Scaling", 0.591607987880707, 0.591607987880707, 0.591607928276062], ["NAME:Translation", 0, 2.38418579101563E-07, -5.7486834526062</pre>

```
],  
"ProjectionType:="      , "Orthographic",  
  
["NAME:ProjectionLimits",  
  
    -2.44362282752991,  
    2.44362282752991,  
  
    -1,  
  
    1,  
    0.625207424163818,  
    10.8721580505371  
]  
]  
)
```

Example Using Vector without Projection Parameters:

```
oEditor.AddViewOrientation("orientation6", 0,  
["NAME:OrientationVectorComponents",  
    ["NAME:Direction",  
        -0.801783978939056,  
        -0.267262011766434,
```

```

        -0.534521996974945
    ],
    ["NAME:Up",
        -0.507091999053955,
        -0.169030994176865,
        0.845155000686646
    ]
]
)

```

VB Syntax	AddViewOrientation <orientationName>, <isGlobalOrientation>, <OrientationVectorComponents <u>or</u> OrientationAnglesArray>, <ProjectionParams>
VB Example	Example Using Angles and Projection Parameters: <pre> oEditor.AddViewOrientation "orientation3", 0, Array("NAME:OrientationAngles", "Psi:=" , -161.565002441406, "Theta:=" , 32.3115005493164, "Phi:=" , 0), Array("NAME:ProjectionParams", Array("NAME:Scaling", 0.591607987880707, </pre>

```
        0.591607987880707,  
  
        0.591607928276062  
    ),  
  
    Array("NAME:Translation",  
  
        0,  
        2.38418579101563E-07,  
        -5.7486834526062  
    ),  
    "ProjectionType:="      , "Orthographic",  
  
    Array("NAME:ProjectionLimits",  
  
        -2.44362282752991,  
        2.44362282752991,  
  
        -1,  
  
        1,
```

```
        0.625207424163818,  
        10.8721580505371  
    )  
)  
  
Example Using Vector without Projection Parameters:  
oEditor.AddViewOrientation "orientation6", 0,  
Array("NAME:OrientationVectorComponents",  
    Array("NAME:Direction",  
        -0.801783978939056,  
        -0.267262011766434,  
        -0.534521996974945  
    ),  
    Array("NAME:Up",  
        -0.507091999053955,  
        -0.169030994176865,  
        0.845155000686646  
    )  
)
```

BreakUDMConnection

Breaks a current UDM connection to SpaceClaim.

UI Access	Break Connection.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	BreakUDMConnection (<SelectionsArray>)		
Python Example	<pre>oEditor.BreakUDMConnection(["NAME:Selections", "Selections:=" , "SpaceClaim1"])</pre>		

VB Syntax	BreakUDMConnection <SelectionsArray>		
VB Example	<pre>oEditor.BreakUDMConnection Array("NAME:Selections", "Selections:=", "")</pre>		

ChangeProperty

Changes the properties of an object in the history tree.

UI Access	Right-click an object in the History Tree and select Properties .

Parameters	Name	Type	Description
	<propertyArgs>	Array	Structured array. The properties vary depending on the object. Due to the number of potential configurations, it is recommended that you generate this script using the UI's Automation tab.
Return Value	None.		

Python Syntax	ChangeProperty(<propertyArgs>)
Python Example	Example Changing the Position of a Box: <pre> oEditor.ChangeProperty(["NAME:AllTabs", ["NAME:Geometry3DCmdTab", ["NAME:PropServers" , "Box1:CreateBox:1"], ["NAME:ChangedProps", ["NAME:Position", "X:=" , "0.35in", "Y:=" , "0.55in", "Z:=" , "0in"]]]]) </pre>

Example Changing a Box's Material and Wireframe Display:

```
oEditor.ChangeProperty(  
[  
  "NAME:AllTabs",  
  [  
    "NAME:Geometry3DAttributeTab",  
    [  
      "NAME:PropServers" ,  
      "Box1"  
    ],  
    [  
      "NAME:ChangedProps",  
      [  
        "NAME:Material",  
        "Value:="      , "\"vacuum\""  
      ],  
      [  
        "NAME:Display Wireframe",  
        "Value:="      , True  
      ]  
    ]  
  ]  
])
```

Example Changing the Model Depth:

```
oEditor.ChangeProperty(  
[  
  "NAME:AllTabs",  
  [  
    "NAME:Maxwell2D",  
    [  
      ]  
    ]  
  ]  
])
```

```

        "NAME:PropServers",
        "Design Settings"
    ],
    [
        "NAME:ChangedProps",
        [
            "NAME:Model settings/Depth",
            "Value:="          , "2.4mm"
        ]
    ]
]
])

```

Example Changing 2D Transient Use of Band Mapping Angle:

```

oEditor.ChangeProperty(
[
    "NAME:AllTabs",
    [
        "NAME:MeshSetupTab",
        [
            "NAME:PropServers",
            "MeshSetup:CylindricalGap1"
        ],
        [
            "NAME:ChangedProps",
            [
                "NAME:Use Band Mapping Angle",
                "Value:="          , True
            ]
        ]
    ]
]
)

oEditor.ChangeProperty(

```

```
[
  "NAME:AllTabs",
  [
    "NAME:MeshSetupTab",
    [
      "NAME:PropServers",
      "MeshSetup:CylindricalGap1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Band Mapping Angle",
        "Value:="          , "2deg"
      ]
    ]
  ]
])
```

Example Edit VirtualPort Excitation by changing component properties:

```
oEditor.ChangeProperty(
[
  "NAME:AllTabs",
  [
    "NAME:Component Data",
    [
      "NAME:PropServers",
      "BoundarySetup:LC1_1_Port1"
    ],
    [
      "NAME:ChangedProps",
      [
        "NAME:Voltage",
        "Value:="          , "10mV"
      ]
    ]
  ]
])
```

```

    ]
  ]
])
...

```

VB Syntax	ChangeProperty <propertyArgs>
VB Example	Example Changing the Position of a Box: <pre> oEditor.ChangeProperty Array("NAME:AllTabs", Array("NAME:Geometry3DCmdTab", Array("NAME:PropServers" , "Box1:CreateBox:1"), Array("NAME:ChangedProps", Array("NAME:Position", "X:=" , "0.35in", "Y:=" , "0.55in", "Z:=" , "0in")))) </pre> Example Changing a Box's Material and Wireframe Display: <pre> oEditor.ChangeProperty(Array("NAME:AllTabs", Array("NAME:Geometry3DAttributeTab", Array("NAME:PropServers", "Box1"))) </pre>

```
        Array("NAME:ChangedProps",  
            Array("NAME:Material",  
                "Value:=", "\"vacuum\"")  
            ),  
        Array("NAME:Display Wireframe",  
            "Value:=", True  
        )  
    )  
)  
)
```

Example Changing the Model Depth:

```
oEditor.ChangeProperty  
Array("NAME:AllTabs",  
    Array("NAME:Maxwell12D",  
        Array("NAME:PropServers"  
            "Design Settings"  
        ),  
        Array("NAME:ChangedProps",  
            Array("NAME:Model settings/Depth",  
                "Value:=", "2.4mm"  
            )  
        )  
    )  
)
```

Example Changing 2D Transient Use of Band Mapping Angle:

```
oEditor.ChangeProperty  
Array("NAME:AllTabs",  
    Array("NAME:MeshSetupTab",  
        Array("NAME:PropServers"  
            "MeshSetup:CylindricalGap1"
```

```
        ),
        Array("NAME:ChangedProps",
            Array("NAME:Use Band Mapping Angle",
                "Value:=", True
            )
        )
    )
)

oEditor.ChangeProperty
Array("NAME:AllTabs",
    Array("NAME:MeshSetupTab",
        Array("NAME:PropServers"
            "MeshSetup:CylindricalGap1"
        ),
        Array("NAME:ChangedProps",
            Array("NAME:Band Mapping Angle",
                "Value:=", "2deg"
            )
        )
    )
)
```

Delete

Deletes the specified object(s).

UI Access	Edit > Delete.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	Delete(<SelectionsArray>)
Python Example	<pre>oEditor.Delete(["NAME:Selections", "Selections:=", "Rectangle1,Rectangle2"])</pre>

VB Syntax	Delete <SelectionsArray>
VB Example	<pre>oEditor.Delete Array("NAME:Selections", "Selections:=", "Rectangle1,Rectangle2")</pre>

GetBodyNamesByPosition

Returns the names of objects that contact a specified point.

UI Access	N/A		
Parameters	Name	Type	Description
	<positionParameters>	Array	Structured array containing position coordinates for active coordinate system. Array("NAME:Parameters", "XPosition:=", <string>, "YPosition:=", <string>, "ZPosition:=", <string>)
Return Value	Array containing string object names.		

Python Syntax	GetBodyNamesByPosition (<positionParameters>)
Python Example	<pre>oEditor.GetBodyNamesByPosition(["NAME:Parameters", "XPosition:=", "0mm", "YPosition:=", "15mm", "ZPosition:=", "0mm"])</pre>

VB Syntax	GetBodyNamesByPosition <positionParameters>
VB Example	<pre>oEditor.GetBodyNamesByPosition Array("NAME:Parameters", _ "XPosition:=", "0mm", _ "YPosition:=", "15mm", _ "ZPosition:=", "0mm")</pre>

GetEdgeByPosition

Returns the ID for edge(s) that contact a specified point.

UI Access	N/A		
Parameters	Name	Type	Description
	<positionParameters>	Array	Structured array.

		<pre>Array("NAME:EdgeParameters", "BodyName:=", <string object name>, "Xposition:=", <string>, "YPosition:=", <string>, "ZPosition:=", <string>)</pre> Note: For 2D XY Designs, ZPosition should be set to "0". For 2D RZ Designs, YPosition should be set to "0".
Return Value	Array containing string edge IDs.	

Python Syntax	GetEdgeByPosition (<positionParameters>)
Python Example	<pre>oEditor.GetEdgeByPosition(["NAME:EdgeParameters", "BodyName:=", "Box1", "Xposition:=", "10mm", "YPosition:=", "0mm", "ZPosition:=", "10mm"])</pre>

VB Syntax	GetEdgeByPosition <positionParameters>
VB Example	<pre>oEditor.GetEdgeByPosition Array("NAME:EdgeParameters", "BodyName:=", "Box1", "Xposition:=", "10mm", "YPosition:=", "0mm", "ZPosition:=", "10mm")</pre>

GetEdgeIDsFromFace

Returns the edge IDs for a specified Face ID.

UI Access	N/A		
Parameters	Name	Type	Description
	<faceID>	Integer	ID of the specified face.
Return Value	Array containing string edge IDs.		

Python Syntax	GetEdgeIDsFromFace (<faceID>)
Python Example	<pre>oEditor.GetEdgeIDsFromFace (20)</pre>

VB Syntax	GetEdgeIDsFromFace <faceID>
VB Example	<pre>oEditor.GetEdgeIDsFromFace 20</pre>

GetEdgeIDsFromObject

Returns the edge IDs for a specified object.

UI Access	N/A		
Parameters	Name	Type	Description
	<object>	String	Object name.
Return Value	Array containing string edge IDs.		

Python Syntax	GetEdgeIDsFromObject (<object>)		
Python Example	<code>oEditor.GetEdgeIDsFromObject ("Box1")</code>		

VB Syntax	GetEdgeIDsFromObject <object>		
VB Example	<code>oEditor.GetEdgeIDsFromObject "Box1"</code>		

GetFaceArea

Returns the area of a specified face.

UI Access	N/A		
Parameters	Name	Type	Description
	<faceID>	Integer	ID of specified face.
Return Value	Long integer of face area.		

Python Syntax	GetFaceArea (<faceID>)
Python Example	oEditor.GetFaceArea(19)

VB Syntax	GetFaceArea <faceID>
VB Example	oEditor.GetFaceArea 19

GetFaceCenter

Returns the center position of a specified face.

UI Access	N/A		
Parameters	Name	Type	Description
	<planarFaceID>	Long	Face ID
Return Value	Array containing string X, Y, Z coordinates.		

Python Syntax	GetFaceCenter (<planarFaceID>)
Python Example	oEditor.GetFaceCenter(12)

VB Syntax	GetFaceCenter <planarFaceID>
VB Example	oEditor.GetFaceCenter 12

GetFaceByPosition

Returns the face ID located at a specified position.

Note:

The coordinates must point to exactly one face, not a vertex or edge where two or more faces join.

UI Access	N/A		
Parameters	Name	Type	Description
	<FaceParameters>	Array	Structured Array. Array ("NAME:FaceParameters", "BodyName:=", <string>, "XPosition:=", <string>, "YPosition:=", <string>, "ZPosition:=", <string>) For 2D XY Designs, ZPosition should be set to "0". For 2D RZ Designs, YPosition should be set to "0".
Return Value	Integer Face ID.		

Python Syntax	GetFaceByPosition(<FaceParameters>)
Python Example	oEditor.GetFaceByPosition(

	<pre>["NAME:FaceParameters", "BodyName:=", "Box1", "XPosition:=", "0.2mm", "YPosition:=", "-0.2mm", "ZPosition:=", "0.4mm"]</pre>

VB Syntax	GetFaceByPosition <FaceParameters>
VB Example	<pre>dim FaceID FaceID = oEditor.GetFaceByPosition Array("NAME:FaceParameters", "BodyName:=", "Box1", "XPosition:=", "0.2mm", "YPosition:=", "-0.2mm", "ZPosition:=", "0.4mm")</pre>

GetFaceIDs

Returns the face IDs associated with a specified object.

UI Access	N/A		
Parameters	Name	Type	Description
	<objectName>	String	Object name.
Return Value	Array containing string face IDs.		

Python Syntax	GetFaceIDs (<objectName>)
Python Example	<code>oEditor.GetFaceIDs ('Box1')</code>

VB Syntax	GetFaceIDs <objectName>
VB Example	<code>oEditor.GetFaceIDs "Box1"</code>

GetGeometryModelerMode

Returns the modeler mode (3D or XY) for the current design. This lets you know whether the current model is 2D or 3D.

UI Access	N/A
Parameters	None.
Return Value	String containing the modeling mode ("3D" or "XY").

Python Syntax	GetGeometryModelerMode()
Python Example	<code>oEditor.GetGeometryModelerMode()</code>

VB Syntax	GetGeometryModelerMode
VB Example	<code>oEditor.GetGeometryModelerMode</code>

GetModelBoundingBox

Returns the bounding box of the current model.

UI Access	N/A
Parameters	None.
Return Value	Array containing string Xmin, Ymin, Zmin, Xmax, Ymax, and Zmax values of the bounding box.

Python Syntax	GetModelBoundingBox()
Python Example	<code>oEditor.GetModelBoundingBox()</code>

VB Syntax	GeModelBoundingBox
VB Example	<code>oEditor.GetModelBoundingBox</code>

GetObjectIDByName

Given an object's name, returns its ID. IDs are used with [CreateEntityList](#).

UI Access	N/A		
Parameters	Name	Type	Description
	<objectName>	String	Object name.
Return Value	Integer object ID.		

Python Syntax	<code>GetObjectIDByName(<objectName>)</code>
Python Example	<code>oEditor.GetObjectIDByName('Box1')</code>

VB Syntax	<code>GetObjectIDByName <objectName></code>
VB Example	<code>oEditor.GetObjectIDByName "Box1"</code>

GetObjectName

Returns an object's name from its specified base index (creation order).

UI Access	N/A		
Parameters	Name	Type	Description
	<index>	Integer	Base index (where '0' is the first item created)
Return Value	String containing object name.		

Python Syntax	<code>GetObjectName(<index>)</code>
Python Example	<code>oEditor.GetObjectName(3)</code>

VB Syntax	<code>GetObjectName <index></code>
VB Example	<code>oEditor.GetObjectName 3</code>

GetObjectByNameByFaceID

Returns an object name given a face ID.

UI Access	N/A		
Parameters	Name	Type	Description
	<FaceID>	Integer	The face ID.
Return Value	String containing object name.		

Python Syntax	GetObjectByNameByFaceID (<FaceID>)
Python Example	<code>oEditor.GetObjectByNameByFaceID(88)</code>

VB Syntax	GetObjectByNameByFaceID <FaceID>
VB Example	<code>oEditor.GetObjectByNameByFaceID 10</code>

GetObjectsByMaterial

Returns a list of objects of a specified material.

UI Access	N/A		
Parameters	Name	Type	Description
	<material>	String	Material name.
Return Value	Array containing string object names.		

Python Syntax	GetObjectsByMaterial (<material>)
Python Example	<code>oEditor.GetObjectsByMaterial('copper')</code>

VB Syntax	GetObjectsByMaterial <material>
VB Example	<code>oEditor.GetObjectsByMaterial "copper"</code>

GetObjectsInGroup

Returns a list of objects in a specified group.

UI Access	N/A		
Parameters	Name	Type	Description
	<group>	String	Group name. One of: "<materialName>" "<assignmentName>" "Non Model" "Solids" "Unclassified" "Sheets" "Lines"
Return Value	Array containing string object names.		

Python Syntax	GetObjectsInGroup (<group>)
Python Example	<code>oEditor.GetObjectsInGroup ('Sheets')</code>

VB Syntax	GetObjectsInGroup <group>
VB Example	<code>oEditor.GetObjectsInGroup "Sheets"</code>

GetMatchedObjectName

Returns all object names containing the input text string.

UI Access	N/A		
Parameters	Name	Type	Description
	<wildcardText>	String	Text string present in object name(s).
Return Value	Array containing string object names.		

Python Syntax	GetMatchedObjectName (<wildcardText>)
Python Example	<code>oEditor.GetMatchedObjectName ('Box*')</code>
	<code>oEditor.GetMatchedObjectName ('?ox?')</code>

VB Syntax	GetMatchedObjectName <wildcardText>
VB Example	<code>oEditor.GetMatchedObjectName "Box*"</code>

GetModelUnits

Returns the model's unit of measure.

UI Access	N/A
Parameters	None.
Return Value	String containing unit of measure.

Python Syntax	GetModelUnits()
Python Example	<code>oEditor.GetModelUnits()</code>

VB Syntax	GetModelUnits
VB Example	<code>oEditor.GetModelUnits</code>

GetNumObjects

Returns the number of objects in a design.

UI Access	N/A
Parameters	None.
Return Value	Integer number of objects.

Python Syntax	GetNumObjects()
Python Example	<code>oEditor.GetNumObjects()</code>

VB Syntax	GetNumObjects
VB Example	<code>oEditor.GetNumObjects()</code>

GetSelections [Model Editor]

Returns an array of currently selected objects.

UI Access	N/A
Parameters	None.
Return Value	Array containing object IDs

Python Syntax	GetSelections()
Python Example	<code>oEditor.GetSelections()</code>

VB Syntax	GetSelections
VB Example	<code>oEditor.GetSelections</code>

GetUserPosition

Returns a user's current coordinates in the 3D Modeler window.

UI Access	N/A		
Parameters	Name	Type	Description
	<prompt>	String	"Enter a point." Then click a point in the 3D Modeler window.
Return Value	Array containing X, Y, Z coordinates.		

Python Syntax	GetUserPosition(<prompt>)		
Python Example	oEditor.GetUserPosition("Enter a point.")		

VB Syntax	GetUserPosition <prompt>		
VB Example	oEditor.GetUserPosition "Enter a point."		

GetVertexIDsFromEdge

Returns vertex IDs associated with a specified edge.

UI Access	N/A		
Parameters	Name	Type	Description
	<edgeID>	Integer	Edge ID.
Return Value	Array containing string vertex IDs.		

Python Syntax	<code>GetVertexIDsFromEdge(<edgeID>)</code>
Python Example	<code>oEditor.GetVertexIDsFromEdge(10)</code>

VB Syntax	<code>GetVertexIDsFromEdge <edgeID></code>
VB Example	<code>oEditor.GetVertexIDsFromEdge 10</code>

GetVertexIDsFromFace

Returns vertex IDs associated with a specified face.

UI Access	N/A		
Parameters	Name	Type	Description
	<faceID>	Integer	Face ID.
Return Value	Array containing string vertex IDs.		

Python Syntax	<code>GetVertexIDsFromFace(<faceID>)</code>
Python Example	<code>oEditor.GetVertexIDsFromFace(10)</code>

VB Syntax	<code>GetVertexIDsFromFace <faceID></code>
VB Example	<code>oEditor.GetVertexIDsFromFace 10</code>

GetVertexIDsFromObject

Returns vertex IDs associated with a specified object.

UI Access	N/A		
Parameters	Name	Type	Description
	< <i>objectName</i> >	String	Object name.
Return Value	Array containing string vertex IDs.		

Python Syntax	GetVertexIDsFromObject(< <i>objectName</i> >)		
Python Example	<code>oEditor.GetVertexIDsFromObject('Box1')</code>		

VB Syntax	GetVertexIDsFromObject < <i>objectName</i> >		
VB Example	<code>oEditor.GetVertexIDsFromObject "Box1"</code>		

GetVertexPosition

Returns an array of coordinates for a specified vertex.

UI Access	N/A		
Parameters	Name	Type	Description
	< <i>vertexID</i> >	Integer	Vertex ID.
Return Value	Array containing X, Y, Z coordinates.		

Python Syntax	GetVertexPosition(<vertexID>)
Python Example	oEditor.GetVertexPosition(1)

VB Syntax	GetVertexPosition <vertexID>
VB Example	oEditor.GetVertexPosition 1

OpenExternalEditor

Launches a SpaceClaim session.

UI Access	Modeler > SpaceClaim Link > Connect to Active Session.		
Parameters	Name	Type	Description
	<SelectionsArray>	Array	Structured array. See: SelectionsArray .
Return Value	None.		

Python Syntax	OpenExternalEditor (<SelectionsArray>)
Python Example	<pre>oEditor.OpenExternalEditor (["NAME:Selections", "Selections:=" , "SpaceClaim1"])</pre>

VB Syntax	OpenExternalEditor <SelectionsArray>
VB Example	<pre>oEditor.OpenExternalEditor Array("NAME:Selections", "Selections:=", "")</pre>

PageSetup

Specifies page settings for printing.

UI Access	File > Page Setup.		
Parameters	Name	Type	Description
	<Parameters>	Array	Structured array. ["NAME:PageSetupData", "margins:=", <SetupArray>]
	<SetupArray>	Array	Structured Array ["left:=", <string>, "right:=", <string>, "top:=", <string>, "bottom:=", <string>)]

Return Value	None.

Python Syntax	PageSetup(<Parameters>)
Python Example	<pre>oEditor.PageSetup(["NAME:PageSetupData", "margins:=", ["left:=", "10mm", "right:=", "10mm", "top:=", "10mm", "bottom:=", "10mm"]])</pre>

VB Syntax	PageSetup <Parameters>
VB Example	<pre>oEditor.PageSetup Array("NAME:PageSetupData", "margins:=", Array("left:=", "10mm", "right:=", "10mm", "top:=", "10mm", "bottom:=", "10mm"))</pre>

RenamePart

Renames an object.

UI Access	Enter new name in Name field.		
Parameters	Name	Type	Description
	<renameParametersArray>	Array	Structured array. Array("NAME:Rename Data", "Old Name:=", <string>, "New Name:=", <string>)
Return Value	None.		

Python Syntax	oEditor.RenamePart(<renameParametersArray>)		
Python Example	<pre>oEditor.RenamePart(['NAME:Rename Data', 'Old Name:=', 'partname', 'New Name:=', 'newpartname',])</pre>		

VB Syntax	<code>oEditor.RenamePart <RenameParametersArray></code>
VB Example	<pre>oEditor.RenamePart Array("NAME:Rename Data", "Old Name:=", "partname", "New Name:=", "newpartname",)</pre>

This page intentionally
left blank.

11 - Output Variable Script Commands

Output variable commands should be executed by the "OutputVariable" module.

First, obtain the output variable module from oDesign and use it for output variable commands:

```
Set oModule = oDesign.GetModule("OutputVariable")  
  
oModule.<CommandName><args>
```

The commands are:

[CreateOutputVariable](#)

[DeleteOutputVariable](#)

[DoesOutputVariableExist](#)

[EditOutputVariable](#)

[GetOutputVariableValue](#)

[GetOutputVariableValue](#)

[SimValueContext](#)

CreateOutputVariable (Maxwell)

Use: Adds a new output variable to the output variable list. Output variables are associated with a name and an expression. The name of an output variable is not permitted to collide with design variables, Sim values, or other output variable names. It cannot have spaces or any arithmetic or other operators. The definitions cannot be cyclic. For example, $A = 2*B$, $B=3*A$ is not allowed.

Command: Maxwell3D>Results>Output Variables, Maxwell2D>Results>Output Variables, or RMXprt>Results>Output Variables.

Syntax: CreateOutputVariable <OutputVarName>, <Expression>, <SolutionName>, <ReportTypeName>, <simValueContext>

Return Value: None

Parameters: <OutputVarName>

Type: <string>

Name of the output variable.

<Expression>

Type: <value>

Value to assign to the variable.

<SolutionName>

Type: <string>

Name of the solution as listed in the **output variable UI**.

For example: "Setup1 : Last Adaptive"

<ReportTypeName>

Type: <string>

The name of the report type as seen in the output variable UI.

<simValueContext>

Type: <variant>

Context for which the output variable expression is being evaluated

Example: Set oModule = oDesign.GetModule("OutputVariable")
oModule.CreateOutputVariable "magforce", "mag(Force1.Force_x)", _
"Setup1 : LastAdaptive", "Magnetostatic", Array()

DeleteOutputVariable

Deletes an existing output variable. The variable can only be deleted if it is not in use by any traces.

UI Access	Maxwell > Results > Output Variables. In the Output Variables window, click Delete.		
Parameters	Name	Type	Description
	<OutputVarName>	String	Name of the output variable.
Return Value	None.		

Python Syntax	DeleteOutputVariable (<OutputVarName>)		
Python Example	<pre>oModule = oDesign.GetModule("OutputVariable") oModule.DeleteOutputVariable ("testNew")</pre>		

VB Syntax	DeleteOutputVariable <OutputVarName>		
VB Example	<pre>Set oModule = oDesign.GetModule("OutputVariable") oModule.DeleteOutputVariable "testNew"</pre>		

DoesOutputVariableExist

Verifies whether or not a named output variable exists.

UI Access	N/A		
Parameters	Name	Type	Description
	<outputVariableName>	String	The output variable name.

Return Value	Boolean True if the variable exists; False if it does not.

Python Syntax	DoesOutputVariableExist(<outputVariableName>)
Python Example	<pre>oProject = oDesktop.GetActiveProject() oDesign = oProject.GetActiveDesign() oModule = oDesign.GetModule("OutputVariable") oModule.DoesOutputVariableExist("MyTestVar")</pre>

VB Syntax	DoesOutputVariableExist(<outputVariableName>)
VB Example	<pre>Set oModule = oDesign.GetModule "OutputVariable" oModule.DoesOutputVariableExist "MyTestVar"</pre>

EditOutputVariable

Changes the name or expression of an existing output variable.

UI Access	N/A		
Parameters	Name	Type	Description
	<OrigVarName>	String	Name of the original output variable.
	<NewExpression>	String	New value to assign to the variable.
	<NewVarName>	String	New name of the variable if any, else pass empty string.

	<i><SolutionName></i>	String	Name of the solution as seen in the output variable UI. For example, "Setup1 : Last Adaptive".
	<i><ReportType></i>	String	The name of the report type as seen in the output variable UI.
	<i><ContextArray></i>	Array	Structured array containing context for which the output variable expression is being evaluated. Array("Context:=", <string>)
Return Value	None		

Python Syntax	EditOutputVariable (<OrigVarName>, <NewExpression>, <NewVarName>, <SolutionName>, <ReportType>, <ContextArray>)
Python Example	<pre>oModule = oDesign.GetModule("OutputVariable") oModule.EditOutputVariable ("test", "normalize(R1_0.V)", "testNew", "TR", "Standard", [])</pre>

VB Syntax	EditOutputVariable <OrigVarName>, <NewExpression>, <NewVarName>, <SolutionName>, <ReportType>, <ContextArray>
VB Example	<pre>Set oModule = oDesign.GetModule("OutputVariable") oModule.EditOutputVariable "test", "dB(S(WavePort1,WavePort1)) ", "testNew", "Setup1 : LastAdaptive", "Modal Solution Data", Array("Domain:=", "Sweep")</pre>

GetOutputVariableValue

Use: Gets the double value of an output variable. Only those expressions that return a double value are supported. The expression is evaluated only for a single point.

Syntax: GetOutputVariableValue(<OutputVarName>, <IntrinsicVariation>, <SolutionName>, <ReportTypeName>, <SimValueContext>)

Return Value: Double value of the output variable .

Parameters: <OutputVarName>

Type: <string>

Name of the output variable.

<IntrinsicVariation>

Type: <string>

A set of intrinsic variable value pairs to use when evaluating the output expression. If no variables are present, a null string must be used. The Eddy Current solution type requires at least the specification of frequency as shown here while the transient solver will require Time as a minimum entry.

For example: "" (Null String)

"freq='60'" (Eddy current example)

<SolutionName>

Type: <string>

Name of the solution as listed in the output variables dialog box as shown in SimValueContext.

For example: "Setup1 : Last Adaptive" or "Setup1 : Transient"

<ReportTypeName>

Type: <string>

The name of the report type as seen in the output variable dialog box.

Note:

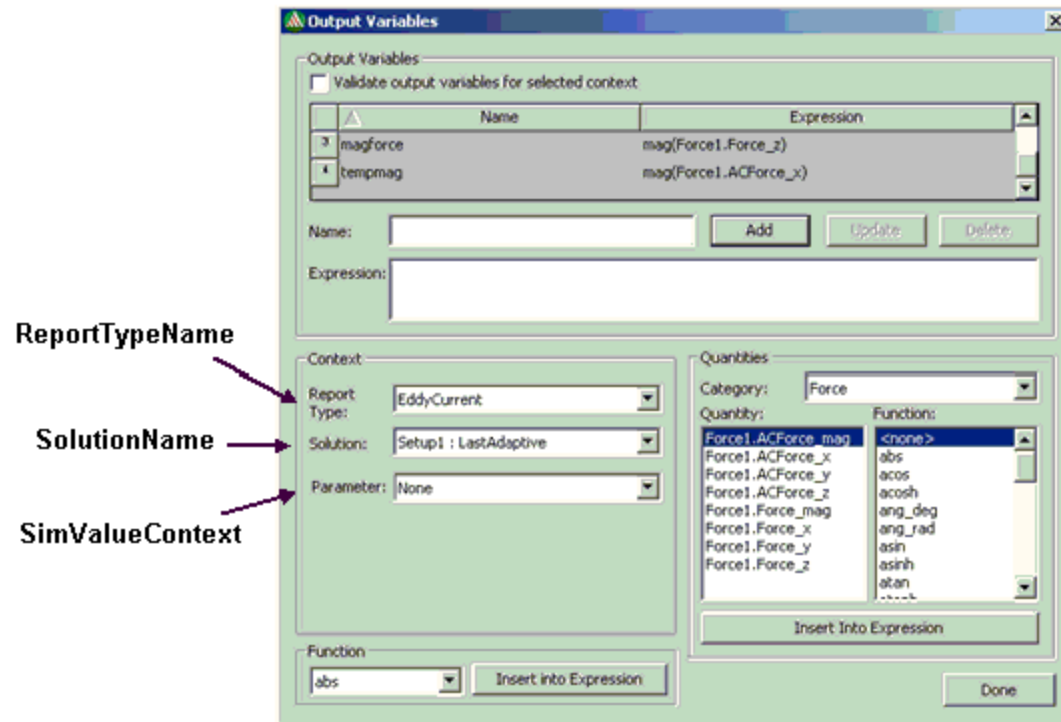
If the Output Variable is based upon a field quantity, then the ReportTypeName must be set to **"Fields"**.

<SimValueContext>

Type: <Array>

Context for which the output variable expression is being evaluated. This section is related to the Parameters or Geometry fields in the UI that specify a context for extracting data quantities. This can be an empty string if there is no context.

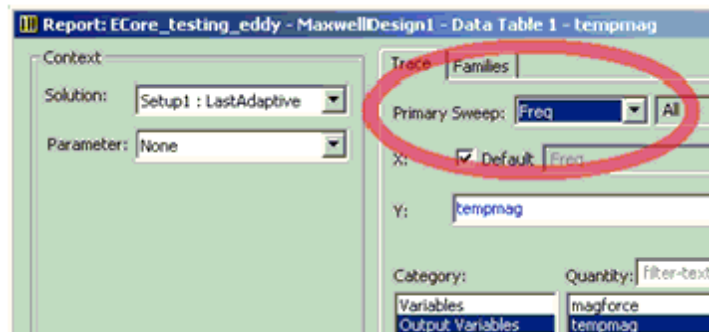
The **Context** section of the Output Variables dialog box contains the information that is required when getting the value of the Output Variable. Note these values when using the dialog box to create an Output Variable.

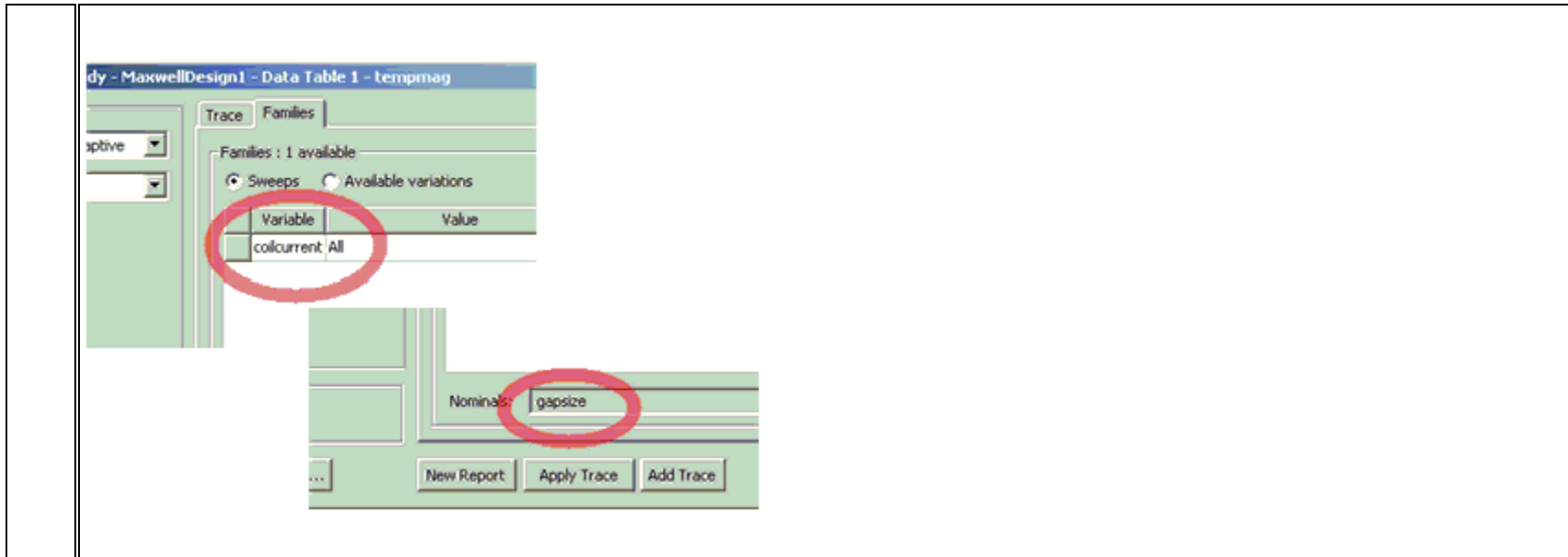


Hint Variation variables can be identified by doing a **Data Table** report and viewing the variable specifications in the table header. For example:

	Freq [Hz]	tempmag coilcurrent='1000' gapsize='-0.001' Setup1 : LastAdaptive
1	60.000000	1.538441

Also, as shown below, the **Trace** and **Families** tabs on the Report dialog box can be used as illustrated to identify variation variables.





Example:

```
Dim Val
```

```
Val= oModule.GetOutputVariableValue("tempmag", "freq='60' coilcurrent='1000' gapsize='-0.001'",  
"Setup1 : LastAdaptive", "EddyCurrent", "")
```

```
Val= oModule.GetOutputVariableValue("tempmag", "coilcurrent='1000' gapsize='-0.001'", "Setup1 :  
LastAdaptive", "Magnetostatic", "")
```

```
Val= oModule.GetOutputVariableValue("BFieldMag", "", "Setup1 : LastAdaptive", "Fields", "")
```

```
Val= oModule.GetOutputVariableValue("MagBatPoint1", "freq='60' coilcurrent='1000' gapsize='-  
0.001' Phase='0deg'", "Setup1 : LastAdaptive", "Fields", Array("Context:=", "Point1"))
```

12 - Reporter Editor Script Commands

Reporter commands should be executed by the oDesign object.

For example:

```
Set oDesign = Project.SetActiveDesign("MaxwellDesign1")
```

```
Set oModule = oDesign.GetModule("ReportSetup")
```

All Report and Trace properties can be edited using the **ChangeProperty** commands. This includes Title properties, General properties, and Background properties such as border color, fonts, X and Y axis scaling, and number display.

Note:

When you execute **Tools > Record Script**, operations performed in the Reporter are automatically recorded.

The list of commands is as follows:

[AddCartesianLimitLine](#)

[AddCartesianXMarker](#)

[AddCartesianYMarker](#)

[AddCartesianYMarkerToStack](#)

[AddDeltaMarker](#)

[AddMarker](#)

[AddNote](#)

[AddTraces](#)

AddVerifEyeAnalysis

[ClearAllMarkers](#)

[ClearAllTraceCharacteristics](#)

[CopyReportDefinitions](#)

[CopyReportData](#)

[CopyTraceDefinitions](#)

[CopyTracesData](#)

[CreateReport](#)

[CreateReportFromFile](#)

[CreateReportFromTemplate](#)

[CreateReportOfAllQuantities](#)

[DeleteMarker](#)

[DeleteAllReports](#)

[DeleteReports](#)

[DeleteTraces](#)

EditQuickEyeAnalysis

EditVerifEyeAnalysis

[FFTONReport](#)

[ExportFieldsToFile](#)

[ExportImageToFile](#)

[ExportModelImageToFile](#)

[ExportModelMeshToFile](#)

[ExportToFile \[Reporter\]](#)

[ExportMarkerTable](#)

[GetAllCategories](#)

[GetAllQuantities](#)

[GetAllReportNames](#)

[GetAvailableDisplayTypes](#)

[GetAvailableReportTypes](#)

[GetAvailableSolutions](#)

[GetChildNames](#)

[GetChildObject](#)

[GetChildTypes](#)

[GetDisplayType](#)

[GetPropertyValue](#)

[GetReportTraceNames](#)

[GetSolutionContexts](#)

[GetSolutionDataPerVariation](#)

[GroupPlotCurvesByGroupingStrategy](#)

[ImportIntoReport](#)

[MovePlotCurvesToGroup](#)

[MovePlotCurvesToNewGroup](#)

[PasteReports](#)

- [PasteTraces](#)
- [RenameReport](#)
- [RenameTrace](#)
- [ResetPlotSettings](#)
- [SavePlotSettingsAsDefault](#)
- [SetPropValue](#)
- [UpdateAllFieldsPlots](#)
- [UpdateAllReports](#)
- [UpdateQuantityFieldsPlots](#)
- [UpdateReports](#)
- [UpdateTraces](#)
- [UpdateTracesContextandSweeps](#)
- [UnGroupPlotCurvesInGroup](#)

AddCartesianLimitLine

Adds a limit line to a report on the X axis.

UI Access	Report2D > Add Limit Line> Specify Points...		
Parameters	Name	Type	Description
	<ReportName>	String	Name of the report.
	<Def>	Array	Structured array: Array("NAME:CartesianLimitLine",

			<pre> Array("NAME:XValues", <integer X values>), "XUnits:=", <string unit of measure for X>, Array("NAME:YValues", <integer Y values>), "YUnits:=", "<string unit of measure for Y>", "YAxis:=", <string name of associated Y axis>) </pre>
Return Value	None.		

Python Syntax	AddCartesianLimitLine (<ReportName>, <Def>)
Python Example	<pre> oModule.AddCartesianLimitLine("Project Outputs", ["NAME:CartesianLimitLine", ["NAME:XValues",0, 2, 5, 7, 10, 15], "XUnits:=", "s", ["NAME:YValues",0.05, 0.3, 0.65, 0.825, 0.95, 1], "YUnits:=", "mV", "YAxis:=", "Y1"]) </pre>

VB Syntax	AddCartesianLimitLine <ReportName>, <Def>
VB Example	<pre>oModule.AddCartesianLimitLine "Project Outputs",</pre>

	<pre>Array("NAME:CartesianLimitLine", Array("NAME:XValues",0, 2, 5, 7, 10, 15), "XUnits:=", "s", Array("NAME:YValues",0.05, 0.3, 0.65, 0.825, 0.95, 1), "YUnits:=", "mV", "YAxis:=", "Y1")</pre>

AddCartesianXMarker

Adds a marker to a report on the X axis.

UI Access	Report2D > Marker > Add X Marker.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report.
	<MarkerName>	String	Marker name, including any trailing number.
	<XValue>	Double	X coordinate.
Return Value	None		

Python Syntax	AddCartesianXMarker (<ReportName>, <MarkerName>, <XValue>)
Python Example	oModule.AddCartesianXMarker ("XY Plot 1", "MX1", 0)

VB Syntax	AddCartesianXMarker <ReportName>, <MarkerName>, <XValue>
VB Example	<code>oModule.AddCartesianXMarker "XY Plot1", "MX1", 0</code>

AddCartesianYMarker

Adds a marker to a report on the Y axis.

UI Access	Report2D > Marker > Add Y Marker.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report.
	<MarkerName>	String	Marker name, including any trailing number.
	<AxisName>	String	Name of axis.
	<YValue>	Double	Y coordinate.
	<CurveName>	String	Name of curve.
Return Value	None		

Python Syntax	AddCartesianYMarker (<ReportName>, <MarkerName>, <AxisName>, <YValue>, <CurveName>)
Python Example	<code>oModule.AddCartesianYMarker("XY Plot 1", "MY1", "Y1", 0, "dB() : Setup1 : Sweep1")</code>

VB Syntax	AddCartesianYMarker <ReportName>, <MarkerName>, <AxisName>, <YValue>, <CurveName>
VB Example	<code>oModule.AddCartesianYMarker "XY Plot 1", "MY1", "Y1", 0, _ "dB() : Setup1 : Sweep1"</code>

AddDeltaMarker

Add markers to calculate differences between two trace points on a plot.

UI Access	Report2D > Marker > Add Delta Marker.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report.
	<MarkerName1>	String	Marker name, including any trailing number, for the first marker.
	<CurveName1>	String	Full trace name for the first marker.
	<PrimarySweepValue1>	String	Frequency, including unit, for the first marker's sweep.
	<MarkerName2>	String	Marker name, including any trailing number, for the second marker.
	<CurveName2>	String	Full trace name for the second marker.
	<PrimarySweepValue2>	String	Frequency, including unit, for the second marker's sweep.
Return Value	None		

Python Syntax	AddDeltaMarker(<ReportName>, <MarkerName1>, <CurveName1>, <PrimarySweepValue1>, <MarkerName2>, <CurveName2>, <PrimarySweepValue2>)
Python Example	<pre>oModule.AddDeltaMarker("S Parameter Plot 1", "m1", "dB(S(Port1 Port1)) : Setup1 : Sweep1 : Cartesian", "8.45GHz", "m2", "dB(S(Port1 Port1)) : Setup1 : Sweep1 : Cartesian", "9.5GHz")</pre>

VB Syntax	AddDeltaMarker <ReportName>, <MarkerName1>, <CurveName1>, <PrimarySweepValue1>, <MarkerName2>, <CurveName2>, <PrimarySweepValue2>
VB Example	<pre>oModule.AddDeltaMarker "S Parameter Plot 1", "m1", "dB(S(Port1 Port1)) : Setup1 : Sweep1 : Cartesian", "8.45GHz", "m2", "dB(S(Port1 Port1)) : Setup1 : Sweep1 : Cartesian", "9.5GHz"</pre>

AddMarker

Adds a marker to a trace on a report.

UI Access	Report2D > Marker > Add Marker.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report.
	<MarkerName>	String	Marker name, including any trailing number.
	<CurveName>	String	Full trace name.
	<PrimarySweepValue>	String	Primary sweep value, including unit.
Return Value	None		

Python Syntax	AddMarker(<ReportName>, <MarkerName>, <CurveName>, <PrimarySweepValue>)
Python Example	<pre>oModule.AddMarker("XY Plot 1", "m5", "GS1.VAL : TR4 : Cartesian", "3.61599999999997s")</pre>

VB Syntax	AddMarker <ReportName>, <MarkerName>, <CurveName>, <PrimarySweepValue>
VB Example	<pre>Set oModule = oDesign.GetModule("ReportSetup") oModule.AddMarker "XY Plot1", "m1", "mag(S(Port1 Port1)) : Setup1: LastAdaptive : Cartesian", "0.3in"</pre>

AddNote

Adds a note at a specified location to a given report.

UI Access	Right-click on the plot and select Add Note .		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report
	<NoteDataArray>	Array	Structured array. Array("NAME:<StringDataName>", <NoteArray>)
	<NoteArray>	Array	Structured array: Array("NAME:<NoteDataSourceName>", "SourceName:=", <string source name>, "HaveDefaultPos:=", <boolean True for position 0,0; False to specify below>, "DefaultXPos:=", <int X position for note; 0 for default>, "DefaultYPos:=", <int Y position for note; 0 for default>,

			"String:=", <string note text>)
Return Value	None		

Python Syntax	AddNote (<ReportName>, <NoteDataArray>)
Python Example	<pre>oModule.AddNote("XY Plot1", ["NAME:NoteDataSource", "SourceName:=", "Note1", "HaveDefaultPos:=", False, "DefaultXPos:=", 1996, "DefaultYPos:=", 3177, "String:=", "This is a note."])</pre>

VB Syntax	AddNote <ReportName> <NoteDataArray>
VB Example	<pre>oModule.AddNote "XY Plot1", _ Array("NAME:NoteDataSource", _ "SourceName:=", "Note1", _ "HaveDefaultPos:=", false, _ "DefaultXPos:=", 1996, _</pre>

	<pre>"DefaultYPos:=", 3177, _ "String:=", "This is a note.")</pre>

AddTraceCharacteristics

Adds a trace characteristics field to the legend on a report.

UI Access	Report2D > Trace Characteristics > All. This opens the Add Trace Characteristics dialog box.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of report.
	<FunctionName>	String	The function name. See the Functions column of the Add Trace Characteristics dialog box.
	<FunctionArgs>	Array	Array containing string values for any function arguments. Pass empty array if no arguments exist. To see which argument(s) a function takes, see the bottom of the Add Trace Characteristics dialog box. For function with one argument: Array(<value>) For function with multiple arguments: Array(<value>, <value>, ...)
	<RangeArgs>	Array	Required. Array containing either string "Full" for a full sweep range, or "Specified" and strings containing the start and end values for the frequency range. For example: Array("Specified", "19.5GHz", "24.4GHz")
Return Value	None.		

Python Syntax	AddTraceCharacteristics (<ReportName>, <FunctionName>, <FunctionArgs>, <RangeArgs>)
Python Example	<pre>oModule.AddTraceCharacteristics("XY Plot 2", "delaytime", ["0"], ["Full"]) oModule.AddTraceCharacteristics("Differential S-parameters", "prms", ["0", "0"], ["Full"]) oModule.AddTraceCharacteristics("Rept2DRectFreq", "distortion", ["0"], ["Specified", "19.5GHz", "20.4GHz"])</pre>

VB Syntax	AddTraceCharacteristics <ReportName>, <FunctionName>, <FunctionArgs>, <RangeArgs>
VB Example	<pre>oModule.AddTraceCharacteristics "XY Plot 2", "delaytime", Array("0"), Array("Full") oModule.AddTraceCharacteristics "Differential S-parameters", "prms", Array("0", "0"), Array("Full") oModule.AddTraceCharacteristics "Rept2DRectFreq", "distortion", Array("0"), Array("Specified", "19.5GHz", "20.4GHz")</pre>

AddTraces

Creates a new trace and adds it to the specified report.

UI Access	Modify Report > Add Trace.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of Report
	<SolutionName>	String	Name of the solution as listed in the Modify Report dialog box.

	<ContextArray>	Array	Context for which the expression is being evaluated. This can be an empty string if there is no context. Array("Domain:=", <DomainType>) <DomainType> ex. "Sweep" or "Time" Array("Context:=", <GeometryType>) <GeometryType> ex. "Infinite Spheren", "Spheren", "Polylinen"
	<FamiliesArray>	Array	Contains sweep definitions for the report. Array("<VariableName>:= ", <ValueArray>) <ValueArray> Array("All") or Array("Value1", "Value2", ..."Valuen") examples of <VariableName> "Freq", "Theta", "Distance"
	<ReportDataArray>	Array	This array contains the report quantity and X, Y, and (Z) axis definitions. Array("X Component:=", <VariableName>, "Y Component:=", <VariableName> <ReportQuantityArray>) <ReportQuantityArray> ex. Array("dB(S(Port1, Port1))")
Return Value	None		

Python Syntax	Add Traces(<ReportName>, <SolutionName>, <ContextArray>, <FamiliesArray>, <ReportDataArray>)
Python Example	<pre>oModule.AddTraces("XY Plot1", "Setup1 : Sweep1", ["Domain:=", "Time", "HoldTime:=", 1, "RiseTime:=", 0, "StepTime:=", 6.24999373748E-012, "Step:=", False, "WindowWidth:=", 1, "WindowType:=", 0, "KaiserParameter:=", 1, "MaximumTime:=", 6.2437437437444E-009], ["Time:=", ["All"], "OverridingValues:=", ["0s", "6.24999373748188e-012s", ...]], ["X Component:=", "Time", "Y Component:=", ["TDRZ (WavePort1)"]], [])</pre>

VB Syntax	Add Traces <ReportName> <SolutionName> <ContextArray> <FamiliesArray> <ReportDataArray>
VB Example	<pre>oModule.AddTraces "XY Plot1", "Setup1 : Sweep1", _ Array("Domain:=", "Time", "HoldTime:=", 1, "RiseTime:=", 0, _ "StepTime:=", 6.24999373748E-012, "Step:=", false, _ "WindowWidth:=", 1, _ "WindowType:=", 0, "KaiserParameter:=", 1, _ "MaximumTime:=", 6.2437437437444E-009), _ Array("Time:=", Array("All"), "OverridingValues:=", _ Array("0s", "6.24999373748188e-012s", ...)), _ Array("X Component:=", "Time", _</pre>

	<pre>"Y Component:=", Array("TDRZ(WavePort1)"), _ Array()</pre>

ChangeProperty[ReportSetup]

Change the properties for a Report.

UI Access	Double-click on a report to change its properties.		
Parameters	Name	Type	Description
	<PropertyArray>	Array	Varies, depending on the properties associated with the select object command.
Return Value	None.		

Python Syntax	ChangeProperty(<PropertyArray>)
Python Example	<pre>oModule = oDesign.GetModule("ReportSetup") oModule.ChangeProperty(["NAME:AllTabs", ["NAME:General", ["NAME:PropServers",</pre>

```
"XY Plot 1:General"
],
[
  "NAME:ChangedProps",
  [
    "NAME:Use Scientific Notation",
    "Value:=", True
  ]
]
]
])

oModule = oDesign.GetModule("ReportSetup")
oModule.ChangeProperty(
  [
    "NAME:AllTabs",
    [
      "NAME:Legend",
      [
        "NAME:PropServers",
        "S Parameter Plot 1:Legend"
```

```
    ],  
    [  
        "NAME:ChangedProps",  
        [  
            "NAME:Legend Name",  
            "Value:=" , "Ansys"  
        ]  
    ]  
]  
]  
])  
  
oModule.ChangeProperty(  
    [  
        "NAME:AllTabs",  
        [  
            "NAME:General",  
            [  
                "NAME:PropServers",  
                "S Parameter Plot 1:General"  
            ],  
        ],  
    ]  
)
```

	<pre> "NAME:ChangedProps", ["NAME:Auto Scale Fonts", "Value:=" , False]]])</pre>

VB Syntax	ChangeProperty <PropertyArray>
VB Example	<pre>oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Cartesian", Array ("NAME:PropServers", _ "XY Plot 1:CartesianDisplayTypeProperty"), Array("NAME:ChangedProps", Array("NAME:Show X Scrollbar", "Value:=", _ true)))) oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:General", Array("NAME:PropServ- ers", _ "XY Plot 1:General"), Array("NAME:ChangedProps", Array("NAME:Use Scientific Notation", "Value:=", _ true))))</pre>

ClearAllMarkers

Clears all markers from a report.

UI Access	Report2D > Markers > Clear All.		
Parameters	Name	Type	Description
	<ReportName>	String	Name of Report
Return Value	None		

Python Syntax	ClearAllMarkers(<ReportName>)
Python Example	<code>oModule.ClearAllMarkers("XY Plot 1")</code>

VB Syntax	ClearAllMarkers <ReportName>
VB Example	<code>oModule.ClearAllMarkers "XY Plot 1"</code>

ClearAllTraceCharacteristics

Clears all trace characteristics from the legend in a report.

UI Access	Report2D > Trace Characteristics > Clear All.		
Parameters	Name	Type	Description
	<PlotName>	String	Name of the plot

Return Value	None

Python Syntax	<code>ClearAllTraceCharacteristics(<PlotName>)</code>
Python Example	<code>oModule.ClearAllTraceCharacteristics("XY Plot 1")</code>

VB Syntax	<code>ClearAllTraceCharacteristics <PlotName></code>
VB Example	<code>oModule.ClearAllTraceCharacteristics "XY Plot 1"</code>

CopyTracesData

Copies trace data for a paste operation.

UI Access	Select a trace in the Project tree, right-click and select Copy Data .		
Parameters	Name	Type	Description
	<ReportName>	String	Name of Report.
	<TraceArray>	Array	Trace definitions from which to copy corresponding data.
Return Value	None.		

Python Syntax	<code>CopyTracesData(<ReportName>, <TracesArray>)</code>
Python Example	<code>oModule.CopyTracesData ("Transmission", ["mag(S (Port1, Port2)) "])</code>

VB Syntax	CopyTracesData <ReportName>, <TracesArray>
VB Example	<pre>oModule.CopyTracesData _ "C11", Array("mag(S(Port1,Port2))")</pre>

CopyReportsData

Copy all data corresponding to the specified reports.

UI Access	Select a report in the Project tree, right-click and select Copy Data		
Parameters	Name	Type	Description
	<ReportsArray>	Array	Names of reports from which to copy data
Return Value	None		

Python Syntax	CopyReportsData (<ReportsArray>)
Python Example	<pre>oModule.CopyReportsData (["Transmission","Reflection"])</pre>

VB Syntax	CopyReportsData <ReportsArray>
VB Example	<pre>oModule.CopyReportsData _ Array("Transmission","Reflection")</pre>

CopyReportDefinitions

Copy the definition of a report for paste operations.

UI Access	Select a report in the Project tree, right-click and select Copy Definition		
Parameters	Name	Type	Description
	<ReportsArray>	Array	Names of reports from which to copy the definitions
Return Value	None		

Python Syntax	CopyReportDefinitions(<ReportsArray>)
Python Example	<pre>oModule.CopyReportDefinitions(["Transmission", "Reflection"])</pre>

VB Syntax	CopyReportDefinitions <ReportsArray>
VB Example	<pre>oModule.CopyReportDefinitionsv _ Array("Transmission", "Reflection")</pre>

CopyTraceDefinitions

Copy trace definitions for a paste operation.

UI Access	Select a trace in the Project tree, right-click and select Copy Definition		
Parameters	Name	Type	Description
	<ReportName>	String	Name of Report

	<table><tr><td><TracesArray></td><td>Array</td><td>Trace definitions to copy</td></tr></table>	<TracesArray>	Array	Trace definitions to copy
<TracesArray>	Array	Trace definitions to copy		
Return Value	None			

Python Syntax	CopyTraceDefinitions(<ReportName>, <TracesArray>)
Python Example	<pre>oModule.CopyTraceDefinitions ("Transmission", ["mag (S (Port1, Port2)) "])</pre>

VB Syntax	CopyTraceDefinitions <ReportName>, <TracesArray>
VB Example	<pre>oModule.CopyTraceDefinitions "Transmission", _ Array ("mag (S (Port1, Port2)) ")</pre>

CreateReportFromFile

Creates a new report from an .rdat file.

UI Access	Right-click on Results > Create Report From File...		
Parameters	Name	Type	Description
	<FilePathName>	String	Path to .rdat file.
Return Value	None.		

Python Syntax	CreateReportFromFile(<FilePathName>)
Python Example	<code>oModule.CreateReportFromFile("C:/Users/MyDir/Documents/Return_Loss.rdat")</code>

VB Syntax	CreateReportFromFile <FilePathName>
VB Example	<code>oModule.CreateReportFromFile "C:/Users/MyDir/Documents/Return_Loss.rdat"</code>

CreateReport (Maxwell)

Use: Creates a new report with a single trace and adds it to the **Results** branch in the project tree. To add more traces, use the **AddTraces** command. To edit the display properties, use the **ChangeProperty** Script command.

Command: Maxwell2D or Maxwell3D>Results>Create<type> Report

Syntax: CreateReport <ReportName> <ReportType> <DisplayType> <SolutionName> <ContextArray> <FamiliesArray><ReportDataArray>

Return Value: None

Parameters: <ReportName>

Type: <string>

Name of Report.

ReportType

Some possible values are:

"TransientAPhiFormulation"

"Fields"

"Noise Vibration"

"Time Averaged Fields"

"Matrix".

"DC R/L Fields".

"AC R/L Fields".

"C Fields"

"Harmonic Force"

DisplayType

"Rectangular Plot", "Rectangular Stacked Plot", "Data Table", "3D Rectangular Plot", "Rectangular Contour Plot".

<TraceArray>

```
Array("NAME:Traces",  
      <OneTraceArray>, <OneTraceArray>, ...)
```

<OneTraceArray>

```
Array("NAME:<TraceName>,"  
      "SolutionName:=", "string",  
      "Context:=", "string",  
      <DisplayTypeDependentData>)
```

<SolutionName>

*Name of the solution as listed in the **Traces** dialog box.*

For example: "Setup1 : Last Adaptive"

<Context> Array

Context for which the output variable expression is being evaluated.

```
Array("Domain:=", <DomainType>)
```

```
<DomainType>
```

ex. "Sweep" or "Time"

This can be an empty string if there is no context.

Example: "Line1" or ""

Field reports usually require a polyline (e.g. "Line1") unless they are integrations..

```
<DisplayTypeDependentData>
```

This data varies according to the display type. See the examples below.

```
<FamiliesArray>
```

Type: Array of strings

Contains sweep definitions for the report.

```
Array("<VariableName>:= ", <ValueArray>)
```

```
<ValueArray>
```

```
Array("All") or Array("Value1", "Value2", ..."Valuen")
```

examples of <VariableName>

```
"Freq", "Theta", "Distance"
```

```
<ReportDataArray>
```

Type: Array of strings

This array contains the report quantity and X, Y, and (Z) axis definitions.

```
Array("X Component:=", <VariableName>, "Y Component:=", <VariableName> | <ReportQuantityArray>)
```

```
<ReportQuantityArray>
```

```
ex. Array("dB(S(Port1, Port1))")
```

Example: oDesign.CreateReport Array("NAME:Rept2DRectTime", _

```
"ReportType:=", "Matrix",
```

```
"DisplayType:=", "Rectangular Plot",
```

```
Array("NAME:Traces",
```

```
Array("NAME:Trace1",
```

```
"SolutionName:=",
```

```
"Setup1 : Adaptive_2",
```

```
"Context:=", "Original",
```

```
"XComponent:=", "Pass",
```

```
"YComponent:=", "C(Box1, Box1)",
```

```
"YAxis:=", 1)))
```

Python Example: oModule.CreateReport("Loss Plot 1", "Transient", "Rectangular Plot", "Setup1 : Transient",

```
[
```

```
"Domain:=" , "Average and RMS",
```

```
"npoles:=" , "2",  
"alignment:=" , "0deg",  
"orientation:=" , "CounterClockwise",  
"losses:=" , "0W",  
"speed:=" , "0rpm",  
"Resistance(PhA):=" , "0ohm",  
"Resistance(PhB):=" , "0ohm",  
"Resistance(PhC):=" , "0ohm",  
"Inductance(PhA):=" , "0H",  
"Inductance(PhB):=" , "0H",  
"Inductance(PhC):=" , "0H"  
],  
[  
"Im:=" , ["All"],  
"Gamma:=" , ["Nominal"],  
"speed_rpm:=" , ["Nominal"],  
"poles:=" , ["Nominal"]  
],  
[  
"X Component:=" , "Im",  
"Y Component:=" , ["AvgCoreLoss"]
```

```
], [])
```

Python Example for Transient A-phi Formulation: oModule.CreateReport("Loss Plot 1", "TransientAPhiFormulation", "Rectangular Plot", "Setup1 : Transient",

```
[  
"Domain:=" , "Sweep"  
],  
[  
"Time:=" , ["All"],  
"$HcRef:=" , ["Nominal"],  
"$MagnetTemperature:=" , ["Nominal"]  
],  
[  
"X Component:=" , "Time",  
"Y Component:=" , ["TerminalVoltage(Voltage1)", "TerminalVoltage(Voltage2)"]  
])
```

[Beta] Python Example for Transient A-phi Formulation, LayoutForce plot: oModule.CreateReport("LayoutForce Plot 1", "TransientAPhiFormulation", "Rectangular Plot", "Setup1 : Transient",

```
[  
"Domain:=" , "Sweep"
```



```
],  
[  
  "Time:=" , ["All"]  
],  
[  
  "X Component:=" , "Time",  
  "Y Component:=" , ["LayoutForcel.Force_mag"]  
])  
```
```

***Python Example for Average\_Surface\_Loss\_Density:***

```
oModule.CreateReport("Calculator Expressions Plot 1", "Time Averaged Fields", "Rectangular
Plot", "Setup1 : Transient",

 [
 "Context:=" , "Point1",
 "PointCount:=" , 1001,
 "Time0:=" , "0s"
],

 [
 "Time:=" , ["All"]
],

 [

```

```
"X Component:=" , "Time",
"Y Component:=" , ["Average_Surface_Loss_Density"]
])
```

### ***Python Example for Transient and Eddy Current Per-Winding Losses:***

```
oModule.CreateReport("Loss Plot 4", "Transient", "Rectangular Plot", "Setup1 : Transient",
[
"Domain:=" , "Sweep"
],
[
"Time:=" , ["All"]
],
[
"X Component:=" , "Time",
"Y Component:=" , ["PerWindingSolidLoss(WindingV1_Solid)"]
])
oModule.AddTraces("Loss Plot 4", "Setup1 : Transient",
[
"Domain:=" , "Sweep"
],
```

```
[
"Time:=" , ["All"]
],
[
"X Component:=" , "Time",
"Y Component:=" , ["PerWindingStrandedLoss(WindingI3_Litz)"]
])
oModule.AddTraces("Loss Plot 4", "Setup1 : Transient",
[
"Domain:=" , "Sweep"
],
[
"Time:=" , ["All"]
],
[
"X Component:=" , "Time",
"Y Component:=" , ["PerWindingStrandedLossAC(WindingI3_Litz)"]
])
```

***Python Example for Harmonic Force Report Type:***

```
oModule.CreateReport("Harmonic Force Plot 5", "Harmonic Force", "3D Rectangular Bar Plot",
"Setup1 : Transient",
[
 "Freq:=" , ["All"],
 "SpatialOrder:=" , ["All"],
 "fractions:=" , ["Nominal"],
 "halfAxial:=" , ["Nominal"],
 "endRegion:=" , ["Nominal"],
 "RPM:=" , ["1500rpm"],
 "ppairs:=" , ["Nominal"]
],
[
 "X Component:=" , "Freq",
 "Y Component:=" , "SpatialOrder",
 "Z Component:=" , ["re(HarmonicForce_a)"]
])
```

### ***Python Example for Thin Layer (Air Gap) Report Type:***

```
oModule.CreateReport("Magnetic Energy Table 1", "Magnetostatic", "Data Table", "Setup1 : LastAd-
aptive", [],
[
```

```
"$Current:=" , ["All"]

],

[
"X Component:=" , "$Current",
"Y Component:=" , ["MagneticEnergy(ThinLayer1)"]
])
```

**VB Example for Thin Layer (Air Gap) Report Type:**

```
oModule.CreateReport "Magnetic Energy Table 1", "Magnetostatic", "Data Table" "Setup1 : LastAd-
aptive", Array(), Array("$Current:=", Array("All")), Array("X Component:=", "$Current", "Y Com-
ponent:=", Array("MagneticEnergy(ThinLayer1)"))
```

CreateReportFromTemplate

Creates a report from a saved template.

|              |                                                                       |        |                         |
|--------------|-----------------------------------------------------------------------|--------|-------------------------|
| UI Access    | [product] > Results > Report Templates > PersonalLib > [TemplateName] |        |                         |
| Parameters   | Name                                                                  | Type   | Description             |
|              | <TemplatePath>                                                        | String | Path to report template |
| Return Value | None                                                                  |        |                         |

|                |                                                                                                        |
|----------------|--------------------------------------------------------------------------------------------------------|
| Python Syntax  | CreateReportFromTemplate( <TemplatePath>)                                                              |
| Python Example | oModule.CreateReportFromTemplate(<br>"C:/MyHFSSProjects/PersonalLib/ReportTemplates/TestTemplate.rpt") |

|                   |                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | CreateReportFromTemplate <TemplatePath>                                                                               |
| <b>VB Example</b> | <pre>oModule.CreateReportFromTemplate "C:/MyHFSSProjects/PersonalLib/" _<br/>"ReportTemplates/TestTemplate.rpt"</pre> |

## CreateReportOfAllQuantities

Creates a report including all quantities in a category. Cannot create a report with expressions.

|                     |                            |        |                                                                 |
|---------------------|----------------------------|--------|-----------------------------------------------------------------|
| <b>UI Access</b>    | NA                         |        |                                                                 |
| <b>Parameters</b>   | Name                       | Type   | Description                                                     |
|                     | <ReportType>               | String | Report type name as input parameter                             |
|                     | <DisplayType>              | String | Display type name as input parameter                            |
|                     | <SolutionName>             | String | Solution name as input parameter                                |
|                     | <SimValueCtxt>             | String | A context name, or array of string that encoded the context(I). |
|                     | <CategoryName>             | String | Category name as input parameter                                |
|                     | <PointSet>                 | Array  | Array of strings(II)                                            |
|                     | <CommonComponentsOfTraces> | Array  | Array of strings(III)                                           |
|                     | <ExtTraceInfo>             | Array  | Array of strings(IV)                                            |
| <b>Return Value</b> | None.                      |        |                                                                 |

|                      |                                                                                                                              |
|----------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b> | CreateReportOfAllQuantities(<ReportNameArg>, <ReportType>, <DisplayType>,<br><SolutionName>, <SimValueCtxt>, <CategoryName>, |
|----------------------|------------------------------------------------------------------------------------------------------------------------------|

|                       |                                                                                                                                                                                                                                               |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                       | <i>&lt;PointSet&gt;, &lt;CommonComponentsOfTraces&gt;, &lt;ExtTraceInfo&gt;</i> )                                                                                                                                                             |
| <b>Python Example</b> | <pre>oModule.CreateReportOfAllQuantities("Smith Chart all", "Modal Solution Data", "Smith Chart", "Setup1 : LastAdaptive", [], "S Parameter", ["Freq:=", ["All"], "offset:=", ["All"], "a:=", ["Nominal"], "b:=", ["Nominal"]], [], [])</pre> |

|                   |                                                                                                                                                                                                                                                                                                |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | CreateReportOfAllQuantities <i>&lt;ReportNameArg&gt;, &lt;ReportType&gt;, &lt;DisplayType&gt;, &lt;SolutionName&gt;, &lt;SimValueCtxt&gt;, &lt;CategoryName&gt;, &lt;PointSet&gt;, &lt;CommonComponentsOfTraces&gt;, &lt;ExtTraceInfo&gt;</i>                                                  |
| <b>VB Example</b> | <pre>oModule.CreateReportOfAllQuantities "Smith Chart all", _ "Modal Solution Data", "Smith Chart", "Setup1 : LastAdaptive", _ Array(), "S Parameter", _ Array("Freq:=", Array("All"), "offset:=", Array("All"), _ "a:=", Array("Nominal"), "b:=", Array("Nominal")), _ Array(), Array()</pre> |

## DeleteMarker

*Use:*Deletes the specified marker.

|                  |                                                                       |
|------------------|-----------------------------------------------------------------------|
| <b>UI Access</b> | [product] > <b>Fields &gt; Fields &gt; Marker &gt; Delete Marker.</b> |
|------------------|-----------------------------------------------------------------------|

|              |              |        |                     |
|--------------|--------------|--------|---------------------|
| Parameters   | Name         | Type   | Description         |
|              | <MarkerName> | String | Name of the marker. |
| Return Value | None.        |        |                     |

|                |                                         |
|----------------|-----------------------------------------|
| Python Syntax  | DeleteMarker(<MarkerName>)              |
| Python Example | <code>oModule.DeleteMarker("m1")</code> |

|            |                                        |
|------------|----------------------------------------|
| VB Syntax  | DeleteMarker <MarkerName>              |
| VB Example | <code>oModule.DeleteMarker "m1"</code> |

## DeleteAllReports

Deletes all existing reports.

|              |                                                                                                                      |
|--------------|----------------------------------------------------------------------------------------------------------------------|
| UI Access    | Right-click the report to delete in the project tree, and then click <b>Delete All Reports</b> on the shortcut menu. |
| Parameters   | None.                                                                                                                |
| Return Value | None.                                                                                                                |

|                |                                         |
|----------------|-----------------------------------------|
| Python Syntax  | DeleteAllReports()                      |
| Python Example | <code>oModule.DeleteAllReports()</code> |



|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | DeleteAllReports                      |
| <b>VB Example</b> | <code>oModule.DeleteAllReports</code> |

## DeleteReports

Deletes an existing report or reports.

|                     |                                                                                                         |       |                                     |
|---------------------|---------------------------------------------------------------------------------------------------------|-------|-------------------------------------|
| <b>UI Access</b>    | Right-click the report to delete in the project tree, and then click <b>Delete</b> on the shortcut menu |       |                                     |
| <b>Parameters</b>   | Name                                                                                                    | Type  | Description                         |
|                     | <ReportNameArray>                                                                                       | Array | Array of report names to be deleted |
| <b>Return Value</b> | None.                                                                                                   |       |                                     |

|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <b>Python Syntax</b>  | DeleteReports(<ReportNameArray>)                        |
| <b>Python Example</b> | <code>oModule.DeleteReports (["Rept2DRectFreq"])</code> |

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteReports <ReportNameArray>                            |
| <b>VB Example</b> | <code>oModule.DeleteReports Array("Rept2DRectFreq")</code> |

## DeleteTraces

Deletes an existing traces or traces.

|                     |                                                                                                        |             |                                                                                                        |
|---------------------|--------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click the Trace to delete in the project tree, and then click <b>Delete</b> on the shortcut menu |             |                                                                                                        |
| <b>Parameters</b>   | <b>Name</b>                                                                                            | <b>Type</b> | <b>Description</b>                                                                                     |
|                     | <TraceSelection>                                                                                       | Array       | Structured array define selections.<br><br>Array("<ReportName>:=", <TracesArray>, <TracesArray>, ... ) |
|                     | <ReportName>                                                                                           | String      | Name of Report                                                                                         |
|                     | <TracesArray>                                                                                          | Array       | Contains the traces to delete within a report<br><br>Array(<Trace>, <Trace>, ...)                      |
|                     | <Trace>                                                                                                | String      | A specific trace that the user wishes to delete                                                        |
| <b>Return Value</b> | None.                                                                                                  |             |                                                                                                        |

|                       |                                                                                                                     |
|-----------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | DeleteTraces(<TraceSelection>)                                                                                      |
| <b>Python Example</b> | <pre>oModule.DeleteTraces (["XY Plot 1:=",<br/>["dB(S(LumpPort1,LumpPort1))"],<br/>"XY Plot 2:=", ["Mag_E"]])</pre> |

|                   |                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteTraces <TraceSelection>                                                                                                        |
| <b>VB Example</b> | <pre>oModule.DeleteTraces Array("XY Plot 1:=", _<br/>Array("dB(S(LumpPort1,LumpPort1))"), )<br/>"XY Plot 2:=", Array("Mag_E"))</pre> |

## ExportImageToFile [Reporter]

Exports a report image in a specified format. In Release 23.1, this command is fully supports -ng (non-graphical) mode.

|                     |              |             |                                                                                                                                            |
|---------------------|--------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A          |             |                                                                                                                                            |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>                                                                                                                         |
|                     | <ReportName> | String      | Name of report to be exported.                                                                                                             |
|                     | <FileName>   | String      | Full path of the exported image file name; with extension of jpg, gif, tiff, tif, bmp, or wrl.                                             |
|                     | <Width>      | Integer     | Image width in pixels; if width or height is less or equal to zero, use the report window width, or 500 pixels if report window is closed. |
|                     | <Height>     | Integer     | Image height in pixels; if width or height is less or equal to zero, use the report window height, or 500 pixels report window is closed.  |
| <b>Return Value</b> | None.        |             |                                                                                                                                            |

|                       |                                                                                                                         |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportImageToFile(<ReportName>, <FileName>, <Width>, <Height>)                                                          |
| <b>Python Example</b> | <pre>oModule.ExportImageToFile(     "Rectangular Contour Plot 1",     "D:/work/2015/UV-Export/pp1.gif",     0, 0)</pre> |

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportImageToFile <ReportName>, <FileName>, <Width>, <Height> |
| <b>VB Example</b> | oModule.ExportImageToFile _                                   |

|  |                                                                           |
|--|---------------------------------------------------------------------------|
|  | "Rectangular Contour Plot 1", _<br>"D:/work/2015/UV-Export/pp1.gif", 0, 0 |
|--|---------------------------------------------------------------------------|

## ExportPlotImageToFile

Deprecated. Use [ExportPlotImageWithViewToFile](#).

Creates field plot exports of existing field plots from a given view points, and with the model being auto-sized automatically for each view.

| UI Access    | N/A                            |        |                                                                         |
|--------------|--------------------------------|--------|-------------------------------------------------------------------------|
| Parameters   | Name                           | Type   | Description                                                             |
|              | <FileName>                     | String | Full path plus file name.                                               |
|              | <FolderName>                   | String | Plot folder name.                                                       |
|              | <ItemName>                     | String | Name of fields to plot.                                                 |
|              | <SetViewTopDownDirectionByRCS> | String | Optional. Name of relative coordinate system to use for the field plot. |
| Return Value | None.                          |        |                                                                         |

| Python Syntax  | ExportPlotImageToFile(<FileName>, <FolderName>, <ItemName>, <SetViewTopDownDirectionByRCS>)         |
|----------------|-----------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.ExportPlotImageToFile(<br/>"C:\TestEPITF2.jpg", "",<br/>"Mag_E2", "RelativeCS1")</pre> |

|                   |                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportPlotImageToFile <FileName>, <FolderName>, <ItemName>, <SetViewTopDownDirectionByRCS>    |
| <b>VB Example</b> | <pre>oModule.ExportPlotImageToFile _ "C:\TestEPITF2.jpg", "", _ "Mag_E2", "RelativeCS1"</pre> |

## ExportReport

### Note:

The ExportReport script command has been replaced by the script command [ExportToFile](#). ExportReport remains in order to retain backward compatibility for existing scripts, but it is strongly recommended that you now use [ExportToFile](#).

Export a report to a data file.

*Command:* None

*Syntax:* ExportReport <ReportName>, <FileName>, <FileExtension>

*Return Value:* None

*Parameters:* <ReportName>

Type: string

<Filename>

Type: string

<FileExtension>

Type: string

*VB Example:*

```
Dim oAnsoftApp
```

```
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("BJTinverter")
Set oDesign = oProject.SetActiveDesign("Nexxim1")
oDesign.ExportReport "Data Table 1", "table_test", "csv"
```

|                |                                                     |
|----------------|-----------------------------------------------------|
| Python Syntax  | ExportReport( <ReportName>, <FileName>)             |
| Python Example | oDesign.ExportReport(<br>"Plot1", "c:\report1.dat") |

## ExportToFile

**Note:**

The ExportToFile script command has replaced the script command [ExportReport](#). ExportReport remains in order to retain backward compatibility for existing scripts, but it is strongly recommended that you now use ExportToFile.

From a data table or plot, generates text format, comma delimited, tab delimited, or .dat type output files.

|                     |                                                                                |         |                                                                                                                                                                                                   |
|---------------------|--------------------------------------------------------------------------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click on report name in the project tree and select <b>Export Data</b> . |         |                                                                                                                                                                                                   |
| <b>Parameters</b>   | Name                                                                           | Type    | Description                                                                                                                                                                                       |
|                     | <ReportName>                                                                   | String  | Name of report to be exported.                                                                                                                                                                    |
|                     | <FileName>                                                                     | String  | Full path of the exported image file name; with extension of<br>.txt - Post processor format file<br>.csv - Comma-delimited data file<br>.tab - Tab-separated file<br>.dat - Ansys plot data file |
|                     | <UnitSpec>                                                                     | String  | For example, "kV, Mhz, yd"                                                                                                                                                                        |
|                     | <UseTraceNumberFormat>                                                         | Boolean | "True", "False"                                                                                                                                                                                   |
| <b>Return Value</b> | None.                                                                          |         |                                                                                                                                                                                                   |

|                      |                                        |
|----------------------|----------------------------------------|
| <b>Python Syntax</b> | ExportToFile(<ReportName>, <FileName>) |
|----------------------|----------------------------------------|

|                       |                                                                                                                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.ExportToFile("rETotal", "C:/Users/Documents/rETotal.csv") oModule.ExportToFile("S Parameter Table 1", "D:/Users/Documents/cfft.csv", False, " kV, MHz, yd ", True)</pre> |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                         |
|-------------------|-------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>ExportToFile &lt;ReportName&gt;, &lt;FileName&gt;</code>          |
| <b>VB Example</b> | <code>oModule.ExportToFile "rETotal", "C:/Documents/rETotal.csv"</code> |

ExportToFile [Reporter]

From a data table or plot, generates text format, comma delimited, tab delimited, .dat, or .rdat type output files.

|              |                                                                           |        |                    |                            |
|--------------|---------------------------------------------------------------------------|--------|--------------------|----------------------------|
| UI Access    | Right-click on report name in the Project tree and select <b>Export</b> . |        |                    |                            |
| Parameters   | Name                                                                      | Type   | Description        |                            |
|              | <ReportName>                                                              | String | Name of the report |                            |
|              | <FileName>                                                                | String | Path and File Name |                            |
|              |                                                                           |        | Supported formats  |                            |
|              |                                                                           |        | .txt               | Post processor format file |
|              |                                                                           |        | .csv               | Comma-delimited data file  |
|              |                                                                           |        | .tab               | Tab-separated file         |
|              |                                                                           |        | .dat               | Ansys plot data file       |
|              |                                                                           |        | .rdat              | Ansys report data file     |
| Return Value | None                                                                      |        |                    |                            |



|                       |                                                                     |
|-----------------------|---------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportToFile (<ReportName>, <FileName>)                             |
| <b>Python Example</b> | <pre>oModule.ExportToFile(<br/>    "Plot1", "c:\report1.dat")</pre> |

|                   |                                                                   |
|-------------------|-------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportToFile <ReportName>, <FileName>                             |
| <b>VB Example</b> | <pre>oModule.ExportToFile<br/>    "Plot1", "c:\report1.dat"</pre> |

## ExportMarkerTable

Exports the marker table to a .csv or .tab file.

|                     |                                                                                               |        |                                   |
|---------------------|-----------------------------------------------------------------------------------------------|--------|-----------------------------------|
| <b>UI Access</b>    | [product] > <b>Fields</b> > <b>Plot Fields</b> > <b>Marker</b> > <b>Export Marker Table</b> . |        |                                   |
| <b>Parameters</b>   | Name                                                                                          | Type   | Description                       |
|                     | <FileName>                                                                                    | String | Name of export file include path. |
| <b>Return Value</b> | None.                                                                                         |        |                                   |

|                       |                                                                      |
|-----------------------|----------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportMarkerTable(<FileName>)                                        |
| <b>Python Example</b> | <pre>oModule.ExportMarkerTable("C:/work/FieldMarkerTable.csv")</pre> |

|                  |                              |
|------------------|------------------------------|
| <b>VB Syntax</b> | ExportMarkerTable <FileName> |
|------------------|------------------------------|

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <b>VB Example</b> | <code>oModule.ExportMarkerTable "C:/work/FieldMarkerTable.csv"</code> |
|-------------------|-----------------------------------------------------------------------|

## FFTONReport

Performs an FFT on a selected report.

|                     |                                                                                           |             |                                                                                                                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click on <b>Results</b> in the project tree, select <b>Perform FFT On Report...</b> |             |                                                                                                                                                                                                             |
| <b>Parameters</b>   | <b>Name</b>                                                                               | <b>Type</b> | <b>Description</b>                                                                                                                                                                                          |
|                     | <code>&lt;ReportName&gt;</code>                                                           | String      | Name of specified report.                                                                                                                                                                                   |
|                     | <code>&lt;WindowName&gt;</code>                                                           | String      | Name of window to apply for FFT. Possible values are "Rectangular", "Tri", "Van Hann", "Hamming", "Blackman", "Lanczos", "Weber", "Welch".                                                                  |
|                     | <code>&lt;Function&gt;</code>                                                             | String      | Function to apply on transformation values. Possible values are "none", "ang_deg", "ang_rad", "arg", "cang_deg", "cang_rad", "dB", "dB1 normalize", "dB20normalize", "dBc", "im", "mag", "normalize", "re". |
| <b>Return Value</b> | None.                                                                                     |             |                                                                                                                                                                                                             |

|                       |                                                                                   |
|-----------------------|-----------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>FFTONReport &lt;ReportName&gt;, &lt;WindowName&gt;, &lt;Function&gt;</code> |
| <b>Python Example</b> | <code>oModule.FFTONReport("XY Plot 1", "Rectangular", "dB")</code>                |

|                   |                                                                                    |
|-------------------|------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>FFTONReport(&lt;ReportName&gt;, &lt;WindowName&gt;, &lt;Function&gt;)</code> |
| <b>VB Example</b> | <code>oModule.FFTONReport "XY Plot 1", "Rectangular", "dB"</code>                  |

## GetAllReportNames

Gets the names of existing reports in a design

|                     |                       |
|---------------------|-----------------------|
| <b>UI Access</b>    | N/A                   |
| <b>Parameters</b>   | None.                 |
| <b>Return Value</b> | Array of report names |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | GetAllReportNames()                      |
| <b>Python Example</b> | <code>oModule.GetAllReportNames()</code> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | GetAllReportNames                      |
| <b>VB Example</b> | <code>oModule.GetAllReportNames</code> |

## GetAllCategories

Get all available category names (not including variable and output-variables) in a solution for a report type and display type, returned as an array of text strings.

|                   |                |             |                                                               |
|-------------------|----------------|-------------|---------------------------------------------------------------|
| <b>UI Access</b>  | NA             |             |                                                               |
| <b>Parameters</b> | <b>Name</b>    | <b>Type</b> | <b>Description</b>                                            |
|                   | <ReportType>   | String      | Report type name.                                             |
|                   | <DisplayType>  | String      | Name of display type.                                         |
|                   | <SolutionName> | String      | Name of solution.                                             |
|                   | <SimValueCtxt> | Array       | A context name, or array of strings that encode the contexts. |

|              |                       |
|--------------|-----------------------|
| Return Value | Array of text strings |
|--------------|-----------------------|

|                |                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | <code>GetAllCategories(&lt;ReportType&gt;, &lt;DisplayType&gt;, &lt;SolutionName&gt;, &lt;SimValueCtxt&gt;)</code>       |
| Python Example | <code>oModule.GetAllCategories("Far Fields",<br/>"Rectangular Plot", "Setup1 : LastAdaptive", "Infinite Sphere1")</code> |

|            |                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | <code>GetAllCategories &lt;ReportType&gt;, &lt;DisplayType&gt;, &lt;SolutionName&gt;, &lt;SimValueCtxt&gt;</code>               |
| VB Example | <code>oModule.GetAllCategories _<br/>"Far Fields", "Rectangular Plot", _<br/>"Setup1 : LastAdaptive", "Infinite Sphere1"</code> |

## GetAllQuantities

Gets all available quantity names in category, returned as an array of text strings.

|            |                |        |                                                                                                                          |
|------------|----------------|--------|--------------------------------------------------------------------------------------------------------------------------|
| UI Access  | NA             |        |                                                                                                                          |
| Parameters | Name           | Type   | Description                                                                                                              |
|            | <ReportType>   | String | Report type name.                                                                                                        |
|            | <DisplayType>  | String | Display type name.                                                                                                       |
|            | <SolutionName> | String | Name of solution.                                                                                                        |
|            | <SimValueCtxt> | Array  | A context name, or array of string that encoded the contexts(l).                                                         |
|            | <CategoryName> | String | A category name as input parameter. a category name returned in GetAllCategories() or "Variables", or "Output Variables" |

|                     |                       |
|---------------------|-----------------------|
| <b>Return Value</b> | Array of text strings |
|---------------------|-----------------------|

|                       |                                                                                                                                                     |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAllQuantities(&lt;ReportType&gt;,&lt;DisplayType&gt;, &lt;SolutionName&gt;, &lt;SimValueCtxt&gt;, &lt;CategoryName&gt;)</code>             |
| <b>Python Example</b> | <pre>oModule.GetAllQuantities(<br/>    "Far Fields", "Rectangular Plot",<br/>    "Setup1 : LastAdaptive",<br/>    "Infinite Sphere1", "Gain")</pre> |

|                   |                                                                                                                                                         |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetAllQuantities &lt;ReportType&gt;,&lt;DisplayType&gt;, &lt;SolutionName&gt;, &lt;SimValueCtxt&gt;, &lt;CategoryName&gt;</code>                  |
| <b>VB Example</b> | <pre>oModule.GetAllQuantities _<br/>    "Far Fields", "Rectangular Plot", _<br/>    "Setup1 : LastAdaptive", )<br/>    "Infinite Sphere1", "Gain"</pre> |

## GetAvailableDisplayTypes

Retrieves all supported display types in report type as an array of text strings.

|                   |              |        |                   |
|-------------------|--------------|--------|-------------------|
| <b>UI Access</b>  | N/A          |        |                   |
| <b>Parameters</b> | Name         | Type   | Description       |
|                   | <ReportType> | String | Report type name. |

|                     |                       |
|---------------------|-----------------------|
| <b>Return Value</b> | Array of text strings |
|---------------------|-----------------------|

|                       |                                                             |
|-----------------------|-------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAvailableDisplayTypes(&lt;ReportType&gt;)</code>   |
| <b>Python Example</b> | <code>oModule.GetAvailableDisplayTypes("Far Fields")</code> |

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetAvailableDisplayTypes &lt;ReportType&gt;</code>   |
| <b>VB Example</b> | <code>oModule.GetAvailableDisplayTypes "Far Fields"</code> |

## GetAvailableReportTypes

Retrieves all available report types in the current Design as an array of text string.

|                     |                       |
|---------------------|-----------------------|
| <b>UI Access</b>    | N/A                   |
| <b>Parameters</b>   | None.                 |
| <b>Return Value</b> | Array of text strings |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAvailableReportTypes()</code>         |
| <b>Python Example</b> | <code>oModule.GetAvailableReportTypes()</code> |

|                  |                                      |
|------------------|--------------------------------------|
| <b>VB Syntax</b> | <code>GetAvailableReportTypes</code> |
|------------------|--------------------------------------|

|                   |                                              |
|-------------------|----------------------------------------------|
| <b>VB Example</b> | <code>oModule.GetAvailableReportTypes</code> |
|-------------------|----------------------------------------------|

## GetAvailableSolutions

Gets all available solutions in report type as an array of text strings.

|                     |                           |        |                   |
|---------------------|---------------------------|--------|-------------------|
| <b>UI Access</b>    | N/A                       |        |                   |
| <b>Parameters</b>   | Name                      | Type   | Description       |
|                     | <i>&lt;ReportType&gt;</i> | String | Report type name. |
| <b>Return Value</b> | Array of text strings     |        |                   |

|                       |                                                          |
|-----------------------|----------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAvailableSolutions(&lt;ReportType&gt;)</code>   |
| <b>Python Example</b> | <code>oModule.GetAvailableSolutions("Far Fields")</code> |

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetAvailableSolutions &lt;ReportType&gt;</code>   |
| <b>VB Example</b> | <code>oModule.GetAvailableSolutions "Far Fields"</code> |

## GetDataExpressions

Returns data expressions.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                       |
|---------------------|-----------------------|
| <b>UI Access</b>    | N/A                   |
| <b>Parameters</b>   | N/A                   |
| <b>Return Value</b> | Array of text strings |

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | GetDataExpressions()                                  |
| <b>Python Example</b> | <pre>expressions = oModule.GetDataExpressions()</pre> |

|                   |                                                                           |
|-------------------|---------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetDataExpressions()                                                      |
| <b>VB Example</b> | <pre>dim expressions<br/>expressions = oModule.GetDataExpressions()</pre> |

## GetDataUnits

Returns text string containing units.



**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                                         |        |                                                           |
|---------------------|-----------------------------------------|--------|-----------------------------------------------------------|
| <b>UI Access</b>    | N/A                                     |        |                                                           |
| <b>Parameters</b>   | Name                                    | Type   | Description                                               |
|                     | <expressionString>                      | String | Can be returned from <a href="#">GetDataExpressions()</a> |
| <b>Return Value</b> | Text string of units; empty if no units |        |                                                           |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | GetDataUnits(<expressionString>)               |
| <b>Python Example</b> | <code>oModule.GetDataUnits(expressions)</code> |

|                   |                                                |
|-------------------|------------------------------------------------|
| <b>VB Syntax</b>  | GetDataUnits(<expressionString>)               |
| <b>VB Example</b> | <code>oModule.GetDataUnits(expressions)</code> |

## GetDesignVariableNames

Returns array of strings containing design variable names.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                  |
|---------------------|------------------|
| <b>UI Access</b>    | NA               |
| <b>Parameters</b>   | NA               |
| <b>Return Value</b> | Array of strings |

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>Python Syntax</b>  | GetDesignVariableNames()                            |
| <b>Python Example</b> | <pre>names = oModule.GetDesignVariableNames()</pre> |

|                   |                                                                   |
|-------------------|-------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetDesignVariableNames()                                          |
| <b>VB Example</b> | <pre>dim names<br/>names = oModule.GetDesignVariableNames()</pre> |

## GetDesignVariableUnits

Returns array of strings containing design variable units.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                                          |        |                                                               |
|---------------------|------------------------------------------|--------|---------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                      |        |                                                               |
| <b>Parameters</b>   | Name                                     | Type   | Description                                                   |
|                     | <varName>                                | String | Can be returned from <a href="#">GetDesignVariableNames()</a> |
| <b>Return Value</b> | Text string of units; empty if no units. |        |                                                               |

|                       |                                                                    |
|-----------------------|--------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetDesignVariableUnits(<varName>)                                  |
| <b>Python Example</b> | <pre>units = oModule.GetDesignVariableUnits('Variable Name')</pre> |

|                   |                                                                                  |
|-------------------|----------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetDesignVariableUnits(<varName>)                                                |
| <b>VB Example</b> | <pre>dim units<br/>units = oModule.GetDesignVariableUnits("Variable Name")</pre> |

## GetDesignVariableValue

Returns a design variable's value.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |              |         |                                                                                      |
|---------------------|--------------|---------|--------------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A          |         |                                                                                      |
| <b>Parameters</b>   | Name         | Type    | Description                                                                          |
|                     | <varName>    | String  | Can be returned from <a href="#">GetDesignVariableNames()</a>                        |
|                     | <siValue>    | Boolean | True to return SI-value; False to return by <a href="#">GetDesignVariableUnits()</a> |
| <b>Return Value</b> | Double value |         |                                                                                      |

|                       |                                                                |
|-----------------------|----------------------------------------------------------------|
| <b>Python Syntax</b>  | GetDesignVariableValue(<varName>, <siValue>)                   |
| <b>Python Example</b> | <pre>value = oModule.GetDesignVariableValue('varName',1)</pre> |

|                   |                                                                              |
|-------------------|------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetDesignVariableValue(<varName>, <siValue>)                                 |
| <b>VB Example</b> | <pre>dim value value = oModule.GetDesignVariableValue("varName", True)</pre> |

## GetDesignVariationKey

Returns a design's Variation Key.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                                  |
|---------------------|----------------------------------|
| <b>UI Access</b>    | N/A                              |
| <b>Parameters</b>   | N/A                              |
| <b>Return Value</b> | String containing variation key. |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | GetDesignVariationKey()                      |
| <b>Python Example</b> | <code>oModule.GetDesignVariationKey()</code> |

|                   |                                                              |
|-------------------|--------------------------------------------------------------|
| <b>VB Syntax</b>  | GetDesignVariationKey()                                      |
| <b>VB Example</b> | <pre>dim key set key = oModule.GetDesignVariationKey()</pre> |

## GetDisplayType (Maxwell)

**Use:** Get Display type of a report.

**Command:** None

**Syntax:** GetDisplayType <ReportName>

**Return Value:** String for display type

**Parameters:**

<ReportName>

Type: string

The name of report to find the display type for.

**Example:** Dim displayType

```
displayType = oModule.GetDisplayType("XY Plot 1")
```

# GetImagDataValues

Returns array of imaginary data values.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|              |                    |         |                                                                                                   |
|--------------|--------------------|---------|---------------------------------------------------------------------------------------------------|
| UI Access    | N/A                |         |                                                                                                   |
| Parameters   | Name               | Type    | Description                                                                                       |
|              | <expressionString> | String  | Can be returned from <a href="#">GetDataExpressions()</a>                                         |
|              | <siValue>          | Boolean | True to return SI-value; False to return with units returned in <a href="#">GetSweepUnits()</a> . |
| Return Value | Array of doubles   |         |                                                                                                   |

|                       |                                                                        |
|-----------------------|------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetImagDataValues(<expressionString>,<siValue>)                        |
| <b>Python Example</b> | <pre>imaginaryvalues = oModule.GetImagDataValues('expression',1)</pre> |

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetImagDataValues(<expressionString>,<siValue>)                                                   |
| <b>VB Example</b> | <pre>dim imaginaryvalues<br/>imaginaryvalues = oModule.GetImagDataValues("expression",True)</pre> |

## GetPerQuantityPrimarySweepValues

Returns per quantity primary sweep values.

### Important:

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                                                                                                        |         |                                                                               |
|---------------------|--------------------------------------------------------------------------------------------------------|---------|-------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                                                                                    |         |                                                                               |
| <b>Parameters</b>   | Name                                                                                                   | Type    | Description                                                                   |
|                     | <expressionString>                                                                                     | String  | Can be returned from <a href="#">GetDataExpressions()</a> .                   |
|                     | <siValue>                                                                                              | Boolean | True to return SI-value; False to return by <a href="#">GetSweepUnits()</a> . |
| <b>Return Value</b> | Array of doubles if <a href="#">IsPerQuantityPrimarySweep()</a> returned True; error if returned False |         |                                                                               |

|                      |                                                          |
|----------------------|----------------------------------------------------------|
| <b>Python Syntax</b> | GetPerQuantitySweepValues(<expressionString>, <siValue>) |
|----------------------|----------------------------------------------------------|

|                |                                                                                          |
|----------------|------------------------------------------------------------------------------------------|
| Python Example | <pre>sweepvalues = oModule.GetPerQuantitySweepValues('0.111,0.201,0.345,0.231', 1)</pre> |
|----------------|------------------------------------------------------------------------------------------|

|            |                                                                                                             |
|------------|-------------------------------------------------------------------------------------------------------------|
| VB Syntax  | GetPerQuantitySweepValues(<expressionString>, <siValue>)                                                    |
| VB Example | <pre>dim sweepvalues sweepvalues = oModule.GetPerQuantitySweepValues("0.111,0.201,0.345,0.231", True)</pre> |

## GetPropertyValue

Returns the value of a single property belonging to a specific *<PropServer>* and *<PropTab>*. This function is available with the Project, Design or Editor objects, including definition editors.

**Tip:** Use the script recording feature and edit a property, and then view the resulting script to see the format for that property.

|            |           |        |                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------|-----------|--------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | N/A       |        |                                                                                                                                                                                                                                                                                                                                                                                                                |
| Parameters | Name      | Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                    |
|            | <PropTab> | String | One of the following, where tab titles are shown in parentheses: <ul style="list-style-type: none"><li>PassedParameterTab ("Parameter Values")</li><li>DefinitionParameterTab (Parameter Defaults")</li><li>LocalVariableTab ("Variables" or "Local Variables")</li><li>ProjectVariableTab ("Project variables")</li><li>ConstantsTab ("Constants")</li><li>BaseElementTab ("Symbol" or "Footprint")</li></ul> |



|                     |                               |        |                                                                                                                                                                                                                                 |
|---------------------|-------------------------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                               |        | <ul style="list-style-type: none"> <li>• ComponentTab ("General")</li> <li>• Component("Component")</li> <li>• CustomTab ("Intrinsic Variables")</li> <li>• Quantities ("Quantities")</li> <li>• Signals ("Signals")</li> </ul> |
|                     | <i>&lt;PropServer&gt;</i>     | String | An object identifier, generally returned from another script method, such as <code>CompInst@R;2;3</code>                                                                                                                        |
|                     | <i>&lt;PropName&gt;</i>       | String | Name of the property.                                                                                                                                                                                                           |
| <b>Return Value</b> | String value of the property. |        |                                                                                                                                                                                                                                 |

|                       |                                                                                                                                                       |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetPropertyValue (&lt;PropTab&gt;, &lt;PropServer&gt;, &lt;PropName&gt;)</code>                                                                 |
| <b>Python Example</b> | <pre>selectionArray = oEditor.GetSelections() for k in selectionArray:     val = oEditor.GetPropertyValue("PassedParameterTab", k, "R")     ...</pre> |

|                   |                                                                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetPropertyValue (&lt;PropTab&gt;, &lt;PropServer&gt;, &lt;PropName&gt;)</code>                                                               |
| <b>VB Example</b> | <pre>selectionArray = oEditor.GetSelections for k in selectionArray:     val = oEditor.GetPropertyValue("PassedParameterTab", k, "R")     ...</pre> |

# GetRealDataValues

Returns array of real data values.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|              |                    |         |                                                                                                   |
|--------------|--------------------|---------|---------------------------------------------------------------------------------------------------|
| UI Access    | N/A                |         |                                                                                                   |
| Parameters   | Name               | Type    | Description                                                                                       |
|              | <expressionString> | String  | Can be returned from <a href="#">GetDataExpressions()</a>                                         |
|              | <siValue>          | Boolean | True to return SI-value; False to return with units returned in <a href="#">GetSweepUnits()</a> . |
| Return Value | Array of doubles   |         |                                                                                                   |

|                |                                                                     |
|----------------|---------------------------------------------------------------------|
| Python Syntax  | GetRealDataValues(<expressionString>,<siValue>)                     |
| Python Example | <code>realvalues = oModule.GetRealDataValues('expression',1)</code> |

|           |                                                 |
|-----------|-------------------------------------------------|
| VB Syntax | GetRealDataValues(<expressionString>,<siValue>) |
|-----------|-------------------------------------------------|

|                   |                                                                                      |
|-------------------|--------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>dim realvalues realvalues = oModule.GetRealDataValues("expression", True)</pre> |
|-------------------|--------------------------------------------------------------------------------------|

## GetReportTraceNames

Gets the names of existing trace names in a plot.

|                     |                                          |        |                         |
|---------------------|------------------------------------------|--------|-------------------------|
| <b>UI Access</b>    | N/A                                      |        |                         |
| <b>Parameters</b>   | Name                                     | Type   | Description             |
|                     | <PlotName>                               | String | Name of specified plot. |
| <b>Return Value</b> | Array of strings containing trace names. |        |                         |

|                       |                                                             |
|-----------------------|-------------------------------------------------------------|
| <b>Python Syntax</b>  | GetReportTraceNames(<PlotName>)                             |
| <b>Python Example</b> | <pre>oModule.GetReportTraceNames("SParameter Plot 1")</pre> |

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <b>VB Syntax</b>  | GetReportTraceNames <PlotName>                             |
| <b>VB Example</b> | <pre>oModule.GetReportTraceNames "SParameter Plot 1"</pre> |

## GetSolutionContexts

Gets all available solution context names in a solution as an array of text strings.

|                   |      |      |             |
|-------------------|------|------|-------------|
| <b>UI Access</b>  | N/A  |      |             |
| <b>Parameters</b> | Name | Type | Description |
|                   |      |      |             |

|                     |                             |        |                    |
|---------------------|-----------------------------|--------|--------------------|
|                     | <i>&lt;ReportType&gt;</i>   | String | Report type name.  |
|                     | <i>&lt;DisplayType&gt;</i>  | String | Display type name. |
|                     | <i>&lt;SolutionName&gt;</i> | String | Name of solution.  |
| <b>Return Value</b> | Array of text strings       |        |                    |

|                       |                                                                                                          |
|-----------------------|----------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetSolutionContexts(&lt;ReportType&gt;, &lt;DisplayType&gt;, &lt;SolutionName&gt;)</code>          |
| <b>Python Example</b> | <pre>oModule.GetSolutionContexts(<br/>    "Far Fields", "Rectangular Plot", "Setup1:LastAdaptive")</pre> |

|                   |                                                                                                                    |
|-------------------|--------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetSolutionContexts &lt;ReportType&gt;, &lt;DisplayType&gt;, &lt;SolutionName&gt;</code>                     |
| <b>VB Example</b> | <pre>oModule.GetSolutionContexts _<br/>    "Far Fields", "Rectangular Plot", _<br/>    "Setup1:LastAdaptive"</pre> |

## GetSolutionDataPerVariation

Obtains solution data for a given report type and solution. You must have already run a simulation.

|                   |                              |             |                                      |
|-------------------|------------------------------|-------------|--------------------------------------|
| <b>UI Access</b>  | N/A                          |             |                                      |
| <b>Parameters</b> | <b>Name</b>                  | <b>Type</b> | <b>Description</b>                   |
|                   | <i>&lt;reportTypeArg&gt;</i> | String      | Report type name as input parameter. |

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                            |                                                                                                        |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|--------------------------------------------------------------------------------------------------------|
|                     | <solutionNameArg>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | String                     | Solution name as input parameter.                                                                      |
|                     | <simValueCtxtArg>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | Structured Array           | Same as ContextArray values created in the relevant CreateReport script.                               |
|                     | <familiesArg>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | Array of Strings           | Same as FamiliesArray values created in the relevant CreateReport script.                              |
|                     | <expressionArg>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | String or Array of Strings | Text string or array of text strings; valid expression, may validate it as the data-table Y-component. |
| <b>Return Value</b> | <p>ARRAY of ISolutionDataResultComInterface objects, containing:</p> <ul style="list-style-type: none"> <li>• <a href="#">GetSweepNames()</a></li> <li>• <a href="#">GetSweepUnits()</a></li> <li>• <a href="#">GetSweepValues()</a></li> <li>• <a href="#">IsPerQuantityPrimarySweep()</a></li> <li>• <a href="#">GetDataExpressions()</a> – should be the same as &lt;expressionArg&gt;</li> <li>• <a href="#">GetPerQuantityPrimarySweepValues()</a></li> <li>• <a href="#">IsDataComplex()</a></li> <li>• <a href="#">GetDataUnits()</a></li> <li>• <a href="#">GetRealDataValues()</a></li> <li>• <a href="#">GetImagDataValues()</a></li> <li>• <a href="#">ReleaseData()</a></li> <li>• <a href="#">GetDesignVariableNames()</a></li> <li>• <a href="#">GetDesignVariableUnits()</a></li> <li>• <a href="#">GetDesignVariableValue()</a></li> <li>• <a href="#">GetDesignVariationKey()</a></li> </ul> |                            |                                                                                                        |

**Note:**

- This command is *not* recordable from the UI, but its parameters are similar to CreateReport, so you may record a CreateReport script to get the parameter values.
- For the returned ISolutionDataResultComInterface object, some of its functions have an optional boolean parameter: SIValue. SIValue defaults to True. When the pass in value is True, return data values will be in Standard International values; when False, return data values will be in the current units.

**Example:** Freq Sweep with [1GHz, 2GHz, 3GHz], GetSweepUnits("Freq") return "GHz"; GetSweepValues("Freq", True) return [1000000, 2000000, 3000000]; GetSweepValues("Freq", False) return [1, 2, 3].

|                       |                                                                                                                                                                                                                                                       |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetSolutionDataPerVariation(reportTypeArg, solutionNameArg, simValueCtxtArg, familiesArg, expressionArg)                                                                                                                                              |
| <b>Python Example</b> | <pre>oModule = oDesign.GetModule("ReportSetup")  arr = oModule.GetSolutionDataPerVariation('Modal Solution Data', 'Setup1 : Sweep', ['Domain:=', 'Sweep'], ['Freq:=', ['All'], 'offset:=', ['All']], ['S (Port1,Port1)', 'dB(S(Port1,Port3))'])</pre> |

|                   |                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSolutionDataPerVariation(reportTypeArg, solutionNameArg, simValueCtxtArg, familiesArg, expressionArg)                                |
| <b>VB Example</b> | <pre>set solutionData = oModule.GetSolutionDataPerVariation( "Modal Solution Data", "Setup1 : Sweep", Array("Domain:=", "Sweep"),</pre> |

```
Array("Freq:=", Array("All"), "offset:=", Array("All")),
Array("S(Port1,Port1)", "dB(S(Port1,Port3))")
)
```

## GetSweepNames

Returns array of text strings containing primary sweep name(s).

### Important:

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                       |
|---------------------|-----------------------|
| <b>UI Access</b>    | N/A                   |
| <b>Parameters</b>   | N/A                   |
| <b>Return Value</b> | Array of text strings |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | GetSweepNames()                                |
| <b>Python Example</b> | <pre>sweepnames = arr[0].GetSweepNames()</pre> |

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSweepNames()                                               |
| <b>VB Example</b> | <pre>dim sweepnames sweepnames = arr[0].GetSweepNames()</pre> |

## GetSweepUnits

Returns text string containing units.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                              |        |                    |
|---------------------|------------------------------|--------|--------------------|
| <b>UI Access</b>    | N/A                          |        |                    |
| <b>Parameters</b>   | Name                         | Type   | Description        |
|                     | <sweepName>                  | String | Primary sweep name |
| <b>Return Value</b> | Text string containing units |        |                    |

|                       |                                                          |
|-----------------------|----------------------------------------------------------|
| <b>Python Syntax</b>  | GetSweepUnits(<sweepName>)                               |
| <b>Python Example</b> | <pre>sweepunits = oModule.GetSweepUnits('Sweep 1')</pre> |

|                   |                                                                             |
|-------------------|-----------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSweepUnits(<sweepName>)                                                  |
| <b>VB Example</b> | <pre>dim sweepunits<br/>sweepunits = oModule.GetSweepUnits("Sweep 1")</pre> |



## GetSweepValues

Returns sweep values.

**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |                  |         |                                                                               |
|---------------------|------------------|---------|-------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A              |         |                                                                               |
| <b>Parameters</b>   | Name             | Type    | Description                                                                   |
|                     | <sweepName>      | String  | Primary sweep name                                                            |
|                     | <siValue>        | Boolean | True to return SI-value; False to return by <a href="#">GetSweepUnits()</a> . |
| <b>Return Value</b> | Array of doubles |         |                                                                               |

|                       |                                                                  |
|-----------------------|------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetSweepValues(<sweepName>, <siValue>)                           |
| <b>Python Example</b> | <pre>sweepvalues = oModule.GetSweepValues('Sweep 1', True)</pre> |

|                   |                                                                                      |
|-------------------|--------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSweepValues(<sweepName>, <siValue>)                                               |
| <b>VB Example</b> | <pre>dim sweepvalues<br/>sweepvalues = oModule.GetSweepValues("Sweep 1", True)</pre> |

## GroupPlotCurvesByGroupingStrategy

Groups curves in a Stacked Plot automatically based on a curve grouping strategy.

|              |                 |        |                                                            |
|--------------|-----------------|--------|------------------------------------------------------------|
| UI Access    | N/A             |        |                                                            |
| Parameters   | Name            | Type   | Description                                                |
|              | <ReportName>    | String | Name of report.                                            |
|              | <GroupStrategy> | String | Strategy for grouping, "Single", "By Trace" or "By Units". |
| Return Value | None.           |        |                                                            |

|                |                                                                                        |  |  |
|----------------|----------------------------------------------------------------------------------------|--|--|
| Python Syntax  | GroupPlotCurvesByGroupingStrategy(<ReportName>, <GroupStrategy>)                       |  |  |
| Python Example | <code>oModule.GroupPlotCurvesByGroupingStrategy("Transient Plot 1", "By Trace")</code> |  |  |

|            |                                                                                       |  |  |
|------------|---------------------------------------------------------------------------------------|--|--|
| VB Syntax  | GroupPlotCurvesByGroupingStrategy <ReportName>, <GroupStrategy>                       |  |  |
| VB Example | <code>oModule.GroupPlotCurvesByGroupingStrategy "Transient Plot 1", "By Trace"</code> |  |  |

## ImportIntoReport

Imports .tab, .csv, and .dat format files into a report.

|            |                                                                             |        |                    |
|------------|-----------------------------------------------------------------------------|--------|--------------------|
| UI Access  | Right-click on report name in the Project tree and select <b>Import....</b> |        |                    |
| Parameters | Name                                                                        | Type   | Description        |
|            | <ReportName>                                                                | String | Name of the Report |

|                         |                                                                                                                                                                                                                                                                                                                                  |                         |                           |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |
|-------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------|---------------------------|--------------------|--|--|--|------|---------------------------|--|--|------|--------------------|--|--|------|----------------------|
|                         | <table><tr><td><b>&lt;FileName&gt;</b></td><td>String</td><td colspan="2">Path and File Name</td></tr><tr><td></td><td></td><td>.csv</td><td>Comma-delimited data file</td></tr><tr><td></td><td></td><td>.tab</td><td>Tab-separated file</td></tr><tr><td></td><td></td><td>.dat</td><td>Ansys plot data file</td></tr></table> | <b>&lt;FileName&gt;</b> | String                    | Path and File Name |  |  |  | .csv | Comma-delimited data file |  |  | .tab | Tab-separated file |  |  | .dat | Ansys plot data file |
| <b>&lt;FileName&gt;</b> | String                                                                                                                                                                                                                                                                                                                           | Path and File Name      |                           |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |
|                         |                                                                                                                                                                                                                                                                                                                                  | .csv                    | Comma-delimited data file |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |
|                         |                                                                                                                                                                                                                                                                                                                                  | .tab                    | Tab-separated file        |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |
|                         |                                                                                                                                                                                                                                                                                                                                  | .dat                    | Ansys plot data file      |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |
| <b>Return Value</b>     | None                                                                                                                                                                                                                                                                                                                             |                         |                           |                    |  |  |  |      |                           |  |  |      |                    |  |  |      |                      |

|                       |                                                                      |
|-----------------------|----------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ImportIntoReport (&lt;ReportName&gt;, &lt;FileName&gt;)</code> |
| <b>Python Example</b> | <code>oDesign.ImportIntoReport ("Plot1", "c:\report1.dat")</code>    |

|                   |                                                                    |
|-------------------|--------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>ImportIntoReport &lt;ReportName&gt;, &lt;FileName&gt;</code> |
| <b>VB Example</b> | <code>oDesign.ImportIntoReport "Plot1", "c:\report1.dat"</code>    |

## IsDataComplex

Returns whether an expression is complex.

### Important:

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                  |    |
|------------------|----|
| <b>UI Access</b> | NA |
|------------------|----|

|              |                                                            |        |                                                             |
|--------------|------------------------------------------------------------|--------|-------------------------------------------------------------|
| Parameters   | Name                                                       | Type   | Description                                                 |
|              | <expressionString>                                         | String | Can be returned from <a href="#">GetDataExpressions()</a> . |
| Return Value | Boolean (True if expression is Complex data; False if not) |        |                                                             |

|                |                                                           |
|----------------|-----------------------------------------------------------|
| Python Syntax  | IsDataComplex(<expressionString>)                         |
| Python Example | <code>oModule.IsDataComplex('.001,.234,.455,.434')</code> |

|            |                                                           |
|------------|-----------------------------------------------------------|
| VB Syntax  | IsDataComplex(<expressionString>)                         |
| VB Example | <code>oModule.IsDataComplex('.001,.234,.455,.434')</code> |

## IsPerQuantityPrimarySweep

Returns whether data expressions have different primary sweep values.

### Important:

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|            |     |
|------------|-----|
| UI Access  | N/A |
| Parameters | N/A |

|                     |                                                                        |
|---------------------|------------------------------------------------------------------------|
| <b>Return Value</b> | Boolean (True if data expressions have different primary sweep values) |
|---------------------|------------------------------------------------------------------------|

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | IsPerQuantityPrimarySweep()                          |
| <b>Python Example</b> | <pre>var = oModule.IsPerQuantityPrimarySweep()</pre> |

|                   |                                                                  |
|-------------------|------------------------------------------------------------------|
| <b>VB Syntax</b>  | IsPerQuantityPrimarySweep()                                      |
| <b>VB Example</b> | <pre>dim var<br/>var = oModule.IsPerQuantityPrimarySweep()</pre> |

## MovePlotCurvesToGroup

In a Stacked Plot move curve(s) from its stack(s) to an existing stack. Here term 'group' is synonymous to 'stack' in the context of cartesian stacked plot.

|                     |              |             |                               |
|---------------------|--------------|-------------|-------------------------------|
| <b>UI Access</b>    | N/A          |             |                               |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>            |
|                     | <ReportName> | String      | Name of report.               |
|                     | <CurveArray> | Array       | Array of curve names to move. |
|                     | <StackName>  | String      | Name of stack to move to.     |
| <b>Return Value</b> | None.        |             |                               |

|               |                                                                |
|---------------|----------------------------------------------------------------|
| <b>Python</b> | MovePlotCurvesToGroup(<ReportName>, <CurveArray>, <StackName>) |
|---------------|----------------------------------------------------------------|

|                       |                                                                                                        |
|-----------------------|--------------------------------------------------------------------------------------------------------|
| <b>Syntax</b>         |                                                                                                        |
| <b>Python Example</b> | <code>oModule.MovePlotCurvesToGroup("XY Stacked Plot 1", ["R2.V : TR", "R2.I : TR"], "Stack 2")</code> |

|                   |                                                                                                                              |
|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>MovePlotCurvesToGroup &lt;ReportName&gt;, &lt;CurveArray&gt;, &lt;StackName&gt;</code>                                 |
| <b>VB Example</b> | <code>oModule.MovePlotCurvesToGroup _<br/>"XY Stacked Plot 1", _<br/>Array("R2.V : TR", "R2.I : TR"), _<br/>"Stack 2"</code> |

## MovePlotCurvesToNewGroup

Move curve(s) from its stack(s) to a new stack. Here term 'group' is synonymous to 'stack' in the context of Cartesian stacked plot.

|                     |              |        |                               |
|---------------------|--------------|--------|-------------------------------|
| <b>UI Access</b>    | N/A          |        |                               |
| <b>Parameters</b>   | Name         | Type   | Description                   |
|                     | <ReportName> | String | Name of report.               |
|                     | <CurveArray> | Array  | Array of curve names to move. |
| <b>Return Value</b> | None.        |        |                               |

|                       |                                                                                |
|-----------------------|--------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>MovePlotCurvesToNewGroup (&lt;ReportName&gt;, &lt;CurveArray&gt;)</code> |
| <b>Python Example</b> | <code>oModule.MovePlotCurvesToNewGroup (</code>                                |

|  |                                                                 |
|--|-----------------------------------------------------------------|
|  | <pre>"XY Stacked Plot 1",<br/>["R2.V : TR", "R2.I : TR"])</pre> |
|--|-----------------------------------------------------------------|

|                   |                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>MovePlotCurvesToNewGroup &lt;ReportName&gt;, &lt;CurveArray&gt;</code>                                 |
| <b>VB Example</b> | <pre>oModule.MovePlotCurvesToNewGroup _<br/>"XY Stacked Plot 1", _<br/>Array("R2.V : TR", "R2.I : TR")</pre> |

## PasteReports

Paste copied reports to results in the current project.

|                     |                        |
|---------------------|------------------------|
| <b>UI Access</b>    | <b>Edit &gt; Paste</b> |
| <b>Parameters</b>   | None.                  |
| <b>Return Value</b> | None.                  |

|                       |                                    |
|-----------------------|------------------------------------|
| <b>Python Syntax</b>  | <code>PasteReports ()</code>       |
| <b>Python Example</b> | <pre>oModule.PasteReports ()</pre> |

|                   |                                 |
|-------------------|---------------------------------|
| <b>VB Syntax</b>  | <code>PasteReports</code>       |
| <b>VB Example</b> | <pre>oModule.PasteReports</pre> |

# PasteTraces

Pastes copied traces to a named plot.

|              |              |        |              |
|--------------|--------------|--------|--------------|
| UI Access    | Paste        |        |              |
| Parameters   | Name         | Type   | Description  |
|              | <ReportName> | String | Name of plot |
| Return Value | None         |        |              |

|                |                                  |
|----------------|----------------------------------|
| Python Syntax  | PasteTraces (<ReportName>)       |
| Python Example | oModule.PasteTraces ("XY Plot1") |

|            |                                |
|------------|--------------------------------|
| VB Syntax  | PasteTraces <ReportName>       |
| VB Example | oModule.PasteTraces "XY Plot1" |

# Release Data

Releases all cached data. After this function is called, all subsequent function calls to the object will fail.



**Important:**

This is a member function of ISolutionDataResultComInterface object, which is the element of the returned array from function [GetSolutionDataPerVariation](#).

|                     |     |
|---------------------|-----|
| <b>UI Access</b>    | N/A |
| <b>Parameters</b>   | N/A |
| <b>Return Value</b> | N/A |

|                       |                                    |
|-----------------------|------------------------------------|
| <b>Python Syntax</b>  | ReleaseData()                      |
| <b>Python Example</b> | <code>oModule.ReleaseData()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | ReleaseData()                      |
| <b>VB Example</b> | <code>oModule.ReleaseData()</code> |

## RenameReport

Renames an existing report.

|                   |                                                                           |        |                 |
|-------------------|---------------------------------------------------------------------------|--------|-----------------|
| <b>UI Access</b>  | Select a report on the Project tree, right-click and select <b>Rename</b> |        |                 |
| <b>Parameters</b> | Name                                                                      | Type   | Description     |
|                   | <OldReportName>                                                           | String | Old Report Name |
|                   | <NewReportName>                                                           | String | New Report Name |

|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                                                          |
|-----------------------|--------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameReport (&lt;OldReportName&gt;, &lt;NewReportName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.RenameReport("XY Plot1", "Reflection")</code>              |

|                   |                                                                        |
|-------------------|------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameReport &lt;OldReportName&gt;, &lt;NewReportName&gt;</code> |
| <b>VB Example</b> | <code>oModule.RenameReport "XY Plot1", "Reflection"</code>             |

## RenameTrace

To rename a trace in a plot

|                     |              |        |                 |
|---------------------|--------------|--------|-----------------|
| <b>UI Access</b>    | N/A          |        |                 |
| <b>Parameters</b>   | Name         | Type   | Description     |
|                     | <ReportName> | String | Name of report. |
|                     | <TraceName>  | String | Name of Trace   |
|                     | <NewName>    | String | New trace name. |
| <b>Return Value</b> | None.        |        |                 |

|                      |                                                                                  |
|----------------------|----------------------------------------------------------------------------------|
| <b>Python Syntax</b> | <code>RenameTrace(&lt;ReportName&gt;, &lt;TraceName&gt;, &lt;NewName&gt;)</code> |
|----------------------|----------------------------------------------------------------------------------|

|                       |                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.RenameTrace ("XY Plot1",<br/>"dB(S(WavePort1,WavePort1))1", "Port1dbS")</pre> |
|-----------------------|--------------------------------------------------------------------------------------------|

|                   |                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <b>RenameTrace &lt;ReportName&gt;, &lt;TraceName&gt;, &lt;NewName&gt;</b>                        |
| <b>VB Example</b> | <pre>oModule.RenameTrace "XY Plot1", _<br/>"dB(S(WavePort1,WavePort1))1", _<br/>"Port1dbS"</pre> |

## ResetPlotSettings

Resets plot settings to defaults.

|                     |                                                                    |             |                         |
|---------------------|--------------------------------------------------------------------|-------------|-------------------------|
| <b>UI Access</b>    | Right-click on a plot, select <b>Edit &gt; Reset Plot Settings</b> |             |                         |
| <b>Parameters</b>   | <b>Name</b>                                                        | <b>Type</b> | <b>Description</b>      |
|                     | <PlotName>                                                         | String      | Name of specified plot. |
| <b>Return Value</b> | None.                                                              |             |                         |

|                       |                                                                   |
|-----------------------|-------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>ResetPlotSettings(&lt;PlotName&gt;)</b>                        |
| <b>Python Example</b> | <pre>oModule.ResetPlotSettings("Differential S-parameters")</pre> |

|                  |                                           |
|------------------|-------------------------------------------|
| <b>VB Syntax</b> | <b>ResetPlotSettings &lt;PlotName&gt;</b> |
|------------------|-------------------------------------------|

|                   |                                                                     |
|-------------------|---------------------------------------------------------------------|
| <b>VB Example</b> | <code>oModule.ResetPlotSettings("Differential S-parameters")</code> |
|-------------------|---------------------------------------------------------------------|

## SavePlotSettingsAsDefault

Saves report plot settings as default.

|                     |                                                       |        |                                        |
|---------------------|-------------------------------------------------------|--------|----------------------------------------|
| <b>UI Access</b>    | <b>Report Templates &gt; Save Settings as Default</b> |        |                                        |
| <b>Parameters</b>   | Name                                                  | Type   | Description                            |
|                     | <PlotName>                                            | String | Name of plot to use for plot defaults. |
| <b>Return Value</b> | None.                                                 |        |                                        |

|                       |                                                             |
|-----------------------|-------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>SavePlotSettingsAsDefault("&lt;PlotName&gt;")</code>  |
| <b>Python Example</b> | <code>oModule.SavePlotSettingsAsDefault ("XY Plot1")</code> |

|                   |                                                            |
|-------------------|------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>SavePlotSettingsAsDefault "&lt;PlotName&gt;"</code>  |
| <b>VB Example</b> | <code>oModule.SavePlotSettingsAsDefault "XY Plot 1"</code> |

## UpdateAllReports

Updates all reports in the **Results** branch in the project tree.

|                   |                                                                                     |
|-------------------|-------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Right-click on <b>Results</b> in the project tree, select <b>Update All Reports</b> |
| <b>Parameters</b> | None                                                                                |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | UpdateAllReports()                      |
| <b>Python Example</b> | <code>oModule.UpdateAllReports()</code> |

|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | UpdateAllReports                      |
| <b>VB Example</b> | <code>oModule.UpdateAllReports</code> |

## UpdateReports

Updates specified reports.

|                     |               |       |                                           |
|---------------------|---------------|-------|-------------------------------------------|
| <b>UI Access</b>    | N/A           |       |                                           |
| <b>Parameters</b>   | Name          | Type  | Description                               |
|                     | <ReportNames> | Array | Array of strings containing report names. |
| <b>Return Value</b> | None.         |       |                                           |

|                       |                                                                 |
|-----------------------|-----------------------------------------------------------------|
| <b>Python Syntax</b>  | UpdateReports(<ReportNames>)                                    |
| <b>Python Example</b> | <code>oModule.UpdateReports (["XY Plot 1", "XY Plot 4"])</code> |

|                  |                             |
|------------------|-----------------------------|
| <b>VB Syntax</b> | UpdateReports <ReportNames> |
|------------------|-----------------------------|

|                   |                                                                    |
|-------------------|--------------------------------------------------------------------|
| <b>VB Example</b> | <code>oModule.UpdateReports Array("XY Plot 1", "XY Plot 4")</code> |
|-------------------|--------------------------------------------------------------------|

## UpdateTraces

Update the traces in a report for which traces are not automatically updated by the Report Traces dialog box, Update Report, Real Time selection.

| UI Access         | In <b>Report</b> dialog, click <b>Apply Traces</b> button |             |                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|-----------------------------------------------------------|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | <b>Name</b>                                               | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                                                |
|                   | <code>&lt;ReportName&gt;</code>                           | String      | Name of Report.                                                                                                                                                                                                                                                                                                                                   |
|                   | <code>&lt;TraceNames&gt;</code>                           | Array       | Array of strings containing trace names.                                                                                                                                                                                                                                                                                                          |
|                   | <code>&lt;SolutionName&gt;</code>                         | String      | Name of the solution.                                                                                                                                                                                                                                                                                                                             |
|                   | <code>&lt;ContextArray&gt;</code>                         | Array       | Context for which the expression is being evaluated. This can be an empty string if there is no context.<br><br><code>Array("Domain:=", &lt;DomainType&gt;)</code><br><br><DomainType> ex. "Sweep" or "Time"<br><br><code>Array("Context:=", &lt;GeometryType&gt;)</code><br><br><GeometryType> ex. "Infinite Spheren", "Spheren",<br>"Polylinen" |
|                   | <code>&lt;FamiliesArray&gt;</code>                        | Array       | Contains sweep definitions for the report.<br><br><code>Array("&lt;VariableName&gt;:= ", &lt;ValueArray&gt;)</code><br><br><ValueArray><br><br><code>Array("All")</code> or <code>Array("Value1", "Value2", ... "Valuen")</code><br><br>examples of <VariableName> "Freq", "Theta", "Distance"                                                    |
|                   | <code>&lt;ReportDataArray&gt;</code>                      | Array       | This array contains the report quantity and X, Y, and (Z) axis definitions.                                                                                                                                                                                                                                                                       |

|                     |                |       |                                                                                                                                                              |
|---------------------|----------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                |       | Array("X Component:=", <VariableName>,<br>"Y Component:=", <VariableName>   <ReportQuantityArray>)<br><ReportQuantityArray> ex. Array("dB(S(Port1, Port1))") |
|                     | <ExtTraceInfo> | Array | Optional. Array defines extended trace information.                                                                                                          |
| <b>Return Value</b> | None.          |       |                                                                                                                                                              |

| Python Syntax         | UpdateTraces(<ReportName>, <SolutionName>, <ContextArray>, <FamiliesArray>, <ReportDataArray>)                                                                                                                                                                                                                                            |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.UpdateTraces("XY Plot 1", ["NEG1.VAL"], "TR4", [   "NAME:Context",   "SimValueContext:=", [2,0,2,0,False,False, -1,1,0,1,1,"",0,0,"CG",False,"0","KP",False,"0","MH",False, "100","TE",False,"100s","TH",False,"40", "TS",False,"0ns","UF",False, "0","WT",False,"0","WW",False,"100"] ], [   "Spectrum:=", ["All"] ],</pre> |

|  |                                                                                                      |
|--|------------------------------------------------------------------------------------------------------|
|  | <pre>[<br/>  "X Component:=", "Spectrum",<br/>  "Y Component:=", ["mag (NEG1.VAL)"]<br/>], [])</pre> |
|--|------------------------------------------------------------------------------------------------------|

| VB Syntax  | UpdateTraces <ReportName>, <SolutionName>, <ContextArray>, <FamiliesArray>, <ReportDataArray>                                                                                                                                                                                                                                                                                                                                                                            |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.UpdateTraces "XY Plot1", Array("dB(S(WavePort1,WavePort1))"), _<br/>  "Setup1 : Sweep1", Array("Domain:=", "Time", "HoldTime:=", 1, _<br/>  "RiseTime:=", 0, "StepTime:=", 0, "Step:=", false, _<br/>  "WindowWidth:=", 1, "WindowType:=", 0, "KaiserParameter:=",1, _<br/>  "MaximumTime:=", 0), Array("Time:=", Array("All")), _<br/>  Array("X Component:=", "Time",_<br/>  "Y Component:=", Array("dB(S(WavePort1,WavePort1))")), _<br/>  Array()</pre> |

# UpdateTracesContextAndSweeps

Edits sweeps and context of multiple traces without affecting their component expressions.

| UI Access  | Modify Report with multiple traces selected. |        |                 |
|------------|----------------------------------------------|--------|-----------------|
| Parameters | Name                                         | Type   | Description     |
|            | <ReportName>                                 | String | Name of Report. |



|                     |                |        |                                                                                                                                       |
|---------------------|----------------|--------|---------------------------------------------------------------------------------------------------------------------------------------|
|                     | <TraceNames>   | Array  | Array of strings containing trace names.                                                                                              |
|                     | <SolutionName> | String | Name of the solution as listed in the <b>Modify Report</b> dialog box.<br><br>For example: "Setup1 : Last Adaptive"                   |
|                     | <ContextArray> | Array  | Context for which the expression is being evaluated. This can be an empty string if there is no context.<br><br>ex. "Sweep" or "Time" |
|                     | <PointSet>     | Array  | Point set for the selected traces, for example, X and Y values for the plot.                                                          |
| <b>Return Value</b> | None           |        |                                                                                                                                       |

|                       |                                                                                                                                                                                                                                                                                                                                                                    |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | UpdateTracesContextAndSweeps(<ReportName>, <TraceNames>, <SolutionName>, <ContextArray>, <PointSet>)                                                                                                                                                                                                                                                               |
| <b>Python Example</b> | <pre>oModule.UpdateTracesContextAndSweeps_ ("Active S Parameter Quick Report", ["dB(ActiveS(Port1:1))", "dB(ActiveS(Port2:1))"], "Setup1 : Sweep1", [], ["Freq=", ["9GHz", "9.05GHz", "9.1GHz", "9.15GHz", "9.2GHz", "9.25GHz", "9.3GHz", "9.35GHz", "9.4GHz", "9.45GHz", "9.5GHz", "9.55GHz", "9.6GHz", "9.65GHz", "9.7GHz", "9.9GHz", "9.95GHz", "10GHz"],</pre> |

|  |                       |
|--|-----------------------|
|  | "offset:=", ["All"])] |
|--|-----------------------|

| VB Syntax  | UpdateTracesContextAndSweeps <ReportName>, <TraceNames>, <SolutionName>, <ContextArray>, <PointSet>                                                                                                                                                                                                                                                                                                                                                                       |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.UpdateTracesContextAndSweeps _ "Active S Parameter Quick Report", _ Array("dB(ActiveS(Port1:1))", "dB(ActiveS(Port2:1))"), _ "Setup1 : Sweep1", Array(), _ Array("Freq:=", _ Array("9GHz", "9.05GHz", "9.1GHz", _ "9.15GHz", "9.2GHz", _ "9.25GHz", "9.3GHz", "9.35GHz", _ "9.4GHz", "9.45GHz", "9.5GHz", _ "9.55GHz", _ "9.6GHz", "9.65GHz", "9.7GHz", _ "9.75GHz", "9.8GHz", "9.85GHz", _ "9.9GHz", "9.95GHz", "10GHz"), _ "offset:=", Array("All"))</pre> |

## UnGroupPlotCurvesInGroup

From a Stacked Plot, ungroups curves in a stack.

|                     |              |        |                   |
|---------------------|--------------|--------|-------------------|
| <b>UI Access</b>    | N/A          |        |                   |
| <b>Parameters</b>   | Name         | Type   | Description       |
|                     | <ReportName> | String | Name of report.   |
|                     | <GroupName>  | String | Stack group name. |
| <b>Return Value</b> | None.        |        |                   |

|                       |                                                                                |
|-----------------------|--------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | UnGroupPlotCurvesInGroup(<ReportName>, <GroupName>)                            |
| <b>Python Example</b> | <code>oModule.UngroupPlotCurvesInGroup("S Parameter Plot 3", "Stack 1")</code> |

|                   |                                                                               |
|-------------------|-------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | UnGroupPlotCurvesInGroup <ReportName>, <GroupName>                            |
| <b>VB Example</b> | <code>oModule.UngroupPlotCurvesInGroup "S Parameter Plot 3", "Stack 1"</code> |

This page intentionally  
left blank.

# 13 - Boundary and Excitation Module Script Commands in Maxwell

Boundary and excitation commands should be executed by the **BoundarySetup** module.

```
Set oModule = oDesign.GetModule("BoundarySetup")
```

```
oModule.CommandName <args>
```

## Conventions Used in this Chapter

<BoundName>

Type: string.

Name of a boundary.

<BoundNameArray>

Type: Array of strings

An array of the names in a boundary/excitation group.

<AssignmentObjects>

***Type: Array of strings.***

An array of object names.

<AssignmentFaces>

Type: Array of integers.

An array of face IDs. The ID of a face can be determined through the user interface using the **Modeler>Measure>Area** command. The face ID is given in the **Measure Information** dialog box.

<LineEndPoint>

```
Array(<double>, <double>, <double>)
<CoordSysArray>
Array("NAME:CoordSysVector",
"Origin:=", <CoordSysPoint>,
"UPos:=", <LineEndPoint>)
```

### Related Topics

[General Commands Recognized by the Boundary/Excitations Module](#)

[Script Commands for Creating and Modifying Boundaries in Maxwell](#)

[Script Commands for Creating and Modifying Excitations in Maxwell](#)

## General Commands Recognized by the Boundary/Excitations Module

[AddAssignmentToBoundary](#)

[DeleteAllBoundaries](#)

[DeleteAllExcitations](#)

[DeleteBoundaries](#)

[GetBoundaryAssignment](#)

[GetBoundaries](#)

[GetBoundariesOfType](#)

[GetCoreLossEffect](#)

[GetCoreLossEffectOnField](#)

[GetDisplacementCurrent](#)[GetEddyEffect](#)[GetExcitations](#)[GetExcitationsOfType](#)[GetHybridRegions](#)[GetHybridRegionsOfType](#)[GetNumBoundaries](#)[GetNumBoundariesOfType](#)[GetNumExcitations](#)[GetNumExcitationsOfType](#)[ReassignBoundary](#)[RemoveAssignmentFromBoundary](#)[RenameBoundary](#)[ReprioritizeBoundaries](#)

## AddAssignmentToBoundary

Adds a new geometry assignment to a boundary.

| UI Access  | N/A          |       |                                                                                             |
|------------|--------------|-------|---------------------------------------------------------------------------------------------|
| Parameters | Name         | Type  | Description                                                                                 |
|            | <Assignment> | Array | Structured array.<br><br>Array("Name:<BoundName>",<br><br>"Objects:=", <AssignmentObjects>, |

|              |                                                                                        |                               |  |  |  |                               |
|--------------|----------------------------------------------------------------------------------------|-------------------------------|--|--|--|-------------------------------|
|              | <table><tr><td></td><td></td><td>"Faces:=", &lt;AssignmentFaces&gt;)</td></tr></table> |                               |  |  |  | "Faces:=", <AssignmentFaces>) |
|              |                                                                                        | "Faces:=", <AssignmentFaces>) |  |  |  |                               |
| Return Value | None.                                                                                  |                               |  |  |  |                               |

|                       |                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddAssignmentToBoundary(<Assignment>)                                                          |
| <b>Python Example</b> | <pre>oModule.AddAssignmentToBoundary((<br/>    ["NAME:PerfE1",<br/>    "Faces:=", [12]])</pre> |

|                   |                                                                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AddAssignmentToBoundary <Assignment>                                                                                                              |
| <b>VB Example</b> | <pre>oModule.AddAssignmentTOBoundary Array("NAME:PerfE1", _<br/>    "Objects:=", Array("Box2", "Box3"), _<br/>    "Faces:=", Array(12, 11))</pre> |

## DeleteAllBoundaries

Deletes all boundaries.

|                     |                                               |
|---------------------|-----------------------------------------------|
| <b>UI Access</b>    | [product] > <b>Boundaries &gt; Delete All</b> |
| <b>Parameters</b>   | None.                                         |
| <b>Return Value</b> | None.                                         |



|                       |                                            |
|-----------------------|--------------------------------------------|
| <b>Python Syntax</b>  | DeleteAllBoundaries()                      |
| <b>Python Example</b> | <code>oModule.DeleteAllBoundaries()</code> |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | DeleteAllBoundaries                      |
| <b>VB Example</b> | <code>oModule.DeleteAllBoundaries</code> |

## DeleteAllExcitations

Deletes all excitations.

|                     |                                                    |
|---------------------|----------------------------------------------------|
| <b>UI Access</b>    | [product] > <b>Excitations</b> > <b>Delete All</b> |
| <b>Parameters</b>   | None.                                              |
| <b>Return Value</b> | None.                                              |

|                       |                                             |
|-----------------------|---------------------------------------------|
| <b>Python Syntax</b>  | DeleteAllExcitations()                      |
| <b>Python Example</b> | <code>oModule.DeleteAllExcitations()</code> |

|                   |                                           |
|-------------------|-------------------------------------------|
| <b>VB Syntax</b>  | DeleteAllExcitations                      |
| <b>VB Example</b> | <code>oModule.DeleteAllExcitations</code> |

## DeleteBoundaries

Deletes the specified boundaries and excitations.

|                     |                                                                                                                           |       |                                    |
|---------------------|---------------------------------------------------------------------------------------------------------------------------|-------|------------------------------------|
| <b>UI Access</b>    | <b>Delete</b> command in the <b>List</b> dialog box. Click <b>[product] &gt; List</b> to open the <b>List</b> dialog box. |       |                                    |
| <b>Parameters</b>   | Name                                                                                                                      | Type  | Description                        |
|                     | <NameArray>                                                                                                               | Array | Array of boundary condition names. |
| <b>Return Value</b> | None.                                                                                                                     |       |                                    |

|                       |                                                   |
|-----------------------|---------------------------------------------------|
| <b>Python Syntax</b>  | DeleteBoundaries(<NameArray>)                     |
| <b>Python Example</b> | oModule.DeleteBoundaries(["PerfE1", "WavePort1"]) |

|                   |                                                       |
|-------------------|-------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteBoundaries <NameArray>                          |
| <b>VB Example</b> | oModule.DeleteBoundaries Array("PerfE1", "WavePort1") |

## GetBoundaryAssignment

Gets a list of face IDs associated with the given boundary or excitation assignment.

|                     |                                  |        |                                               |
|---------------------|----------------------------------|--------|-----------------------------------------------|
| <b>UI Access</b>    | N/A                              |        |                                               |
| <b>Parameters</b>   | Name                             | Type   | Description                                   |
|                     | <BoundaryName>                   | String | Name of the specified boundary or excitation. |
| <b>Return Value</b> | Array of face IDs or object IDs. |        |                                               |

|                      |                                       |
|----------------------|---------------------------------------|
| <b>Python Syntax</b> | GetBoundaryAssignment(<BoundaryName>) |
|----------------------|---------------------------------------|

|                       |                                                    |
|-----------------------|----------------------------------------------------|
| <b>Python Example</b> | <code>oModule.GetBoundaryAssignment("Rad1")</code> |
|-----------------------|----------------------------------------------------|

|                   |                                                         |
|-------------------|---------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetBoundaryAssignment &lt;BoundaryName&gt;</code> |
| <b>VB Example</b> | <code>oModule.GetBoundaryAssignment "Rad1"</code>       |

## GetBoundaries

Gets boundary names in the current design.

|                     |                          |
|---------------------|--------------------------|
| <b>UI Access</b>    | N/A                      |
| <b>Parameters</b>   | None.                    |
| <b>Return Value</b> | Array of boundary names. |

|                       |                                      |
|-----------------------|--------------------------------------|
| <b>Python Syntax</b>  | <code>GetBoundaries()</code>         |
| <b>Python Example</b> | <code>oModule.GetBoundaries()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | <code>GetBoundaries</code>         |
| <b>VB Example</b> | <code>oModule.GetBoundaries</code> |

## GetBoundariesOfType

Gets boundary names of the given type.

|                  |     |
|------------------|-----|
| <b>UI Access</b> | N/A |
|------------------|-----|

|              |                                            |        |                        |
|--------------|--------------------------------------------|--------|------------------------|
| Parameters   | Name                                       | Type   | Description            |
|              | <BoundaryType>                             | String | Name of boundary type. |
| Return Value | Array of boundary names of the given type. |        |                        |

|                |                                                      |
|----------------|------------------------------------------------------|
| Python Syntax  | GetBoundariesOfType(<BoundaryType>)                  |
| Python Example | <code>oModule.GetBoundariesOfType("PerfectE")</code> |

|            |                                                     |
|------------|-----------------------------------------------------|
| VB Syntax  | GetBoundariesOfType <BoundaryType>                  |
| VB Example | <code>oModule.GetBoundariesOfType "PerfectE"</code> |

## GetCoreLossEffect

**Use:** Returns whether the core loss is included in the simulation for the object. Applicable to 2D and 3D Transient and to 3D Eddy Current solution types.

**Command:** None

**Syntax:** GetCoreLossEffect(<ObjectName>)

**Return Value:** Returns true or false based on the check box selection. True if checked.

**Parameters:** <ObjectName>

Type:<string>

Object name.

For example: "inner\_arm"

**Example:** `oModule.GetCoreLossEffect("inner_arm")`

### GetCoreLossEffectOnField

**Use:** Returns whether the core loss effect on the field is being considered. Applicable to 2D and 3D Transient solution types.

**Command:** None

**Syntax:** `GetCoreLossEffectOnField()`

**Return Value:** Returns true or false based on the check box selection. True if checked.

**Parameters:** None

**Example:** `oModule.GetCoreLossEffectOnField()`

### GetDisplacementCurrent

**Use:** Returns whether the displacement current calculation is activated for the object. Applicable to Eddy Current solution type.

**Command:** None

**Syntax:** `GetDisplacementCurrent(<ObjectName>)`

**Return Value:** Returns true or false based on the check box selection. True if checked.

**Parameters:** `<ObjectName>`

Type: `<string>`

Object name.

For example: `"coil1"`

**Example:** `oModule.GetDisplacementCurrent("coil1")`

### GetEddyEffect

**Use:** Returns whether the eddy effect is activated for the object. Applicable to 2D and 3D Transient and to Eddy Current solution types.

**Command:** None

**Syntax:** GetEddyEffect(<ObjectName>)

**Return Value:** Returns true or false based on the given object name. True if checked.

**Parameters:** <ObjectName>

Type:<string>

Object name.

For example: "inner\_arm"

**Example:** oModule.GetEddyEffect("outer\_arm")

## GetExcitations

Gets excitation port and terminal names for a model.

|              |                                                       |
|--------------|-------------------------------------------------------|
| UI Access    | N/A                                                   |
| Parameters   | None.                                                 |
| Return Value | Array of excitation name paired with excitation type. |

|                |                          |
|----------------|--------------------------|
| Python Syntax  | GetExcitations()         |
| Python Example | oModule.GetExcitations() |

|            |                        |
|------------|------------------------|
| VB Syntax  | GetExcitations         |
| VB Example | oModule.GetExcitations |

## GetExcitationsOfType

Gets excitation names of the given type.

|                     |                                              |        |                          |
|---------------------|----------------------------------------------|--------|--------------------------|
| <b>UI Access</b>    | N/A                                          |        |                          |
| <b>Parameters</b>   | Name                                         | Type   | Description              |
|                     | <ExcitationType>                             | String | Name of excitation type. |
| <b>Return Value</b> | Array of excitation names of the given type. |        |                          |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | GetExcitationsOfType(<ExcitationType>)    |
| <b>Python Example</b> | oModule.GetExcitationsOfType("Wave Port") |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | GetExcitationsOfType <ExcitationType>    |
| <b>VB Example</b> | oModule.GetExcitationsOfType "Wave Port" |

## GetNumBoundaries

Gets the number of boundaries in a design.

|                     |                               |
|---------------------|-------------------------------|
| <b>UI Access</b>    | N/A                           |
| <b>Parameters</b>   | None.                         |
| <b>Return Value</b> | Integer number of boundaries. |

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | GetNumBoundaries()                      |
| <b>Python Example</b> | <code>oModule.GetNumBoundaries()</code> |

|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | GetNumBoundaries                      |
| <b>VB Example</b> | <code>oModule.GetNumBoundaries</code> |

## GetNumBoundariesOfType

Gets the number of boundaries of the given type.

|                     |                               |        |                          |
|---------------------|-------------------------------|--------|--------------------------|
| <b>UI Access</b>    | N/A                           |        |                          |
| <b>Parameters</b>   | Name                          | Type   | Description              |
|                     | < <i>BoundaryType</i> >       | String | Specified boundary type. |
| <b>Return Value</b> | Integer number of boundaries. |        |                          |

|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <b>Python Syntax</b>  | GetNumBoundariesOfType(< <i>BoundaryType</i> >)         |
| <b>Python Example</b> | <code>oModule.GetNumBoundariesOfType("PerfectE")</code> |

|                   |                                                        |
|-------------------|--------------------------------------------------------|
| <b>VB Syntax</b>  | GetNumBoundariesOfType < <i>BoundaryType</i> >         |
| <b>VB Example</b> | <code>oModule.GetNumBoundariesOfType "PerfectE"</code> |



## GetNumExcitations

Gets the number of excitations in a design.

|                     |                                |
|---------------------|--------------------------------|
| <b>UI Access</b>    | N/A                            |
| <b>Parameters</b>   | None.                          |
| <b>Return Value</b> | Integer number of excitations. |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | GetNumExcitations()                      |
| <b>Python Example</b> | <code>oModule.GetNumExcitations()</code> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | GetNumExcitations                      |
| <b>VB Example</b> | <code>oModule.GetNumExcitations</code> |

## GetNumExcitationsOfType

Gets the number of excitations of the given type, including all defined modes and terminals of ports.

|                     |                                |        |                               |
|---------------------|--------------------------------|--------|-------------------------------|
| <b>UI Access</b>    | N/A                            |        |                               |
| <b>Parameters</b>   | Name                           | Type   | Description                   |
|                     | <ExcitationType>               | String | Specified type of excitation. |
| <b>Return Value</b> | Integer number of excitations. |        |                               |

|                       |                                            |
|-----------------------|--------------------------------------------|
| <b>Python Syntax</b>  | GetNumExcitationsOfType(<ExcitationType>)  |
| <b>Python Example</b> | oModule.GetNumExcitationsOfType("Voltage") |

|                   |                                           |
|-------------------|-------------------------------------------|
| <b>VB Syntax</b>  | GetNumExcitationsOfType <ExcitationType>  |
| <b>VB Example</b> | oModule.GetNumExcitationsOfType "Voltage" |

**ReassignBoundary**

Specifies a new geometry assignment for a boundary.

|                     |                                                                                         |             |                                                                                                                          |
|---------------------|-----------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click <b>Boundaries</b> > <b>Reassign</b> or <b>Excitations</b> > <b>Reassign</b> |             |                                                                                                                          |
| <b>Parameters</b>   | <b>Name</b>                                                                             | <b>Type</b> | <b>Description</b>                                                                                                       |
|                     | <AssignmentArray>                                                                       | Array       | Structured array.<br><br>Array("Name:<BoundName>",<br>"Objects:=", <AssignmentObjects>,<br>"Faces:=", <AssignmentFaces>) |
| <b>Return Value</b> | None.                                                                                   |             |                                                                                                                          |

|                       |                                     |
|-----------------------|-------------------------------------|
| <b>Python Syntax</b>  | ReassignBoundary(<AssignmentArray>) |
| <b>Python Example</b> | oModule.ReassignBoundary(<br>[      |

|  |                                               |
|--|-----------------------------------------------|
|  | <pre>"NAME:PerfH12", "Faces:=", [17] ])</pre> |
|--|-----------------------------------------------|

|                   |                                                                                     |
|-------------------|-------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ReassignBoundary <AssignmentArray>                                                  |
| <b>VB Example</b> | <pre>oModule.ReassignBoundary _ Array("NAME:PerfH12", _ "Faces:=", Array(17))</pre> |

## RemoveAssignmentFromBoundary

Removes a geometry assignment from a boundary.

|                     |                                                                                    |       |                                                                                                                                                 |
|---------------------|------------------------------------------------------------------------------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click on a boundary or an excitation, select <b>Remove from Assignment</b> . |       |                                                                                                                                                 |
| <b>Parameters</b>   | Name                                                                               | Type  | Description                                                                                                                                     |
|                     | <AssignmentArray>                                                                  | Array | <p>Structured array.</p> <pre>Array("Name:&lt;BoundName&gt;", "Objects:=", &lt;AssignmentObjects&gt;, "Faces:=", &lt;AssignmentFaces&gt;)</pre> |
| <b>Return Value</b> | None.                                                                              |       |                                                                                                                                                 |

|                      |                                                 |
|----------------------|-------------------------------------------------|
| <b>Python Syntax</b> | RemoveAssignmentFromBoundary(<AssignmentArray>) |
|----------------------|-------------------------------------------------|

|                       |                                                                                                                                      |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.RemoveAssignmentFromBoundary(<br/><br/>    [<br/>        "NAME:Rad1",<br/>        "Faces:=", [11]<br/>    ]<br/>)</pre> |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RemoveAssignmentFromBoundary &lt;AssignmentArray&gt;</code>                                    |
| <b>VB Example</b> | <pre>oModule.RemoveAssignmentFromBoundary _<br/>Array("NAME:Rad1", _<br/>"Faces:=", Array(11))</pre> |

**RenameBoundary**

Renames a boundary or excitation.

|                     |                                                                                                |             |                                     |
|---------------------|------------------------------------------------------------------------------------------------|-------------|-------------------------------------|
| <b>UI Access</b>    | Right-click a boundary in the project tree, and then click <b>Rename</b> on the shortcut menu. |             |                                     |
| <b>Parameters</b>   | <b>Name</b>                                                                                    | <b>Type</b> | <b>Description</b>                  |
|                     | <OldName>                                                                                      | String      | Name of the boundary to be renamed. |
|                     | <NewName>                                                                                      | String      | New name for the boundary.          |
| <b>Return Value</b> | None.                                                                                          |             |                                     |

|                      |                                                               |
|----------------------|---------------------------------------------------------------|
| <b>Python Syntax</b> | <code>RenameBoundary(&lt;OldName&gt;, &lt;NewName&gt;)</code> |
|----------------------|---------------------------------------------------------------|

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>Python Example</b> | <code>oModule.RenameBoundary("Rad2", "Rad3")</code> |
|-----------------------|-----------------------------------------------------|

|                   |                                                              |
|-------------------|--------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameBoundary &lt;OldName&gt;, &lt;NewName&gt;</code> |
| <b>VB Example</b> | <code>oModule.RenameBoundary "Rad2", "Rad3"</code>           |

## ReprioritizeBoundaries

Specifies the order in which the boundaries and excitations are recognized by the solver. The first boundary in the list has the highest priority.

**Note:** this command is only valid if all defined boundaries and excitations appear in the list. All ports must be listed before any other boundary type.

|                     |                                                    |       |                                                                                                                  |
|---------------------|----------------------------------------------------|-------|------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | <b>[product] &gt; Boundaries &gt; Reprioritize</b> |       |                                                                                                                  |
| <b>Parameters</b>   | Name                                               | Type  | Description                                                                                                      |
|                     | <NewOrderArray>                                    | Array | Structured array.<br><br><code>Array("NAME:NewOrder",<br/><br/>&lt;BoundName&gt;, &lt;BoundName&gt;, ...)</code> |
| <b>Return Value</b> | None.                                              |       |                                                                                                                  |

|                       |                                                                                        |
|-----------------------|----------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ReprioritizeBoundaries(&lt;NewOrderArray&gt;)</code>                             |
| <b>Python Example</b> | <pre>oModule.ReprioritizeBoundaries(<br/>    ["NAME:NewOrder",<br/>    "Imped1",</pre> |

|  |                                     |
|--|-------------------------------------|
|  | <pre>"PerfE1",<br/>"PerfH1"])</pre> |
|--|-------------------------------------|

|                   |                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ReprioritizeBoundaries <NewOrderArray>                                                               |
| <b>VB Example</b> | <pre>oModule.ReprioritizeBoundaries Array("NAME:NewOrder", _<br/>"Imped1", "PerfE1", "PerfH1")</pre> |

## Script Commands for Creating and Modifying Boundaries in Maxwell

Following are script commands for creating and modifying boundaries that are recognized by the "BoundarySetup" module. In the following commands, all named data can be included/excluded as desired and may appear in any order.

- [AssignZeroTangentialHField](#)
- [EditZeroTangentialHField](#)
- [AssignFluxTangential](#)
- [EditFluxTangential](#)
- [AssignInsulating](#)
- [EditInsulating](#)
- [AssignSymmetry](#)
- [EditSymmetry](#)
- [AssignIndependent](#)
- [AssignDependent \(2D\)](#)
- [AssignDependent \(3D\)](#)
- [EditIndependent](#)

- [EditDependent](#)
- [AssignRadiation](#)
- [EditRadiation](#)
- [AssignImpedance](#)
- [EditImpedance](#)
- [AssignTangentialHField](#)
- [EditTangentialHField](#)
- [AssignVectorPotential](#)
- [EditVectorPotential](#)
- [AssignCylindricalHField](#)
- [EditCylindricalHField](#)
- [AssignResistiveSheet](#)
- [EditResistiveSheet](#)
- [AssignTouching](#)
- [EditTouching](#)
- [AssignThinLayer](#)
- [EditThinLayer](#)

## AssignCylindricalHField

**Use:** Creates an H-Field boundary on a cylindrical face.

**Command:** Maxwell>Boundaries>Assign>Tangential H Field

**Syntax:** AssignCylindricalHField <CylindricalHFieldArray>

**Return Value:** None

**Parameters:** <CylindricalHFieldArray>

```
Array("NAME:<BoundName>",
```

```
"ComponentPhiReal:=", <value>,
"ComponentPhiImg:=", <value>,
"ComponentZReal:=", <value>,
"ComponentZImg:=", <value>,
"ReverseZ:=", <bool>,
"Origin:=", <LineEndPoint>
"Objects:=", <AssignmentObjects>,
"Faces:=", <AssignmentObjects>)
```

**Example:** oModule.AssignCylindricalHField Array("NAME:TangentialHField4", "ComponentPhiReal:=", \_  
"1", "ComponentPhiImg:=", "2", "ComponentZReal:=", "3", \_ "ComponentZImg:=", "4", "ReverseZ:=", \_  
false, \_ "ReferencePosDefined:=", true, "Origin:=", \_ Array("1.36568542494924mm", "0.6mm",  
"0mm"), \_  
"Faces:=", Array(75))

### AssignDependent (2D)

**Use:** Creates a dependent boundary.

**Command:** Maxwell2D>Boundaries>Assign>Matching>Dependent

**Syntax:** AssignDependent ([DependentArray])

**Return Value:** None

**Parameters:** <DependentArray>

```
["NAME:<BoundName>",
```

Either:



- "Edges:=" , <Edge ID list>, the IDs of the edges which the boundary is assigned to.
- "Objects:=" , <list of objects>, the name(s) of the polylines the boundary is assigned to.

```
"ReverseU:=" , <bool>,
"Independent:=" , <string, name of independent>,
"RelationIsSame:=" , <bool>
(where:bool == True for Hdep = Hind, False for Hdep = -Hind, and so on)
]
```

**Example command applied to polyline:**

```
oModule.AssignDependent(["NAME:Dependent1",
"Objects:=" , ["Dependent"],
"ReverseU:=" , False,
"Independent:=" , "Independent1",
"RelationIsSame:=" , False])
```

**Example command applied to edge:**

```
oModule.AssignDependent(["NAME:Dependent1",
"Edges:=" , [8],
"ReverseU:=" , True,
"Independent:=" , "Independent1",
"RelationIsSame:=" , False])
```

## AssignDependent (3D)

**Use:** Creates a dependent boundary.

**Command:** Maxwell3D > Boundaries > Assign > Matching > Dependent

**Syntax:** AssignDependent ([DependentArray])

**Return Value:** None

**Parameters:** <DependentArray>

```
[("NAME:<BoundName>",
```

Either:

- "Faces:=" , <Face ID list>, the IDs of the faces which the boundary is assigned to.
- "Objects:=" <list of objects>, the name(s) of the sheets the boundary is assigned to.

<CoordSysArray>, ["NAME:CoordSysVector", "Origin:=", <list of coordinates>, "UPos:=", <list of coordinates>,]

"ReverseV:=", <bool>,

"Independent:=", <string, name of independent>,

"RelationIsSame:=", <bool>,

*//(bool == True for Hdep = Hind, bool == False for Hdep = -Hind, and so on)]*

**Example command applied to sheet:**

```
oModule.AssignDependent(["NAME:Dependent1",
"Objects:=" , ["DependentSheet"],
["NAME:CoordSysVector","Origin:=" , ["0mm","0mm","0mm"],
"UPos:=" , ["3.67394039744206e-15mm","60mm","0mm"]],
"ReverseV:=" , True,
"Independent:=" , "Independent1",
"RelationIsSame:=" , False])
```

**Example command applied to face:**

```
oModule.AssignDependent(["NAME:Dependent1",
"Faces:=" , [9],
["NAME:CoordSysVector","Origin:=" , ["0.6mm","-0.6mm","1mm"],
"UPos:=" , ["-0.8mm","-0.6mm","0mm"]],
"ReverseV:=" , False,
"Independent:=" , "Independent1",
"RelationIsSame:=" , False])
```

## AssignFluxTangential

Assigns a Flux Tangential boundary condition to the selected face(s).

|                     |                                   |        |                                      |
|---------------------|-----------------------------------|--------|--------------------------------------|
| <b>UI Access</b>    | Boundaries>Assign>Flux Tangential |        |                                      |
| <b>Parameters</b>   | Name                              | Type   | Description                          |
|                     | <NAME>                            | string | Name of the flux tangential boundary |
|                     | <Faces>                           | list   | List of face identifiers             |
| <b>Return Value</b> | None                              |        |                                      |

|                       |                                                                                                                     |  |  |
|-----------------------|---------------------------------------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | AssignFluxTangential (<NAME>, <Faces>)                                                                              |  |  |
| <b>Python Example</b> | <pre>oModule.AssignFluxTangential(     [         "NAME:FluxTangential2",         "Faces:=", [285,313]     ]) </pre> |  |  |

|                   |                                                                                                        |  |  |
|-------------------|--------------------------------------------------------------------------------------------------------|--|--|
| <b>VB Example</b> | <pre>oModule.AssignFluxTangential Array("NAME:FluxTangential2",     "Faces:=", Array(285, 313)) </pre> |  |  |
|-------------------|--------------------------------------------------------------------------------------------------------|--|--|

## AssignImpedance (Maxwell)

**Use:** Creates an impedance boundary.

**Command:** Maxwell3D or Maxwell2D > Boundaries > Assign > Impedance

**Syntax:** AssignImpedance <ImpedanceArray>

**Return Value:** None

**Parameters:** <ImpedanceArray>

```
Array("NAME:<BoundName>",
"Faces:=", <AssignmentFaces>
"Conductivity:=", <value>,
"Permeability:=", <value>,
"UseMaterial:=", <true or false>,
"MaterialName:=", <Material name>,
"NonlinearCoefficient:=", <number>,
"IsPermeabilityNonlinear:=", <true or false>,
"Objects:=", <AssignmentObjects>,
```

**Example:** oModule.AssignImpedance Array("NAME:Impedance5", "Faces:=", Array(29), "Conductivity:=", \_  
"2128000", "Permeability:=", "1", "UseMaterial:=", true, "MaterialName:=", \_"Alnico5", "Non-  
linearCoefficient:=", 5, "IsPermeabilityNonlinear:=", true)

### AssignIndependent (Maxwell)

**Use:** Creates an independent boundary.

**Command:** Maxwell3D or Maxwell2D>Boundaries>Assign>Matching>Independent

**Syntax:** AssignIndependent <IndependentArray>

**Return Value:** None

**Parameters:** <IndependentArray>

```
Array("NAME:<BoundName>",
<CoordSysArray>,
"ReverseV:=", <bool>,
Either:
```

- For 2D - <Edge ID list>: the IDs of the edges which the boundary is assigned to. The value is a list of edge numbers.
- For 3D - "Faces:=", <AssignmentFaces>,
- <Object>: the name(s) of the objects the boundary is assigned to.

For 2D, objects are Polylines. For 3D, objects can be sheets.

**Example AssignIndependent 2D:**

```
oModule.AssignIndependent(["NAME:Independent1",
"Edges:=" , [20], "ReverseU:=" , False])
```

**Example command applied to polyline:**

```
oModule.AssignIndependent(["NAME:Independent1",
"Objects:=" , ["Independent"],"ReverseU:=" , False])
```

**Example AssignIndependent 3D:**

```
oModule.AssignIndependent(["NAME:Independent1",
"Faces:=" , [25],["NAME:CoordSysVector",
"Origin:=" , ["0mm","0mm","82.1090449266692mm"],
"UPos:=" , ["0mm","60mm","82.1090449266692mm"]],
"ReverseV:=" , False])
```

**Example command applied to sheet:**

```
oModule.AssignIndependent(["NAME:Independent1",
"Objects:=" , ["IndependentSheet"],
["NAME:CoordSysVector","Origin:=" , ["0mm","0mm","0mm"],
"UPos:=" , ["60mm","0mm","0mm"]],
"ReverseV:=" , True])
```

## AssignInsulating

**Use:** Creates an insulating boundary.

**Command:** Maxwell3D>Boundaries>Assign>Insulating

**Syntax:** AssignInsulating <InsulatingArray>

**Return Value:** None

**Parameters:** <InsulatingArray>

```
Array("Name:<BoundName>",
 "Objects:=", <AssignmentObjects>,
 "Faces:=", <AssignmentFaces>
 "Permittivity:=", <double value>
 "Thickness:=", <value><unit>
```

**Example:** oModule.AssignInsulating Array("NAME:Insulating2", "Faces:=", Array(9), "Permittivity:=", "3", "Thickness:=", "5mm")

## AssignRadiation (Maxwell)

**Use:** Creates a radiation boundary.

**Command:** Maxwell3D>Boundaries>Assign>Radiation

**Syntax:** AssignRadiation <RadiationArray>

**Return Value:** None

**Parameters:** <RadiationArray>

```
Array("NAME:<BoundName>",
```

```
"Objects:=", <AssignmentObjects>,
```

```
"Faces:=", <AssignmentFaces>)
```

**Example:** `oModule.AssignRadiation Array("NAME:Radiation1", _ "Faces:=", Array(12, 11))`

## AssignResistiveSheet

**Use:** For Maxwell 3D Magnetostatic, Eddy Current and Transient designs, creates a resistive sheet boundary between two conductors.

For 3D Eddy Current and Transient designs, the user specifies a Boundary name, boundary face ID list, and a resistance value with unit.

For 3D Magnetostatic designs, the user specifies a Boundary name, boundary face ID list, and nonlinear anode and cathode coefficients.

**Note:** For more information on the nonlinear resistive sheet boundary condition, see *Assigning a Resistive Sheet Boundary for the 3D Magnetostatic Solver* in the Maxwell Help.

| UI Access    | Maxwell 3D > Boundaries > Assign > Resistive Sheet |         |                                                                                                                                                                                           |
|--------------|----------------------------------------------------|---------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                               | Type    | Description                                                                                                                                                                               |
|              | <Name>                                             | String  | Boundary name                                                                                                                                                                             |
|              | <Objects>                                          | List    | Name of objects or faces to apply boundary condition; it can be a list of names or integers.                                                                                              |
|              | <Nonlinear>                                        | Boolean | If set to True, designates a nonlinear resistive sheet boundary condition, and user must specify A, B, C, and D inputs (double values) for anode and cathode coefficients (see examples). |
|              | <Resistance>                                       | String  | Specifies resistance value with unit; not available if Nonlinear set to True.                                                                                                             |
| Return Value | None                                               |         |                                                                                                                                                                                           |

For 3D Magnetostatic, Eddy Current, and Transient designs

| Python Syntax  | AssignResistiveSheet ([<NAME>, <Objects>, <Nonlinear>,<Resistance>])                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.AssignResistiveSheet (     [         "NAME:ResistiveSheet2",         "Faces:=", [93],         "Nonlinear:=", False,         "Resistance:=", "2ohm"     ]) oModule.AssignResistiveSheet (     [         "NAME:ResistiveSheet1",         "Objects:="          , ["Circle1"],         "Nonlinear:="        , True,         "AnodeParA:="        , "3000000000",         "AnodeParB:="        , "5",         "AnodeParC:="        , "11000000000000000",         "AnodeParD:="        , "2",         "CathodeParA:="      , "3000000000",         "CathodeParB:="      , "10",         "CathodeParC:="      , "11000000000000000",</pre> |



|  |                                             |
|--|---------------------------------------------|
|  | <pre>"CathodeParD:="      , "2"    ])</pre> |
|--|---------------------------------------------|

|                   |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AssignResistiveSheet Array (<NAME>, <Faces>, <Nonlinear>, <Resistance>)                                                                     |
| <b>VB Example</b> | <pre>oModule.AssignResistiveSheet Array("NAME:ResistiveSheet4", "Faces:=", Array(93), "Non- linear:=", False, "Resistance:=", "2ohm")</pre> |

## AssignSymmetry (Maxwell)

**Use:** Creates a symmetry boundary.

**Command:** Maxwell3D or Maxwell2D>Boundaries>Assign>Symmetry

**Syntax:** AssignSymmetry <SymmetryArray>

**Return Value:** None

**Parameters:** <SymmetryArray>

```
Array("NAME:<BoundName>",
 "IsOdd:=", <bool>, //true for odd, false for even
 "Objects:=", <AssignmentObjects>,
 "Faces:=", <AssignmentFaces>)
```

**Example:** oModule.AssignSymmetry Array("NAME:Symmetry1", \_ "IsOdd:=", true, "Faces:=", Array(35))

## AssignTangentialHField

**Use:** Creates an H-Field boundary on a planar face.

**Command:** Maxwell3D>Boundaries>Assign>Tangential H Field

**Syntax:** AssignTangentialHField <TangentialHFieldArray>

**Return Value:** None

**Parameters:** <TangentialHFieldArray>

```
Array("NAME:<BoundName>",
 "ComponentXReal:=", <value>,
 "ComponentXImg:=", <value>,
 "ComponentYReal:=", <value>,
 "ComponentYImg:=", <value>,
 <CoordSysArray>
 "ReverseV:=", <bool>,
 "Objects:=", <AssignmentObjects>,
 "Faces:=", <AssignmentObjects>)
```

**Example:** oModule.AssignTangentialHField Array("NAME:TangentialHField2", "ComponentXReal:=", \_  
"1", "ComponentXImg:=", "2", "ComponentYReal:=", "3", \_ "ComponentYImg:=", "4", Array  
("NAME:CoordSysVector", \_ "Origin:=", Array( "-0.4mm", "-1.4mm", "1mm"), "UPos:=", \_  
Array("-0.4mm", "-0.8mm", "1mm")), "ReverseV:=", true, \_ "Faces:=", Array(7))

**AssignThinLayer**

**Use:** For Maxwell 3D AC Conduction, 3D DC Conduction, 3D Magnetostatic, 3D Eddy Current, and 3D Transient designs, creates a thin layer boundary between two conductors.

|           |                                         |
|-----------|-----------------------------------------|
| UI Access | Maxwell 3D>Boundaries>Assign>Thin Layer |
|-----------|-----------------------------------------|

|                     | Name             | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------|------------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     | <ThinLayerArray> | Array | <ul style="list-style-type: none"> <li>For 3D AC Conduction designs: Array containing the boundary name, boundary face ID array, and permittivity, conductivity, thickness value with unit.</li> <li>For 3D DC Conduction designs: Array containing the boundary name, boundary face ID array, and conductivity and thickness value with unit.</li> <li>For 3D Magnetostatic, and 3D Transient (T-Omega only): Create an air gap boundary between two non-conductors. Array containing the boundary name, and thickness value with unit.</li> <li>For 3D Eddy Current: Array containing the boundary name, AirGap (bool); If AirGap=False, include boundary face ID array, conductivity, relative permeability, and thickness value with unit. If AirGap=true, include only thickness value with unit.</li> </ul> |
| <b>Parameters</b>   |                  |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
| <b>Return Value</b> | None             |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

### For 3D AC Conduction designs

|                       |                                                                                                                                                                                                        |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AssignThinLayer ([<NAME>, <Faces>, <Permittivity>, <Conductivity>, <Thickness>])                                                                                                                       |
| <b>Python Example</b> | <pre>oModule.AssignThinLayer (     [         "NAME:ThinLayer1",         "Faces:=", [40],         "Permittivity:=", "1"         "Conductivity:=", "1S_per_m"         "Thickness:=", "1mm"     ] )</pre> |
| <b>VB Syntax</b>      | AssignThinLayer Array (<NAME>, <Faces>, <Permittivity>, <Conductivity>, <Thickness>)                                                                                                                   |

|                   |                                                                                                                                                                 |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.AssignThinLayer Array("NAME:ThinLayer1", "Faces:=", Array40),<br/>"Permittivity:=", "1", "Conductivity:=", "1S_per_m", "Thickness:=", "1mm")</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|

**For 3D DC Conduction designs**

|                       |                                                                                                                                                                                                     |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>AssignThinLayer ([&lt;NAME&gt;, &lt;Faces&gt;, &lt;Conductivity&gt;, &lt;Thickness&gt;])</b>                                                                                                     |
| <b>Python Example</b> | <pre>oModule.AssignThinLayer(<br/>    [<br/>        "NAME:ThinLayer1",<br/>        "Faces:=", [40],<br/>        "Conductivity:=", "1S_per_m"<br/>        "Thickness:=", "1mm"<br/>    ]<br/>)</pre> |
| <b>VB Syntax</b>      | <b>AssignThinLayer Array (&lt;NAME&gt;, &lt;Faces&gt;, &lt;Conductivity&gt;, &lt;Thickness&gt;)</b>                                                                                                 |
| <b>VB Example</b>     | <pre>oModule.AssignThinLayer Array("NAME:ThinLayer1", "Faces:=", Array40),<br/>"Conductivity:=", "1S_per_m", "Thickness:=", "1mm")</pre>                                                            |

**For 3D Magnetostatic and 3D Transient designs**

|                       |                                                                                                                               |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>AssignThinLayer ([&lt;NAME&gt;, &lt;Thickness&gt;])</b>                                                                    |
| <b>Python Example</b> | <pre>oModule.AssignThinLayer(<br/>    [<br/>        "NAME:ThinLayer1",<br/>        "Thickness:=", "1mm"<br/>    ]<br/>)</pre> |

|                   |                                                                                     |
|-------------------|-------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AssignThinLayer Array (<NAME>, <Thickness>)                                         |
| <b>VB Example</b> | <code>oModule.AssignThinLayer Array("NAME:ThinLayer1", "Thickness:=", "1mm")</code> |

### For 3D Eddy Current designs

|                       |                                                                                                                                                                                                                                                        |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AssignThinLayer ([<NAME>, <Faces>, <AirGap>, <Conductivity>, <RelativePermeability>, <Thickness>])                                                                                                                                                     |
| <b>Python Example</b> | <pre>oModule.AssignThinLayer(     [         "NAME:ThinLayer1",         "Faces:=", [40],         "AirGap:=", False,         "Conductivity:=", "500000000000S_per_m"         "RelativePermeability:=", "1"         "Thickness:=", "0.02mm"     ]) </pre> |

|                   |                                                                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AssignThinLayer Array (<NAME>, <Faces>, <AirGap>, <Conductivity>, <RelativePermeability>, <Thickness>)                                                                                                 |
| <b>VB Example</b> | <pre>oModule.AssignThinLayer Array("NAME:ThinLayer1", "Faces:=", Array(40), "AirGap:=", False, "Conductivity:=", "500000000000S_per_m", "RelativePermeability:=", "1", "Thickness:=", "0.02mm") </pre> |

## AssignTouching

**Use:** For Maxwell 3D Transient designs with translational motion with a moving part touching stationary part, assigns a touching boundary condition to a face of the moving part that touches a stationary part.

|              |                                       |        |                          |
|--------------|---------------------------------------|--------|--------------------------|
| UI Access    | Maxwell 3D>Boundaries>Assign>Touching |        |                          |
| Parameters   | Name                                  | Type   | Description              |
|              | <NAME>                                | string | The name of the boundary |
|              | <Faces>                               | list   | Array of face IDs        |
| Return Value | None                                  |        |                          |

|                |                                                                                                                         |  |  |
|----------------|-------------------------------------------------------------------------------------------------------------------------|--|--|
| Python Syntax  | AssignTouching (<NAME>, <Faces>)                                                                                        |  |  |
| Python Example | <pre>oModule.AssignTouching(<br/>    [<br/>        "NAME:Touching3",<br/>        "Faces:=", [210]<br/>    ]<br/>)</pre> |  |  |

|            |                                                                                      |  |  |
|------------|--------------------------------------------------------------------------------------|--|--|
| VB Syntax  | AssignTouching (<NAME>, <Faces>)                                                     |  |  |
| VB Example | <pre>oModule.AssignTouching<br/>Array("NAME:Touching3", "Faces:=", Array(210))</pre> |  |  |

## AssignVectorPotential

**Use:** For Maxwell 2D Eddy Current, Magnetostatic, and Transient designs, assigns a vector potential boundary condition to the specified edges.

|           |                                               |
|-----------|-----------------------------------------------|
| UI Access | Maxwell 2D>Boundaries>Assign>Vector Potential |
|-----------|-----------------------------------------------|

|                     |                    |             |                                                                                                                   |
|---------------------|--------------------|-------------|-------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b>   | <b>Name</b>        | <b>Type</b> | <b>Description</b>                                                                                                |
|                     | <NAME>             | string      | The name of the boundary                                                                                          |
|                     | <Edges>            | list        | Array of edge IDs                                                                                                 |
|                     | <Phase>            | string      | Phase angle with unit (for 2D Eddy Current designs only)                                                          |
|                     | <Value>            | string      | Vector potential value in weber/m                                                                                 |
|                     | <CoordinateSystem> | string      | Name of the reference coordinate system if the vector potential value is a spatial dependent function such as z+2 |
| <b>Return Value</b> | None               |             |                                                                                                                   |

|                       |                                                                                                                                                                                                                 |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AssignVectorPotential (<NAME>, <Edges>, <Phase>, <Value>, <CoordinateSystem>)                                                                                                                                   |
| <b>Python Example</b> | <pre>oModule.AssignVectorPotential (     [         "NAME:VectorPotential2",         "Edges:=", [21],         "Phase:=", "60deg",         "Value:=", "Z+2",         "CoordinateSystem:=", "Global"     ] )</pre> |

|                   |                                                                                                                                                                             |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AssignVectorPotential (<NAME>, <Edges>, <Phase>, <Value>, <CoordinateSystem>)                                                                                               |
| <b>VB Example</b> | <pre>oModule.AssignVectorPotential Array("NAME:VectorPotential3",     "Edges:=", Array(21), "Phase:=", "60deg", "Value:=", "Z+2"),     "CoordinateSystem:=", "Global"</pre> |

## AssignZeroTangentialHField

**Use:** Creates a zero tangential H-Field boundary.

**Command:** Maxwell>Boundaries>Assign>Zero Tangential H Field

**Syntax:** AssignZeroTangentialHField <ZeroTangentialHFieldArray>

**Return Value:** None

**Parameters:** <ZeroTangentialHFieldArray>

```
Array("NAME:<BoundName>",
 "Objects:=", <AssignmentObjects>,
 "Faces:=", <AssignmentFaces>)
```

**Example:** oModule.AssignZeroTangentialHField Array("NAME:ZeroTangentialHField1", "Faces:=", Array(7))

## EditCylindricalHField

**Use:** Edits a cylindrical H-Field boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditCylindricalHField <BoundName> <CylindricalHFieldArray>

**Return Value:** None

## Script Commands for Creating and Modifying Excitations

### EditDependent (Maxwell)

**Use:** Modifies a dependent boundary.

**Command:** Double-click the boundary in the project tree to modify its settings.



**Syntax:** EditDependent <BoundName> <DependentArray>

**Return Value:** None

**Note:** Refer to [AssignDependent \(2D\)](#) and [AssignDependent \(3D\)](#) examples for syntax details.

## EditFluxTangential

Edits a Flux Tangential boundary condition name.

|                     |                                                                                        |        |                 |
|---------------------|----------------------------------------------------------------------------------------|--------|-----------------|
| <b>UI Access</b>    | Double-click the Flux Tangential Boundaries item in the Project tree to edit the name. |        |                 |
| <b>Parameters</b>   | Name                                                                                   | Type   | Description     |
|                     | <NAME>                                                                                 | string | The edited name |
| <b>Return Value</b> | None                                                                                   |        |                 |

|                       |                                                                                                                     |
|-----------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditFluxTangential_1 (<NAME>)                                                                                       |
| <b>Python Example</b> | <pre>oModule.EditFluxTangential("FluxTangential1",<br/>    [<br/>        "NAME:FluxTangential_New"<br/>    ])</pre> |

|                   |                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.EditFluxTangential "FluxTangential1",<br/>Array("NAME:FluxTangential_New")</pre> |
|-------------------|-----------------------------------------------------------------------------------------------|

## EditImpedance (Maxwell)

**Use:** Edits an impedance boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditImpedance <BoundName> <ImpedanceArray>

**Return Value:** None

**Parameters:** <BoundName>, string name of boundary

<ImpedanceArray>

Array("NAME:<BoundName>",

"Faces:=", <AssignmentFaces>)

"Conductivity:=", <value>,

"Permeability:=", <value>,

"UseMaterial:=", <true or false>,

"MaterialName:=", <Material name>,

"NonlinearCoefficient:=", <number>,

"IsPermeabilityNonlinear:=", <true or false>,

"Objects:=", <AssignmentObjects>,

**Example:** (UseMaterial= true)

```
oModule.EditImpedance "Impedance5", Array("NAME:Impedance5", "UseMaterial:=", true, "MaterialName:=", "Ceramic8D", "IsPermeabilityNonlinear:=", true, "NonlinearCoefficient:=", 6, "Conductivity:=", "0.0001")
```

**Example:** (use constant; UseMaterial false)

```
oModule.EditImpedance "Impedance5", Array("NAME:Impedance5", "UseMaterial:=", false, "Permeability:=", "2", "Conductivity:=", "3")
```

## EditIndependent (Maxwell)

**Use:** Modifies an independent boundary.

**Command:** Double-click the boundary in the Project Manager to modify its settings.

**Syntax:** Edit <BoundName> <PrimaryArray>

**Return Value:** None

**Note:** Refer to [AssignIndependent](#) examples for syntax details.

## EditInsulating

**Use:** Edits an insulating boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditInsulating <BoundName> <InsulatingArray>

**Return Value:** None

**Parameters:** <BoundName> , string name of boundary

<InsulatingArray>

```
Array("Name:<BoundName>",
 "Permittivity:=", <double value>
 "Thickness:=", <value><unit>
```

**Example:** oModule.EditInsulating "Insulating2", Array("NAME:Insulating2", "Permittivity:=", "5", "Thickness:=", "7meter")

## EditRadiation (Maxwell)

**Use:** Edits a radiation boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditRadiation <BoundName> <RadiationArray>

**Return Value:** None

**EditResistiveSheet**

**Use:** For Maxwell 3D Magnetostatic, Eddy Current, Transient designs, edits a resistive sheet boundary. A nonlinear boundary condition can be used in a Maxwell 3D Magnetostatic design.

|              |                                                           |        |                                                                                                                                                                                                                                                                                                                                                               |
|--------------|-----------------------------------------------------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Double-click the boundary in the project tree to edit it. |        |                                                                                                                                                                                                                                                                                                                                                               |
| Parameters   | Name                                                      | Type   | Description                                                                                                                                                                                                                                                                                                                                                   |
|              | <BoundName>                                               | String | The name of the boundary.                                                                                                                                                                                                                                                                                                                                     |
|              | <ResistiveSheetArray>                                     | Array  | For 3D Eddy Current and Transient designs: Array containing the boundary name, and resistance value with unit.<br><br>For 3D Magnetostatic designs: Array containing the boundary name, and nonlinear boundary condition. If the nonlinear boundary condition is selected, the user has to specify inputs (double values) for anode and cathode coefficients. |
| Return Value | None                                                      |        |                                                                                                                                                                                                                                                                                                                                                               |

**For 3D Magnetostatic, Eddy Current, and Transient designs**

|                |                                                                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EditResistiveSheet (<BoundName>, [<NAME>, <Nonlinear>, <Resistance>])                                                                                                             |
| Python Example | <pre>oModule.EditResistiveSheet ("ResistiveSheet1",<br/><br/>    [<br/>        "NAME:ResistiveSheet1",<br/>        "Nonlinear:=", False,<br/>        "Resistance:=", "3ohm"</pre> |

```
])
```

With a nonlinear boundary condition:

```
oModule.EditResistiveSheet("ResistiveSheet1",
```

```
[
```

```
 "NAME:ResistiveSheet1",
```

```
 "Nonlinear:=", True,
```

```
 "AnodeParA:=", "2000000000",
```

```
 "AnodeParB:=", "7.1",
```

```
 "AnodeParC:=", "1200000000000000",
```

```
 "AnodeParD:=", "6",
```

```
 "CathodeParA:=", "1000000000",
```

```
 "CathodeParB:=", "12",
```

```
 "CathodeParC:=", "1200000000000000",
```

```
 "CathodeParD:=", "4"
```

```
])
```

|                   |                                                                                                                                        |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EditResistiveSheet &lt;BoundName&gt;, Array (&lt;NAME&gt;, &lt;Nonlinear&gt;,&lt;Resistance&gt;)</code>                          |
| <b>VB Example</b> | <pre>oModule.EditResistiveSheet "ResistiveSheet2", Array("NAME:ResistiveSheet2", "Non- linear:=", False, "Resistance:=", "3ohm")</pre> |

## EditSymmetry

Modifies a symmetry boundary.

|              |                                                                              |        |                                                                                  |
|--------------|------------------------------------------------------------------------------|--------|----------------------------------------------------------------------------------|
| UI Access    | Double-click the symmetry boundary in the project tree to edit its settings. |        |                                                                                  |
| Parameters   | Name                                                                         | Type   | Description                                                                      |
|              | <BoundaryName>                                                               | String | Name of the symmetry boundary to be edited.                                      |
|              | <SymmetryArray>                                                              | Array  | Structured array.<br><br>Array("NAME:<BoundName>",<br>"IsPerfectE:=", <boolean>) |
| Return Value | None.                                                                        |        |                                                                                  |

|                |                                                                                                  |
|----------------|--------------------------------------------------------------------------------------------------|
| Python Syntax  | EditSymmetry(<BoundaryName>, <SymmetryArray>)                                                    |
| Python Example | <pre>oModule.EditSymmetry("Sym1",<br/>["NAME:Sym1After",<br/>  "IsPerfectE:=", True<br/>])</pre> |

|            |                                                                         |
|------------|-------------------------------------------------------------------------|
| VB Syntax  | EditSymmetry <BoundaryName>, <SymmetryArray>                            |
| VB Example | <pre>oModule.EditSymmetry "Sym1", _<br/>Array("NAME:Sym1After", _</pre> |

|  |                      |
|--|----------------------|
|  | "IsPerfectE=", true) |
|--|----------------------|

## EditTangentialHField

**Use:** Edits a tangential H-Field boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditTangentialHField <BoundName> <TangentialHFieldArray>

**Return Value:** None

## EditThinLayer

**Use:** For Maxwell 3D AC Conduction, 3D DC Conduction, 3D Magnetostatic, 3D Eddy Current, and 3D Transient designs, edit a thin layer boundary.

| UI Access  | Double-click the boundary in the project tree to edit it. |        |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------|-----------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                      | Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|            | <BoundName>                                               | String | The name of the boundary.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|            | <ThinLayerArray>                                          | Array  | <ul style="list-style-type: none"> <li>For 3D AC Conduction designs: Array containing the boundary name, boundary face ID array, and permittivity, conductivity, thickness value with unit.</li> <li>For 3D DC Conduction designs: Array containing the boundary name, boundary face ID array, and conductivity and thickness value with unit.</li> <li>For 3D Magnetostatic and 3D Transient (T-Omega only): An air gap boundary between two non-conductors. Array containing the boundary name, and thickness value with unit.</li> <li>For 3D Eddy Current: Array containing the boundary name, AirGap (bool); If AirGap=False, include conductivity, relative permeability, and thickness value with unit. If AirGap=true, include only thickness value with unit.</li> </ul> |

|              |      |
|--------------|------|
| Return Value | None |
|--------------|------|

**For 3D AC Conduction and 3D DC Conduction designs**

|                |                                                                                                                                                                                                                                            |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EditThinLayer ( <i>BoundName</i> [<NAME>, <Faces>, <Permittivity(AC Conduction)>, <Conductivity>, <Thickness>])                                                                                                                            |
| Python Example | <pre>oModule.EditThinLayer("ThinLayer1"<br/>    [<br/>        "NAME:ThinLayer1",<br/>        "Faces:=", [40],<br/>        "Permittivity:=", "1"<br/>        "Conductivity:=", "1S_per_m"<br/>        "Thickness:=", "1mm"<br/>    ])</pre> |
| VB Syntax      | EditThinLayer <i>BoundName</i> Array (<NAME>, <Faces>, <Permittivity>, <Conductivity>, <Thickness>)                                                                                                                                        |
| VB Example     | <pre>oModule.EditThinLayer Array("NAME:ThinLayer1", "Faces:=", Array40),<br/>"Permittivity:=", "1", "Conductivity:=", "1S_per_m", "Thickness:=", "1mm")</pre>                                                                              |

**For 3D Magnetostatic and 3D Transient designs**

|                |                                                                                                                                    |
|----------------|------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EditThinLayer ( <i>BoundName</i> [<NAME>, <Thickness>])                                                                            |
| Python Example | <pre>oModule.EditThinLayer("ThinLayer1"<br/>    [<br/>        "NAME:ThinLayer1",<br/>        "Thickness:=", "1mm"<br/>    ])</pre> |



|                   |                                                                                   |
|-------------------|-----------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | EditThinLayer <i>BoundName</i> Array (<NAME>, <Thickness>)                        |
| <b>VB Example</b> | <code>oModule.EditThinLayer Array("NAME:ThinLayer1", "Thickness:=", "1mm")</code> |

### For 3D Eddy Current designs

|                       |                                                                                                                                                                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditThinLayer ( <i>BoundName</i> [<NAME>, <AirGap>, <Conductivity>, <RelativePermeability>, <Thickness>])                                                                                                                                |
| <b>Python Example</b> | <pre>oModule.EditThinLayer ("ThinLayer1"     [         "NAME:ThinLayer1",         "AirGap:=", False,         "Conductivity:=", "500000000000S_per_m"         "RelativePermeability:=", "1"         "Thickness:=", "0.02mm"     ] )</pre> |

|                        |                                                                                                                                                                                               |
|------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syn-<br/>tax</b> | EditThinLayer <i>BoundName</i> Array (<NAME>, <AirGap>, <Conductivity>, <RelativePermeability>, <Thickness>)                                                                                  |
| <b>VB<br/>Example</b>  | <code>oModule.EditThinLayer "ThinLayer1", Array("NAME:ThinLayer1", "AirGap:=", False, "Conductivity:=", "500000000000S_per_m", "RelativePermeability:=", "1", "Thickness:=", "0.02mm")</code> |

### EditTouching

**Use:** For Maxwell 3D Transient designs with translational motion with a moving part touching stationary part, edits a touching boundary condition.

|                  |                                                                                                                    |
|------------------|--------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b> | Double-click a touching boundary condition in the Project Manager to open the <b>Touching Boundary</b> dialog box. |
|------------------|--------------------------------------------------------------------------------------------------------------------|

|              |        |        |                          |
|--------------|--------|--------|--------------------------|
| Parameters   | Name   | Type   | Description              |
|              | <NAME> | string | The name of the boundary |
| Return Value | None   |        |                          |

|                |                                                                                         |
|----------------|-----------------------------------------------------------------------------------------|
| Python Syntax  | EditTouching (<NAME>)                                                                   |
| Python Example | <pre>oModule.EditTouching(<br/>    [<br/>        "NAME:Touching3"<br/>    ]<br/>)</pre> |

|            |                                                             |
|------------|-------------------------------------------------------------|
| VB Syntax  | EditTouching (<NAME>)                                       |
| VB Example | <pre>oModule.EditTouching<br/>Array("NAME:Touching3")</pre> |

## EditVectorPotential

**Use:** For Maxwell 2D Eddy Current, Magnetostatic, and Transient designs, edits a vector potential boundary condition.

|            |                                                                                                                                    |        |                                                          |
|------------|------------------------------------------------------------------------------------------------------------------------------------|--------|----------------------------------------------------------|
| UI Access  | Double-click a vector potential boundary condition in the Project Manager to open the <b>Vector Potential Boundary</b> dialog box. |        |                                                          |
| Parameters | Name                                                                                                                               | Type   | Description                                              |
|            | <NAME>                                                                                                                             | string | The name of the boundary                                 |
|            | <Edges>                                                                                                                            | list   | Array of edge IDs                                        |
|            | <Phase>                                                                                                                            | string | Phase angle with unit (for 2D Eddy Current designs only) |

|                     |                    |        |                                                                                                                   |
|---------------------|--------------------|--------|-------------------------------------------------------------------------------------------------------------------|
|                     | <Value>            | string | Vector potential value in weber/m                                                                                 |
|                     | <CoordinateSystem> | string | Name of the reference coordinate system if the vector potential value is a spatial dependent function such as z+2 |
| <b>Return Value</b> | None               |        |                                                                                                                   |

|                       |                                                                                                                                                                                                              |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditVectorPotential (<NAME>, <Edges>, <Phase>, <Value>, <CoordinateSystem>)                                                                                                                                  |
| <b>Python Example</b> | <pre>oModule.EditVectorPotential(     [         "NAME:VectorPotential2",         "Edges:=", [21],         "Phase:=", "60deg",         "Value:=", "z+2",         "CoordinateSystem:=", "Global"     ]) </pre> |

|                   |                                                                                                                                                                            |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | EditVectorPotential (<NAME>, <Edges>, <Phase>, <Value>, <CoordinateSystem>)                                                                                                |
| <b>VB Example</b> | <pre>oModule.EditVectorPotential Array("NAME:VectorPotential3",     "Edges:=", Array(21), "Phase:=", "60deg", "Value:=", "z+2",     "CoordinateSystem:=", "Global") </pre> |

## EditZeroTangentialHField

**Use:** Edits a zero tangential H-Field boundary.

**Command:** Double-click the boundary in the project tree to edit it.

**Syntax:** EditZeroTangentialHField <BoundName>, <PerfectEArray>

**Return Value:** None

## Script Commands for Creating and Modifying Excitations in Maxwell

Following are script commands for creating and modifying excitations that are recognized by the "BoundarySetup" module. In the following commands, all named data can be included/excluded as desired and may appear in any order.

- [AddTerminalsToWinding](#)
- [AssignCharge](#)
- [AssignCoilTerminal](#)
- [AssignCoilTerminalGroup](#)
- [AssignCurrent](#)
- [AssignCurrentDensity](#)
- [AssignCurrentDensityGroup](#)
- [AssignCurrentDensityTerminal](#)
- [AssignCurrentDensityTerminalGroup](#)
- [AssignCurrentGroup](#)
- [AssignFloating](#)
- [AssignSink](#)
- [AssignVoltage](#)
- [AssignVoltageAPhi](#)
- [AssignVolumeChargeDensity](#)
- [AssignVoltageDrop](#)
- [AssignVoltageDropGroup](#)
- [AssignVoltageGroup](#)
- [Assign Winding Group](#)

- [EditCharge](#)
- [EditCoilTerminal](#)
- [EditCurrent](#)
- [EditCurrentDensity](#)
- [EditCurrentDensityTerminal](#)
- [EditExternalCircuit](#)
- [EditFloating](#)
- [EditSink](#)
- [EditVoltage](#)
- [EditVoltageAPhi](#)
- [EditVolumeChargeDensity](#)
- [EditVoltageDrop](#)
- [EditWindingGroup](#)
- [SetCoreLoss](#)
- [SetEddyEffect](#)
- [SetMagnetizationCompute](#)
- [SetMinimumTimeStep](#)

## AddTerminalsToWinding

**Use:** Adds existing terminal(s) to the selected winding.

**Command:** Right-click a winding item in the project tree, and then select **Add Terminals**.

**Syntax:** AddTerminalsToWinding <AddTerminalsToWindingArray>

**Return Value:** None

**Parameters:** < AddTerminalsToWindingArray >

```
Array(" NAME:AddTerminalsToWinding", <BoundListArray>)
```

```
<BoundListArray> Array("NAME:BoundaryList",<CoilTerminalArray>)
```

*Example:*

```
oModule.AddTerminalstoWinding Array("NAME:AddTerminalsToWinding", _
Array("NAME:BoundaryList", Array("NAME:CoilTerminal1", _
"Point out of terminal:=", false, "Conductor number:=", _
"1", "Winding:=", "Winding1", "Faces:=", Array(12)), _
Array("NAME:CoilTerminal2", "Point out of terminal:=", _
false, "Conductor number:=", "1", "Winding:=", _
"Winding1", "Faces:=", Array(7))))
```

### AssignCharge

**Use:** Creates a charge excitation.

**Command:** Maxwell>Excitations>Assign>Charge

**Syntax:** AssignCharge <ChargeArray>

**Return Value:** None

**Parameters:** <ChargeArray>

```
Array("NAME:<BoundName>",
"Value:=", <value>,
"Objects:=", <AssignmentObjects>,
"Faces:=", <AssignmentFaces>)
```

**Example:** oModule.AssignCharge Array("NAME:Charge1", "Value:=", \_ "1", "Faces:=", Array(11))

### AssignCoilGroup

**Use:** For Maxwell 2D designs, creates a group of coil excitations. The coil excitations can be assigned on 2D objects.

|              |                                                                                                                                                                                                                                                                                                                                                            |        |                                                      |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|------------------------------------------------------|
| UI Access    | Select the desired objects (coils) on the model or in the History tree. Then, select <b>Maxwell 2D&gt; Excitations&gt;Assign&gt;Coil</b> to open the <b>Coil Excitation</b> dialog where you can enter the base Name for the group of excitations being created, the number of conductors, and the polarity to be assigned to each of the new excitations. |        |                                                      |
| Parameters   | Name                                                                                                                                                                                                                                                                                                                                                       | Type   | Description                                          |
|              | <NAME>                                                                                                                                                                                                                                                                                                                                                     | string | The name of the coil group                           |
|              | <Objects>                                                                                                                                                                                                                                                                                                                                                  | list   | List of objects to which the excitations are applied |
|              | <Conductor number>                                                                                                                                                                                                                                                                                                                                         | int    | The number of conductors                             |
|              | <Polarity type>                                                                                                                                                                                                                                                                                                                                            | string | Positive or Negative                                 |
| Return Value | None                                                                                                                                                                                                                                                                                                                                                       |        |                                                      |

|                       |                                                                                                                                                                                                                                                                 |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>AssignCoilGroup (CoilGroupArray, &lt;NAME&gt;, &lt;Objects&gt;, &lt;Conductor number&gt;, &lt;Polarity type&gt;)</code>                                                                                                                                   |
| <b>Python Example</b> | <pre>oModule.AssignCoilGroup(["terminal_A_plus", "terminal_A_plus_1"],     [         "NAME:terminal_A_plus",         "Objects:=" , ["terminal_A_plus","terminal_A_plus_1"],         "Conductor number:=" , "10",         "PolarityType:=", Positive    ])</pre> |

|                   |                                                                                                                                                                                                                                     |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>AssignCoilGroup CoilGroupArray, &lt;NAME&gt;, &lt;Objects&gt;, &lt;Conductor number&gt;, &lt;Polarity type&gt;</code>                                                                                                         |
| <b>VB Example</b> | <pre>OModule.AssignCoilGroup Array("terminal_A_plus", "terminal_A_plus_1"), Array(NAME:terminal_A_plus", "Objects:=", Array("terminal_A_plus","terminal_A_plus_1"), "Conductor number:=" , "10", "Polarity type:=", Positive)</pre> |

## AssignCoilTerminal

**Use:** Assigns 2D terminals

**Command:** Maxwell>Excitations>Assign>Coil Terminal

**Syntax:** AssignCoilTerminal <TerminalArray>

**Parameters:** <TerminalArray>

```
Array("NAME:AssignTerminals",
 Array("Name:SourceList", Array("Name:<SourceName>",
 "Excitation:=", <NetObject>, "Objects:=", <Assignment 2D>)),
 Array("Name:SinkList", Array("Name:<SinkName>",
 "Excitation:=", <NetObject>, "Objects:=", <Assignment 2D>)),
 "Name>DeleteList", <Name Array>)
```

**Return Value:** None

```
Example: OModule.Assign2DTerminals Array("NAME:AssignTerminals", Array("NAME:SourceList", Array
("NAME:Source2", "Net:=", _ "Box2", "Objects:=", Array("Rectangle1"))), _ Array("NAME:SinkList",
Array("NAME:Sink1", "Net:=", _ "Box1", "Objects:=", Array("Rectangle2"))), _ "DeleteList:=", "")

oModule.AssignPerfectH Array("NAME:PerfH1", "Faces:=", _ Array(12))
```

## AssignCoilTerminalGroup

**Use:** For Maxwell 3D designs, creates a group of coil terminal excitations. The coil terminals can be assigned on the faces of 3D objects or directly on sheet objects.

|           |                                                                                                                                                                                                                                   |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access | Select the desired objects (coils) on the model or in the History tree. Then, select <b>Maxwell 3D&gt; Excitations&gt;Assign&gt;Coil Terminal</b> to open the <b>Coil Terminal Excitation</b> dialog where you can enter the base |
|-----------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|



|              |                                                                                                                                                                                          |        |                                                                     |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|---------------------------------------------------------------------|
|              | Name for the group of excitations being created, the number of conductors, and use the Swap Direction button to set the current direction to be assigned to each of the new excitations. |        |                                                                     |
| Parameters   | Name                                                                                                                                                                                     | Type   | Description                                                         |
|              | <NAME>                                                                                                                                                                                   | string | The name of the coil terminal group                                 |
|              | <Objects>                                                                                                                                                                                | list   | List of faces or sheet objects to which the excitations are applied |
|              | <Conductor number>                                                                                                                                                                       | int    | The number of conductors                                            |
|              | <Point out of terminal>                                                                                                                                                                  | bool   | True sets the current direction out of the terminal                 |
| Return Value | None                                                                                                                                                                                     |        |                                                                     |

|                |                                                                                                                                                                                                                                                                            |  |  |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| Python Syntax  | <i>AssignCoilTerminalGroup (TerminalGroupArray, &lt;NAME&gt;, &lt;Objects&gt;, &lt;Conductor number&gt;, &lt;Point out of terminal&gt;)</i>                                                                                                                                |  |  |
| Python Example | <pre>oModule.AssignCoilTerminalGroup(["CoilTerminal_1", "CoilTerminal_2"],     [         "NAME:CoilTerminal_1",         "Objects:=" , ["terminal_A_plus","terminal_A_plus_1"],         "Conductor number:=" , "10",         "Point out of terminal:=", False     ]) </pre> |  |  |

|            |                                                                                                                                                                               |  |  |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| VB Syntax  | <i>AssignCoilTerminalGroup TerminalGroupArray, &lt;NAME&gt;, &lt;Objects&gt;, &lt;Conductor number&gt;, &lt;Point out of terminal&gt;</i>                                     |  |  |
| VB Example | <pre>OModule.AssignCoilTerminalGroup Array("CoilTerminal_1", "CoilTerminal_2"), Array(NAME:CoilTerminal_1", "Objects:=", Array("terminal_A_plus","terminal_A_plus_1"), </pre> |  |  |

|  |                                                                |
|--|----------------------------------------------------------------|
|  | "Conductor number:=" , "10", "Point out of terminal:=", False) |
|--|----------------------------------------------------------------|

AssignCurrent

Creates a current source.

|              |                            |        |                                                                                          |
|--------------|----------------------------|--------|------------------------------------------------------------------------------------------|
| UI Access    | Excitations>Assign>Current |        |                                                                                          |
| Parameters   | Name                       | Type   | Description                                                                              |
|              | <NAME>                     | string | The name of the excitation                                                               |
|              | <Faces>                    | list   | List of faces to which the excitation is applied                                         |
|              | <Current>                  | string | Current value with unit                                                                  |
|              | <IsSolid>                  | bool   | True if type is Solid; False if Stranded                                                 |
|              | <CurrentExcitationModel>   | int    | 0 for Single Potential<br>1 for Double Potentials<br>2 for Double Potentials with Ground |
|              | <Point out of terminal>    | bool   | True sets the current direction out of the terminal                                      |
| Return Value | None                       |        |                                                                                          |

|                |                                                                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AssignCurrent_1 (<NAME>, <Faces>, <Current>, <IsSolid>, <CurrentExcitationModel>, <Point out of terminal>)                                                               |
| Python Example | <pre>oModule.AssignCurrent(<br/>    [<br/>        "NAME:Current6",<br/>        "Faces:=" , [260],<br/>        "Current:=" , "0mA",<br/>        "IsSolid:=" , True,</pre> |

|  |                                                                                    |
|--|------------------------------------------------------------------------------------|
|  | <pre> "CurrentExcitationModel:=", 0, "Point out of terminal:=", False     ])</pre> |
|--|------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre> oModule.AssignCurrent Array("NAME:Current6", "Faces:=", Array(952), "Current:=", "0mA", "IsSolid:=", true, "CurrentExcitationModel:=", 0, "Point out of terminal:=", false)</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## AssignCurrentDensity

**Use:** Creates a current density excitation.

**Command:** Maxwell>Excitations>Assign>Current Density

**Syntax:** AssignCurrentDensity <CurrentDensityArray>

**Return Value:** None

**Parameters:** <CurrentDensityArray>

```

Array("NAME:<BoundName>",
"CurrentDensityX:=", <value>,
"Phase:=", <value>,//for eddy current solution type
"CurrentDensityY:=", <value>,
"CurrentDensityZ:=", <bool>,
"CoordinateSystem Name:=", <string>,
"CoordinateSystem Type:=", <string>,
"Objects:=", <AssignmentObjects>)
```

**Example:** oModule.AssignCurrentDensity Array("NAME:CurrentDensity1", "CurrentDensityX=", \_  
"1", "CurrentDensityY=", "2", "CurrentDensityZ=", \_ "3", "CoordinateSystem Name=", "Global",  
\_ "CoordinateSystem Type=", "Cartesian", "Objects=", \_ Array("Box1"))

**Example:** oModule.AssignCurrentDensity Array("NAME:CurrentDensity1", "Phase=", "12", \_ "Cur-  
rentDensityX=", "1", "CurrentDensityY=", \_  
"2", "CurrentDensityZ=", "3", \_  
"CoordinateSystem Name=", "Global",  
"CoordinateSystem Type=", "Cartesian", "Objects=", \_ Array("Box1"))

## AssignCurrentDensityGroup

**Use:** Creates a group of current density excitations.

**Command:** Maxwell>Excitations>Assign>Current Density

**Syntax:** AssignCurrentDensityGroup <BoundNameArray> <CurrentDensityArray>

**Return Value:** None

## AssignCurrentDensityTerminal

**Use:** Creates a current density terminal excitation.

**Command:** Maxwell>Excitations>Assign>Current Density Terminal

**Syntax:** AssignCurrentDensityTerminal <CurrentDensityTerminalArray>

**Return Value:** None

**Parameters:** <CurrentDensityTerminalArray>

```
Array("Name:<BoundName>",
"Objects=", <AssignmentObjects>,
"Faces=", <AssignmentFaces>)
```

**Example:** `oModule.AssignCurrentDensityTerminal Array("NAME:CurrentDensityTerminal1", "Faces:=",  
Array(7))`

## **AssignCurrentDensityTerminalGroup**

**Use:** Creates a group of current density terminal excitations.

**Command:** `Maxwell>Excitations>Assign>Current Density Terminal`

**Syntax:** `AssignCurrentDensityTerminalGroup <BoundNameArray> <CurrentDensityTerminalArray>`

**Return Value:** None

## **AssignCurrentGroup**

**Use:** Creates a group of current excitations.

**Command:** `Maxwell>Excitations>Assign>Current`

**Syntax:** `AssignCurrentGroup <BoundNameArray> <CurrentArray>`

**Return Value:** None

## **AssignFloating**

**Use:** Creates a floating excitation.

**Command:** `Maxwell>Boundaries>Assign>Floating`

**Syntax:** `AssignFloating <FloatingArray>`

**Return Value:** None

**Parameters:** `<FloatingArray>`

```
Array("NAME:<BoundName>",
 "Value:=", <value>,
 "Objects:=", <AssignmentObjects>,
 "Faces:=", <AssignmentFaces>)
```

**Example:** `oModule.AssignFloating Array("NAME:Floating1", _ "Value:=", "1", "Faces:=", Array(11))`

## AssignSink (Maxwell)

**Use:** Creates a sink.

**Command:** Maxwell>Excitations>Assign>Sink

**Syntax:** AssignSink <SinkArray>

**Return Value:** None

**Parameters:** <SinkArray>

```
Array("NAME:<SinkName>",
 "Faces:=", <AssignmentFaces>)
```

**Example:** `oModule.AssignSink Array("NAME:Sink1", _  
"Faces:=", Array(12))`

## AssignVoltage (Maxwell)

**Use:** Creates a voltage excitation.

**Command:** Maxwell>Excitations>Assign>Voltage

**Syntax:** AssignVoltage <SourceArray>

**Return Value:** None

**Parameters:** <SourceArray>

```
Array("NAME:<SourceName>",
 "Faces:=", <AssignmentFaces>)
```

**Example:** `oModule.AssignVoltage Array("NAME:Source1", _`

```
"Faces:=", Array(12))
```

## AssignVoltageAPhi

Assigns a voltage excitation for 3D transient A-Phi solutions.

| UI Access    | Excitations>Assign>Voltage          |        |                                                                                          |
|--------------|-------------------------------------|--------|------------------------------------------------------------------------------------------|
| Parameters   | Name                                | Type   | Description                                                                              |
|              | <NAME>                              | string | The name of the voltage excitation                                                       |
|              | <Objects>                           | list   | List of objects to which the excitation is applied                                       |
|              | <VoltageAPhi>                       | string | Voltage value with unit                                                                  |
|              | <VoltageAPhiInitialCurrent>         | string | Initial current value with unit                                                          |
|              | <VoltageAPhiHasInitialCurrent>      | bool   | True if Initial Current is used.                                                         |
|              | <VoltageAPhiExcitationModel>        | int    | 0 for Single Potential<br>1 for Double Potentials<br>2 for Double Potentials with Ground |
|              | <VoltageAPhi_Point_out_of_terminal> | bool   | True sets the voltage direction out of the terminal                                      |
| Return Value | None                                |        |                                                                                          |

| Python Syntax  | AssignVoltageAPhi (<NAME>, <Objects>, <VoltageAPhi>, <VoltageAPhiInitialCurrent>, <VoltageAPhiHasInitialCurrent>, <VoltageAPhiExcitationModel>, <VoltageAPhi_Point_out_of_terminal>) |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.AssignVoltageAPhi (     [         "NAME:Voltage4",         "Objects:=",      , ["Cylinder1"],         "VoltageAPhi:=",  , "0mV",</pre>                                  |

|  |                                                                                                                                                                                              |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"VoltageAPhiInitialCurrent:=", "0mA",<br/>"VoltageAPhiHasInitialCurrent:=", False,<br/>"VoltageAPhiExcitationModel:=", 0,<br/>"VoltageAPhi_Point_out_of_terminal:=", False<br/>])</pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.AssignVoltageAPhi Array("NAME:Voltage4", "Objects:=", Array("Cylinder1"),<br/>"VoltageAPhi:=", "0mV", "VoltageAPhiInitialCurrent:=", "0mA",<br/>"VoltageAPhiHasInitialCurrent:=", false, "VoltageAPhiExcitationModel:=", 0,<br/>"VoltageAPhi_Point_out_of_terminal:=", false)</pre> |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**AssignVoltageDrop**

**Use:** Creates a voltage drop excitation.

**Command:** Maxwell>Excitations>Assign>Voltage Drop

**Syntax:**AssignVoltageDrop <VoltageDropArray>

**Return Value:** None

**Parameters:** <VoltageDropArray>

```
Array("NAME:<BoundName>",
"Voltage Drop:=", <value>,
"Point out of terminal:=", <bool>,
"Objects:=", <AssignmentObjects>,
"Faces:=", <AssignmentFaces>)
```



**Example:** `oModule.AssignVoltageDrop Array("NAME:VoltageDrop1", _ "Voltage Drop:=", "1kV", "Point out of terminal:=", _ true, "Faces:=", Array(7))`

## AssignVoltageDropGroup

**Use:** Creates a group of voltage drop excitations.

**Command:** Maxwell>Excitations>Assign>Voltage Drop

**Syntax:** AssignVoltageDropGroup <BoundNameArray> <VoltageDropArray>

**Return Value:** None

## AssignVoltageGroup

**Use:** Creates a group of voltage excitations. The size of the bound name array must be identical to the size of assignment. This command is not supported for the **Electric** solution types.

**Command:** Maxwell>Excitations>Assign>Voltage

**Syntax:** AssignVoltageGroup <BoundNameArray> <VoltageArray>

**Return Value:** None

**Example:** `oModule.AssignVoltageGroup Array("VoltageSrc2_1", _ "VoltageSrc2_2"), Array("NAME:VoltageSrc2_1", _ "Voltage:=", "0V", "Faces:=", Array(12, 11))`

## AssignVolumeChargeDensity

**Use:** Creates a volume charge density excitation.

**Command:** Maxwell>Excitations>Assign>Volume Charge Density

**Syntax:** AssignVolumeChargeDensity <VolumeChargeDensityArray>

**Return Value:** None

**Parameters:** <VolumeChargeDensityArray>

`Array("NAME:<BoundName>",`

```
"Value:=", <value>,
"CoordinateSystem :=", <string>,
"Objects:=", <AssignmentObjects>)
```

**Example:** oModule.AssignVolumeChargeDensity Array("NAME:VolumeChargeDenstiy1", "Value:=", "1", \_  
"CoordinateSystem:=", "", "Objects:=", Array("Box1"))

**Example:** oModule.AssignVolumeChargeDensity Array("NAME:x", \_ "Value:=", "y", "Coordin-  
ateSystem:=", "Global", \_ "Objects:=", Array("Box2"))

### AssignWindingGroup

**Use:** Creates a winding group.

**Command:** Maxwell>Excitations>Add Winding

**Syntax:** AssignWindingGroup <WindingGroupArray >

**Return Value:** None

*Parameters:*

<WindingGroupArray>

```
Array("NAME:<BoundName>",
"Type:=", <WindingType>,
"IsSolid:=", <bool>, //true for solid, false for stranded
"Current:=", <value>,
"Resistance:=", <value>,
"Inductance:=", <value>,
"Voltage:=", <value>,
```

)

*Example:*  
oModule.AssignWindingGroup Array("NAME:Winding1", \_ "Type:=", "Voltage", "IsSolid:=", \_false, \_  
"Current:=", "4A", "Resistance:=", "50hm", \_  
"Inductance:=", "6mH", "Voltage:=", "7V")

**Example:** oModule.AssignWindingGroup Array("NAME:Winding2", \_ "Type:=", "Current", "IsSolid:=",  
false, "Current:=", \_ "1A", "Resistance:=", "00hm", "Inductance:=", "0mH", \_ "Voltage:=", "0V")

**Example:** oModule.AssignWindingGroup Array("NAME:Winding3", \_ "Type:=", "External", "IsSolid:=",  
true, "Current:=", \_ "1A", "Resistance:=", "00hm", "Inductance:=", "0mH", \_ "Voltage:=", "0V")

**EditCharge**

**Use:** Edits a charge excitation.

**Command:** Double-click the excitation in the project tree to edit it.

**Syntax:**EditCharge <BoundName> <ChargeArray>

**Return Value:** None

**EditCoilTerminal**

**Use:** Edits a coil terminal excitation.

**Command:** Double-click the excitation in the project tree to edit it.

**Syntax:**EditCoilTerminal <BoundName> <CoilTerminalArray>

**Return Value:** None

**EditCurrent**

Modifies a current source.

|           |                                                                                                                                                                                       |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access | In the Project Manager tree, right click on <b>Excitations &gt; Current</b> or in the Modeler window, right click on a planar entity and select <b>Assign Excitation &gt; Current</b> |
|-----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                     |                          |             |                                                                                                                                                             |
|---------------------|--------------------------|-------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b>   | <b>Name</b>              | <b>Type</b> | <b>Description</b>                                                                                                                                          |
|                     | <NAME>                   | string      | The name of the excitation                                                                                                                                  |
|                     | <Current>                | string      | Current value with unit                                                                                                                                     |
|                     | <IsSolid>                | bool        | True if type is Solid; False if Stranded. Note that this parameter is not available for a 3D Eddy Current A- $\Phi$ design because only Solid is available. |
|                     | <CurrentExcitationModel> | int         | 0 for Single Potential<br>1 for Double Potentials<br>2 for Double Potentials with Ground                                                                    |
|                     | <Point out of terminal>  | bool        | True sets the current direction out of the terminal                                                                                                         |
| <b>Return Value</b> | None                     |             |                                                                                                                                                             |

|                       |                                                                                                                                                                                                                                             |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditCurrent_1 (<NAME>, <Current>, <IsSolid>, <CurrentExcitationModel>, <Point out of terminal>)                                                                                                                                             |
| <b>Python Example</b> | <pre>oModule.EditCurrent("Current7",<br/>[<br/>    "NAME:Current7",<br/>    "Current:="          , "0mA",<br/>    "IsSolid:="          , True,<br/>    "CurrentExcitationModel:=", 1,<br/>    "Point out of terminal:=", False<br/>])</pre> |

|                   |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | EditCurrent <Name>, <CurrentArray>                                                                                                          |
| <b>VB Example</b> | <pre>oModule.EditCurrent "Current7", Array("NAME:Current7",<br/>"Current:=", "0mA", "IsSolid:=", true, "CurrentExcitationModel:=", 1,</pre> |

|  |                                   |
|--|-----------------------------------|
|  | "Point out of terminal:=", false) |
|--|-----------------------------------|

## EditCurrentDensity

**Use:** Edits a current density excitation.

**Command:** Double-click the excitation in the project tree to edit it.

**Syntax:** EditCurrentDensity <BoundName> <CurrentDensityArray>

**Return Value:** None

## EditCurrentDensityTerminal

**Use:** Edits a current density terminal excitation.

**Note:** Double-click the excitation in the project tree to edit it.

**Syntax:** EditCurrentDensityTerminal <BoundName> <CurrentDensityTerminalArray>

**Return Value:** None

## EditExternalCircuit

**Use:** Edits the external circuit for the winding. The SourceNameArray and SourceTypeArray parameters must be of the same size and be a one-to-one match.

**Command:** Maxwell>Excitations>External Circuit>Edit External Circuit

**Syntax:** EditExternalCircuit <fileName> <SourceNameArray> <SourceTypeArray>

**Return Value:** None

**Parameters:** <SourceNameArray> Array(<string>)

<SourceTypeArray> Array(<srcTypeEnum>)<srcTypeEnum>

1- Time dependent 2-Position dependent 3- Speed dependent

**Example:** oModule.EditExternalCircuit "C:\TestProjects\Tdslink\extnlckt00.ckt", Array("VC1", \_  
"VSA1\_0", "VSA1\_1", "VSA1\_2"), Array(1, 2, 1, 3)

EditFloating

*Use:* Edits a floating excitation.

*Command:* Double-click the excitation in the project tree to edit it.

*Syntax:*EditFloating <BoundName> <FloatingArray>

*Return Value:* None

EditSink

*Use:* Edits a sink excitation.

*Command:* Double-click the excitation in the project tree to edit it.

*Syntax:*EditSink <BoundName> <SinkArray>

*Return Value:* None

EditVoltage

Modifies a voltage source.

|            |                                                                         |        |                                                                                                   |
|------------|-------------------------------------------------------------------------|--------|---------------------------------------------------------------------------------------------------|
| UI Access  | Double-click the excitation in the project tree to modify its settings. |        |                                                                                                   |
| Parameters | Name                                                                    | Type   | Description                                                                                       |
|            | <BoundaryName>                                                          | String | Name of the voltage source to be edited.                                                          |
|            | <VoltageArray>                                                          | Array  | Structured array.<br><br>Array ("NAME:<BoundName>",<br>"Voltage:=", <value>,<br><DirectionArray>) |
|            | <DirectionArray>                                                        | Array  | Structured array.                                                                                 |

|                     |       |  |                                                                                                                  |
|---------------------|-------|--|------------------------------------------------------------------------------------------------------------------|
|                     |       |  | <pre> Array("NAME:Direction",       "Start:=", &lt;LineEndPoint&gt;,       "End:=", &lt;LineEndPoint&gt;) </pre> |
| <b>Return Value</b> | None. |  |                                                                                                                  |

|                       |                                                                                                                                                                                                             |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>EditVoltage(&lt;BoundaryName&gt;, &lt;VoltageArray&gt;)</code>                                                                                                                                        |
| <b>Python Example</b> | <pre> oModule.EditVoltage("Voltage1",     ["NAME:Voltage1After",      "Voltage:=", "1000mV",      ["NAME:Direction",       "Start:=", [-0.4, -1.2, 0],       "End:=", [-1.4, -1.2, 0]      ]     ] ) </pre> |

|                   |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EditVoltage &lt;BoundaryName&gt;, &lt;VoltageArray&gt;</code>                                                                         |
| <b>VB Example</b> | <pre> oModule.EditVoltage "Voltage1", _ Array("NAME:Voltage1After", _       "Voltage:=", "1000mV", _       Array("NAME:Direction", _ </pre> |

|  |                                                                                  |
|--|----------------------------------------------------------------------------------|
|  | <pre>"Start:=", Array(-0.4, -1.2, 0), _<br/>"End:=", Array(-1.4, -1.2, 0))</pre> |
|--|----------------------------------------------------------------------------------|

**EditVoltageAPhi**

Edits a voltage excitation for 3D transient A-Phi solutions.

|                     |                                                                        |        |                                                                                          |
|---------------------|------------------------------------------------------------------------|--------|------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Double-click the Voltage Excitations item in the Project tree to edit. |        |                                                                                          |
| <b>Parameters</b>   | Name                                                                   | Type   | Description                                                                              |
|                     | <NAME>                                                                 | string | The name of the voltage excitation                                                       |
|                     | <VoltageAPhi>                                                          | string | Voltage value with unit                                                                  |
|                     | <VoltageAPhiInitialCurrent>                                            | string | Initial current value with unit                                                          |
|                     | <VoltageAPhiHasInitialCurrent>                                         | bool   | True if Initial Current is used.                                                         |
|                     | <VoltageAPhiExcitationModel>                                           | int    | 0 for Single Potential<br>1 for Double Potentials<br>2 for Double Potentials with Ground |
|                     | <VoltageAPhi_Point_out_of_terminal>                                    | bool   | True sets the voltage direction out of the terminal                                      |
| <b>Return Value</b> | None                                                                   |        |                                                                                          |

|                       |                                                                                                                                                                         |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditVoltageAPhi (<NAME>, <VoltageAPhi>, <VoltageAPhiInitialCurrent>, <VoltageAPhiHasInitialCurrent>, <VoltageAPhiExcitationModel>, <VoltageAPhi_Point_out_of_terminal>) |
| <b>Python Example</b> | <pre>oModule.EditVoltageAPhi ("Voltage1",<br/>[</pre>                                                                                                                   |



|  |                                                                                                                                                                                                                        |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "NAME:Voltage1", "VoltageAPhi:=" , "0mV", "VoltageAPhiInitialCurrent:=", "0mA", "VoltageAPhiHasInitialCurrent:=", True, "VoltageAPhiExcitationModel:=", 1, "VoltageAPhi_Point_out_of_terminal:=", True ]) </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                 |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre> oModule.EditVoltageAPhi "Voltage1", Array("NAME:Voltage1", "VoltageAPhi:=", "0mV", "VoltageAPhiInitialCurrent:=", "0mA", "VoltageAPhiHasInitialCurrent:=", false, "VoltageAPhiExcitationModel:=", 0, "VoltageAPhi_Point_out_of_terminal:=", false) </pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## EditVoltageDrop

**Use:** Edits a voltage drop excitation.

**Command:** Double-click the excitation in the project tree to edit it.

**Syntax:** EditVoltageDrop <BoundName> <VoltageDropArray>

**Return Value:** None

## EditVolumeChargeDensity

**Use:** Edits a volume charge density excitation.

**Command:** Double-click the excitation in the project tree to edit it.

**Syntax:** EditVolumeChargeDensity <BoundName> <VolumeChargeDensityArray>

**Return Value:** None

## EditWindingGroup

**Use:** Edits a winding group.

**Command:** Double-click the winding group item in the project tree to edit it.

**Syntax:** EditWindingGroup <BoundName> <WindingGroupArray>

**Return Value:** None

## SetCoreLoss

**Use:** Turns on the core loss effect. Please list all the objects that need to turn on the core loss effect. If an object is not in the list, then it is treated as core loss effect "off". This setting only takes effect if the object has the corresponding core loss definition in the material library.

**Command:** Maxwell>Excitation>Set Core Loss

**Syntax:** SetCoreLoss <ObjectNameArray>

**Return Value:** None

**Parameters:** < ObjectNameArray >

Array("<objectName>"), <bool>, //true to apply core loss effect on field, false means do not apply core loss effect. Default is false.

<objectName> string

**Example:** oModule.SetCoreLoss Array("Box1", "Box2"), true

## SetEddyEffect

**Use:** Sets the eddy effect for an excitation.

**Command:** Maxwell > Excitations > Set Eddy Effects

**Syntax:** SetEddyEffect <EddyEffectSettingArray>

**Return Value:** None

**Parameters:**

<EddyEffectSettingArray>

```

 Array("NAME:Eddy Effect Setting", <EddyEffectVectorArray>)
<EddyEffectVectorArray>
 Array("NAME:EddyEffectVector", Array<EddyEffectDataArray>)
<EddyEffectDataArray>
 Array(("NAME:Data",
 "Object Name:=", <string>,
 "Eddy Effect:=", <bool>,
 "DisplacementCurrent:=", <bool>))

```

**Example:**

```

oModule.SetEddyEffect Array("NAME:Eddy Effect Setting", _
Array("NAME:EddyEffectVector", Array("NAME:Data", _
"Object Name:=", "Box1", "Eddy Effect:=", true, _
"Displacement Current:=", true), Array("NAME:Data", _
"Object Name:=", "Box1_1", "Eddy Effect:=", true, _
"Displacement Current:=", false)))

```

**Example:**

```

oModule.SetEddyEffect Array("NAME:Eddy Effect Setting", _
Array("NAME:EddyEffectVector", Array("NAME:Data", _
"Object Name:=", "Box1", "Eddy Effect:=", true), _
Array("NAME:Data", "Object Name:=", "Box1_1", _
"Eddy Effect:=", false)))

```

**SetMagnetizationCompute**

**Use:** Specifies Magnetization Computation Points settings.

**Command:** Maxwell > Excitations > Set Magnetization Computation

**Syntax:** SetMagnetizationCompute <MagnetizationComputationArray>

### **Return Value:** None

**Parameters:** <MagnetizationComputationArray>

```
Array("NAME:<Magnetization>",
 "IsTarget:=", <bool>,
 "ForDemagComputation:=", <bool>,
 "Project:=", <string>,
 "Design:=", <string>,
 "Soln:=", <string>,
 Array("NAME:Params",)
 "ForceSourceToSolve:=", <bool>,
 "PreservePartnerSoln:=", <bool>,
 "PathRelativeTo:=", <string>,)
```

**Example:** Set oModule = oDesign.GetModule("BoundarySetup")  
oModule.SetMagnetizationCompute Array("NAME:Magnetization", "IsTarget:=", true, "ForDemagComputation:=", true, "Project:=", "test\_magnetization\_link.aedt", "Design:=", "mag\_hystereis\_only\_source", "Soln:=", "Setup1 : Transient", Array("NAME:Params", "\$Temp:=", "20cel", "Cur:=", "0A"), "ForceSourceToSolve:=", false, "PreservePartnerSoln:=", false, "PathRelativeTo:=", "TargetProject")

### **Other BoundarySetup Module Script Commands**

Following are additional script commands that are recognized by the **BoundarySetup** module:

**Note:**

In the following commands, all named data can be included/excluded as desired and may appear in any order.

- [AssignWindingGroup](#)
- [EditWindingGroup](#)
- [AddTerminalsToWinding](#)
- [EditExternalCircuit](#)
- [SetCoreLoss](#)
- [SetEddyEffect](#)
- [SetMinimumTimeStep](#)

**SetMinimumTimeStep**

**Use:** Sets the minimum time step for an external circuit excitation.

**Command:** Maxwell>Excitations>External Circuit>Set Minimum Time Step

**Syntax:** SetMinimumTimeStep <value>

**Return Value:** None

**Parameters:** None

**Example:** oModule.SetMinimumTimeStep "1ps"

**ShowWindow**

**Use:** Sets up a Y connection among voltage windings. It will connect their negative voltage terminals to a common node of the circuit. Only one Y Connection is allowed per design.

**Syntax:** SetupYConnection <YConnectionArray>

**Return Value:** None

**Parameters:** <YConnectionArray>

Type: Array of strings

When the *YConnectionArray* is empty, it is used to delete the Y Connection.

*Array(<YConnection>)*

*<YConnection>*

Type: Array of strings

*Array("Name:YConnection", "Windings:=", <WindingNames>)*

*<WindingNames>*

Type: String

A list of winding names separated by commas

***VB Example:***

Add Y Connection:

```
oModule.SetupYConnection Array(Array("NAME:YConnection", "Windings:=", "Wind-
ing1,Winding2,Winding3,Winding4"))
```

Delete Y Connection:

```
oModule.SetupYConnection
```

| Python Syntax  | SetupYConnection(<YConnectionArray>)                               |
|----------------|--------------------------------------------------------------------|
| Python Example | <u>Add Y Connection</u><br><br>oModule.SetupYConnection(<br>[<br>[ |

|  |                                                                                                                                             |
|--|---------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "NAME:YConnection", "Windings:=" , "Winding1,Winding2,Winding3,Winding4"  ] ])  Delete Y Connection oModule.SetupYConnection() </pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------|

## SetPermeabilityComputation

**Use:** Enables a nonlinear analysis that will calculate permeability data, which will then be available for [3D Eddy Current T-Ω design simulation](#).

**Command:** In the Project Manager tree, right click **Excitations > Set Permeability Computation**

**Syntax:** SetPermeabilityComputation <ObjArray>

**Return Value:** None

**Parameters:** <ObjectArray>

List of objects with nonlinear BH curve material

### Example:

```

oModule = oDesign.GetModule("BoundarySetup")
oModule.SetPermeabilityComputation(
[
"NAME:MuLinkSrcObjs",
"Objs:=" , ["Core_1st","Core_2nd","Magnet"]

```

] )



## 14 - Mesh Operations Module Script Commands

Commands for mesh setup and operations should be executed by the "MeshSetup" module.

```
Set oModule = oDesign.GetModule("MeshSetup")
```

```
oModule.CommandName <args>
```

### Conventions Used in this Chapter

<OpName>

Type: <string>

Name of a mesh operation.

<AssignmentObjects>

**Type: Array of strings**

An array of object names.

<AssignmentFaces>

Type: Array of integers.

An array of face IDs. The ID of a face can be determined through the user interface using the **3DModeler>Measure>Area** command. The face ID is given in the **Measure Information** dialog box.

**The topics for this section include:**

[General Commands Recognized by the Mesh Operations Module](#)

[Script Commands for Creating and Modifying Mesh Operations](#)

# General Commands Recognized by the Mesh Operations Module

General commands recognized by the Mesh Operations Module:

[DeleteOp](#)

[GetMeshOpAssignment](#)

[GetOperationName](#)

[RenameOp](#)

## DeleteOp

Deletes the specified mesh operations.

|              |                                               |       |                                |
|--------------|-----------------------------------------------|-------|--------------------------------|
| UI Access    | Delete command in the <b>List</b> dialog box. |       |                                |
| Parameters   | Name                                          | Type  | Description                    |
|              | <NameArray>                                   | Array | Array of mesh operation names. |
| Return Value | None.                                         |       |                                |

|                |                                                        |
|----------------|--------------------------------------------------------|
| Python Syntax  | DeleteOp(<NameArray>)                                  |
| Python Example | oModule.DeleteOp(["Length1", "SkinDepth1", "Length2"]) |

|            |                                                   |
|------------|---------------------------------------------------|
| VB Syntax  | DeleteOp <NameArray>                              |
| VB Example | oModule.DeleteOp Array("Length1", "SkinDepth1", _ |

|  |            |
|--|------------|
|  | "Length2") |
|--|------------|

## GetOperationNames

Gets the names of mesh operations defined in a design.

|                     |                                                   |        |                           |
|---------------------|---------------------------------------------------|--------|---------------------------|
| <b>UI Access</b>    | N/A                                               |        |                           |
| <b>Parameters</b>   | Name                                              | Type   | Description               |
|                     | <OperationType>                                   | String | Specified operation type. |
| <b>Return Value</b> | Array of strings containing mesh operation names. |        |                           |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | GetOperationNames(<OperationType>)        |
| <b>Python Example</b> | oModule.GetOperationNames("Length Based") |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | GetOperationNames <OperationType>        |
| <b>VB Example</b> | oModule.GetOperationNames "Length Based" |

## RenameOp

Renames a mesh operation.

|                   |                                                                                                        |        |                                  |
|-------------------|--------------------------------------------------------------------------------------------------------|--------|----------------------------------|
| <b>UI Access</b>  | Right-click the mesh operation in the project tree, and then click <b>Rename</b> on the shortcut menu. |        |                                  |
| <b>Parameters</b> | Name                                                                                                   | Type   | Description                      |
|                   | <OldName>                                                                                              | String | Old name for the mesh operation. |
|                   | <NewName>                                                                                              | String | New name for the mesh operation. |

|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameOp(&lt;OldName&gt;, &lt;NewName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.RenameOp("SkinDepth1", "NewName")</code>  |

|                   |                                                        |
|-------------------|--------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameOp &lt;OldName&gt;, &lt;NewName&gt;</code> |
| <b>VB Example</b> | <code>oModule.RenameOp "SkinDepth1", "NewName"</code>  |

## Script Commands for Creating and Modifying Mesh Operations

Script commands for creating and modifying mesh operations are as follows:

"AssignCylindricalGapOp " on the facing page

"AssignCylindricalGapOp " on the facing page

[AssignLengthOp](#)

[AssignModelResolutionOp](#)

"AssignSkinDepthLayerSetting " on page 14-17

[AssignSkinDepthOp](#)

[AssignTrueSurfOp](#)

"EditCylindricalGapOp " on page 14-22

"EditDensityControlOp" on page 14-24

[EditEdgeCutLayerOp](#)[EditLengthOp](#)[EditModelResolutionOp](#)[EditSkinDepthOp](#)[EditTrueSurfOp](#)

"InitialMeshSettings " on page 14-35

## AssignCylindricalGapOp

### 3D Designs

For 3D designs, assigns a cylindrical gap 3D mesh operation to enable use of Clone Mesh and associated Band Mapping Angle. For Maxwell 2D designs, [see below](#).

=====

Assigns a cylindrical gap 3D mesh operation to enable use of Clone Mesh and associated Band Mapping Angle.

| UI Access  | Maxwell3D > Mesh > Assign Mesh Operation > Cylindrical Gap Treatment |       |                                                                                                                                                                                                                                                             |
|------------|----------------------------------------------------------------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                 | Type  | Description                                                                                                                                                                                                                                                 |
|            | <CylindricalGapOpParams>                                             | Array | Structured array.<br><br>Array("NAME:<OpName>",<br>"CloneMesh:=", <bool>,<br>"BandMappingAngle:=", <value> <i>[use if CloneMesh is "true"]</i><br>"MovingSideLayers:=", <integer>,<br>"StaticSideLayers:=", <integer>,<br>"Objects:=", <AssignmentObjects>) |

|              |       |
|--------------|-------|
| Return Value | None. |
|--------------|-------|

|                |                                                                                                                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AssignCylindricalGapOp(<CylindricalGapOpParams>)                                                                                                                                                                         |
| Python Example | <pre>oModule.AssignCylindricalGapOp ([     "NAME:CylindricalGap1",     "CloneMesh:=", True,     "BandMappingAngle:=", 3deg,     "MovingSideLayers:=", 1,     "StaticSideLayers:=", 3,     "Objects:=", ["Box2"] ])</pre> |

|            |                                                                                                                                                                         |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | AssignCylindricalGapOp <CylindricalGapOpParams>                                                                                                                         |
| VB Example | <pre>oModule.AssignCylindricalGapOp _ Array("NAME:CylindricalGap1", _     "CloneMesh:=", true, _     "BandMappingAngle:=", 3deg, _     "MovingSideLayers:=", 1, _</pre> |

|  |                                                                                                                                                                                          |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "StaticSideLayers:=", 3, _ "Objects:=", Array("Box2")) oModule.AssignCylindricalGapOp _ Array("NAME:CylindricalGap1", _ "CloneMesh:=", false, _ "Objects:=", Array("Box2")) </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## Maxwell 2D Designs

For Maxwell 2D Designs, assigns a cylindrical gap 2D mesh operation.

| UI Access    | Maxwell 2D > Mesh > Assign Mesh Operation > Cylindrical Gap Treatment |       |                                                                                                                                                                                                                                                                                 |
|--------------|-----------------------------------------------------------------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                  | Type  | Description                                                                                                                                                                                                                                                                     |
|              | <CylindricalGapOpParams>                                              | Array | Structured array.<br><br><pre> Array("NAME:&lt;OpName&gt;", "UseBandMappingAngle:=" , &lt;bool&gt;, "BandMappingAngle:=", &lt;value&gt; <i>[use if UseBandMappingAngle is "true". Range is 0.0005deg through 3deg inclusive]</i> "Objects:=", &lt;AssignmentObjects&gt;) </pre> |
| Return Value | None.                                                                 |       |                                                                                                                                                                                                                                                                                 |

|                |                                                  |
|----------------|--------------------------------------------------|
| Python Syntax  | AssignCylindricalGapOp(<CylindricalGapOpParams>) |
| Python Example | oModule.AssignCylindricalGapOp                   |

|  |                                                                                                                                                                    |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>( [<br/>  "NAME:CylindricalGap1",<br/>  "UseBandMappingAngle:=" , True,<br/>  "BandMappingAngle:="      , "3deg",<br/>  "Objects:=", ["Circle2"]<br/>])</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | AssignCylindricalGapOp <CylindricalGapOpParams>                                                                                                                                                  |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AssignCylindricalGapOp<br/>Array("NAME:CylindricalGap1",<br/>  "UseBandMappingAngle:=" , True,<br/>  "BandMappingAngle:="      , "3deg",<br/>  "Objects:=", Array("Circle2"))</pre> |

AssignDensityControlOp

Assigns a density control mesh operation to specify refinement of clone mesh.

| UI Access  | Maxwell3D > Mesh > Assign Mesh Operation > Inside Selection > Clone Mesh Density |       |                                                 |
|------------|----------------------------------------------------------------------------------|-------|-------------------------------------------------|
| Parameters | Name                                                                             | Type  | Description                                     |
|            | <DensityControlParams>                                                           | Array | Structured array.<br><br>Array("NAME:<OpName>", |



|                     |       |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|---------------------|-------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |       |  | <pre>"RefineInside:=", &lt;bool&gt;, "Objects:=", &lt;AssignmentObjects&gt;, "RestrictMaxElemLength:=", &lt;bool&gt;, "MaxElemLength:=", &lt;value&gt;, "RestrictLayersNum:=", &lt;bool&gt;, "LayersNum:=", &lt;integer&gt;)</pre> <p>RefineInside</p> <p>If true, Objects should be specified. Implies apply restrictions to tetrahedra inside the object.</p> <p>If false, Faces and/or Objects can be specified. Implies apply restrictions to triangles on the surface of the face or object.</p> <p>RestrictMaxElemLength</p> <p>If true, MaxElemLength should be specified.</p> <p>RestrictLayersNum</p> <p>If true, LayersNum should be specified.</p> |
| <b>Return Value</b> | None. |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | AssignDensityControlOp(<DensityControlParams>) |
| <b>Python Example</b> | <pre>oModule.AssignDensityControlOp( [</pre>   |

|  |                                                                                                                                                                                                                                                                                                      |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"NAME:Clone Mesh Density3",<br/>"RefineInside:="          , True,<br/>"Objects:="                , ["Inner_arm"],<br/>"RestrictMaxElemLength:=", True,<br/>"MaxElemLength:="         , "0.015mm",<br/>"RestrictLayersNum:="     , True,<br/>"LayersNum:="              , "15"<br/><br/>])</pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | AssignDensityControlOp <DensityControlParams>                                                                                                                                                                                                                                                   |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AssignDensityControlOp _<br/>Array("NAME:Clone Mesh Density3", _<br/>"RefineInside:=", true, _<br/>"Objects:=", Array("Inner_arm"), _<br/>"RestrictMaxElemLength:=", true, _<br/>"MaxElemLength:=", "0.015mm", _<br/>"RestrictLayersNum:=", true, _<br/>"LayersNum:=", "15")</pre> |

## AssignEdgeCutLayerOp

Assigns the edge cut mesh operation to specify refinement of layer mesh.

| UI Access    | Maxwell 3D > Mesh > Assign Mesh Operation > Inside Selection > Edge Cut Based |       |                                                                                                                       |
|--------------|-------------------------------------------------------------------------------|-------|-----------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                          | Type  | Description                                                                                                           |
|              | <EdgeCutLayerOpParams>                                                        | Array | Structured array.<br><br>Array("NAME:<OpName>",<br>"Objects:=", <AssignmentObjects>,<br>"Layer Thickness:=", <value>) |
| Return Value | None.                                                                         |       |                                                                                                                       |

| Python Syntax  | AssignEdgeCutLayerOp(<EdgeCutLayerOpParams>)                                                                                                                                                          |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.AssignEdgeCutLayerOp(<br/>    [<br/>        "NAME:EdgeCut1",<br/>        "Objects:="                , ["Rotor"],<br/>        "Layer Thickness:="        , "0.55mm"<br/>    ]<br/>)</pre> |

| VB Syntax  | AssignEdgeCutLayerOp <EdgeCutLayerOpParams> |
|------------|---------------------------------------------|
| VB Example | <pre>oModule.AssignEdgeCutLayerOp _</pre>   |

|  |                                                                                                           |
|--|-----------------------------------------------------------------------------------------------------------|
|  | <pre>Array("NAME:EdgeCut1", _<br/>"Objects:=", Array("Rotor"), _<br/>"Layer Thickness:=", "0.55mm")</pre> |
|--|-----------------------------------------------------------------------------------------------------------|

### AssignLengthOp

Assigns length-based operations to the selection.

| UI Access  | Maxwell 3D > Mesh > Assign Mesh Operation > On Selection   Inside Selection > Length Based |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------|--------------------------------------------------------------------------------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                                       | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|            | <LengthOpParams>                                                                           | Array | <p>Structured array.</p> <pre>Array("NAME:&lt;OpName&gt;",<br/>"RefineInside:=", &lt;bool&gt;,<br/>"Objects:=", &lt;AssignmentObjects&gt;,<br/>"Faces:=", &lt;AssignmentFaces&gt;,<br/>"Edges:=", &lt;AssignmentEdges&gt;,<br/>"RestrictElem:=", &lt;bool&gt;<br/>"NumMaxElem:=", &lt;integer&gt;<br/>"RestrictLength:=", &lt;bool&gt;<br/>"MaxLength:=", &lt;value&gt;<br/>"ApplyToInitialMesh:=", &lt;bool&gt;)</pre> <p>RefineInside</p> <p>If true, Objects should be specified. Implies apply restrictions to tet-</p> |

|                     |       |  |                                                                                                                                                                                                                                                                                                                                       |
|---------------------|-------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |       |  | <p>rahedra inside the object.</p> <p>If false, Edges, Faces and/or Objects can be specified. Implies apply restrictions to triangles on the surface of the face or object.</p> <p>RestrictElem</p> <p><i>If true</i>, NumMaxElem should be specified.</p> <p>RestrictLength</p> <p><i>If true</i>, MaxLength should be specified.</p> |
| <b>Return Value</b> | None. |  |                                                                                                                                                                                                                                                                                                                                       |

| Python Syntax         | AssignLengthOp(<LengthOpParams>)                                                                                                                                                                                                                                 |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.AssignLengthOp ([     "NAME:Length1",     "RefineInside:=", True,     "Objects:=", ["Box1"],     "RestrictElem:=", True,     "NumMaxElem:=", 1000,     "RestrictLength:=", True,     "MaxLength:=", "1mm"     "ApplyToInitialMesh:=", True ])</pre> |

|  |                                                                                                                                                                                                                                                |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>oModule.AssignLengthOp ([   "NAME:Length1",   "RefineInside:=", False,   "Edges:=", [58,73],   "RestrictElem:=", True,   "NumMaxElem:=", 1000,   "RestrictLength:=", True,   "MaxLength:=", "1mm"   "ApplyToInitialMesh:=", True ])</pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | AssignLengthOp <LengthOpParams>                                                                                                                                     |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AssignLengthOp Array("NAME:Length1", _   "RefineInside:=", true,   "Objects:=", Array("Box1"),   "RestrictElem:=", true,   "NumMaxElem:=", 1000,</pre> |

|  |                                                                                                                                                                                                                                                                                                                                                |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "RestrictLength:=", true, "MaxLength:=", "1mm") "ApplyToInitialMesh:=", True  oModule.AssignLengthOp Array("NAME:Length1", _     "RefineInside:=", false,     "Edges:=", Array(58,73),     "RestrictElem:=", true,     "NumMaxElem:=", 1000,     "RestrictLength:=", true,     "MaxLength:=", "1mm") "ApplyToInitialMesh:=", True </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## AssignModelResolutionOp

Assigns a model resolution name, value and unit for mesh operations, or specify to UseAutoFeaturelength. If UseAutoFeature length is true, the Defeature length is not used.

| UI Access  | Maxwell 3D > Mesh > Assign Mesh Operation > Model Resolution |       |                                                                                                                                                                  |
|------------|--------------------------------------------------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                         | Type  | Description                                                                                                                                                      |
|            | <ModelResParams>                                             | Array | Structured array.<br><br><pre> Array("NAME:&lt;ModelResName&gt;",     "Objects:=", &lt;array of object names&gt;,     "UseAutoLength:=", &lt;boolean&gt;, </pre> |

|              |                                                                                                       |                                        |  |  |  |                                        |
|--------------|-------------------------------------------------------------------------------------------------------|----------------------------------------|--|--|--|----------------------------------------|
|              | <table><tr><td></td><td></td><td>"DefeatureLength:=", "&lt;value&gt;&lt;units&gt;")</td></tr></table> |                                        |  |  |  | "DefeatureLength:=", "<value><units>") |
|              |                                                                                                       | "DefeatureLength:=", "<value><units>") |  |  |  |                                        |
| Return Value | None.                                                                                                 |                                        |  |  |  |                                        |

|                       |                                                                                                                                                                                           |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>AssignModelResolutionOp(&lt;ModelResParams&gt;)</b>                                                                                                                                    |
| <b>Python Example</b> | <pre>oModule.AssignModelResolutionOp ([     "NAME:ModelResolution1",     "Objects:=", ["waveguide"],     "UseAutoLength:=", True,     "DefeatureLength:=", "71.5891053163818mil" ])</pre> |

|                   |                                                                                                                                                                                          |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <b>AssignModelResolutionOp &lt;ModelResParams&gt;</b>                                                                                                                                    |
| <b>VB Example</b> | <pre>oModule.AssignModelResolutionOp _ Array("NAME:ModelResolution1", _ "Objects:=", Array( "waveguide"), _ "UseAutoLength:=", true, _ "DefeatureLength:=", "71.5891053163818mil")</pre> |



## AssignSkinDepthLayerSetting

**Use:** Assigns a skin-depth layer settings to the selected edges.

**Command:** Click **Mesh > Assign Mesh Operation > On Selection > Skin Depth Based**

**Syntax:** AssignSkinDepthLayerSetting Array("NAME:<LayerSettingName>", "Enabled:=", <SettingEnabled>, "Edges:=", Array(<EdgeIDArray>), "SkinDepth:=", "<SkinDepthValue>", "NumLayers:=", "<LayerNumber>")

**Return Value:** None

**Parameters:** <LayerSettingName>

The name of the skin depth layer setting.

<SettingEnabled>

Boolean: "Enabled:=", true

<EdgeIDArray>

Array of edge IDs. "Edges:=", Array(10, 12, 13)

<SkinDepthValue>

Skin depth value with unit. "SkinDepth:=", "1mm"

<LayerNumber>

Integer, the number of layers. "NumLayers:=", "2"

**Example:** oModule.AssignSkinDepthLayerSetting Array("NAME:SkinDepthLayer1",  
"Enabled:=", true  
"Edges:=", Array(10, 12, 13),  
"SkinDepth:=", "1mm",  
"NumLayers:=", 2)

### AssignSkinDepthOp

Assigns a skin-depth based operations to the selection.

| UI Access    | Maxwell 3D > Mesh > Assign Mesh Operation > On Selection > Skin Depth Based |       |                                                                                                                                                                                                                                                                                                          |
|--------------|-----------------------------------------------------------------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                        | Type  | Description                                                                                                                                                                                                                                                                                              |
|              | <SkinDepthOpParams>                                                         | Array | Structured array.<br><br>Array("NAME:<OpName>",<br>"Faces:=", <AssignmentFaces>,<br>"RestrictElem:=", <bool>,<br>"NumMaxElem:=", <int>,<br>"SkinDepth:=", <value>,<br>"SurfTriMaxLength:=", <value>,<br>"NumLayers:=", <int>)<br><br>RestrictElem<br><br><i>If true, NumMaxElem should be specified.</i> |
| Return Value | None.                                                                       |       |                                                                                                                                                                                                                                                                                                          |

| Python Syntax  | AssignSkinDepthOp(<SkinDepthOpParams>) |
|----------------|----------------------------------------|
| Python Example | oModule.AssignSkinDepthOp<br><br>( [   |

|  |                                                                                                                                                                     |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "NAME:SkinDepth1", "Faces:=", [7], "RestrictElem:=", True, "NumMaxElem:=", 1000, "SkinDepth:=", "1mm", "SurfTriMaxLength:=", "1mm", "NumLayers:=", 2 ] </pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | AssignSkinDepthOp <SkinDepthOpParams>                                                                                                                                                                                                       |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.AssignSkinDepthOp Array("NAME:SkinDepth1", _     "Faces:=", Array(7), _     "RestrictElem:=", true, _     "NumMaxElem:=", 1000, _     "SkinDepth:=", "1mm", _     "SurfTriMaxLength:=", "1mm", _     "NumLayers:=", 2) </pre> |

## AssignTrueSurfOp

Assigns a true surface-based mesh operation on the selection.

|           |                                                                   |
|-----------|-------------------------------------------------------------------|
| UI Access | Maxwell 3D > Mesh > Assign Mesh Operation > Surface Approximation |
|-----------|-------------------------------------------------------------------|

|              |                    |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|--------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name               | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|              | <TrueSurfOpParams> | Array | <p>Structured array.</p> <pre>Array("NAME:&lt;OpName&gt;",<br/>      "Faces:=", &lt;AssignmentFaces&gt;,<br/>      "SurfDevChoice:=", &lt;RadioOption&gt;,<br/>      "SurfDev:=", &lt;value&gt;,<br/>      "NormalDevChoice:=", &lt;RadioOption&gt;,<br/>      "NormalDev:=", &lt;value&gt;,<br/>      "AspectRatioChoice:=", &lt;RadioOption&gt;,<br/>      "AspectRatio:=", &lt;double&gt;)</pre> <p>&lt;RadioOption&gt;</p> <p><i>Type: &lt;int&gt;</i></p> <p>0: Ignore</p> <p>1: Use defaults</p> <p>2: Specify the value</p> |
| Return Value | None.              |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|               |                                      |
|---------------|--------------------------------------|
| Python Syntax | AssignTrueSurfOp(<TrueSurfOpParams>) |
|---------------|--------------------------------------|

|                       |                                                                                                                                                                                                                                                                                 |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.AssignTrueSurfOp<br/><br/>([<br/>    "NAME:TrueSurf1",<br/>    "Faces:=", [9],<br/>    "SurfDevChoice:=", 2,<br/>    "SurfDev:=", "0.04123105626mm",<br/>    "NormalDevChoice:=", 2,<br/>    "NormalDev:=", "15deg",<br/>    "AspectRatioChoice:=", 1<br/>])</pre> |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| <b>VB Syntax</b>  | <b>AssignTrueSurfOp &lt;TrueSurfOpParams&gt;</b>                                                                                                                                                                                                                           |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.AssignTrueSurfOp Array("NAME:TrueSurf1",<br/>    "Faces:=", Array(9), _<br/>    "SurfDevChoice:=", 2, _<br/>    "SurfDev:=", "0.04123105626mm", _<br/>    "NormalDevChoice:=", 2, _<br/>    "NormalDev:=", "15deg", _<br/>    "AspectRatioChoice:=", 1)</pre> |

EditCylindricalGapOp

Modifies an existing cylindrical gap 3D mesh operation to enable/disable use of Clone Mesh and associated Band Mapping Angle.

Modifies an existing cylindrical gap 2D mesh operation to enable/disable use of Band Mapping Angle.

|              |                                                                        |        |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|--------------|------------------------------------------------------------------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Double-click the operation in the project tree to modify its settings. |        |                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| Parameters   | Name                                                                   | Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                                     |
|              | <OpName>                                                               | String | Name of the operation to be edited.                                                                                                                                                                                                                                                                                                                                                                                             |
|              | <CylindricalGapOpParams>                                               | Array  | Structured array for 3D mesh.<br><br>Array ("NAME:<OpName>",<br>"CloneMesh:=", <bool>,<br>"BandMappingAngle:=", <value> <i>[use if CloneMesh is "true"]</i><br>"MovingSideLayers:=", <integer>,<br>"StaticSideLayers:=", <integer>)<br><br>Structured array for 2D mesh.<br><br>Array ("NAME:<OpName>",<br>"UseBandMappingAngle:=", <bool>,<br>"BandMappingAngle:=", <real>) <i>[range is 0.0005deg through 3deg inclusive]</i> |
| Return Value | None                                                                   |        |                                                                                                                                                                                                                                                                                                                                                                                                                                 |

| Python Syntax  | EditCylindricalGapOp(<OpName>, <CylindricalGapOpParams>)                                                                                                                                                                                                                                                                                                                                                        |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditCylindricalGapOp("CylindricalGap1",     ["NAME:CylindricalGap1",     "CloneMesh:=", True,     "BandMappingAngle:=", 3deg,     "MovingSideLayers:=", 1,     "StaticSideLayers:=", 3] )</pre> <p><b>Maxwell 2D Example</b></p> <pre>oModule.EditCylindricalGapOp("CylindricalGap1",     ["NAME:CylindricalGap1",     "UseBandMappingAngle:=", True,     "BandMappingAngle:=", 3deg,     ])</pre> |

| VB Syntax  | EditCylindricalGapOp <OpName>, <CylindricalGapOpParams>                                                                                                                            |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditCylindricalGapOp "CylindricalGap1",     Array("NAME:CylindricalGap1",     "CloneMesh:=", true,     "BandMappingAngle:=", 3deg,     "MovingSideLayers:=", 1,</pre> |

|  |                                                                                                                                                                                                                                |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"StaticSideLayers:=", 3)  <b>Maxwell 2D Example</b>  oModule.EditCylindricalGapOp "CylindricalGap1", _     Array("NAME:CylindricalGap1",         "UseBandMappingAngle:=", true,         "BandMappingAngle:=", 3deg)</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**EditDensityControlOp**

Edits an existing density control operation. This cannot be used to modify assignments. Instead, the mesh operation should be deleted and a new one created.

| UI Access  | Double-click the operation in the project tree to modify its settings. |        |                                                                                                                                                                                                                                                           |
|------------|------------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                   | Type   | Description                                                                                                                                                                                                                                               |
|            | <OpName>                                                               | String | Name of the operation to be edited.                                                                                                                                                                                                                       |
|            | <DensityControlParams>                                                 | Array  | Structured array.                                                                                                                                                                                                                                         |
|            |                                                                        |        | <pre>Array("NAME:&lt;OpName&gt;",     "RefineInside:=", &lt;bool&gt;,     "RestrictMaxElemLength:=", &lt;bool&gt;,     "MaxElemLength:=", &lt;value&gt;,     "RestrictLayersNum:=", &lt;bool&gt;,     "LayersNum:=", &lt;integer&gt;)  RefineInside</pre> |



|                     |       |  |                                                                                                                                                                                                                                                                                                                                                                                                        |
|---------------------|-------|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |       |  | <p>If true, Objects should be specified. Implies apply restrictions to tetrahedra inside the object.</p> <p>If false, Faces and/or Objects can be specified. Implies apply restrictions to triangles on the surface of the face or object.</p> <p>RestrictMaxElemLength</p> <p>If true, MaxElemLength should be specified.</p> <p>RestrictLayersNum</p> <p>If true, LayersNum should be specified.</p> |
| <b>Return Value</b> | None. |  |                                                                                                                                                                                                                                                                                                                                                                                                        |

| Python Syntax         | EditDensityControlOp(<OpName>, <DensityControlOpParams>)                                                                                                                                                                                                                   |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.EditDensityControlOp("Clone Mesh Density1", [ "NAME:Clone Mesh Density1", "RefineInside:="          , True, "RestrictMaxElemLength:=", True, "MaxElemLength:="        , "0.02mm", "RestrictLayersNum:="     , True, "LayersNum:="             , "15" ])</pre> |

|                   |                                                                                                                                                                                                                                                                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EditDensityControlOp &lt;OpName&gt;, &lt;DensityControlOpParams&gt;</code>                                                                                                                                                                                              |
| <b>VB Example</b> | <pre>oModule.EditDensityControlOp "Clone Mesh Density1", _<br/>Array("NAME:Clone Mesh Density1", _<br/>"RefineInside:=", true, _<br/>"RestrictMaxElemLength:=", true, _<br/>"MaxElemLength:=", "0.015mm", _<br/>"RestrictLayersNum:=", true, _<br/>"LayersNum:=", "15")</pre> |

## EditEdgeCutLayerOp

Edits an existing edge cut operation. This cannot be used to modify assignments. Instead, the mesh operation should be deleted and a new one created.

|                     |                                                                        |        |                                                                                  |
|---------------------|------------------------------------------------------------------------|--------|----------------------------------------------------------------------------------|
| <b>UI Access</b>    | Double-click the operation in the project tree to modify its settings. |        |                                                                                  |
| <b>Parameters</b>   | Name                                                                   | Type   | Description                                                                      |
|                     | <OpName>                                                               | String | Name of the operation to be edited.                                              |
|                     | <EdgeCutLayerOpParams>                                                 | Array  | Structured array.<br><br>Array("NAME:<OpName>",<br>"Layer Thickness:=", <value>) |
| <b>Return Value</b> | None.                                                                  |        |                                                                                  |

| Python Syntax  | EditEdgeCutLayerOp(<OpName>, <EdgeCutLayerOpParams>)                                                        |
|----------------|-------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditEdgeCutLayerOp("EdgeCut1", [ "NAME:EdgeCut1", "Layer Thickness:="      , "0.66mm" ])</pre> |

| VB Syntax  | EditEdgeCutLayerOp <OpName>, <EdgeCutLayerOpParams>                                                         |
|------------|-------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditEdgeCutLayerOp "EdgeCut1", _ Array("NAME:EdgeCut1", _ "Layer Thickness:=", "0.66mm")</pre> |

## EditLengthOp

Edits an existing length-based operation. This cannot be used to modify assignments. Instead, the mesh operation should be deleted and a new one created.

| UI Access  | Double-click the operation in the project tree to modify its settings. |        |                                                                                                                 |
|------------|------------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                   | Type   | Description                                                                                                     |
|            | <OpName>                                                               | String | Name of the operation to be edited.                                                                             |
|            | <LengthOpParams>                                                       | Array  | Structured array.                                                                                               |
|            |                                                                        |        | <pre>Array("NAME:&lt;OpName&gt;", "RefineInside:=", &lt;bool&gt;, "Objects:=", &lt;AssignmentObjects&gt;,</pre> |

|              |       |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|--------------|-------|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              |       |  | <pre>"Faces:=", &lt;AssignmentFaces&gt;, "Edges:=", &lt;AssignmentEdges&gt;, "RestrictElem:=", &lt;bool&gt;, "NumMaxElem:=", &lt;integer&gt;, "RestrictLength:=", &lt;bool&gt;, "MaxLength:=", &lt;value&gt; "ApplyToInitialMesh:=", &lt;bool&gt;)</pre> <p>RefineInside</p> <p>If true, Objects should be specified. Implies apply restrictions to tetrahedra inside the object.</p> <p>If false, Edges, Faces and/or Objects can be specified. Implies apply restrictions to triangles on the surface of the face or object.</p> <p>RestrictElem</p> <p>If true, NumMaxElem should be specified.</p> <p>RestrictLength</p> <p>If true, MaxLength should be specified.</p> |
| Return Value | None. |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |

|                |                                          |
|----------------|------------------------------------------|
| Python Syntax  | EditLengthOp(<OpName>, <LengthOpParams>) |
| Python Example | oModule.EditLengthOpK("Length1",         |

|  |                                                                                                                                                                                                                                                                                 |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>[<br/>  "NAME:Length1",<br/>  "RefineInside:=", True,<br/>  "Objects:=", ["Circle2", "Rectangle1"],<br/>  "RestrictElem:=", False,<br/>  "NumMaxElem:=", 1000,<br/>  "RestrictLength:=", True,<br/>  "MaxLength:=", "2mm",<br/>  "ApplyToInitialMesh:=", False<br/>]</pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | EditLengthOp <OpName>, <LengthOpParams>                                                                                                                                                                                                                                                         |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditLengthOpK "Length1", Array("NAME:Length1", _<br/>  "RefineInside:=", false,<br/>  "Objects:=", Array("Box1"),<br/>  "RestrictElem:=", false,<br/>  "NumMaxElem:=", 1000,<br/>  "RestrictLength:=", true,<br/>  "MaxLength:=", "2mm"<br/>  "ApplyToInitialMesh:=", False)</pre> |

## EditModelResolutionOp

Assigns a model resolution name, value and unit for mesh operations. If UseAutoLength is true, the Defeature length is not used.

|              |                                                                        |        |                                                                                                                                   |
|--------------|------------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Double-click the operation in the project tree to modify its settings. |        |                                                                                                                                   |
| Parameters   | Name                                                                   | Type   | Description                                                                                                                       |
|              | <OpName>                                                               | String | Name of the operation to be edited.                                                                                               |
|              | <ModelResParams>                                                       | Array  | Structured array.<br><br>Array ("NAME:<ModelResName>",<br>"UseAutoLength:=", <boolean>,<br>"DefeatureLength:=", "<value><units>") |
| Return Value | None.                                                                  |        |                                                                                                                                   |

|                |                                                                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EditModelResolutionOp(<OpName>, <ModelResParams>                                                                                                                                                |
| Python Example | <pre>oModule.EditModelResolutionOp("ModelResolution1",<br/>[<br/>    "NAME:ModelResolution1",<br/>    "UseAutoLength:=", False,<br/>    "DefeatureLength:=", "71.5891053163818mil"<br/>])</pre> |

|                   |                                                                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EditModelResolutionOp &lt;OpName&gt;, &lt;ModelResParams&gt;</code>                                                                                             |
| <b>VB Example</b> | <pre>oModule.EditModelResolutionOp "ModelResolution1", _ Array("NAME:ModelResolution1", "UseAutoLength:=", false, _ "DefeatureLength:=", "71.5891053163818mil")</pre> |

## EditSkinDepthOp

Modifies an existing skin-depth based mesh operation. Assignments cannot be changed using this command. To change the assignment, you must delete operation and create it using a new assignment.

|                     |                                                                        |             |                                                                                                                                                                                                                                                                                                                                |
|---------------------|------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Double-click the operation in the project tree to modify its settings. |             |                                                                                                                                                                                                                                                                                                                                |
| <b>Parameters</b>   | <b>Name</b>                                                            | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                             |
|                     | <OpName>                                                               | String      | Name of the operation to be edited.                                                                                                                                                                                                                                                                                            |
|                     | <SkinDepthOpParams>                                                    | Array       | Structured array.<br><br><pre>Array("NAME:&lt;OpName&gt;",       "RestrictElem:=", &lt;bool&gt;,       "NumMaxElem:=", &lt;int&gt;,       "SkinDepth:=", &lt;value&gt;,       "SurfTriMaxLength:=", &lt;value&gt;,       "NumLayers:=", &lt;int&gt;)</pre> RestrictElem<br><br><i>If true, NumMaxElem should be specified.</i> |
| <b>Return Value</b> | None.                                                                  |             |                                                                                                                                                                                                                                                                                                                                |

| Python Syntax  | EditSkinDepthOp(<OpName>, <SkinDepthOpParams>)                                                                                                                                                                   |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditSkinDepthOp("SkinDepth1",<br/>[<br/>    "NAME:SkinD",<br/>    "RestrictElem:=", False,<br/>    "SkinDepth:=", "2mm",<br/>    "SurfTriMaxLength:=", "1mm",<br/>    "NumLayers:=", 2<br/>])</pre> |

| VB Syntax  | EditSkinDepthOp <OpName>, <SkinDepthOpParams>                                                                                                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditSkinDepthOp "SkinDepth1",<br/>    Array("NAME:SkinD", _<br/>        "RestrictElem:=", false, _<br/>        "SkinDepth:=", "2mm", _<br/>        "SurfTriMaxLength:=", "1mm", _<br/>        "NumLayers:=", 2)</pre> |



## EditTrueSurfOp

Modifies an existing true surface approximation-based mesh operation. Assignments cannot be changed using this command. To change the assignment, delete this operation and create it using a new assignment.

| UI Access    | Double-click the operation in the project tree to modify its settings. |        |                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|--------------|------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                   | Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|              | <OpName>                                                               | String | Name of the operation to be edited.                                                                                                                                                                                                                                                                                                                                                                                                                          |
|              | <TrueSurfOpParams>                                                     | Array  | <p>Structured array.</p> <pre>Array("NAME:&lt;OpName&gt;",       "SurfDevChoice:=", &lt;RadioOption&gt;,       "SurfDev:=", &lt;value&gt;,       "NormalDevChoice:=", &lt;RadioOption&gt;,       "NormalDev:=", &lt;value&gt;,       "AspectRatioChoice:=", &lt;RadioOption&gt;,       "AspectRatio:=", &lt;double&gt;)</pre> <p>&lt;RadioOption&gt;</p> <p><i>Type: &lt;int&gt;</i></p> <p>0: Ignore</p> <p>1: Use defaults</p> <p>2: Specify the value</p> |
| Return Value | None.                                                                  |        |                                                                                                                                                                                                                                                                                                                                                                                                                                                              |

| Python Syntax  | EditTrueSurfOp(<OpName>, <TrueSurfOpParams>)                                                                                                                                                                                                   |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditTrueSurfOp("TrueSurf2",<br/>[<br/>    "NAME:trusurf",<br/>    "SurfDevChoice:=", 2,<br/>    "SurfDev:=", "0.03mm",<br/>    "NormalDevChoice:=", 1,<br/>    "AspectRatioChoice:=", 2,<br/>    "AspectRatio:=", 10<br/>])</pre> |

| VB Syntax  | EditTrueSurfOp <OpName>, <TrueSurfOpParams>                                                                                                                                                                                           |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditTrueSurfOp "TrueSurf2",<br/>    Array("NAME:trusurf", _<br/>        "SurfDevChoice:=", 2, _<br/>        "SurfDev:=", "0.03mm", _<br/>        "NormalDevChoice:=", 1, _<br/>        "AspectRatioChoice:=", 2, _</pre> |

```
"AspectRatio:=", 10)
```

## InitialMeshSettings

Assigns a true surface-based mesh operation to the selection.

| UI Access    | Mesh > Initial Mesh Settings             |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|------------------------------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                     | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|              | <i>&lt;InitialMeshSettingsParams&gt;</i> | Array | Structured array.<br><br>Array("NAME:<MeshSettings>",<br>"CurvedSurfaceApproxChoice:=", <"UseSlider"<br>or "Manual Settings"><br>"SliderMeshSettings:=", <integer 1 through<br>9>,<br>"UseLegacyFaceterForTauVolumeMesh:=",<br><boolean>,<br>"DynamicSurfaceResolution:=", <boolean>,<br>"UseFlexMeshingForTAUvolumeMesh:=",<br><boolean>,<br>"UseAlternativeMeshMethodsAsFallBack:=",<br><boolean>,<br>"AllowPhiForLayeredGeometry:=", <boolean>) |
| Return Value | None.                                    |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Python Syntax | InitialMeshSettings( <i>&lt;InitialMeshSettingsParams&gt;</i> ) |
|---------------|-----------------------------------------------------------------|

**Python Example**

```
oModule.InitialMeshSettings
([
 "NAME:MeshSettings",
 [
 "NAME:GlobalSurfApproximation",
 "CurvedSurfaceApproxChoice:=", "UseSlider",
 "SliderMeshSettings:=" , 5
],
 [
 "NAME:GlobalCurvilinear",
 "Apply:=", False
],
 [
 "NAME:GlobalModelRes",
 "UseAutoLength:=", True
],
 [
 "NAME:GlobalLengthMeshSetup",
 "RefineInside:=", True,
 "ID:=", -1,
```

```

 "Type:=", "LengthBased",
 "RestrictElem:=", False,
 "NumMaxElem:=", "1000",
 "RestrictLength:=", True,
 "MaxLength:=", "45mm",
 "ApplyToInitialMesh:=", True,
 "IsGlobal:=", True
],
 "MeshMethod:=", "Auto",
 "UseLegacyFaceterForTAUVolumeMesh:=", False,
 "DynamicSurfaceResolution:=", False,
 "UseFlexMeshingForTAUvolumeMesh:=", False,
 "UseAlternativeMeshMethodsAsFallBack:=", False,
 "AllowPhiForLayeredGeometry:=", True
]
)

```

|                   |                                                                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <b>InitialMeshSettings &lt;InitialMeshSettingsParams&gt;</b>                                                                                                                            |
| <b>VB Example</b> | <pre> oModule.InitialMeshSettings Array("NAME:MeshSettings", Array("NAME:G- lobalSurfApproximation",     "CurvedSurfaceApproxChoice:=", "UseSlider", "SliderMeshSettings:=", 5), </pre> |

```
Array("NAME:GlobalCurvilinear", "Apply:=", false),
Array("NAME:GlobalModelRes", "UseAutoLength:=", true), "MeshMethod:=", "Auto",
"UseLegacyFaceterForTauVolumeMesh:=", false,
"DynamicSurfaceResolution:=", false,
"UseFlexMeshingForTAUvolumeMesh:=", true,
"UseAlternativeMeshMethodsAsFallBack:=", false,
"AllowPhiForLayeredGeometry:=", true)
oModule.InitialMeshSettings Array("NAME:MeshSettings",
Array("NAME:GlobalSurfApproximation",
"CurvedSurfaceApproxChoice:=",
 "ManualSettings",
 "SurfDevChoice:=", 2,
 "SurfDev:=", "0.01mm",
 "NormalDevChoice:=", 2,
 "NormalDev:=", "22.5deg",
 "AspectRatioChoice:=", 2, "AspectRatio:=", "10"),
Array("NAME:GlobalCurvilinear", "Apply:=", false),
Array("NAME:GlobalModelRes", "UseAutoLength:=", true),
Array("GlobalLengthMeshSetup", "RefineInside:=", True, "ID:=", -1,
"Type:=", "LengthBased", "RestrictElem:=", False, "NumMaxElem:=", "1000",
"RestrictLength:=", True, "MaxLength:=", "45mm",
```

|                                                                                                                                                                                                                                                                                                              |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>"ApplyToInitialMesh:=", True, "IsGlobal:=", True)  "MeshMethod:=", "Auto",  "UseLegacyFaceterForTauVolumeMesh:=", false,  "DynamicSurfaceResolution:=", false,  "UseFlexMeshingForTAUvolumeMesh:=", true,  "UseAlternativeMeshMethodsAsFallBack:=", false,  "AllowPhiForLayeredGeometry:=", true)</pre> |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

This page intentionally  
left blank.



## 15 - Analysis Setup Module Script Commands

Maxwell analysis setup commands should be executed by the Analysis module, referred to in Maxwell scripts as the "AnalysisSetup" module.

Set oModule = oDesign.GetModule("AnalysisSetup")

[ClearLinkedData](#)

[CopySetup](#)

[DeleteSetups](#)

[EditSetup](#)

[ExportCircuit](#)

[ExportIcepak](#)

[ExportSolnData](#)

[GetSetupCount](#)

[GetSetups](#)

[GetSweepCount](#)

[GetSweeps](#)

[InsertSetup \(Maxwell\)](#)

[PasteSetup](#)

[PasteSweep](#)

[RenameSetup](#)

[ResetAllToTimeZero](#)

[ResetSetupToTimeZero](#)

[RevertAllToInitial](#)

[RevertAllToInitialTemperature](#)

[RevertSetupToInitial](#)

[RevertSetupToInitialTemperature](#)

## ClearLinkedData (Module)

Clear the linked data of the specified solution setups. (This command is similar to the ClearLinkedData command for the *design* level, which clears the linked data for all solution setups in the design.)

|              |                                                                                                                                                                                                                 |       |                                                                    |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|--------------------------------------------------------------------|
| UI Access    | Project Manager > <i>{Design name}</i> > Analysis > right-click <i>{Setup name}</i> > Clear Linked Data<br>or, with a solution setup selected in the Project Manager:<br>Maxwell > Analysis > Clear Linked Data |       |                                                                    |
| Parameters   | Name                                                                                                                                                                                                            | Type  | Description                                                        |
|              | <SetupNameArray>                                                                                                                                                                                                | Array | Specify the name of the setups whose linked data are to be cleaned |
| Return Value | None                                                                                                                                                                                                            |       |                                                                    |

|                |                                                  |
|----------------|--------------------------------------------------|
| Python Syntax  | ClearLinkedData(<SetupNameArray>)                |
| Python Example | <code>oModule.ClearLinkedData(["setup1"])</code> |

|           |                                  |
|-----------|----------------------------------|
| VB Syntax | ClearLinkedData <SetupNameArray> |
|-----------|----------------------------------|

|                   |                                                      |
|-------------------|------------------------------------------------------|
| <b>VB Example</b> | <code>oModule.ClearLinkedData Array("setup1")</code> |
|-------------------|------------------------------------------------------|

## CopySetup

Copy the specified Optimetrics setup.

|                     |             |        |                    |
|---------------------|-------------|--------|--------------------|
| <b>UI Access</b>    | NA          |        |                    |
| <b>Parameters</b>   | Name        | Type   | Description        |
|                     | <SetupName> | String | Name of the setup. |
| <b>Return Value</b> | None.       |        |                    |

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | <code>CopySetup (&lt;SetupName&gt;)</code>            |
| <b>Python Example</b> | <code>oModule.CopySetup ("OptimizationSetup1")</code> |

|                   |                                                     |
|-------------------|-----------------------------------------------------|
| <b>VB Syntax</b>  | <code>CopySetup &lt;SetupName&gt;</code>            |
| <b>VB Example</b> | <code>oModule.CopySetup "OptimizationSetup1"</code> |

## DeleteSetups

Deletes one or more solution setups, which are specified by an array of solution setup names.

|                   |                                                                                                                                                                       |       |                                |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|--------------------------------|
| <b>UI Access</b>  | Right-click a solution setup in the project tree and then click <b>Delete</b> on the shortcut menu, or delete selected solution setups in the <b>List</b> dialog box. |       |                                |
| <b>Parameters</b> | Name                                                                                                                                                                  | Type  | Description                    |
|                   | <SetupArray>                                                                                                                                                          | Array | Array of solution setup names. |

|              |       |
|--------------|-------|
| Return Value | None. |
|--------------|-------|

|                |                                                         |
|----------------|---------------------------------------------------------|
| Python Syntax  | DeleteSetups (<SetupArray>)                             |
| Python Example | <code>oModule.DeleteSetups(["Setup1", "Setup2"])</code> |

|            |                                                             |
|------------|-------------------------------------------------------------|
| VB Syntax  | DeleteSetups <SetupArray>                                   |
| VB Example | <code>oModule.DeleteSetups Array("Setup1", "Setup2")</code> |

EditSetup [Maxwell]

Modifies an existing solution setup.

|              |                                                                                   |        |                                                                                                                                                                 |
|--------------|-----------------------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Double-click a solution setup in the Project Manager tree to modify its settings. |        |                                                                                                                                                                 |
| Parameters   | Name                                                                              | Type   | Description                                                                                                                                                     |
|              | <SetupName>                                                                       | String | Name of the solve setup being edited.                                                                                                                           |
|              | <Attributes>                                                                      | Array  | Structured array.<br><br>Array("NAME:<NewSetupName>", <NamedParameters>)<br><br>See the "InsertSetup [Maxwell]" on page 15-22 command for details and examples. |
| Return Value | None                                                                              |        |                                                                                                                                                                 |

**Note:** For Eddy Current designs, `Frequency` also supports use of variables and expressions.

| Python Syntax  | EditSetup (<SetupName>, <Attributes>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <p>3D Eddy Current with Permeability Link:</p> <pre>oModule.EditSetup("EddyCurrent",<br/>[<br/>    "NAME:EddyCurrent",<br/>    "Enabled:="          , True,<br/>    [<br/>        "NAME:MeshLink",<br/>        "ImportMesh:="    , False<br/>    ],<br/>    "MaximumPasses:="    , 4,<br/>    "MinimumPasses:="    , 2,<br/>    "MinimumConvergedPasses:=", 1,<br/>    "PercentRefinement:=" , 30,<br/>    "SolveFieldOnly:="   , False,<br/>    "PercentError:="     , 0.1,<br/>    "SolveMatrixAtLast:=" , True,<br/>    "UseNonLinearIterNum:=" , False,<br/>    [<br/>        "NAME:ExpressionCache",<br/>        [<br/></pre> |

```
"NAME:CacheItem",
"Title:=" , "eddy_loss1",
"Expression:=" , "eddy_loss",
"Intrinsics:=" , "Phase='\0deg'",
"ReportType:=" , "Fields",
[
 "NAME:ExpressionContext"
]
],
"UseCacheFor:=" , ["Pass"],
"UseIterativeSolver:=" , False,
"RelativeResidual:=" , 0.0001,
"NonLinearResidual:=" , 0.0001,
"SmoothBHCurve:=" , False,
"Frequency:=" , "200Hz",
"HasSweepSetup:=" , False,
"UseHighOrderShapeFunc:=", False,
"UseMuLink:=" , True,
"ImportMeshForMuLink:=" , True,
```

|  |                                                 |
|--|-------------------------------------------------|
|  | <pre>"LossAdaptiveCtrl:="      , "0.3" l)</pre> |
|--|-------------------------------------------------|

| VB Syntax  | EditSetup <SetupName>, <Attributes>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <p>3D Eddy Current with Permeability Link</p> <pre>oModule.EditSetup "Setup1", Array("NAME:Setup1", "Enabled:=", true, Array("NAME:MeshLink", "ImportMesh:=", false), "MaximumPasses:=", 10, "MinimumPasses:=", 2, "MinimumConvergedPasses:=", 1, "PercentRefinement:=", 30, "SolveFieldOnly:=", false, "PercentError:=", 1, "SolveMatrixAtLast:=", true, "UseNonLinearIterNum:=", false, "UseCacheFor:=", Array("Pass"), "UseIterativeSolver:=", false, "RelativeResidual:=", 1E-05, "NonLinearResidual:=", 0.0001, "SmoothBHCurve:=", true, "Frequency:=", "60Hz",  "HasSweepSetup:=", true, Array("NAME:SweepRanges", Array("NAME:Subrange", "RangeType:=", "LinearStep", "RangeStart:=", "10Hz", "RangeEnd:=", "1000Hz", "RangeStep:=", "10Hz"), Array("NAME:Subrange", "RangeType:=", "SinglePoints",</pre> |

|  |                                                                                                                                                                                                                                   |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"RangeStart:=", "1000Hz",<br/>"RangeEnd:=", "1000Hz",<br/>"SaveSingleField:=", false)),<br/>"SaveAllFields:=", true,<br/>"UseHighOrderShapeFunc:=", false,<br/>"UseMuLink:=", true,<br/>"ImportMeshForMuLink:=", true)</pre> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**VB Example: 3D ElectricTransient EditSetup:**

```
oModule.EditSetup "Setup1",
Array("NAME:Setup1",
"Tolerance:=", 0.005,
Array("NAME:Data",
"SaveField:=", true,
"Stop:=", "100s",
"InitialStep:=", "0.01s",
"MaxStep:=", "0.5s"),
"Initial Voltage:=", "0V",
"NumberOfOutputVars:=", 0)
```

**VB Example: 3D Transient EditSetup:**

```
oModule.EditSetup "Setup1",
Array("NAME:Setup1", "Enabled:=", true,
Array("NAME:MeshLink", "ImportMesh:=", false),
"NonlinearSolverResidual:=", "0.005",
"ScalarPotential:=", "Second Order",
"SmoothBHCurve:=", false,
"StopTime:=", "10000000ns",
"TimeStep:=", "2000000ns",
"OutputError:=", false,
"OutputPerObjectCoreLoss:=", true,
"OutputPerObjectSolidLoss:=", false,
```



```
"UseControlProgram:=", false,
"ControlProgramName:=", " ",
"ControlProgramArg:=", " ",
"CallCtrlProgAfterLastStep:=", false,
"FastReachSteadyState:=", false,
"AutoDetectSteadyState:=", false,
"IsGeneralTransient:=", true,
"IsHalfPeriodicTransient:=", false,
"SaveFieldsType:=", "None",
"UseNonLinearIterNum:=", false,
"CacheSaveKind:=", "Count",
"NumberSolveSteps:=", 1,
"RangeStart:=", "0s",
"RangeEnd:=", "0.1s")
```

### **VB Example: 3D ACConduction EditSetup:**

```
oModule.EditSetup "Setup1",
Array("NAME:Setup1", "Enabled:=", true,
Array("NAME:MeshLink", "ImportMesh:=", false),
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseNonLinearIterNum:=", true,
"MinIterNum:=", 1,
"MaxIterNum:=", 100,
"CacheSaveKind:=", "Delta",
"ConstantDelta:=", "0s",
"UseIterativeSolver:=", false,
"RelativeResidual:=", 1E-06,
"NonLinearResidual:=", 0.001,
"Frequency:=", "60Hz",
"HasSweepSetup:=", false)
```

### **VB Example: 2D DCConduction EditSetup:**

```
oModule.EditSetup "Setup1",
Array("NAME:Setup1",
"MaximumPasses:=", 20,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
```

```
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

**VB Example: 2D Transient EditSetup:**

```
Set oModule = oDesign.GetModule("AnalysisSetup")
oModule.EditSetup "Setup1",
Array("NAME:Setup1",
"Enabled:=", true,
"NonlinearSolverResidual:=", "0.0001",
"TimeIntegrationMethod:=", "BackwardEuler",
"SmoothBHCurve:=", true,
"StopTime:=", "0.2s",
"TimeStep:=", "0.0002s",
"OutputError:=", false,
"OutputPerObjectCoreLoss:=", true,
"OutputPerObjectSolidLoss:=", false,
"UseControlProgram:=", false,
"ControlProgramName:=", " ",
"ControlProgramArg:=", " ",
"CallCtrlProgAfterLastStep:=", false,
"FastReachSteadyState:=", true,
"AutoDetectSteadyState:=", true,
"FrequencyOfAddedVoltageSource:=", "50Hz",
```

```
"StopCriterion:=", 0.005,
"IsGeneralTransient:=", false,
"IsHalfPeriodicTransient:=", true,
"HasSweepSetup:=", true,
"SaveFieldsType:=", "Custom",
"Sweep Ranges:=",
[
"SweepSetupType:=", "LinearStep",
"StartValue:=", "0",
"StopValue:=", "200ms",
"StepSize:=", "25ms"
]
```

### ***Python Example: Transient EditSetup:***

```
oModule.EditSetup("Setup1",
[
"NAME:Setup1",
"Enabled:=", True,
"NonlinearSolverResidual:=", "0.9",
"ScalarPotential:=", "Second Order",
"TimeIntegrationMethod:=", 0,
"SmoothBHCurve:=", True,
"StopTime:=", "200ms",
"TimeStep:=", "10ms",
"OutputError:=", False,
```

```
"UseControlProgram:=" , False,
"ControlProgramName:=" , " ",
"ControlProgramArg:=" , " ",
"CallCtrlProgAfterLastStep:=", False,
"FastReachSteadyState:=", False,
"AutoDetectSteadyState:=", False,
"IsGeneralTransient:=" , True,
"IsHalfPeriodicTransient:=", False,
"HasSweepSetup:=" , True,
"SaveFieldsType:=" , "Every N Steps",
"N Steps:=", , "$MyStepCount"
])
```

## ExportCircuit

*Use:* Export equivalent circuit data.

*Command:* Right-click a setup in the project tree or the **Analysis** folder and choose **Export Circuit**.

*Syntax:* ExportCircuit <Solution>, <Variation>, <FileName>, <ExportSettings>, <ModelName>, <Freq>

*Return Value:* none

*Parameters:* <Solution>

<SetupName>:<SolutionName>

<SetupName>

Type: <string>

Name of the setup where circuit is being exported

<SolutionName>

Type: <String>

Name of the solution.

<Variation>

Type: <string>

The variation where circuit is being exported

<FileName>

Type: <string>

The name of the file where circuit is being exported

<ModelName>

Type: <String>

Model name or name of the sub circuit (Optional). If not specified then <FileName> is considered as model name.

<Freq>

Type: <double>

Sweep frequency in hz.

<ExportSettings>

Array("NAME:CircuitData", "MatrixName:=", \_

<ReduceMatrix>, "NumberOfCells:=", <NumCell>, "UserHasChangedSettings:=", <User-ChangedSettings>, "IncludeCap:=", <IncludeCap>, "IncludeCond:=", <IncludeCond>, Array("NAME:CouplingLimits", <CouplingLimitsArray>, "IncludeDCR:=", <IncludeDCR>, "IncludeDCL:=", <IncludeDCL>, "IncludeACR:=", <IncludeACR>, "IncludeACL:=", <IncludeACL>, "ADDResistance:=", <AddResistance>)

**Parameters:**

<ReduceMatrix>

Type: <string>

One of the reduced matrix setup or "Original"

<NumCell>

Type: <string>

Number of cells in export. Can be a variable.

<UserChangedSettings>

Type: <bool>

Whether user changed settings or use default settings.wirebond

<IncludeCap>

Type: <bool>

Flag indicates whether to export Capacitance matrix.

<IncludeCond>

Type: <bool>

Flag indicates whether to export Conductance matrix.

<IncludeDCR>

Type: <bool>

Flag indicates whether to export DC resistance matrix.

<IncludeDCL>

Type: <bool>

Flag indicates whether to export DC Inductance matrix.

<IncludeACR>

Type: <bool>

Flag indicates whether to export AC resistance matrix.

<IncludeACL>

Type: <bool>

Flag indicates whether to export AC inductance matrix.

<AddResistance>

Type: <bool>

Adds the DC and AC resistance.

### Note:

You cannot export both AC and DC matrices unless **AddResistance** is selected.

<CouplingLimitsArray>

Array("NAME:CouplingLimits", "CouplingLimitType:=", <CouplingLimitType>, \_

<CouplingLimitsParameters>, 0.01, "CondFraction:=", 0.01)

<CouplingLimitType> = "None"

Argument not needed

<CouplingLimitType> = "ByFraction"

<CouplingLimitsParameters>

"CapFraction:=", <Fraction>, "IndFraction:=", <Fraction>,

"ResFraction:=", <Fraction>, "CondFraction:=", <Fraction>,



**Parameters:**

&lt;Fraction&gt;

Type: &lt;double&gt;

Fraction of the self term

```
<CouplingLimitType> = "ByValue"
```

```
<CouplingLimitsParameters>
```

```
"CapLimit:=", <Limit>, "IndLimit:=", <Limit>, "ResLimit:=", <Limit>,
```

```
"CondLimit:=", <Limit>,
```

**Parameters:**

&lt;Limit&gt;

Type: &lt;string&gt;

Value of the limit.

*VB Example:*

```
oModule.ExportCircuit
"Setup1 : LastAdaptive", "", "C:/Project/Q3D/FourNets.cir", Array("NAME:CircuitData", _
"MatrixName:=", "Original", "NumberOfCells:=", "1", "UserHasChangedSettings:=", true, _
"IncludeCap:=", true, "IncludeCond:=", true, Array("NAME:CouplingLimits", "CouplingLimitType:=",
_
"By Fraction", "CapFraction:=", 0.01, "IndFraction:=", 0.01, "ResFraction:=", 0.01, _
"CondFraction:=", 0.01), "IncludeDCR:=", false, IncludeDCL:=", false, "IncludeACR:=", false, _
"IncludeACL:=", false, "ADDResistance:=", true), "", 20000000000000
```

## ExportIcepak

**Use:** Exports Icepak data to the specified output directory.

**Command:** Right-click a setup under **Analysis** in the project tree, and select **Create Icepak Design**. From the dialog box, select an appropriate solution setup and variation, enter the path to the output directory, and click the **OK** button.

**Syntax:** ExportIcepak <SetupName>, <Variation>, <Output Directory>

**Return Value:** None

**Parameters:** <SetupName>

Type: string

Setup name.

<Variation>

Type: string

Variation values.

<Output Directory>

Type: string

Path to the output directory in which the Icepak design files will be saved.

**Example:** oModule.ExportIcepak "Setup1", "air\_gap='1mm' diameter='120mm'", "D:\Work\Icepak\"

## ExportSolnData

**Use:** Exports solution data to a file.

**Command:** Right-click a parameter setup under **Parameters** in the project tree, and select **ViewSolution**. From the dialog box, select an appropriate solution type, and click the **Export Solution** button.

**Syntax:** ExportSolnData <SetupName>, <SolutionName>, <IsPostProcessed>, <Variation>, <ExportFileName>,

Return Value: None

**Parameters:** <SetupName>

```

 Type: string
 Setup name.
<SolutionName>
 Type: string
 Solution setup name.
<IsPostProcessed>
 Type: bool
 Is post processed (true/false).
<Variation>
 Type: string
 Variation values.
<ExportFileName>
 Type: string
 Export file name.
```

**Example:** oModule.ExportSolnData "Setup1", "Matrix1", false, "", \_  
"C:/Maxwell/Projects/Export/OutputSolution.txt"

# GetSetupCount

Gets the number of analysis setups in a design.

|            |       |
|------------|-------|
| UI Access  | N/A   |
| Parameters | None. |

|                     |                                      |
|---------------------|--------------------------------------|
| <b>Return Value</b> | Integer containing number of setups. |
|---------------------|--------------------------------------|

|                       |                                       |
|-----------------------|---------------------------------------|
| <b>Python Syntax</b>  | GetSetupCount ()                      |
| <b>Python Example</b> | <code>oModule.GetSetupCount ()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | GetSetupCount                      |
| <b>VB Example</b> | <code>oModule.GetSetupCount</code> |

## GetSetups

Gets the names of analysis setups in a design.

|                     |                               |
|---------------------|-------------------------------|
| <b>UI Access</b>    | N/A                           |
| <b>Parameters</b>   | None.                         |
| <b>Return Value</b> | Array of analysis setup names |

|                       |                                   |
|-----------------------|-----------------------------------|
| <b>Python Syntax</b>  | GetSetups ()                      |
| <b>Python Example</b> | <code>oModule.GetSetups ()</code> |

|                  |           |
|------------------|-----------|
| <b>VB Syntax</b> | GetSetups |
|------------------|-----------|

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Example</b> | <code>oModule.GetSetups</code> |
|-------------------|--------------------------------|

## GetSweepCount

Gets the number of sweeps in a specified analysis setup.

|                     |                                                          |        |                          |
|---------------------|----------------------------------------------------------|--------|--------------------------|
| <b>UI Access</b>    | N/A                                                      |        |                          |
| <b>Parameters</b>   | Name                                                     | Type   | Description              |
|                     | <SetupName>                                              | String | Name of specified setup. |
| <b>Return Value</b> | Integer containing number of sweeps for the named setup. |        |                          |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetSweepCount (&lt;SetupName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.GetSweepCount ("Setup1")</code>  |

|                   |                                              |
|-------------------|----------------------------------------------|
| <b>VB Syntax</b>  | <code>GetSweepCount &lt;SetupName&gt;</code> |
| <b>VB Example</b> | <code>oModule.GetSweepCount "Setup1"</code>  |

## GetSweeps

Gets the names of all sweeps in a given analysis setup.

|                   |             |        |                          |
|-------------------|-------------|--------|--------------------------|
| <b>UI Access</b>  | N/A         |        |                          |
| <b>Parameters</b> | Name        | Type   | Description              |
|                   | <SetupName> | String | Name of specified setup. |

|                     |                       |
|---------------------|-----------------------|
| <b>Return Value</b> | Array of sweep names. |
|---------------------|-----------------------|

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | GetSweeps (<SetupName>)                   |
| <b>Python Example</b> | <code>oModule.GetSweeps ("Setup1")</code> |

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>VB Syntax</b>  | GetSweeps <SetupName>                   |
| <b>VB Example</b> | <code>oModule.GetSweeps "Setup1"</code> |

## InsertSetup [Maxwell]

This command adds a new solution setup. The information on this page is specific to the implementation of the InsertSetup command for Maxwell 2D and 3D designs.

The parameters depend on the following model characteristics:

- whether the design is 2D or 3D
- which type of solution you are setting up (*SetupType* = "[Magnetostatic](#)", "[EddyCurrent](#)", "[Transient](#)", "[Electrostatic](#)", "[DCConduction](#)", "[ElectroDCConduction](#)", "[ElectricTransient](#)", and "[ACConduction](#)")
- whether you are importing the mesh from a source design
- whether there are any variables mapped between the source and target designs (when importing a mesh)

Certain options, when selected, activate additional parameters. For example, Solver limiting parameters ( **Minimum** and **Maximum**) are only applicable if the **Number of Nonlinear Iterations** is checked.

|                  |                                                                             |
|------------------|-----------------------------------------------------------------------------|
| <b>UI Access</b> | <b>Maxwell 3D or Maxwell 2D &gt; Analysis Setup &gt; Add Solution Setup</b> |
|------------------|-----------------------------------------------------------------------------|

| Parameters   | Name         | Type   | Description                                                                                                                                                                                                                                                                                                                                                                                                                       |
|--------------|--------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <SetupName>  | String | <p>Type of the solve setup being added.</p> <p>Possible values for 3D designs are "Magnetostatic", "EddyCurrent", "EddyCurrentAPhiFormulation", "Transient", "TransientAPhiFormulation", "Electrostatic", "DCConduction", "ElectroDCConduction", "ElectricTransient", "ACConduction".</p> <p>Possible values for 2D designs are "Magnetostatic", "EddyCurrent", "Transient", "Electrostatic", "DCConduction", "ACConduction".</p> |
|              | <Attributes> | Array  | <p>Structured array.</p> <p>Array("NAME:&lt;NewSetupName&gt;", &lt;NamedParameters&gt;)</p> <p>The named parameters vary according to the setup type and settings selected for a setup. See the typical examples below for each solution type. The parameters correspond to fields in the Setup window.</p>                                                                                                                       |
| Return Value | None         |        |                                                                                                                                                                                                                                                                                                                                                                                                                                   |

|                |                                                                                                                                                                                                    |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | InsertSetup (<SetupName>, <Attributes>)                                                                                                                                                            |
| Python Example | <p>3D <a href="#">Eddy Current with Permeability Computation</a></p> <pre>oModule.InsertSetup("EddyCurrent", [     "NAME:EddyCurrent",     "Enabled:=", True,     [         "NAME:MeshLink",</pre> |

```
"ImportMesh:=" , False
],
"MaximumPasses:=" , 4,
"MinimumPasses:=" , 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=" , 30,
"SolveFieldOnly:=" , False,
"PercentError:=" , 0.1,
"SolveMatrixAtLast:=" , True,
"UseNonLinearIterNum:=" , False,
"UseCacheFor:=" , ["Pass"],
"UseIterativeSolver:=" , False,
"RelativeResidual:=" , 1E-05,
"NonLinearResidual:=" , 0.0001,
"RelaxationFactor:=" , 1,
"SmoothBHCurve:=" , False,
"Frequency:=" , "150kHz",
"HasSweepSetup:=" , False,
"UseHighOrderShapeFunc:=", False,
"UseMuLink:=" , True,
```



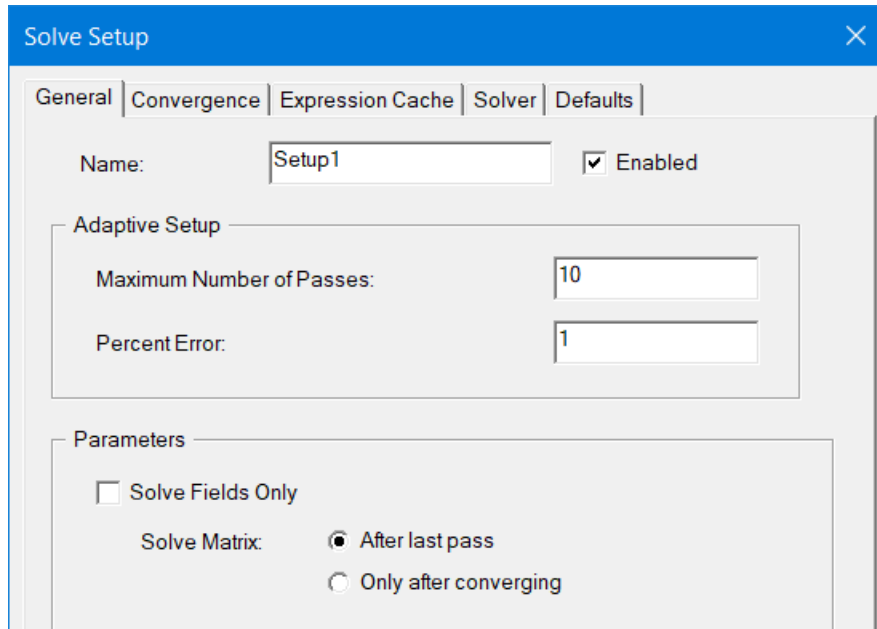
|                   |                                                                                                                                                                                                                                                                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                   | <pre> "ImportMeshForMuLink:=" , True, "LossAdaptiveCtrl:=" , "0.5", "Project:=" , "This Project*", "Product:=" , "Maxwell", "Design:=" , "Wireless_Charging_PM_Source", "Soln:=" , "Setup1 : LastAdaptive", [     "NAME:Params" ], "ForceSourceToSolve:=" , True, "PreservePartnerSoln:=" , True, "PathRelativeTo:=" , "TargetProject" ]) </pre> |
| <b>VB Syntax</b>  | <b>InsertSetup &lt;SetupName&gt;, &lt;Attributes&gt;</b>                                                                                                                                                                                                                                                                                         |
| <b>VB Example</b> | <p>3D <a href="#">Eddy Current with Permeability Computation:</a></p> <pre> oModule.InsertSetup "Setup1", Array("NAME:Setup1", "Enabled:=", true, Array("NAME:MeshLink", "ImportMesh:=", false), "MaximumPasses:=", 10, "MinimumPasses:=", 2, "MinimumConvergedPasses:=", 1, "PercentRefinement:=", 30, "SolveFieldOnly:=", false, </pre>        |

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "PercentError:=", 1, "SolveMatrixAtLast:=", true, "UseNonLinearIterNum:=", false, "UseCacheFor:=", Array("Pass"), "UseIterativeSolver:=", false, "RelativeResidual:=", 1E-05, "NonLinearResidual:=", 0.0001, "SmoothBHCurve:=", true, "Frequency:=", "60Hz",  "HasSweepSetup:=", true, Array("NAME:SweepRanges", Array("NAME:Subrange", "RangeType:=", "LinearStep", "RangeStart:=", "10Hz", "RangeEnd:=", "1000Hz", "RangeStep:=", "10Hz"), Array("NAME:Subrange", "RangeType:=", "SinglePoints", "RangeStart:=", "1000Hz", "RangeEnd:=", "1000Hz", "SaveSingleField:=", false)), "SaveAllFields:=", true, "UseHighOrderShapeFunc:=", false, "UseMuLink:=", true, "ImportMeshForMuLink:=", true) </pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**Tip:** For the Maxwell solver, there are numerous parameter commands for which the availability is determined by whether or not the design is 2D or 3D and by the solution type. Listing all of these commands in this topic would be unwieldy and difficult to navigate. The easiest way to identify all the parameters for a certain solver is to record a *.py* or *.vbs* script of the solver setup in the software, and then view the output, which will list all the parameter commands used that setup. The resulting script can then be easily edited and run. See [Recording a Script](#) for more details.

To understand how the parameter commands correspond to settings in the Solve Setup dialog, let's look at a script for a Maxwell 3D [Magnetostatic](#) design:

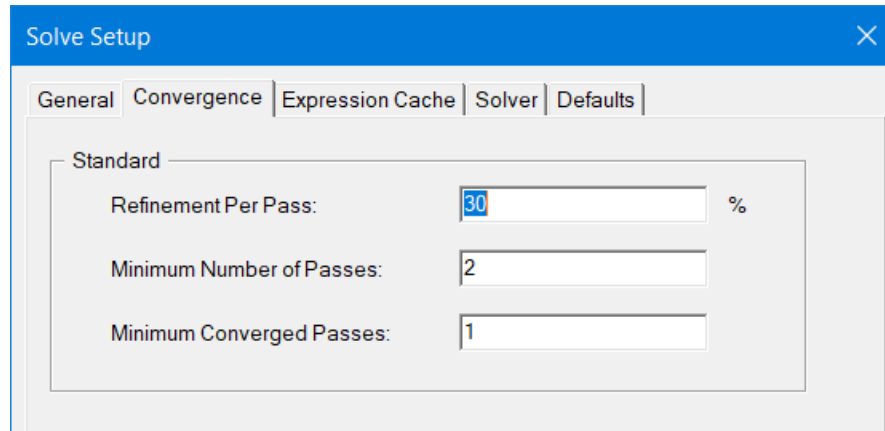
```
oModule.InsertSetup "Magnetostatic", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false,
"UseNonLinearIterNum:=", True,
"MinIterNum:=", 5,
"MaxIterNum:=", 10,
"NonLinearResidual:=", 0.001,
"MuNonLinearBH:=", true,
"ComputeHc:=", false,
"HcNonLinearBH:=", true,
"UserOverrideMu:=", true,
"UserExePath:=", "E:\views\projects\ucp.exe",
"UserExeArg:=", "-3D")
```



Some of the Array attributes correspond to these fields on the **General** tab of the Setup.

```
oModule.InsertSetup "Magnetostatic", Array("NAME:Setup1",
"MaximumPasses:=", 10,
...
"SolveMatrixAtLast:=", true,
```

Other parameters correspond to fields in the **Convergence** tab of the Setup.



```
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
```

Still other parameters correspond to fields in the **Solver** tab of the Setup.

```
"NonLinearResidual:=", 0.001,
UseNonLinearIterNum:=" , True,
"MinIterNum:=" , 5,
"MaxIterNum:=" , 10,
```

The examples that follow demonstrate the commands for different setups.

### Example:2D [Magnetostatic](#) solve setup

```
oModule.InsertSetup "Magnetostatic", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
```

```
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseNonLinearIterNum:=", True,
"MinIterNum:=", 5,
"MaxIterNum:=", 10,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false,
"NonLinearResidual:=", 0.0001,
"MuNonLinearBH:=", true,
"ComputeHc:=", false,
"HcNonLinearBH:=", true
"UserOverrideMu:=", true,
"UserExePath:=", "E:\views\projects\ucp.exe",
"UserExeArg:=", "-3D")
```

### Python Example: 3D [Eddy Current](#) solve setup:

```
oModule.InsertSetup("EddyCurrent",
[
 "NAME:EddyCurrent_03",
 "Enabled:=", True,
 [
 "NAME:MeshLink",
 "ImportMesh:=", False
],
 "MaximumPasses:=", 4,
 "MinimumPasses:=", 2,
```

```
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=" , 30,
"SolveFieldOnly:=" , False,
"PercentError:=" , 0.1,
"SolveMatrixAtLast:=" , True,
"UseNonLinearIterNum:=" , False,
[
 "NAME:ExpressionCache",
 [
 "NAME:CacheItem",
 "Title:=" , "eddy_loss1",
 "Expression:=" , "eddy_loss",
 "Intrinsics:=" , "Phase=\'0deg\'",
 "ReportType:=" , "Fields",
 [
 "NAME:ExpressionContext"
]
]
],
"UseCacheFor:=" , ["Pass"],
"UseIterativeSolver:=" , False,
```

```
"RelativeResidual:=" , 0.0001,
"NonLinearResidual:=" , 0.0001,
"SmoothBHCurve:=" , False,
"Frequency:=" , "200Hz",
"HasSweepSetup:=" , False,
"UseHighOrderShapeFunc:=", False,
"UseMuLink:=" , False,
"LossAdaptiveCtrl:=" , "0.3"
])
```

### Python Example: 2D [Eddy Current](#) solve setup with two-way thermal coupling

```
oModule.InsertSetup("EddyCurrent",
[
"NAME:Setup1",
"Enabled:=" , True,
[
"NAME:MeshLink",
"ImportMesh:=" , False
],
"MaximumPasses:=" , 10,
"MinimumPasses:=" , 2,
```



```
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=" , 30,
"SolveFieldOnly:=" , False,
"PercentError:=" , 1,
"SolveMatrixAtLast:=" , True,
"UseNonLinearIterNum:=" , False,
"NonLinearResidual:=" , 0.0001,
"RelaxationFactor:=" , 0.4, # only available for 2D designs with two-way thermal coupling
"SmoothBHCurve:=" , False,
"Frequency:=" , "60Hz",
"UseLossConvergence:=" , False,
"HasSweepSetup:=" , False
])
```

**Note:** For Eddy Current designs, `Frequency` also supports use of variables and expressions.

**VB Example:3D [Eddy Current](#) solve setup:**

```
oModule.InsertSetup "EddyCurrent", Array("NAME:Setup1",
"Enabled:=", true,
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseNonLinearIterNum:=", True,
"MinIterNum:=", 5,
"MaxIterNum:=", 10,
"UseIterativeSolver:=", true,
"RelativeResidual:=", 0.00013,
"ComputeForceDensity:=", false,
"ComputePowerLoss:=", false,
"ThermalFeedback:=", true,
"Frequency:=", "60Hz",
```

**Note:** For Eddy Current designs, Frequency also supports use of variables and expressions.

```
"HasSweepSetup:=", true,
Array("NAME:SweepRanges", Array("NAME:Subrange", "RangeType:=", "LinearStep", "RangeStart:=",
"10Hz", "RangeEnd:=", "1000Hz", "RangeStep:=", "10Hz"),
Array("NAME:Subrange", "RangeType:=", "SinglePoints", "RangeStart:=", "1000Hz", "RangeEnd:=",
"1000Hz", "SaveSingleField:=", false)), "SaveAllFields:=", true,
"UseHighOrderShapeFunc:=", false)
```

**Example: 3D [Transient](#) solve setup:**

```

oModule.InsertSetup "Transient",
Array("NAME:Setup1", "Enabled:=", true,
"NonlinearSolverResidual:=", 0.005,
"ScalarPotential:=", "First Order",
"TimeIntegrationMethod:=", -1,
Array("NAME:PrevSoln", "Project:=", "This Project*",
"Product:=", "Maxwell",
"Design:=", "NL_V_ED1_Link",
"Soln:=", "Setup1 : LastAdaptive",
Array("NAME:Params"), "ForceSourceToSolve:=", false,
"PreservePartnerSoln:=", false,
"PathRelativeTo:=", "TargetProject"),
"StopTime:=", "10000000ns",
"TimeStep:=", "2000000ns",
"OutputError:=", true,
"OutputPerObjectCoreLoss:=", true,
"OutputPerObjectSolidLoss:=", false,
"UseControlProgram:=", false,
"ControlProgramName:=", "",
"ControlProgramArg:=", " ",
"CallCtrlProgAfterLastStep:=", false,
"FastReachSteadyState:=", true,
"FrequencyOfAddedVoltageSource:=", "60Hz",
"AutoDetectSteadyState:=", false,
"StopCriterion:=", "0.005",
"IsGeneralTransient:=", true,
"IsHalfPeriodicTransient:=", false,
"SaveFieldsType:=", "Every N Steps", "N Steps:=", "10", "Steps From:=", "0s", "Steps To:=",
"10000000ns",
"UseNonLinearIterNum:=", , True,
"MinIterNum:=", , 5,
"MaxIterNum:=", , 10,

```

```
"CacheSaveKind:=", "Count", "NumberSolveSteps:=", 1, "RangeStart:=", "0s", "RangeEnd:=",
"0.1s")
```

**Example: 2D [Transient](#) solve setup**

```
oModule.InsertSetup "Transient",
Array("NAME:Setup1",
"Enabled:=", true,
"NonlinearSolverResidual:=", "0.0001",
"TimeIntegrationMethod:=", "RungeKutta",
"StopTime:=", "0.02s",
"TimeStep:=", "0.0005s",
"OutputError:=", false,
"OutputPerObjectCoreLoss:=", true,
"OutputPerObjectSolidLoss:=", false,
"UseControlProgram:=", false,
"ControlProgramName:=", "",
"ControlProgramArg:=", " ",
"CallCtrlProgAfterLastStep:=", false,
"FastReachSteadyState:=", false,
"FrequencyOfAddedVoltageSource:=", "60Hz",
"AutoDetectSteadyState:=", false,
"StopCriterion:=", "0.005",
"IsGeneralTransient:=", true,
"IsAutoSubDivision:=", true,
"HasSweepSetup:=", true,
"SweepSetupType:=", "LinearStep",
"StartValue:=", "0s",
"StopValue:=", "0.1s",
"StepSize:=", "0.005s",
"UseNonLinearIterNum:=", , True,
"MinIterNum:=", , 5,
"MaxIterNum:=", , 10,
"UseAdaptiveTimeStep:=", false,
"InitialTimeStep:=", "0.002s",
"MinTimeStep:=", "0.001s",
"MaxTimeStep:=", "0.003s",
"TimeStepErrTolerance:=", 0.0001)
```

### Example: 3D [Electrostatic](#) solve setup

```
oModule.InsertSetup "Electrostatic", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

### Example: 2D [Electrostatic](#) solve setup:

```
oModule.InsertSetup "Electrostatic", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

### Example: 3D [DC Conduction](#) solve setup

```
oModule.InsertSetup "DCConduction", Array("NAME:Setup2",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
```

```
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

**Example: 2D [DC Conduction](#) solve setup:**

```
oModule.InsertSetup "DCConduction", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

**Example: 3D Electro[DC Conduction](#) solve setup**

```
oModule.InsertSetup "ElectroDCConduction", Array("NAME:Setup3",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false)
```

### Example: 3D [ElectricTransient](#) solve setup

```
oModule.InsertSetup "ElectricTransient", Array("NAME:Setup1",
"Tolerance:=", 0.005,
Array("NAME:Data",
"SaveField:=", true,
"Stop:=", "100s",
"InitialStep:=", "0.01s",
"MaxStep:=", "5s"),
"Initial Voltage:=", "0V",
"NumberOfOutputVars:=", 0)
```

### VB Example: 3D [AC Conduction](#) solve setup

```
oModule.InsertSetup "ACConduction", Array("NAME:Setup2", "Enabled:=", true,
Array("NAME:MeshLink", "ImportMesh:=", false),
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseNonLinearIterNum:=", true,
"MinIterNum:=", 1,
"MaxIterNum:=", 100,
"UseIterativeSolver:=", false,
"RelativeResidual:=", 1E-06,
"NonLinearResidual:=", 0.001,
"Frequency:=", "60Hz",
"HasSweepSetup:=", false)
```



**Python Example:3D [AC Conduction](#) solve setup**

```
oModule.InsertSetup("ACConduction",
[
 "NAME:Setup2",
 "Enabled:=" , True,
 [
 "NAME:MeshLink",
 "ImportMesh:=" , False
],
 "MaximumPasses:=" , 10,
 "MinimumPasses:=" , 2,
 "MinimumConvergedPasses:=", 1,
 "PercentRefinement:=" , 30,
 "SolveFieldOnly:=" , False,
 "PercentError:=" , 1,
 "SolveMatrixAtLast:=" , True,
 "UseNonLinearIterNum:=" , True,
 "MinIterNum:=" , 1,
 "MaxIterNum:=" , 100,
 "UseIterativeSolver:=" , False,
 "RelativeResidual:=" , 1E-06,
```

```
"NonLinearResidual:=" , 0.001,
"Frequency:=" , "60Hz",
"HasSweepSetup:=" , False
])
```

### Example: 2D [AC Conduction](#) solve setup:

```
oModule.InsertSetup "ACConduction", Array("NAME:Setup1",
"MaximumPasses:=", 10,
"MinimumPasses:=", 2,
"MinimumConvergedPasses:=", 1,
"PercentRefinement:=", 30,
"SolveFieldOnly:=", false,
"PercentError:=", 1,
"SolveMatrixAtLast:=", true,
"UseOutputVariable:=", false,
"PreAdaptMesh:=", false,
"Frequency:=", "60Hz",
"HasSweepSetup:=", false)
```

## PasteSetup

*Use:* Paste a solve setup.

*Syntax:* PasteSetup

*Return Value:* None

*VB Example:* oModule.PasteSetup

## PasteSweep

Pastes a copied sweep.

|                     |                                             |        |                                                    |
|---------------------|---------------------------------------------|--------|----------------------------------------------------|
| <b>UI Access</b>    | Right-click on a setup, select <b>Paste</b> |        |                                                    |
| <b>Parameters</b>   | Name                                        | Type   | Description                                        |
|                     | <SetupName>                                 | String | Name of setup where the copied sweep is pasted to. |
| <b>Return Value</b> | None.                                       |        |                                                    |

|                       |                                            |
|-----------------------|--------------------------------------------|
| <b>Python Syntax</b>  | PasteSweep (<SetupName>)                   |
| <b>Python Example</b> | <code>oModule.PasteSweep ("Setup1")</code> |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | PasteSweep <SetupName>                   |
| <b>VB Example</b> | <code>oModule.PasteSweep "Setup1"</code> |

## RenameSetup

Renames an existing solution setup.

|                     |                                                                                                        |        |                                           |
|---------------------|--------------------------------------------------------------------------------------------------------|--------|-------------------------------------------|
| <b>UI Access</b>    | Right-click a solution setup in the Project Manager and then click <b>Rename</b> on the shortcut menu. |        |                                           |
| <b>Parameters</b>   | Name                                                                                                   | Type   | Description                               |
|                     | <OldSetupName>                                                                                         | String | Name of the solution setup being renamed. |
|                     | <NewSetupName>                                                                                         | String | New name for the solution setup.          |
| <b>Return Value</b> | None.                                                                                                  |        |                                           |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | RenameSetup (<OldSetupName>, <NewSetupName>) |
| <b>Python Example</b> | oModule.RenameSetup ("Setup1", "MySetup")    |

|                   |                                            |
|-------------------|--------------------------------------------|
| <b>VB Syntax</b>  | RenameSetup <OldSetupName>, <NewSetupName> |
| <b>VB Example</b> | oModule.RenameSetup "Setup1", "MySetup"    |

## ResetAllToTimeZero

Forces the next solve to start from time 0 for all setups. Applies only to the Transient solution type.

**Command:** Right-click **Analysis** in the project tree, and select **Revert to Time Zero**.

**Syntax:**ResetAllToTimeZero

**Return Value:** None

Parameters: None

**Example:** oModule.ResetAllToTimeZero

## ResetSetupToTimeZero

**Use:** Forces the next solve to start from time 0 for a given setup. Applies only to the Transient solution type.

**Command:** Right-click a setup in the project tree, and select **Revert to Time Zero**.

**Syntax:**ResetSetupToTimeZero <SetupName>

**Return Value:** None

**Parameters:** <SetupName>

Type: string

Setup name

**Example:** oModule.ResetSetupToTimeZero "Setup1"

## RevertAllToInitial

Marks the current mesh for all solution setups as invalid. This will force the next simulation to begin with the initial mesh.

|              |                                                    |
|--------------|----------------------------------------------------|
| UI Access    | Maxwell > Analysis Setup > Revert to Initial Mesh. |
| Parameters   | None.                                              |
| Return Value | None.                                              |

|                |                               |
|----------------|-------------------------------|
| Python Syntax  | RevertAllToInitial ()         |
| Python Example | oModule.RevertAllToInitial () |

|            |                            |
|------------|----------------------------|
| VB Syntax  | RevertAllToInitial         |
| VB Example | oModule.RevertAllToInitial |

## RevertAllToInitialTemperature

Reverts all setups to their initial temperature. Used when coupling with thermal designs.

|           |                                                                                                                                                                      |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access | Maxwell > Set Object Temperature > Enable Feedback. Then in the Project Manager, right-click <b>Analysis</b> > Revert to Initial Temperature. Repeat for all setups. |
|-----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                     |       |
|---------------------|-------|
| <b>Parameters</b>   | None. |
| <b>Return Value</b> | None. |

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | RevertAllToInitialTemperature()                      |
| <b>Python Example</b> | <code>oModule.RevertAllToInitialTemperature()</code> |

|                   |                                                    |
|-------------------|----------------------------------------------------|
| <b>VB Syntax</b>  | RevertAllToInitialTemperature                      |
| <b>VB Example</b> | <code>oModule.RevertAllToInitialTemperature</code> |

## RevertSetupToInitial

Marks the current mesh for a solution setup as invalid. This will force the next simulation to begin with the initial mesh.

|                     |                                                                                                        |        |                          |
|---------------------|--------------------------------------------------------------------------------------------------------|--------|--------------------------|
| <b>UI Access</b>    | Right-click a solution setup in the Project Manager and then click <b>Rename</b> on the shortcut menu. |        |                          |
| <b>Parameters</b>   | Name                                                                                                   | Type   | Description              |
|                     | <SetupName>                                                                                            | String | Name of specified setup. |
| <b>Return Value</b> | None.                                                                                                  |        |                          |

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>Python Syntax</b>  | RevertSetupToInitial (<SetupName>)                  |
| <b>Python Example</b> | <code>oModule.RevertSetupToInitial("Setup1")</code> |

|                   |                                                    |
|-------------------|----------------------------------------------------|
| <b>VB Syntax</b>  | RevertSetupToInitial <SetupName>                   |
| <b>VB Example</b> | <code>oModule.RevertSetupToInitial "Setup1"</code> |

## RevertSetupToInitialTemperature

Reverts a specified setup to its initial temperature. Used when coupling with thermal designs.

|                     |                                                                                                                                                               |        |                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-------------------------|
| <b>UI Access</b>    | <b>Maxwell &gt; Set Object Temperature &gt; Enable Feedback.</b> Then in the Project Manager, right-click <b>Analysis &gt; Revert to Initial Temperature.</b> |        |                         |
| <b>Parameters</b>   | Name                                                                                                                                                          | Type   | Description             |
|                     | <setupName>                                                                                                                                                   | String | Name of solution setup. |
| <b>Return Value</b> | None                                                                                                                                                          |        |                         |

|                       |                                                                |
|-----------------------|----------------------------------------------------------------|
| <b>Python Syntax</b>  | RevertSetupToInitialTemperature(<setupName>)                   |
| <b>Python Example</b> | <code>oModule.RevertSetupToInitialTemperature('Setup1')</code> |

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <b>VB Syntax</b>  | RevertSetupToInitialTemperature <setupName>                   |
| <b>VB Example</b> | <code>oModule.RevertSetupToInitialTemperature "Setup1"</code> |

This page intentionally  
left blank.



## 16 - Optimetrics Module Script Commands

Optimetrics script commands should be executed by the `Optimetrics` module.

```
Set oModule = oDesign.GetModule("Optimetrics")
```

```
oModule.CommandName <args>
```

### Conventions Used in this Chapter

<VarName>

Type: <string>

Name of a variable.

<VarValue>

Type: <string>

Value with unit (i.e., <value>, but cannot be an expression).

<StartV>

Type: <VarValue>

The starting value of a variable.

<StopV>

Type: <VarValue>

The stopping value of a variable.

<MinV>

Type: <VarValue>

The minimum value of a variable.

<MaxV>

Type: <VarValue>

The maximum value of a variable.

<IncludeVar>

Type: <bool>

Specifies whether the variable is included in the analysis.

<StartingPoint>

```
Array("NAME:StartingPoint", "<VarName>:=",
 <VarValue>, ... "<VarName>:=", <VarValue>)
```

<SaveField>

Type: <bool>

Specifies whether HFSS will remove the non-nominal field solution.

<MaxIter>

Type: <int>

Maximum iteration allowed in an analysis.

<PriorSetup>

Type: <string>

The name of the embedded parametric setup.

<Precede>

Type: <bool>

If true, the embedded parametric setup will be solved before the analysis begins.

If false, the embedded parametric setup will be solved during each iteration of the analysis.

<Constraint>

```
Array("NAME:LCS",
 "lc=", Array("<VarName>:=",
 "<Coeff>, ..."<VarName>:=", <Coeff>, "rel:=", <Cond>, "rhs:=", <Rhs>), ...
 "lc=", Array("<VarName>:=", <Coeff>, ..."
 <VarName>:=", <Coeff>, "rel:=", <Cond>, "rhs:=",
 <Rhs>))
```

<Coeff>

Type: <double>

Coefficient for a variable in the linear constraint.

<Cond>

Type: <string>

Inequality condition.

<Rhs>

Type: <double>

Inequality value.

<OptiGoalSpec>

```
"Solution:=", <Soln>, "Calculation:=", <Calc>,
"Context:=", <Geometry>
Array("NAME:Ranges",
 "Range:", Array("Var:=",
 <VarName>, "Type:=", <RangeType>, "Start:=",
 <StartV>, "Stop:=", <StopV>), ...
 "Range:", Array("Var:=", <VarName>, "Type:=",
 <RangeType>, "Start:=", <StartV>, "Stop:=",
 <StopV>))
```

<Soln>

Type: <string>

Name of the solution.

<Calc>

Type: <string>

An expression that is composed of a basic solution quantity and an output variable.

<ContextName>

Type: <string>

Name of context needed in the evaluation of <Calc>.

<Geometry>

Type: <string>

Name of geometry needed in the evaluation of <Calc>.

<RangeType>

Type: <string>

if "r", start and stop values specify a range for the variable.

if "s", start values specify the single value for the variable.

[EditSetup](#)

[EditSetup \[Optimization\]](#)

[EditSetup \[Sensitivity\]](#)

[EditSetup \[Statistical\]](#)

[GetPropNames \[Optimetrics\]](#)

[GetPropValue \[Optimetrics\]](#)

[GetSetupNames \[Optimetrics\]](#)

[GetSetupNamesByType \[Optimetrics\]](#)

[InsertSetup \[Parametric\]](#)

[InsertSetup \[Optimization\]](#)

[InsertSetup \[Sensitivity\]](#)

[InsertSetup \[Statistical\]](#)

[PasteSetup \[Optimetrics\]](#)

[RenameSetup \[Optimetrics\]](#)

[SetPropValue \[Optimetrics\]](#)

[SolveSetup \[Optimetrics\]](#)

**The topics for this section include:**

[General Commands Recognized by the Optimetrics Module](#)

[Parametric Script Commands](#)

[Optimization Script Commands](#)

[Sensitivity Script Commands](#)

[Statistical Script Commands](#)

## General Commands Recognized by the Optimetrics Module

Following are general script commands recognized by the **Optimetrics** module:

[CopySetup](#)

[DistributedAnalyzeSetup](#)

[EditSetup](#)

[ExportDXConfigFile](#)

[ExportOptimetricsProfile](#)

[ExportOptimetricsResult](#)

[ExportParametricResults](#)

[GetOptimetricResult](#)

[GetPropNames \[Optimetrics\]](#)

[GetPropValue \[Optimetrics\]](#)

[GetSetupNames \[Optimetrics\]](#)

[GetSetupNamesByType \[Optimetrics\]](#)[ImportSetup](#)[PasteSetup \[Optimetrics\]](#)[RenameSetup \[Optimetrics\]](#)[SetPropValue \[Optimetrics\]](#)[SolveSetup \[Optimetrics\]](#)[SolveAllSetup](#)

## CopySetup

Copy the specified Optimetrics setup.

|                     |             |        |                    |
|---------------------|-------------|--------|--------------------|
| <b>UI Access</b>    | NA          |        |                    |
| <b>Parameters</b>   | Name        | Type   | Description        |
|                     | <SetupName> | String | Name of the setup. |
| <b>Return Value</b> | None.       |        |                    |

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | CopySetup (<SetupName>)                               |
| <b>Python Example</b> | <code>oModule.CopySetup ("OptimizationSetup1")</code> |

|                   |                                                     |
|-------------------|-----------------------------------------------------|
| <b>VB Syntax</b>  | CopySetup <SetupName>                               |
| <b>VB Example</b> | <code>oModule.CopySetup "OptimizationSetup1"</code> |

### DeleteSetups [Optimetrics]

Deletes the specified Optimetrics setups.

|              |                                                                                              |                  |                         |
|--------------|----------------------------------------------------------------------------------------------|------------------|-------------------------|
| UI Access    | Right-click the setup in the project tree, and then click <b>Delete</b> on the shortcut menu |                  |                         |
| Parameters   | Name                                                                                         | Type             | Description             |
|              | <NameArray>                                                                                  | Array of Strings | An Array of Setup Names |
| Return Value | None                                                                                         |                  |                         |

|                |                                                            |
|----------------|------------------------------------------------------------|
| Python Syntax  | DeleteSetups (<NameArray>)                                 |
| Python Example | <code>oModule.DeleteSetups (["OptimizationSetup1"])</code> |

|            |                                                               |
|------------|---------------------------------------------------------------|
| VB Syntax  | DeleteSetups <NameArray>                                      |
| VB Example | <code>oModule.DeleteSetups Array("OptimizationSetup1")</code> |

### DistributedAnalyzeSetup

Distributes all variable value instances within a parametric sweep to different machines already specified from within the user interface

|            |                                                                                           |         |                                                                                                     |
|------------|-------------------------------------------------------------------------------------------|---------|-----------------------------------------------------------------------------------------------------|
| UI Access  | Right-click the parametric setup name in the project tree and select Distribute Analysis. |         |                                                                                                     |
| Parameters | Name                                                                                      | Type    | Description                                                                                         |
|            | <ParametricSetupName>                                                                     | String  | Name of the Setup                                                                                   |
|            | isBlocking                                                                                | Boolean | An optional arg that defaults to <code>true</code> . If it's <code>false</code> , the command imme- |



|                     |      |  |                                                                                                                                                                           |
|---------------------|------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |      |  | diately returns while the analysis or analyses run in the background. The status of the analyses can be checked with the <code>AreThereSimulationsRunning</code> command. |
| <b>Return Value</b> | None |  |                                                                                                                                                                           |

|                       |                                                                  |  |  |
|-----------------------|------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | DistributedAnalyzeSetup (<ParametricSetupName>)                  |  |  |
| <b>Python Example</b> | <code>oModule.DistributedAnalyzeSetup("ParametricSetup1")</code> |  |  |

|                   |                                                                 |  |  |
|-------------------|-----------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | DistributedAnalyzeSetup <ParametricSetupName>                   |  |  |
| <b>VB Example</b> | <code>oModule.DistributedAnalyzeSetup "ParametricSetup1"</code> |  |  |

## ExportDXConfigFile

Create an xml file with the setup information for Design Xplorer

|                     |                                                                                                                                |             |                                                    |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------|
| <b>UI Access</b>    | Right click on the Design Xplorer setup in the project tree and choose <b>Export External Connector Addin Configuration...</b> |             |                                                    |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                    | <b>Type</b> | <b>Description</b>                                 |
|                     | <SetupName>                                                                                                                    | String      | Must be one of existing DesignExplorer setup names |
|                     | <FileName>                                                                                                                     | String      | Must be a valid file path and name                 |
| <b>Return Value</b> | None                                                                                                                           |             |                                                    |

|                       |                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportDXConfigFile (<SetupName>, <FileName>)                                                   |
| <b>Python Example</b> | <pre>oModule.ExportDXConfigFile ("DesignXplorerSetup1",<br/>"c:/exportdir/DXSetup1.xml")</pre> |

|                   |                                                                                                |
|-------------------|------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportDXConfigFile <SetupName>, <FileName>                                                     |
| <b>VB Example</b> | <pre>oModule.ExportDXConfigFile ("DesignXplorerSetup1",<br/>"c:/exportdir/DXSetup1.xml")</pre> |

## ExportOptimetricsProfile

Export Optimetrics profile data

|                     |                                                                                                                                                                                                                        |             |                                                                                                                   |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|-------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right click on the Optimetrics setup in the project tree and choose <b>View Analysis Result...</b> On the <b>Post Analysis Display</b> dialog box, click the <b>Profile</b> tab and click on the <b>Export</b> button. |             |                                                                                                                   |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                            | <b>Type</b> | <b>Description</b>                                                                                                |
|                     | <SetupName>                                                                                                                                                                                                            | String      | Must be one of the existing Parametric, Optimization, Sensitivity, Statistical or DesignXplorer setup names       |
|                     | <FileName>                                                                                                                                                                                                             | String      | Must be a valid file path and name with extension of csv, tab, dat, or txt                                        |
|                     | [profileNum]                                                                                                                                                                                                           | String      | Must be a numeric string. Optional: defaulted to last profile number. It should be a zero indexed profile number. |
| <b>Return Value</b> | None                                                                                                                                                                                                                   |             |                                                                                                                   |

|                       |                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportOptimetricsProfile (<SetupName>, <FileName>, [profileNum])                           |
| <b>Python Example</b> | <pre>oModule.ExportOptimetricsProfile ("StatisticalSetup1", "c:/exportdir/test.csv")</pre> |

|                   |                                                                                          |
|-------------------|------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportOptimetricsProfile <SetupName>, <FileName>, [profileNum]                           |
| <b>VB Example</b> | <pre>oModule.ExportOptimetricsProfile "StatisticalSetup1", "c:/exportdir/test.csv"</pre> |

## ExportOptimetricsResult

Export an existing Optimization, Sensitivity, Statistical or DesignXplorer result. (Does not export Parametric results.)

|                     |                                                                                                                                                                                                                                                              |             |                                                                                                                                                     |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right click on the desired Optimetrics setup in the project tree and choose <b>View Analysis Result...</b> On the <b>Post Analysis Display</b> dialog box, click the <b>Result</b> tab, then select <b>Table</b> view, and click on the <b>Export</b> button |             |                                                                                                                                                     |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                                  | <b>Type</b> | <b>Description</b>                                                                                                                                  |
|                     | <SetupName>                                                                                                                                                                                                                                                  | String      | Must be one of the existing Optimization, Sensitivity, Statistical, or DesignXplorer setup names                                                    |
|                     | <FileName>                                                                                                                                                                                                                                                   | String      | Must be a valid file path and name with extension of csv, tab, dat, or txt..                                                                        |
|                     | [useFullOutputName]                                                                                                                                                                                                                                          | Boolean     | Optional: defaulted to false. If set to true values will be printed with units. This parameter is ignored for Optimization and Statistical results. |
| <b>Return Value</b> | None                                                                                                                                                                                                                                                         |             |                                                                                                                                                     |

|                       |                                                                        |
|-----------------------|------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportOptimetricsResult (<SetupName>, <FileName>, [useFullOutputName]) |
| <b>Python Example</b> | oModule.ExportOptimetricsResult (                                      |

|  |                                                                   |
|--|-------------------------------------------------------------------|
|  | <code>"StatisticalSetup1", "c:/exportdir/test.csv", false)</code> |
|--|-------------------------------------------------------------------|

|                   |                                                                                                      |
|-------------------|------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>ExportOptimetricsResult &lt;SetupName&gt;, &lt;FileName&gt;, [useFullOutputName]</code>        |
| <b>VB Example</b> | <code>oModule.ExportOptimetricsResult "StatisticalSetup1", "c:/exportdir/test.csv",<br/>false</code> |

## ExportParametricResults

Export existing Parametric results.

|                     |                                                                                                                                                                                                                                                              |             |                                                                           |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|---------------------------------------------------------------------------|
| <b>UI Access</b>    | Right click on the desired Parametric setup in the project tree and choose <b>View Analysis Result...</b> On the <b>Post Analysis Display</b> dialog box, click the <b>Result</b> tab, then select <b>Table</b> view, and click on the <b>Export</b> button. |             |                                                                           |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                                  | <b>Type</b> | <b>Description</b>                                                        |
|                     | <SetupName>                                                                                                                                                                                                                                                  | String      | Must be one of the existing Parametric setup names                        |
|                     | <FileName>                                                                                                                                                                                                                                                   | String      | Must be a valid file path and name with extension of csv, tab, dat or txt |
|                     | <bOutputUnits>                                                                                                                                                                                                                                               | Boolean     | If set to true, values will be printed with units                         |
| <b>Return Value</b> | None                                                                                                                                                                                                                                                         |             |                                                                           |

|                       |                                                                                                        |
|-----------------------|--------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ExportParametricResults (&lt;SetupName&gt;, &lt;FileName&gt;, &lt;bOutputUnits&gt;)</code>       |
| <b>Python Example</b> | <code>oModule.ExportParametricResults (<br/>"ParametricSetup1", "c:/exportdir/test.csv", False)</code> |

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportParametricResults <SetupName>, <FileName>, <bOutputUnits>                                   |
| <b>VB Example</b> | <pre>oModule.ExportParametricResults<br/>"ParametricSetup1", "c:/exportdir/test.csv", false</pre> |

## GetSetupNames [Optimetrics]

Gets a list of Optimetrics setup names

|                     |                                                                |      |             |
|---------------------|----------------------------------------------------------------|------|-------------|
| <b>UI Access</b>    | NA                                                             |      |             |
| <b>Parameters</b>   | Name                                                           | Type | Description |
|                     | None                                                           |      |             |
| <b>Return Value</b> | IAnsoftCollectionObj – a collection of Optimetrics setup names |      |             |

|                       |                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetSetupNames()                                                                                |
| <b>Python Example</b> | <pre>oModule = oDesign.GetModule("Optimetrics")<br/>setupNames = oModule.GetSetupNames()</pre> |

|                   |                                                                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSetupNames()                                                                                                                                 |
| <b>VB Example</b> | <pre>Set oModule_opt = oDesign.GetModule("Optimetrics")<br/>Set opt_setup_list = oModule.GetSetupNames()<br/>numsetups = setupnames.Count</pre> |

## GetSetupNamesByType [Optimetrics]

Gets a list of Optimetrics setup names by type.

|                     |                                                     |        |                                                              |
|---------------------|-----------------------------------------------------|--------|--------------------------------------------------------------|
| <b>UI Access</b>    | NA                                                  |        |                                                              |
| <b>Parameters</b>   | Name                                                | Type   | Description                                                  |
|                     | <Optimetrics type>                                  | String | Examples: parametric, optimization, statistical, sensitivity |
| <b>Return Value</b> | Array of Optimetrics setup names of the given type. |        |                                                              |

|                       |                                                                                                  |
|-----------------------|--------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetSetupNamesByType (<Optimetrics type>)                                                         |
| <b>Python Example</b> | <pre>for name in oModule.GetSetupNamesByType("optimization")     AddInfoMessage(str(name))</pre> |

|                   |                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetSetupNamesByType <Optimetrics type>                                                       |
| <b>VB Example</b> | <pre>For each name in oModule.GetSetupNamesByType("optimization")     MsgBox name Next</pre> |

## ImportSetup

Import an Optimetric setup from a file.

|                     |                 |             |                                                                                                                                                                                                         |
|---------------------|-----------------|-------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | NA              |             |                                                                                                                                                                                                         |
| <b>Parameters</b>   | <b>Name</b>     | <b>Type</b> | <b>Description</b>                                                                                                                                                                                      |
|                     | <SetupTypeName> | String      | Must be one of "OptiParametric", "OptiOptimization", "OptiSensitivity", "OptiStatistical", or "OptiDesignExplorer".                                                                                     |
|                     | <SetupInfo>     | Array       | Array("NAME:<SetupName>", "FilePath")<br><br><SetupName><br><br>Type: <string><br><br>Name of the setup.<br><br><FilePath><br><br>Type : <string: file path><br><br>Must be a valid file path and name. |
| <b>Return Value</b> | None            |             |                                                                                                                                                                                                         |

|                       |                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ImportSetup (<SetupTypeName>, <SetupInfo>)                                                                     |
| <b>Python Example</b> | <pre>oModule.ImportSetup ("OptiStatistical", ["NAME:StatisticalSetup1", "c:/importdir/mySetupInfoFile"])</pre> |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | ImportSetup <SetupTypeName>, <SetupInfo> |
| <b>VB Example</b> | oModule.ImportSetup "OptiStatistical",   |

|  |                                                                                     |
|--|-------------------------------------------------------------------------------------|
|  | <pre>Array("NAME:StatisticalSetup1",<br/><br/>"c:/importdir/mySetupInfoFile")</pre> |
|--|-------------------------------------------------------------------------------------|

## PasteSetup [Optimetrics]

Pastes the specified Optimetrics setup.

|                     |             |             |                    |
|---------------------|-------------|-------------|--------------------|
| <b>UI Access</b>    | NA          |             |                    |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b> |
|                     | <SetupName> | String      | Name of the Setup  |
| <b>Return Value</b> | None        |             |                    |

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | PasteSetup (<SetupName>)                             |
| <b>Python Example</b> | <pre>oModule.PasteSetup ("OptimizationSetup1")</pre> |

|                   |                                                    |
|-------------------|----------------------------------------------------|
| <b>VB Syntax</b>  | PasteSetup <SetupName>                             |
| <b>VB Example</b> | <pre>oModule.PasteSetup "OptimizationSetup1"</pre> |

## RenameSetup [Optimetrics]

Renames the specified Optimetrics setup.



|              |                                                                                               |        |                                    |
|--------------|-----------------------------------------------------------------------------------------------|--------|------------------------------------|
| UI Access    | Right-click the setup in the project tree, and then click <b>Rename</b> on the shortcut menu. |        |                                    |
| Parameters   | Name                                                                                          | Type   | Description                        |
|              | <OldName>                                                                                     | String | The name that needs to be replaced |
|              | <NewName>                                                                                     | String | Replacement name                   |
| Return Value | None                                                                                          |        |                                    |

|                |                                                                          |  |  |
|----------------|--------------------------------------------------------------------------|--|--|
| Python Syntax  | RenameSetup (<OldName> <NewName>)                                        |  |  |
| Python Example | <code>oModule.RenameSetup ("OptimizationSetup1" "MyOptimization")</code> |  |  |

|            |                                                                        |  |  |
|------------|------------------------------------------------------------------------|--|--|
| VB Syntax  | RenameSetup <OldName> <NewName>                                        |  |  |
| VB Example | <code>oModule.RenameSetup "OptimizationSetup1" "MyOptimization"</code> |  |  |

## SolveSetup [Optimetrics]

Solves the specified Optimetrics setup.

|              |                                                                                                |         |                                                                                                                                                                                                                                                                             |
|--------------|------------------------------------------------------------------------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Right-click the setup in the project tree, and then click <b>Analyze</b> on the shortcut menu. |         |                                                                                                                                                                                                                                                                             |
| Parameters   | Name                                                                                           | Type    | Description                                                                                                                                                                                                                                                                 |
|              | <SetupName>                                                                                    | String  | Name of the setup to be solved                                                                                                                                                                                                                                              |
|              | isBlocking                                                                                     | Boolean | An optional arg that defaults to <code>true</code> . If it's <code>false</code> , the command immediately returns while the analysis or analyses run in the background. The status of the analyses can be checked with the <code>AreThereSimulationsRunning</code> command. |
| Return Value | None                                                                                           |         |                                                                                                                                                                                                                                                                             |

|                       |                                                        |
|-----------------------|--------------------------------------------------------|
| <b>Python Syntax</b>  | SolveSetup (<SetupName>)                               |
| <b>Python Example</b> | <code>oModule.SolveSetup ("OptimizationSetup1")</code> |

|                   |                                                      |
|-------------------|------------------------------------------------------|
| <b>VB Syntax</b>  | SolveSetup <SetupName>                               |
| <b>VB Example</b> | <code>oModule.SolveSetup "OptimizationSetup1"</code> |

## SolveAllSetup

Solves all Optimetrics setups

|                     |                                                                                                         |             |                                                                                                                                                                                                                                                                             |
|---------------------|---------------------------------------------------------------------------------------------------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click on <b>Optimetrics</b> in Project Manager and select <b>Analyze&gt;All</b> from context menu |             |                                                                                                                                                                                                                                                                             |
| <b>Parameters</b>   | <b>Name</b>                                                                                             | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                                                          |
|                     | isBlocking                                                                                              | Boolean     | An optional arg that defaults to <code>true</code> . If it's <code>false</code> , the command immediately returns while the analysis or analyses run in the background. The status of the analyses can be checked with the <code>AreThereSimulationsRunning</code> command. |
| <b>Return Value</b> | None                                                                                                    |             |                                                                                                                                                                                                                                                                             |

|                       |                                      |
|-----------------------|--------------------------------------|
| <b>Python Syntax</b>  | SolveAllSetup()                      |
| <b>Python Example</b> | <code>oModule.SolveAllSetup()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | SolveAllSetup                      |
| <b>VB Example</b> | <code>oModule.SolveAllSetup</code> |

## Parametric Script Commands

[EditSetup \[Parametric\]](#)

[ExportParametricSetupTable](#)

[GenerateVariationData \[Parametric\]](#)

[InsertSetup \[Parametric\]](#)

### EditSetup [Parametric]

Modifies an existing parametric setup

|                     |                                                                                                   |        |                                                                                      |
|---------------------|---------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click the setup in the project tree, and then click <b>Properties</b> on the shortcut menu. |        |                                                                                      |
| <b>Parameters</b>   | Name                                                                                              | Type   | Description                                                                          |
|                     | <SetupName>                                                                                       | String | Name of the Setup                                                                    |
|                     | <ParametricParams>                                                                                | List   | List that defines the parameters of the parametric setup; examples are listed below. |
| <b>Return Value</b> | None                                                                                              |        |                                                                                      |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | EditSetup (<SetupName>, <ParametricParams>)  |
| <b>Python Example</b> | See <a href="#">EditSetup [Optimization]</a> |

|                   |                                              |
|-------------------|----------------------------------------------|
| <b>VB Syntax</b>  | EditSetup <SetupName>, <ParametricParams>    |
| <b>VB Example</b> | See <a href="#">EditSetup [Optimization]</a> |

## ExportParametricSetupTable

Exports the parametric setup table as a CSV file.

|                     |                                                                               |        |                            |
|---------------------|-------------------------------------------------------------------------------|--------|----------------------------|
| <b>UI Access</b>    | Double-click parametric setup. Select <b>Table</b> tab. Click <b>Export</b> . |        |                            |
| <b>Parameters</b>   | Name                                                                          | Type   | Description                |
|                     | <SetupName>                                                                   | String | Name of the setup.         |
|                     | <filePath>                                                                    | String | Full path for file export. |
| <b>Return Value</b> | None                                                                          |        |                            |

|                       |                                                                                                           |
|-----------------------|-----------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportParametricSetupTable (<SetupName>, <filePath>)                                                      |
| <b>Python Example</b> | <pre>oModule.ExportParametricSetupTable('ParametricSetup1', 'E:/Files/ParametricSetup1_ Table.csv')</pre> |

|                  |                                                                                           |
|------------------|-------------------------------------------------------------------------------------------|
| <b>VB Syntax</b> | ExportParametricSetupTable <SetupName>, <filePath>                                        |
| <b>VB</b>        | <pre>obj.ExportParametricSetupTable "ParametricSetup1", "E:/Files/ParametricSetup1_</pre> |

|                |            |
|----------------|------------|
| <b>Example</b> | Table.csv" |
|----------------|------------|

## GenerateVariationData [Parametric]

Generate variation data before parametric solve for CAD integrated project

**Command:** Right click on the parametric setup in the project tree and choose "Generate Variation Data"

**Syntax:** GenerateVariationData <SetupName>

**Return Value:** None

**Parameters:** <SetupName>

Name of the setup.

*VB Example:*

```
oModule.GenerateVariationData "ParametricSetup1"
```

## InsertSetup [Parametric]

Inserts a new parametric setup.

| UI Access         | Right-click the <b>Optimetrics</b> folder in the project tree, and then click <b>Add&gt; Parametric</b> on the shortcut menu. |       |                                                                                                                                                                                                                                                  |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | Name                                                                                                                          | Type  | Description                                                                                                                                                                                                                                      |
|                   | <Parametric Params>                                                                                                           | Array | Array("NAME:<SetupName>", "SaveFields:=",<br><SaveField>, <StartingPoint>, "Sim. Setups:=",<br><SimSetups>,<br><SweepDefs>, <SweepOps>,<br>Array("NAME:Goals", Array("NAME:Goal",<br><OptiGoalSpec>), ... Array("NAME:Goal",<br><OptiGoalSpec>)) |

|                     |             |                  |                                                                                                                                                                                                                                                                  |
|---------------------|-------------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     | <SetupName> | String           | Name of the parametric setup.                                                                                                                                                                                                                                    |
|                     | <SimSetups> | Array of Strings | An array of Twin Builder solution setup names.                                                                                                                                                                                                                   |
|                     | <SweepDefs> | Array            | Array("NAME:Sweeps",<br>Array("NAME:SweepDefinition", "Variable:=",<br><VarName>, "Data:=", <SweepData>,<br>"Synchronize:=", <SyncNum>), ...<br>Array("NAME:SweepDefinition", "Variable:=",<br><VarName>, "Data:=", <SweepData>,<br>"Synchronize:=", <SyncNum>)) |
|                     | <SweepData> | String           | "<SweepType>, <StartV>, <StopV>, <StepV>"                                                                                                                                                                                                                        |
|                     | <SweepType> | String           | The type of sweep data.                                                                                                                                                                                                                                          |
|                     | <SyncNum>   | Integer          | <i>SweepDatas</i> with the same value are synchronized.                                                                                                                                                                                                          |
|                     | <SweepOps>  |                  | Array("NAME:Sweep Operations", "<OpType>:=",<br>Array(<VarValue>, ..., <VarValue>), ...<br><OpType>:=, Array(<VarValue>, ..., <VarValue>))                                                                                                                       |
|                     | <OpType>    | String           | The sweep operation type.                                                                                                                                                                                                                                        |
| <b>Return Value</b> | None        |                  |                                                                                                                                                                                                                                                                  |

|                       |                                                    |
|-----------------------|----------------------------------------------------|
| <b>Python Syntax</b>  | InsertSetup ("OptiParametric", <ParametricParams>) |
| <b>Python Example</b> | oModule.InsertSetup ("OptiParametric",<br>[        |

```
"NAME:ParametricSetup1",
"SaveFields:=", true,
[
 "NAME:StartingPoint"
],
"Sim. Setups:=", ["Setup1"],
[
 "NAME:Sweeps",
 [
 "NAME:SweepDefinition",
 "Variable:=", "$width",
 "Data:=", "LIN 12mm 17mm 2.5mm",
 "OffsetF1:=", false,
 "Synchronize:=", 0
],
 [
 "NAME:SweepDefinition",
 "Variable:=", "$length",
 "Data:=", "LIN 8mm 12mm 2mm",
 "OffsetF1:=", false,
 "Synchronize:=", 0
]
]
```

```
]
],
[
 "NAME:Sweep Operations"
],
[
 "NAME:Goals",
 [
 "NAME:Goal",
 "Solution:=", "Setup1 : LastAdaptive",
 "Calculation:=", "returnloss",
 "Context:=", ""
]
 "NAME:Ranges",
 "Range:=",
 [
 "Var:=", "Freq", "Type:=", "s"
 "Start:=", "8GHz",
 "Stop:=", "8GHz"
]
]
```



```
],
[
 "NAME:Goal",
 "Solution:=", "Setup1 : LastAdaptive"
 "Calculation:=", "reflect",
 "Context:=", "",
 [
 "NAME:Ranges",
 "Range:=",
 [
 "Var:=", "Freq",
 "Type:=", "s",
 "Start:=", "8GHz",
 "Stop:=", "8GHz"
]
]
]
]
])
```

Example: Parametric Setup with Harmonic Distortions

```
oModule.InsertSetup("OptiParametric",
[
 "NAME:ParametricSetup1",
 "IsEnabled:" , True,
 [
 "NAME:ProdOptiSetupDataV2",
 "SaveFields:" , False,
 "CopyMesh:" , False,
 "SolveWithCopiedMeshOnly:", False
],
 "InterpolationPoints:" , 0,
 "HarmonicDistortionNum:", 10,
 [
 "NAME:StartingPoint"
],
 "Sim. Setups:" , ["Setup1"],
 [
 "NAME:Sweeps"
],
 [
```

|  |                                                                                        |
|--|----------------------------------------------------------------------------------------|
|  | <pre>        "NAME:Sweep Operations"      ],     [         "NAME:Goals"     ] ])</pre> |
|--|----------------------------------------------------------------------------------------|

| VB Syntax         | InsertSetup "OptiParametric", <ParametricParams>                                                                                                                                                                                                                                                                                              |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.InsertSetup "OptiParametric", Array("NAME:ParametricSetup1", _ "SaveFields:=", true, _ Array("NAME:StartingPoint"), _ "Sim. Setups:=", Array("Setup1"), _ Array("NAME:Sweeps", _ Array("NAME:SweepDefinition", _ "Variable:=", "\$width", _ "Data:=", "LIN 12mm 17mm 2.5mm", _ "OffsetFl:=", false, _ "Synchronize:=", 0),</pre> |

```

Array("NAME:SweepDefinition", _
 "Variable:=", "$length", _
 "Data:=", "LIN 8mm 12mm 2mm", _
 "OffsetFl:=", false, _
 "Synchronize:=", 0)),
Array("NAME:Sweep Operations"), _
Array("NAME:Goals", _
Array("NAME:Goal", _
 "Solution:=", "Setup1 : LastAdaptive", _
 "Calculation:=", "returnloss", _
 "Context:=", "", _
 Array("NAME:Ranges", _
 "Range:=", Array("Var:=", "Freq", "Type:=", "s", _
 "Start:=", "8GHz", "Stop:=", "8GHz"))), _
 Array("NAME:Goal", _
 "Solution:=", "Setup1 : LastAdaptive", _
 "Calculation:=", "reflect", _
 "Context:=", "", _
 Array("NAME:Ranges", _

```

```
"Range:=", Array("Var:=", "Freq", "Type:=", "s", _
"Start:=", "8GHz", "Stop:=", "8GHz"))))
```

## Optimization Script Commands

[EditSetup \[Optimization\]](#)

[InsertSetup \[Optimization\]](#)

### EditSetup [Optimization]

Modifies an existing optimization setup.

| UI Access    | Right-click the setup in the Project Manager tree, and then click <b>Properties</b> from the shortcut menu |        |                                                              |
|--------------|------------------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------|
| Parameters   | Name                                                                                                       | Type   | Description                                                  |
|              | <SetupName>                                                                                                | String | Name of the Setup                                            |
|              | <OptimizationParams>                                                                                       | List   | Parameters that define the setup; examples are listed below. |
| Return Value | None                                                                                                       |        |                                                              |

| Python Syntax  | EditSetup (<SetupName>, <OptimizationParams>)                                                                                                                                                     |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditSetup("OptimizationSetup1", [     "NAME:OptimizationSetup1",     "UseFastCalculationUpdateAlgo:=", False,     "FastCalcOptCtrledByUser:=", False,     "IsEnabled:=", True,</pre> |

```
"SaveSolutions:=", False,
[
 "NAME:StartingPoint"
],
"Optimizer:=", "Quasi Newton",
[
 "NAME:AnalysisStopOptions",
 "StopForNumIteration:=", True,
 "StopForElapsTime:=", False,
 "StopForSlowImprovement:=", False,
 "StopForGrdTolerance:=", False,
 "MaxNumIteration:=", 1000,
 "MaxSolTimeInSec:=", 3600,
 "RelGradientTolerance:=", 0,
 "MinNumIteration:=", 10
],
"CostFuncNormType:=", "L2",
"PriorPSetup:=", "",
"PreSolvePSetup:=", True,
```

```
[
 "NAME:Variables"
],
[
 "NAME:LCS"
],
[
 "NAME:Goals",
 [
 "NAME:Goal",
 "ReportType:=", "Standard",
 "Solution:=", "TR",
 [
 "NAME:SimValueContext",
 "SimValueContext:=", [1,0,2,0,False,False,-1,1,0,1,1,""],0,0
]
],
 "Calculation:=", "acosh(Time)",
 "Name:=", "Time",
 [
 "NAME:Ranges",
```

```
 "Range:=", ["Var:=", "Time","Type:=", "a"]
],
 "Condition:=", "==",
 [
 "NAME:GoalValue",
 "GoalValueType:=", "Independent",
 "Format:=", "Real/Imag",
 "bG:=", ["v:=", "[1;]"]
],
 "Weight:=", "[1;]"
],
"Acceptable_Cost:=", 0,
"Noise:=", 0.0001,
"UpdateDesign:=", False,
"UpdateIteration:=", 5,
"KeepReportAxis:=", True,
"UpdateDesignWhenDone:=", True
])
```

Example with Harmonic Distortions



```
oModule.EditSetup("ParametricSetup1",
[
 "NAME:ParametricSetup1",
 "IsEnabled:=" , True,
 [
 "NAME:ProdOptiSetupDataV2",
 "SaveFields:=" , False,
 "CopyMesh:=" , False,
 "SolveWithCopiedMeshOnly:=", False
],
 "InterpolationPoints:=" , 0,
 "HarmonicDistortionNum:=", 12,
 [
 "NAME:StartingPoint"
],
 "Sim. Setups:=" , ["Setup1"],
 [
 "NAME:Sweeps"
],
 [
 "NAME:Sweep Operations"
```

|  |                                                                       |
|--|-----------------------------------------------------------------------|
|  | <pre>    ],<br/>    [<br/>        "NAME:Goals"<br/>    ]<br/>])</pre> |
|--|-----------------------------------------------------------------------|

| VB Syntax  | EditSetup <SetupName>, <OptimizationParams>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditSetup "OptimizationSetup1",<br/>Array("NAME:OptimizationSetup1", "UseFastCalculationUpdateAlgo:=", _<br/>    false, "FastCalcOptCtrledByUser:=", false,<br/>"IsEnabled:=", true, "SaveSolutions:=", _<br/>    false, Array("NAME:StartingPoint"), "Optimizer:=",<br/>"Quasi Newton", Array("NAME:AnalysisStopOptions", "StopForNumIteration:=", _<br/>    true, "StopForElapsTime:=", false,<br/>"StopForSlowImprovement:=", false, "StopForGrdTolerance:=", _<br/>    false, "MaxNumIteration:=", 1000, "MaxSolTimeInSec:=",<br/>3600, "RelGradientTolerance:=", _<br/>    0, "MinNumIteration:=", 10), "CostFuncNormType:=",<br/>"L2", "PriorPSetup:=", "", "PreSolvePSetup:=", _</pre> |

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>true, Array("NAME:Variables"), Array("NAME:LCS"), Array("NAME:Goals", Array("NAME:Goal", "ReportType:=", _     "Standard", "Solution:=", "TR", Array("NAME:SimValueContext",     "SimValueContext:=", Array( _     1, 0, 2, 0, false, false, -1, 1, 0, 1, 1, "", 0, 0)),     "Calculation:=", "Time", "Name:=", _     "Time", Array("NAME:Ranges", "Range:=", Array("Var:=", "Time", "Type:=", "a")), "Condition:=", _     "=="", Array("NAME:GoalValue", "GoalValueType:=",     "Independent", "Format:=", _     "Real/Imag", "bG:=", Array("v:=", "[1;]")),     "Weight:=", "[1;]")), "Acceptable_Cost:=", _     0, "Noise:=", 0.0001, "UpdateDesign:=", false,     "UpdateIteration:=", 5, "KeepReportAxis:=", _     true, "UpdateDesignWhenDone:=", true)</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

InsertSetup [Optimization]

Use: Inserts a new optimization setup.

| UI Access  | Right-click the <b>Optimetrics</b> folder in the project tree, and then click <b>Add &gt; Optimization</b> on the shortcut menu. |       |                                                                                           |
|------------|----------------------------------------------------------------------------------------------------------------------------------|-------|-------------------------------------------------------------------------------------------|
| Parameters | Name                                                                                                                             | Type  | Description                                                                               |
|            | <OptimizationParams>                                                                                                             | Array | Array("NAME:<SetupName>", "SaveFields:=",<br><SaveField>, <StartingPoint>, "Optimizer:=", |

|                        |          |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------------------|----------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                        |          | <pre> &lt;Optimizer&gt;, "MaxIterations:=", &lt;MaxIter&gt;, "PriorPSetup:=", &lt;PriorSetup&gt;, "PreSolvePSetup:=", &lt;Preceed&gt;, &lt;OptimizationVars&gt;, &lt;Constraint&gt;, Array("NAME:Goals", Array("NAME:Goal", &lt;OptiGoalSpec&gt;, &lt;OptimizationGoalSpec&gt;)), ... Array("NAME:Goal", &lt;OptiGoalSpec&gt;, &lt;OptimizationGoalSpec&gt;)), "Acceptable_Cost:=", &lt;AcceptableCost&gt;, "Noise:=", &lt;Noise&gt;, "UpdateDesignWhenDone:=", &lt;UpdateDesign&gt; </pre> |
| <OptimizationVars>     | Array    | <pre> Array("NAME:Variables", "VarName:=", Array("i:=", &lt;IncludeVar&gt;, "Min:=", &lt;MinV&gt;, "Max:=", &lt;MaxV&gt;, "MinStep:=", &lt;MinStepV&gt;, "MaxStep:=", &lt;MaxStepV&gt;)), ..... "VarName:=", Array("i:=", &lt;IncludeVar&gt;, "Min:=", &lt;MinV&gt;, "Max:=", &lt;MaxV&gt;, "MinStep:=", &lt;MinStepV&gt;, "MaxStep:=", &lt;MaxStepV&gt;)) </pre>                                                                                                                           |
| <MinStepV>             | VarValue | The minimum step of the variable.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <MaxStepV>             | VarValue | The maximum step of the variable.                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <AcceptableCost>       | Double   | The acceptable cost value for the optimizer to stop.                                                                                                                                                                                                                                                                                                                                                                                                                                        |
| <Noise>                | Double   | The noise of the design.                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
| <UpdateDesign>         | Boolean  | Specifies whether or not to apply the optimal variation to the design after the optimization is done.                                                                                                                                                                                                                                                                                                                                                                                       |
| <OptimizationGoalSpec> | Array    | <pre> "Condition:=", &lt;OptimizationCond&gt;, </pre>                                                                                                                                                                                                                                                                                                                                                                                                                                       |

|                     |                    |        |                                                                                                                                                              |
|---------------------|--------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                    |        | Array("NAME:GoalValue", "GoalValeType:=",<br><GoalValueType>,<br>"Format:=", <GoalValueFormat>, "bG:=",<br>Array("v:=", <GoalValue>)), "Weight:=", <Weight>) |
|                     | <OptimizationCond> | String | Either "<=", "==", or ">="                                                                                                                                   |
|                     | <GoalValueType>    | String | Either "Independent" or "Dependent"                                                                                                                          |
|                     | <GoalValueFormat>  | String | Either "Real/Imag" or "Mag/Ang".                                                                                                                             |
|                     | <GoalValue>        | String | Value in string. Value can be a real number, complex number, or expression.                                                                                  |
| <b>Return Value</b> | None               |        |                                                                                                                                                              |

<MinStepV>

Type : <VarValue>

The minimum step of the variable.

<MaxStepV>

Type: <VarValue>

The maximum step of the variable.

<AcceptableCost>

Type: <double>

The acceptable cost value for the optimizer to stop.

<Noise>

Type: <double>

The noise of the design.

<UpdateDesign>

Type: <bool>

Specifies whether or not to apply the optimal variation to the design after the optimization is done.

<OptimizationGoalSpec>

"Condition:=", <OptimizationCond>,

Array("NAME:GoalValue", "GoalValueType:=",

<GoalValueType>,

"Format:=", <GoalValueFormat>, "bG:=",

Array("v:=", <GoalValue>)), "Weight:=", <Weight>)

<OptimizationCond>

Type: <string>

Either "<=", "==" or ">="

<GoalValueType>

Type: <string>

Either "Independent" or "Dependent"

<GoalValueFormat>

Type: <string>

Either "Real/Imag" or "Mag/Ang".

<GoalValue>

Type: <string>

Value in string. Value can be a real number, complex number, or expression.

| Python Syntax         | InsertSetup ("OptiOptimization", <OptimizationParams>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oModule.InsertSetup("OptiOptimization", ["NAME:OptimizationSetup1", "SaveFields:=", false, _ ["NAME:StartingPoint", "\$length:=", "8mm", "\$width:=", "14.5mm"], "Optimizer:=", "Quasi Newton", "MaxIterations:=", 100, "PriorPSetup:=", "ParametricSetup1", "PreSolvePSetup:=", true, ["NAME:Variables", "\$length:=", ["i:=", true, "Min:=", "6mm", "Max:=", "18mm", "MinStep:=", "0.001mm", "MaxStep:=", "1.2mm"], "\$width:=", ["i:=", true, "Min:=", "6.5mm", "Max:=", "19.5mm", "MinStep:=", "0.001mm", "MaxStep:=", "1.3mm"]],</pre> |

```
["NAME:LCS"],
["NAME:Goals",
["NAME:Goal",
"Solution:=", "Setup1 : LastAdaptive",
"Calculation:=", "reflect",
"Context:=", "",
["NAME:Ranges",
"Range:=", ["Var:=", "Freq",
"Type:=", "s",
"Start:=", "8GHz", "Stop:=", "8GHz"]],
"Condition:=", "<=",
["NAME:GoalValue",
"GoalValueType:=", "Independent",
"Format:=", "Real/Imag",
"bG:=", ["v:=", "[0.0001]"]],
"Weight:=", "[1]"]],
"Acceptable_Cost:=", 0.0002,
"Noise:=", 0.0001,
"UpdateDesign:=", true,
```



|  |                                                                                                  |
|--|--------------------------------------------------------------------------------------------------|
|  | <pre> "UpdateIteration:=", 5, "KeepReportAxis:=", true, "UpdateDesignWhenDone:=", true l) </pre> |
|--|--------------------------------------------------------------------------------------------------|

| VB Syntax  | InsertSetup "OptiOptimization", <OptimizationParams>                                                                                                                                                                                                                                                                                                                                                                                                                 |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.InsertSetup "OptiOptimization", _ Array("NAME:OptimizationSetup1", _ "SaveFields:=", false, _ Array("NAME:StartingPoint", "\$length:=", "8mm", _ "\$width:=", "14.5mm"), _ "Optimizer:=", "Quasi Newton", _ "MaxIterations:=", 100, _ "PriorPSetup:=", "ParametricSetup1", _ "PreSolvePSetup:=", true, _ Array("NAME:Variables", _ "\$length:=", Array("i:=", true, "Min:=", "6mm", _ "Max:=", "18mm", _ "MinStep:=", "0.001mm", "MaxStep:=", _ </pre> |

```
"1.2mm"), _
"$width:=", Array("i:=", true, "Min:=", _
"6.5mm", "Max:=", "19.5mm", _
"MinStep:=", "0.001mm", "MaxStep:=", _
"1.3mm")), _
Array("NAME:LCS"), _
Array("NAME:Goals", _
Array("NAME:Goal", _
"Solution:=", "Setup1 : LastAdaptive", _
"Calculation:=", "reflect", _
"Context:=", "", _
Array("NAME:Ranges", _
"Range:=", Array("Var:=", "Freq", _
"Type:=", "s", _
"Start:=", "8GHz", "Stop:=", "8GHz")), _
"Condition:=", "<=", _
Array("NAME:GoalValue", _
"GoalValueType:=", "Independent", _
"Format:=", "Real/Imag", _
```

```

"bG:=", Array("v:=", "[0.0001]")), _
"Weight:=", "[1]")),
"Acceptable_Cost:=", 0.0002, _
"Noise:=", 0.0001, _
"UpdateDesign:=", true, _
"UpdateIteration:=", 5, _
"KeepReportAxis:=", true, _
"UpdateDesignWhenDone:=", true)

```

## InsertSetup for System Coupling

Accesses the [System Coupling Setup](#) window for setting up exchange of data\cosimulation with other Ansys solvers.

| UI Access    | Right click on <b>Optimetrics</b> in the Project Manager tree and select <b>Add &gt; System Coupling Setup</b> |        |                                                                                                                                    |
|--------------|----------------------------------------------------------------------------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                                                           | Type   | Description                                                                                                                        |
|              | "OptiSystemCoupling"                                                                                           | string | Designates setup as System Coupling                                                                                                |
|              | <NAME>                                                                                                         | string | Specifies the name of the system coupling setup                                                                                    |
|              | <IsEnabled>                                                                                                    | bool   | Internal command required for system coupling                                                                                      |
|              | <SolveSetupParameters>                                                                                         | Array  | Specifies the parameters of the magnetic simulation that will be used in the system coupling; see the Python example that follows. |
| Return Value | <SystemCouplingQuantityData>                                                                                   | Array  | Sets the quantities that will be exchanged with Maxwell during the system coupling; see the Python example that follows.           |
|              | None                                                                                                           |        |                                                                                                                                    |

**Note:** Some of the commands listed in the examples that follow are generated when a script is recorded in the Maxwell software interface. These internal commands are required. ANSYS, Inc. recommends [recording a script](#) from the interface, and then editing that script to expand functionality.

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | InsertSetup ("OptiSystemCoupling", <NAME>, <IsEnabled>, <SolveSetupParameters>,<SystemCouplingQuantityData>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Python Example | <pre>oModule.InsertSetup("OptiSystemCoupling", [     "NAME:SystemCouplingSetup2",     "IsEnabled:="         , True,     [         "NAME:ProdOptiSetupDataV2",         "SaveFields:="             , True,         "CopyMesh:="             , False,         "SolveWithCopiedMeshOnly:=", True     ],     [         "NAME:StartingPoint"     ],     "Sim. Setups:="         , ["Setup1"],      [         "NAME:Sweeps",         [             "NAME:SweepDefinition",             "Variable:="                 , "SimTime",             "Data:="                 , "5e-05s",             "OffsetF1:="                 , False,             "Synchronize:="                 , 0         ]     ] ])</pre> |

```

]
],
 [
 "NAME:Sweep Operations"
],
 "CostFunctionName:=" , "Cost",
 "CostFuncNormType:=" , "L2",

 [
 "NAME:CostFunctionGoals"
],
 [
 "NAME:ProdSystemCouplingContextData",
 "SimSetup:=" , "Setup1"
],
 [
 "NAME:ProdSystemCouplingQuantityData",
 "IsIncludeTimeChecked:=", True,
 "TimeVariable:=" , "SimTime",
 "ParameterVariablePairs:=", ["Current:=", "SimTime"],
 "IncludedParameters:=" , [
 "Current:=" , True,
 "Electrical Conductivity:=", False,
 "Lorentz Force:=" , False,
 "Surface Loss:=" , False,
 "System Coupling Time:=", False,
 "Temperature:=" , False,
 "Voltage:=" , True,
]
]

```

|  |                                                                                         |
|--|-----------------------------------------------------------------------------------------|
|  | <pre>        "Volume Loss:=" , False]     ],     [         "NAME:Goals"     ] ] )</pre> |
|--|-----------------------------------------------------------------------------------------|

| VB Syntax      | InsertSetup "OptiSystemCoupling",(<NAME>, <IsEnabled>, <SystemCouplingQuantityData>, <IncludedParameters>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.InsertSetup "OptiSystemCoupling"_     Array("NAME:SystemCouplingSetup2", "IsEnabled:=", _ true, _     Array("NAME:ProdOptiSetupDataV2", "SaveFields:=", true, "CopyMesh:=", false,     "SolveWithCopiedMeshOnly:=", _ true), _     Array("NAME:StartingPoint"), "Sim. Setups:=", Array("Setup1"), _     Array("NAME:Sweeps", Array("NAME:SweepDefinition", "Variable:=", _ "SimTime", _     "Data:=", "2e-05s", "OffsetF1:=", false, "Synchronize:=", 0), _     Array("NAME:SweepDefinition", "Variable:=", _ "SimCurrent", "Data:=", "4A", "Off-     setF1:=", false, "Synchronize:=", 0)), _     Array("NAME:Sweep Operations"), "CostFunctionName:=", _ "Cost", "CostFun-     cNormType:=", "L2", _     Array("NAME:CostFunctionGoals"), Array("NAME:ProdSystemCouplingContextData", _     "SimSetup:=", _ "Setup1"), _     Array("NAME:ProdSystemCouplingQuantityData", "IsIncludeTimeChecked:=", _ true,</pre> |

|  |                                                                                                                                                                                                                                                                                                                                                                                                    |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "TimeVariable:=", "SimTime", "ParameterVariablePairs:=", Array("Current:=", _ "SimCurrent"),      "IncludedParameters:=", Array("Current:=", true, "Electrical Conductivity:=", _ false,_      "Lorentz Force:=", false, "Surface Loss:=", false, "System Coupling Time:=", _ false,_      "Temperature:=", false, "Voltage:=", true, "Volume Loss:=", false)),_ Array("NAME:Goals")) </pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## InitializeSystemCoupling

Used for the System Coupling workflow. The function in the interface is accessed by right clicking a *SystemCouplingSetupn* nested under Optimetrics in the Project Manager Tree, and selecting **Generate Configuration Files**.

When you select **Generate Configuration Files**, a System Coupling configuration (.sp) file and .a py file are generated. The .py file contains this line *oDesign.InitializeSystemCoupling("Optimetrics:SystemCouplingSetupn")* and must be preserved.

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| <b>Python Syntax</b>  | InitializeSystemCoupling()                      |
| <b>Python Example</b> | <code>oDesign.InitializeSystemCoupling()</code> |

## Sensitivity Script Commands

[EditSetup \[Sensitivity\]](#)

[InsertSetup \[Sensitivity\]](#)

### EditSetup [Sensitivity]

Modifies an existing sensitivity setup.

|              |                                                                                           |        |                   |
|--------------|-------------------------------------------------------------------------------------------|--------|-------------------|
| UI Access    | Right-click the setup in the project tree, and then click Properties on the shortcut menu |        |                   |
| Parameters   | Name                                                                                      | Type   | Description       |
|              | <SetupName>                                                                               | String | Name of the Setup |
|              |                                                                                           |        |                   |
| Return Value | None                                                                                      |        |                   |

|                |                                                                                                                                                                                                                                                                        |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EditSetup (<SetupName>, <SensitivityParams>)                                                                                                                                                                                                                           |
| Python Example | <pre>oModule.EditSetup("OptimizationSetup1", [ "NAME:OptimizationSetup1", "UseFastCalculationUpdateAlgo:=", False, "FastCalcOptCtrledByUser:=", False, "IsEnabled:=", True, "SaveSolutions:=", False, [ "NAME:StartingPoint" ], "Optimizer:=", "Quasi Newton", [</pre> |



```
"NAME:AnalysisStopOptions",
"StopForNumIteration:=", True,
"StopForElapsTime:=", False,
"StopForSlowImprovement:=", False,
"StopForGrdTolerance:=" , False,
"MaxNumIteration:=", 1001,
"MaxSolTimeInSec:=", 3600,
"RelGradientTolerance:=", 0,
"MinNumIteration:=", 10
],
"CostFuncNormType:=", "L2",
"PriorPSetup:=", "",
"PreSolvePSetup:=", True,
[
"NAME:Variables"
],
[
"NAME:LCS"
],
[
"NAME:Goals",
```

```
[
"NAME:Goal",
"ReportType:=", "Standard",
"Solution:=", "TR3",
[
"NAME:SimValueContext",
"SimValueContext:=", [1,0,2,0,False,False,-1,1,0,1,1,"",0,0]
],
"Calculation:=", "mag(DIFF1.VAL)",
"Name:=", "DIFF1.VAL",
[
"NAME:Ranges",
"Range:=", [
"Var:=", "Time","Type:=", "a"]
],
"Condition:=", "==",
[
"NAME:GoalValue",
"GoalValueType:=", "Independent",
```

|  |                                                                                                                                                                                                                                                                                                |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Format:=", "Real/Imag",<br/>"bG:=", ["v:=", "[1;]"]<br/>],<br/>"Weight:=", "[1;]"<br/>]<br/>],<br/>"Acceptable_Cost:=", 0,<br/>"Noise:=", 0.0001,<br/>"UpdateDesign:=", False,<br/>"UpdateIteration:=", 5,<br/>"KeepReportAxis:=", True,<br/>"UpdateDesignWhenDone:=", True<br/>])</pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | EditSetup <SetupName>, <SensitivityParams>                                                                                                                                        |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EditSetup("OptimizationSetup1",<br/>Array(<br/>"NAME:OptimizationSetup1",<br/>"UseFastCalculationUpdateAlgo:=", False,<br/>"FastCalcOptCtrledByUser:=", False,</pre> |

```
"IsEnabled:=", True,
"SaveSolutions:=", False,
Array(
 "NAME:StartingPoint"
) ,
"Optimizer:=", "Quasi Newton",
Array(
 "NAME:AnalysisStopOptions",
 "StopForNumIteration:=" , True,
 "StopForElapsTime:=", False,
 "StopForSlowImprovement:=", False,
 "StopForGrdTolerance:=", False,
 "MaxNumIteration:=", 1001,
 "MaxSolTimeInSec:=", 3600,
 "RelGradientTolerance:=", 0,
 "MinNumIteration:=" , 10
) ,
"CostFuncNormType:=", "L2",
"PriorPSetup:=", "",
```

```
"PreSolvePSetup:=", True,
Array(
 "NAME:Variables"
) ,
Array(
 "NAME:LCS"
) ,
Array(
 "NAME:Goals",
 Array(
 "NAME:Goal",
 "ReportType:=", "Standard",
 "Solution:=", "TR3",
 Array(
 "NAME:SimValueContext",
 "SimValueContext:=",
 Array(1,0,2,0,False,False,-1,1,0,1,1,"",0,0)
),
 "Calculation:=", "mag(DIFF1.VAL)",
 "Name:=", "DIFF1.VAL",
 Array(

```

```
"NAME:Ranges",
"Range:=", Array("Var:=", "Time","Type:=", "a")
,
"Condition:=", "==",
Array(
"NAME:GoalValue",
"GoalValueType:=", "Independent",
"Format:=", "Real/Imag",
"bG:=", Array("v:=", "Array(1;]")
,
"Weight:=", "Array(1;]"
)
,
"Acceptable_Cost:=", 0,
"Noise:=", 0.0001,
"UpdateDesign:=", False,
"UpdateIteration:=", 5,
"KeepReportAxis:=", True,
"UpdateDesignWhenDone:=", True
```

))

**InsertSetup [Sensitivity]**

Inserts a new sensitivity setup.

| UI Access         | Right-click <b>Optimetrics</b> in the project tree, and then click <b>Add&gt;Sensitivity</b> on the shortcut menu. |               |                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------|--------------------------------------------------------------------------------------------------------------------|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | Name<br><SensitivityParams>                                                                                        | Type<br>Array | Description<br>Array("NAME:<SetupName>", "SaveFields:=",<br><SaveField>, <StartingPoint>, "MaxIterations:=",<br><MaxIter>, "PriorPSetup:=", <PriorSetup>,<br>"PreSolvePSetup:=", <Preceed>, <SensitivityVars>,<br><Constraint>,<br>Array("NAME:Goals", Array("NAME:Goal",<br><OptiGoalSpec>), ..., Array("NAME:Goal",<br><OptiGoalSpec>)), "Primary Goal:=". <PrimaryGoalID>,<br>"PrimaryError:=", <PrimaryError>) |
|                   | <SensitivityVars>                                                                                                  | Array         | Description<br>Array("NAME:Variables",<br>"VarName:=", Array("i:=", <IncludeVar>,<br>"Min:=", <MinV>, "Max:=", <MaxV>,<br>"IDisp:=", <InitialDisp>),...<br>"VarName:=", Array("i:=", <IncludeVar>,<br>"Min:=", <MinV>, "Max:=", <MaxV>,<br>"IDisp:=", <InitialDisp>))                                                                                                                                              |

|                |                                                                                                                                                                                                                                       |                                         |                                                    |               |          |                                                    |                |        |                                         |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|----------------------------------------------------|---------------|----------|----------------------------------------------------|----------------|--------|-----------------------------------------|
|                | <table><tr><td>&lt;InitialDisp&gt;</td><td>VarValue</td><td>Index of the Primary goal. Index starts from zero.</td></tr><tr><td>&lt;PrimaryError&gt;</td><td>Double</td><td>Error associated with the Primary goal.</td></tr></table> |                                         |                                                    | <InitialDisp> | VarValue | Index of the Primary goal. Index starts from zero. | <PrimaryError> | Double | Error associated with the Primary goal. |
|                | <InitialDisp>                                                                                                                                                                                                                         | VarValue                                | Index of the Primary goal. Index starts from zero. |               |          |                                                    |                |        |                                         |
| <PrimaryError> | Double                                                                                                                                                                                                                                | Error associated with the Primary goal. |                                                    |               |          |                                                    |                |        |                                         |
| Return Value   | None                                                                                                                                                                                                                                  |                                         |                                                    |               |          |                                                    |                |        |                                         |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | InsertSetup ("OptiSensitivity", <SensitivityParams>)                                                                                                                                                                                                                                                                                                                                                                                                                    |
| Python Example | <pre>oModule.InsertSetup(<br/>    "OptiSensitivity", _<br/>    ["NAME:SensitivitySetup1", _<br/>    "SaveFields:=", true, _<br/>    ["NAME:StartingPoint"], _<br/>    "MaxIterations:=", 20, _<br/>    "PriorPSetup:=", "", _<br/>    "PreSolvePSetup:=", true, _<br/>    ["NAME:Variables"], _<br/>    ["NAME:LCS"], _<br/>    "NAME:Goals", _<br/>    ["NAME:Goal", _<br/>    "Solution:=", "Setup1 : LastAdaptive", _<br/>    "Calculation:=", "returnloss", _</pre> |



|  |                                                                                                                                                                                                                                                                                                                                                                                                                       |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Context:=", "", _ ["NAME:Ranges", _ "Range:=", ["Var:=", "Freq", "_ Type:=", "s", _ "Start:=", "8GHz", "Stop:=", "8GHz"]]], _ ["NAME:Goal", _ "Solution:=", "Setup1 : LastAdaptive", _ "Calculation:=", "reflect", _ "Context:=", "", _ ["NAME:Ranges", _ "Range:=", ["Var:=", "Freq", _ "Type:=", "s", _ "Start:=", "8GHz", "Stop:=", "8GHz"]]]], _     "Primary Goal:=", 1, _ "PrimaryError:=", 0.001])</pre> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | InsertSetup "OptiSensitivity", <SensitivityParams>                                    |
|------------|---------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.InsertSetup "OptiSensitivity", _ Array("NAME:SensitivitySetup1", _</pre> |

```

"SaveFields:=", true, _
Array("NAME:StartingPoint"), _
"MaxIterations:=", 20, _
"PriorPSetup:=", "", _
"PreSolvePSetup:=", true, _
Array("NAME:Variables"), _
Array("NAME:LCS"), _
Array("NAME:Goals", _
Array("NAME:Goal", _
"Solution:=", "Setup1 : LastAdaptive", _
"Calculation:=", "returnloss", _
"Context:=", "", _
Array("NAME:Ranges", _
"Range:=", Array("Var:=", "Freq", " _
Type:=", "s", _
"Start:=", "8GHz", "Stop:=", "8GHz"))), _
Array("NAME:Goal", _
"Solution:=", "Setup1 : LastAdaptive", _
"Calculation:=", "reflect", _

```

```

"Context:=", "", _
Array("NAME:Ranges", _
"Range:=", Array("Var:=", "Freq", _
"Type:=", "s", _
"Start:=", "8GHz", "Stop:=", "8GHz"))), _
"Primary Goal:=", 1, _
"PrimaryError:=", 0.001)

```

## Statistical Script Commands

[EditSetup \[Statistical\]](#)

[InsertSetup Statistical](#)

### EditSetup [Statistical]

Modifies an existing statistical setup.

|                     |                                                                                              |             |                    |
|---------------------|----------------------------------------------------------------------------------------------|-------------|--------------------|
| <b>UI Access</b>    | Right-click the setup in the project tree, and click <b>Properties</b> on the shortcut menu. |             |                    |
| <b>Parameters</b>   | <b>Name</b>                                                                                  | <b>Type</b> | <b>Description</b> |
|                     | <SetupName>                                                                                  | String      | Name of the Setup  |
|                     |                                                                                              |             |                    |
| <b>Return Value</b> | None                                                                                         |             |                    |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | EditSetup (<SetupName>, <StatisticalParams>) |
| <b>Python Example</b> | See EditSetup [Optimization]                 |

|                   |                                            |
|-------------------|--------------------------------------------|
| <b>VB Syntax</b>  | EditSetup <SetupName>, <StatisticalParams> |
| <b>VB Example</b> | See EditSetup [Optimization]               |

*Example:*

```

Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("optiguides")
Set oDesign = oProject.SetActiveDesign("HFSSModel1")
Set oModule = oDesign.GetModule("Optimetrics")

oModule.EditSetup "StatisticalSetup1", Array("NAME:StatisticalSetup1",
Array("NAME:ProdOptiSetupData",
"SaveFields=", true, "CopyMesh=", false),
Array("NAME:StartingPoint", "$length=", "7.824547736mm",

```

```
"$width:=", "14.8570192mm"),
"MaxIterations:=", 50,
"PriorPSetup:=", "",

Array("NAME:Variables",
"$length:=", Array("i:=", true,
"int:=", false,
"Dist:=", "Uniform",
"Tol:=", "10%", "StdD:=", "0.2mm", "Min:=", "-3",
"Max:=", "3", "Shape:=", "1", "Scale:=", "0.04mm",
"Location:=", "0.4mm",
"Dataset:=", "", "LatinHypercube:=", "true", "VarMin:=", "0.2mm", "VarMax:=", "0.6mm", "Prob:=",
"0.01",
"Mean:=", "0.4mm"),

"$width:=", Array("i:=", true,
"int:=", false,
"Dist:=", "Gaussian",
"Tol:=", "10%",
"StdD:=", "0.2mm",
"Min:=", "-3", "Max:=", "3",
"Shape:=", "1",
"Scale:=", "0.04mm",
"Location:=", "0.4mm",
"Dataset:=", "",
"LatinHypercube:=", "true",
"VarMin:=", "0.2mm", "VarMax:=", "0.6mm",
"Prob:=", "0.02",
"Mean:=", "0.4mm")),

Array("NAME:Goals", Array("NAME:Goal",
"ReportType:=", "Modal Solution Data",
"Solution:=", "Setup1 : PortOnly",
Array("NAME:SimValueContext", "Domain:=", "Sweep"),
"Calculation:=", "returnloss",
"Name:=", "returnloss",
```

```
Array("NAME:Ranges",
"Range:=", Array("Var:=", "Freq",
"Type:=", "s",
"Start:=", "8.2GHz", "Stop:=", "0"))),

Array("NAME:Goal",
"ReportType:=", "Modal Solution Data",
"Solution:=", "Setup1 : PortOnly",
Array("NAME:SimValueContext",
"Domain:=", "Sweep"),
"Calculation:=", "reflect",
"Name:=", "reflect",
Array("NAME:Ranges",
"Range:=", Array("Var:=", "Freq",
"Type:=", "s",
"Start:=", "8.2GHz", "Stop:=", "0")))))
```

InsertSetup [Statistical]

Inserts a new statistical setup.

| UI Access  | Right-click <b>Optimetrics</b> in the project tree, and then click <b>Add&gt;Statistical</b> on the shortcut menu. |       |                                                                                                                                                                                                                                                                                                           |
|------------|--------------------------------------------------------------------------------------------------------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                                                               | Type  | Description                                                                                                                                                                                                                                                                                               |
|            | <StatisticalParams>                                                                                                | Array | Array("NAME:<SetupName>", "SaveFields:=",<br><br><SaveField>, <StartingPoint>, "MaxIterations:=",<br><br><MaxIter>, "PriorPSetup:=", <PriorSetup>,<br><br>"PreSolvePSetup:=", <Preceed>, <StatisticalVars>,<br><br>Array("NAME:Goals", Array("NAME:Goal",<br><br><OptiGoalSpec>), ..., Array("NAME:Goal", |

|                     |                   |          |                                                                                                                                                                                                                                                                                                                                                          |
|---------------------|-------------------|----------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                   |          | <OptiGoalSpec>)))                                                                                                                                                                                                                                                                                                                                        |
|                     | <StatisticalVars> | Array    | Array("NAME:Variables",<br>"VarName:=", Array("i:=", <IncludeVar>, "Dist:=",<br><DistType>, "Tol:=", <Tolerance>,<br>"StdD:=", <StdD>, "Min:=", <MinCutoff>, "Max:=",<br><MaxCutoff>, ...<br>"VarName:=", Array("i:=", <IncludeVar>, "Dist:=",<br><DistType>, "Tol:=", <Tolerance>, "StdD:=",<br><StdD>, "Min:=", <MinCutoff>, "Max:=",<br><MaxCutoff>)) |
|                     | <DistType>        | String   | Distribution can be "Gaussian" or "Uniform".                                                                                                                                                                                                                                                                                                             |
|                     | <Tolerance>       | VarValue | The tolerance for the variable when distribution is Uniform                                                                                                                                                                                                                                                                                              |
|                     | <StdD>            | VarValue | The standard deviation for the variable when distribution is Gaussian.                                                                                                                                                                                                                                                                                   |
|                     | <MinCutoff>       | Double   | The minimum cut-off for the variable when distribution is Gaussian.                                                                                                                                                                                                                                                                                      |
|                     | <MaxCutoff>       | Double   | The maximum cut-off for the variable when distribution is Gaussian.                                                                                                                                                                                                                                                                                      |
| <b>Return Value</b> | None              |          |                                                                                                                                                                                                                                                                                                                                                          |

|                       |                                                                                                           |
|-----------------------|-----------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | InsertSetup ("OptiStatistical", <StatisticalParams>)                                                      |
| <b>Python Example</b> | <pre>oModule.InsertSetup( "OptiStatistical", _ ["NAME:StatisticalSetup1", _ "SaveFields:=", true, _</pre> |

```

["NAME:StartingPoint"],_
"MaxIterations:=", 50,_
"PriorPSetup:=", "", _
["NAME:Variables"], _
["NAME:Goals", _
["NAME:Goal", _
"Solution:=", "Setup1 : LastAdaptive", _
"Calculation:=", "returnloss", _
"Context:=", "", _
["NAME:Ranges", _
"Range:=", ["Var:=", "Freq", _
"Type:=", "s",_
"Start:=", "8GHz", "Stop:=", "8GHz"]]],_
["NAME:Goal",_
"Solution:=", "Setup1 : LastAdaptive",_
"Calculation:=", "reflect",_
"Context:=", "", _
["NAME:Ranges",_
"Range:=", ["Var:=", "Freq", "Type:=",_

```



|  |                                               |
|--|-----------------------------------------------|
|  | "s", "Start:=", "8GHz", "Stop:=", "8GHz"]]]]) |
|--|-----------------------------------------------|

| VB Syntax         | InsertSetup "OptiStatistical", <StatisticalParams>                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre> oModule.InsertSetup "OptiStatistical", _ Array("NAME:StatisticalSetup1", _ "SaveFields:=", true, _ Array("NAME:StartingPoint"), _ "MaxIterations:=", 50, _ "PriorPSetup:=", "", _ Array("NAME:Variables"), _ Array("NAME:Goals", _ Array("NAME:Goal", _ "Solution:=", "Setup1 : LastAdaptive", _ "Calculation:=", "returnloss", _ "Context:=", "", _ Array("NAME:Ranges", _ "Range:=", Array("Var:=", "Freq", _ "Type:=", "s", _ "Start:=", "8GHz", "Stop:=", "8GHz"))), _ </pre> |

|                                                                                                                                                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>Array("NAME:Goal",_ "Solution:=", "Setup1 : LastAdaptive",_ "Calculation:=", "reflect",_ "Context:=", "", _ Array("NAME:Ranges",_ "Range:=", Array("Var:=", "Freq", "Type:=",_ "s", "Start:=", "8GHz", "Stop:=", "8GHz"))))</pre> |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### *Example:*

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("OptimTee")
```

```
Set oDesign = oProject.SetActiveDesign("TeeModel")

oDesign.ChangeProperty Array("NAME:AllTabs",
Array("NAME:LocalVariableTab",
Array("NAME:PropServers", "LocalVariables"),
Array("NAME:ChangedProps",
Array("NAME:offset",
Array("NAME:Statistical", "Included:=", true))))))

Set oModule = oDesign.GetModule("Optimetrics")

oModule.InsertSetup "OptiStatistical", Array("NAME:StatisticalSetup1",
Array("NAME:ProdOptiSetupData",
"SaveFields:=", false, "CopyMesh:=", false),
Array("NAME:StartingPoint", "offset:=", "0in"),
"MaxIterations:=", 50, "PriorPSetup:=", "",
Array("NAME:Variables",
"offset:=", Array("i:=", true,
"int:=", false,
"Dist:=", "Gaussian",
"Tol:=", "10%",
"StdD:=", ".5in",
"Min:=", "-3",
"Max:=", "3",
"Shape:=", "1",
"Scale:=", "0in",
"Location:=", "0in",
"Dataset:=", "",
"LatinHypercube:=", "true",
"VarMin:=", "-1in",
"VarMax:=", "1in",
"Prob:=", "0.01",
"Mean:=", "0in")),
Array("NAME:Goals", Array("NAME:Goal",
```

```
"ReportType:=", "Modal Solution Data",
"Solution:=", "Setup1 : LastAdaptive", Array("NAME:SimValueContext"),

"Calculation:=", "Power11",
"Name:=", "Power11",
Array("NAME:Ranges",
"Range:=", Array("Var:=", "Freq", "Type:=", "d",
"DiscreteValues:=", "10GHz")))))
```

### For Q3D Extractor and Circuit the command details are as follows:

Inserts a new statistical setup.

**Command:** Right-click **Optimetrics** in the project tree, and then click **Add>Statistical** on the shortcut menu.

**Syntax:** InsertSetup "OptiStatistical", <StatisticalParams>

**Return Value:** None

**Parameters:** <StatisticalParams>

```
Array("NAME:<SetupName>", "SaveFields:=",
<SaveField>, <StartingPoint>, "MaxIterations:=",
<MaxIter>, "PriorPSetup:=", <PriorSetup>,
"PreSolvePSetup:=", <Preceed>, <StatisticalVars>,
Array("NAME:Goals", Array("NAME:Goal",
<OptiGoalSpec>), ..., Array("NAME:Goal",
<OptiGoalSpec>))),
<StatisticalVars>
```

```
Array("NAME:Variables",
 "VarName:=", Array("i:=", <IncludeVar>, "Dist:=",
 <DistType>, "Tol:=", <Tolerance>,
 "StdD:=", <StdD>, "Min:=", <MinCutoff>, "Max:=",
 <MaxCutoff>, ...
 "VarName:=", Array("i:=", <IncludeVar>, "Dist:=",
 <DistType>, "Tol:=", <Tolerance>, "StdD:=",
 <StdD>, "Min:=", <MinCutoff>, "Max:=",
 <MaxCutoff>))
```

**Parameters:**

<DistType>

Type : <string>

Distribution can be "Gaussian" or "Uniform".

<Tolerance>

Type: <VarValue>

The tolerance for the variable when distribution is Uniform.

<StdD>

Type: <VarValue>

The standard deviation for the variable when distribution is Gaussian.

<MinCutoff>

Type: <double>

The minimum cut-off for the variable when distribution is Gaussian.

<MaxCutoff>

Type: <double>

The maximum cut-off for the variable when distribution is Gaussian.

```
Example: oModule.InsertSetup "OptiStatistical", _
Array("NAME:StatisticalSetup1", _
 "SaveFields:=", true, _
 Array("NAME:StartingPoint"), _
 "MaxIterations:=", 50, _
 "PriorPSetup:=", "", _
 Array("NAME:Variables"), _
 Array("NAME:Goals", _
 Array("NAME:Goal", _
 "Solution:=", "Setup1 : LastAdaptive", _
 "Calculation:=", "returnloss", _
 "Context:=", "", _
 Array("NAME:Ranges", _
 "Range:=", Array("Var:=", "Freq", _
 "Type:=", "s", _
 "Start:=", "8GHz", "Stop:=", "8GHz"))), _
```

```
Array("NAME:Goal", _
"Solution:=", "Setup1 : LastAdaptive", _
"Calculation:=", "reflect", _
"Context:=", "", _
Array("NAME:Ranges", _
"Range:=", Array("Var:=", "Freq", "Type:=", _
"s", "Start:=", "8GHz", "Stop:=", "8GHz")))))
```

This page intentionally  
left blank.



## 17 - Solutions Module Script Commands

Solutions commands should be executed by the "Solutions" module.

Set oModule = oDesign.GetModule("Solutions")

oModule.CommandName <args>

[DeleteSolutionVariation](#)

[ExportNetworkData](#)

"ExportTransientData " on page 17-11

"FFTONReport " on page 17-12

"GetAdaptiveSettings " on page 17-13

"GetAllSourceMagnitudes " on page 17-14

"GetAllSourceModes " on page 17-14

"GetAllSourcePhases " on page 17-15

"GetAllSources " on page 17-16

"GetAntennaParameters " on page 17-16

"GetFieldType " on page 17-17

"GetIncludePortPostProcessing " on page 17-18

"GetLayerAndNetOfMeshBody " on page 17-18

[GetMeshBodiesOnLayerAndNet](#)

"GetMultipactionBreakdown " on page 17-23

"GetNetworkDataSolution " on page 17-24

"GetNetworkDataSolutionDefinition " on page 17-25

"GetNetworkPostprocSetup " on page 17-25

"GetTerminalExcitationType " on page 17-26

"GetTransientSolveTimes " on page 17-27

[GetValidSolutionList](#)

[TDROnReport](#)

# ExportLayoutLayerGridField

Export the maximum field value on the top/bottom surface of a layer within a user-defined grid.

| UI Access  | Maxwell > Export Layout Field Data > Grid Max Field Data |         |                                                                                                                            |
|------------|----------------------------------------------------------|---------|----------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                     | Type    | Description                                                                                                                |
|            | fieldExpression                                          | string  | Field type for calculation, supports "Mag_J", "Mag_B", and "Mag_H"                                                         |
|            | layoutComponent                                          | string  | Specifies the layout component to be used for the export                                                                   |
|            | layerName                                                | string  | Specifies which layer to perform the calculation. Use "All-Layers" to create an export field that includes all the layers. |
|            | isTopFace                                                | boolean | True for calculating on the top face of the layer. False for specifying the bottom face.                                   |
|            | gridSize                                                 | string  | Specifies the length of each cell of the user-defined grid.                                                                |
|            | solutionName                                             |         | Maxwell design solution name, for example, "Setup1:Transient"                                                              |
|            | context                                                  | Array   | Structured array to specify Time and other geometry variables; for example, ["Time:":"1ms"]                                |
|            | exportFile                                               | string  | Specifies the path of export file. Supports macros like \$ProjectDir                                                       |
|            | threshold                                                | string  | Specifies the minimum field value for the data to be exported                                                              |
|            | fieldUnit                                                | string  | Specifies unit for exporting the field data                                                                                |
|            | lengthUnit                                               | string  | Specifies the unit for exporting the grid coordinates.                                                                     |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ExportLayoutLayerGridField (fieldExpression, layoutComponent, layerName, isTopFace, gridSize, solutionName, context, exportFile, threshold, fieldUnit, lengthUnit)</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
| <b>Python Example</b> | <pre>oProject = oDesktop.SetActiveProject("SimpleGrid") oDesign = oProject.SetActiveDesign("Maxwell3DDesign1") oModule = oDesign.GetModule("Solutions") oModule.ExportLayoutLayerGridField("Mag_J", "LC1_1", "PWR", True, "5mm", "Setup1: Transient", [     "Time:=" , "20.00000ms" ], "\$ProjectDir/Mag_J.json", "0", "A_per_m2", "mm")  oModule.ExportLayoutLayerGridField("Mag_B", "LC1_1", "PWR", True, "5mm", "Setup1: Transient", [     "Time:=" , "20.00000ms" ], "\$ProjectDir/Mag_B.json", "0", "mTesla", "mm")  oModule.ExportLayoutLayerGridField("Mag_H", "LC1_1", "PWR", True, "5mm", "Setup1: Transient", [</pre> |

```
"Time:=" , "20.00000ms"
], "$ProjectDir/Mag_H.json", "0", "kA_per_m", "mm")
```

| VB Syntax  | ExportLayoutLayerGridField fieldExpression, layoutComponent, layerName, isTopFace, gridSize, solutionName, context, exportFile, threshold, fieldUnit, lengthUnit                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>Set oProject = oDesktop.SetActiveProject("SimpleGrid")<br/>Set oDesign = oProject.SetActiveDesign("Maxwell3DDesign1")<br/>Set oModule = oDesign.GetModule("Solutions")<br/>oModule.ExportLayoutLayerGridField "Mag_B", "LC1_1", "PWR", true, "5", _<br/>"Setup1 : Transient", Array("Time:=", "20.00000ms"), _<br/>"\$ProjectDir/gird_mag_flux.json", "0", "mTesla", "mm"<br/><br/>oModule.ExportLayoutLayerGridField "Mag_H", "LC1_1", "All_Layers", true, "5", _<br/>"Setup1 : Transient", Array("Time:=", "20.00000ms"), _<br/>"\$ProjectDir/gird_mag_field_strength.json", "0", "kA_per_meter", "mm"<br/><br/>oModule.ExportLayoutLayerGridField "Mag_J", "LC1_1", "GND", true, "5", _<br/>"Setup1 : Transient", Array("Time:=", "20.00000ms"), _<br/>"\$ProjectDir/grid_current_density.json", "0", "A_per_m2", "mm"</pre> |

## ExportLayoutViaCurrent

Export the current between signal layers for all the vias in a layout component.

| UI Access    | Maxwell > Export Layout Field Data > Export Via Current |        |                                                                                              |
|--------------|---------------------------------------------------------|--------|----------------------------------------------------------------------------------------------|
| Parameters   | Name                                                    | Type   | Description                                                                                  |
|              | layoutComponent                                         | string | Name of layout component to be used for the export                                           |
|              | solutionName                                            | string | Maxwell design solution name, e.g. Setup1:Transient                                          |
|              | context                                                 | Array  | Structured array to specify Time and other geometry variables; for example, ["Time:=":"1ms"] |
|              | exportFile                                              | string | Specifies the path of export file. Supports macros like \$ProjectDir                         |
|              | threshold                                               | string | Specifies the minimum field value for the data to be exported                                |
|              | fieldUnit                                               | string | Specifies unit for exporting the field data                                                  |
|              | lengthUnit                                              | string | Specifies the unit for exporting the grid coordinates.                                       |
| Return Value | None                                                    |        |                                                                                              |

| Python Syntax  | ExportLayoutViaCurrent(layoutComponent, solutionName, context, exportFile, threshold, fieldUnit, lengthUnit)                                                                                                                                                                               |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oProject = oDesktop.SetActiveProject("ViaTest") oDesign = oProject.SetActiveDesign("Maxwell3DDesign1") oModule = oDesign.GetModule("Solutions") oModule.ExportLayoutViaCurrent("LC1_1", "Setup1:Transient", ["Time:=", "20ms"], "\$ProjectDir/current.json", "0mA", "mA", "mm")</pre> |

| VB Syntax | ExportLayoutViaCurrent layoutComponent, solutionName, context, exportFile, threshold, fieldUnit, lengthUnit |
|-----------|-------------------------------------------------------------------------------------------------------------|
|-----------|-------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>Set oProject = oDesktop.SetActiveProject("ViaTest")  Set oDesign = oProject.SetActiveDesign("Maxwell3DDesign1")  Set oModule = oDesign.GetModule("Solutions")  oModule.ExportLayoutViaCurrent "LC1_1", "Setup1 : Transient", Array("Time:=", _ "20ms"), "\$ProjectDir/current.json", "0mA", "mA", "mm"</pre> |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## ExportLayoutViaCurrentDensity

Export the average current density between signal layers for all the vias in a layout component.

|                     |                                                                                      |        |                                                                                              |
|---------------------|--------------------------------------------------------------------------------------|--------|----------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | <b>Maxwell &gt; Export Layout Field Data &gt; Export Via Average Current Density</b> |        |                                                                                              |
| <b>Parameters</b>   | Name                                                                                 | Type   | Description                                                                                  |
|                     | layoutComponent                                                                      | string | Name of layout component to be used for the export                                           |
|                     | solutionName                                                                         | string | Maxwell design solution name, e.g. Setup1:Transient                                          |
|                     | context                                                                              | Array  | Structured array to specify Time and other geometry variables; for example, ["Time:=":"1ms"] |
|                     | exportFile                                                                           | string | Specifies the path of export file. Supports macros like \$ProjectDir                         |
|                     | threshold                                                                            | string | Specifies the minimum field value for the data to be exported                                |
|                     | fieldUnit                                                                            | string | Specifies unit for exporting the field data                                                  |
|                     | lengthUnit                                                                           | string | Specifies the unit for exporting the grid coordinates.                                       |
| <b>Return Value</b> | None                                                                                 |        |                                                                                              |

|                      |                                                                                                         |
|----------------------|---------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b> | ExportLayoutViaCurrentDensity(layoutComponent, solutionName, context, exportFile, threshold, fieldUnit, |
|----------------------|---------------------------------------------------------------------------------------------------------|

|                       |                                                                                                                                                                                                                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                       | lengthUnit)                                                                                                                                                                                                                                                                                                    |
| <b>Python Example</b> | <pre>oProject = oDesktop.SetActiveProject("SimpleVia") oDesign = oProject.SetActiveDesign("Maxwell3DDesign1") oModule = oDesign.GetModule("Solutions") oModule.ExportLayoutViaCurrentDensity("LC1_1", "Setup1:Transient", ["Time:- :=", "20ms"], "\$ProjectDir/current_density.json", "0mA", "mA", "mm")</pre> |

|                   |                                                                                                                                                                                                                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportLayoutViaCurrentDensity layoutComponent, solutionName, context, exportFile, threshold, fieldUnit, lengthUnit                                                                                                                                                                                                                      |
| <b>VB Example</b> | <pre>Set oProject = oDesktop.SetActiveProject("ViaTest") Set oDesign = oProject.SetActiveDesign("Maxwell3DDesign1") Set oModule = oDesign.GetModule("Solutions") oModule.ExportLayoutViaCurrentDensity "LC1_1", "Setup1 : Transient", Array("Time:=", _ "20.00000ms"), "\$ProjectDir/current_density.json", "0", "A_per_m2", "mm"</pre> |

## DeleteSolutionVariation

Deletes all solution data for specific solutions and design variations. This is obsolete and is supported only for backward compatibility. You should use DeleteFullVariation. See also [DeleteVariation](#).

|                  |                                                                                                               |
|------------------|---------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b> | Right-click on <b>Results</b> , select <b>Browse Solutions...</b> , click <b>Delete</b> button in the dialog. |
|------------------|---------------------------------------------------------------------------------------------------------------|

|              |                               |       |                                                                                   |
|--------------|-------------------------------|-------|-----------------------------------------------------------------------------------|
| Parameters   | Name                          | Type  | Description                                                                       |
|              | < <i>SoluParams</i> >         | Array | Structured array.<br><br>Array(<DataSpecifierArray>, ...)                         |
|              | < <i>DataSpecifierArray</i> > | Array | Structured array.<br><br>Array(<DesignVariationKey>, <SetupName>, <SolutionName>) |
| Return Value | None.                         |       |                                                                                   |

|                |                                                                                                                                                  |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | DeleteSolutionVariation(< <i>SoluParams</i> >)                                                                                                   |
| Python Example | <pre>oModule.DeleteSolutionVariation([<br/>    ["width='2in'", "Setup1", "Adaptive_1"],<br/>    ["width='2in'", "Setup1", "Sweep1"]<br/>])</pre> |

|            |                                                                                                                                                               |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | DeleteSolutionVariation < <i>SoluParams</i> >                                                                                                                 |
| VB Example | <pre>oModule.DeleteSolutionVariation Array( _<br/>    Array("width='2in'", "Setup1", "Adaptive_1"), _<br/>    Array("width='2in'", "Setup1", "Sweep1"))</pre> |



## ExportNetworkData

Exports matrix solution data to a file.

| UI Access  | N/A                  |         |                                                                                                                                                                                                                             |
|------------|----------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                 | Type    | Description                                                                                                                                                                                                                 |
|            | <DesignVariationKey> | String  | Design variation key. Pass empty string for the current nominal variation.                                                                                                                                                  |
|            | <SolnSelectionArray> | Array   | Array of selected solutions.<br><br><code>Array(&lt;SolnSelector&gt;, &lt;SolnSelector&gt;, ...)</code><br><br>If more than one array entry, this indicates a combined Interpolating sweep.                                 |
|            | <SolnSelector>       | String  | Solution setup name and solution name, separated by a colon.                                                                                                                                                                |
|            | <FileFormat>         | Integer | File format value.<br><br>2 : Tab delimited spreadsheet format (.tab)<br>3 : Touchstone (.sNp)<br>4 : CitiFile (.cit)<br>7 : Matlab (.m)<br>8 : Terminal Z0 spreadsheet                                                     |
|            | <OutFile>            | String  | Full path to the file to write out.                                                                                                                                                                                         |
|            | <FreqsArray>         | Array   | The frequencies to export. The <FreqsArray> argument contains a vector (e.g. "1GHz", "2GHz", ...) to use, or "all". To export all frequencies, use Array("all"). If no frequencies are specified, all frequencies are used. |
|            | <DoRenorm>           | Boolean | Specifies whether to renormalize the data before export.                                                                                                                                                                    |
|            | <RenormImped>        | Double  | Real impedance value in ohms, for renormalization. Required in syntax, but ignored if DoRenorm is false.                                                                                                                    |
|            | <DataType>           | String  | Optional. Type: "S", "Y", or "Z". The matrix to export.                                                                                                                                                                     |

|              |                                     |         |                                                                                                                                                              |
|--------------|-------------------------------------|---------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <i>&lt;pass&gt;</i>                 | Integer | Optional. The pass to export. This is ignored if the sourceName is a frequency sweep. Leaving out this value or specifying -1 gets all passes.               |
|              | <i>&lt;ComplexFormat&gt;</i>        | Integer | Optional. Type: "0", "1", or "2"<br><br>The format to use for the exported data.<br><br>0 = Magnitude/Phase.<br><br>1 = Real/Imaginary.<br><br>2 = db/Phase. |
|              | <i>&lt;Precision&gt;</i>            | Integer | Optional. Touchstone number of digits precision. Default if not specified is 15.                                                                             |
|              | <i>&lt;UseExportFreqs&gt;</i>       | Boolean | Specifies whether to use export frequencies.                                                                                                                 |
|              | <i>&lt;IncludeGammaComments&gt;</i> | Boolean | Specifies whether to include Gamma and Impedance comments.                                                                                                   |
|              | <i>&lt;SupportNonStdExport&gt;</i>  | Boolean | Specifies whether to support non-standard Touchstone extensions for mixed reference impedance.                                                               |
| Return Value | None.                               |         |                                                                                                                                                              |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ExportNetworkData( <i>&lt;DesignVariationKey&gt;</i> , <i>&lt;SolnSelectionArray&gt;</i> , <i>&lt;SolnSelector&gt;</i> , <i>&lt;FileFormat&gt;</i> , <i>&lt;OutFile&gt;</i> , <i>&lt;FreqsArray&gt;</i> , <i>&lt;DoRenorm&gt;</i> , <i>&lt;RenormImped&gt;</i> , [Optional <i>&lt;DataType&gt;</i> ], [Optional <i>&lt;pass&gt;</i> ], [Optional <i>&lt;ComplexFormat&gt;</i> ], [Optional <i>&lt;Precision&gt;</i> ], [Optional <i>&lt;UseExportFreqs&gt;</i> ], [Optional <i>&lt;IncludeGammaComments&gt;</i> ], [Optional <i>&lt;SupportNonStdExport&gt;</i> ]) |
| Python Example |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |

|           |                                                                                                                                                                              |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax | ExportNetworkData <i>&lt;DesignVariationKey&gt;</i> , <i>&lt;SolnSelectionArray&gt;</i> , <i>&lt;SolnSelector&gt;</i> , <i>&lt;FileFormat&gt;</i> , <i>&lt;OutFile&gt;</i> , |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                                                                  |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                   | <code>&lt;FreqsArray&gt;, &lt;DoRenorm&gt;, &lt;RenormImped&gt;, [Optional &lt;DataType&gt;], [Optional &lt;pass&gt;], [Optional &lt;ComplexFormat&gt;], [Optional &lt;Precision&gt;], [Optional &lt;UseExportFreqs&gt;], [Optional &lt;IncludeGammaComments&gt;], [Optional &lt;SupportNonStdExport&gt;]</code> |
| <b>VB Example</b> | <pre>oModule.ExportNetworkData "width='2in'", _     Array("Setup1:Sweep1"), 2, "c:\mydir\out.tab", _     Array("all"), false, 0  oModule.ExportNetworkData "width='2in'", _     Array("Setup1:Sweep1", "Setup1:Sweep2"), 3, _     "c:\mydir\out.s2p", Array(1.0e9, 1.5e9, 2.0e9), _     true, 50.0</pre>         |

## ExportTransientData

Exports transient solution data to a file.

| UI Access         | HFSS > Results > Solution Data..., then click <b>Export</b> . |        |                                                                            |
|-------------------|---------------------------------------------------------------|--------|----------------------------------------------------------------------------|
| <b>Parameters</b> | Name                                                          | Type   | Description                                                                |
|                   | <code>&lt;SoluName&gt;</code>                                 | String | Name of specified solution.                                                |
|                   | <code>&lt;Variation&gt;</code>                                | String | Design variation key. Pass empty string for the current nominal variation. |
|                   | <code>&lt;FileName&gt;</code>                                 | String | Full path of output file.                                                  |
|                   | <code>&lt;DataType&gt;</code>                                 | String | Optional. Type of output data.                                             |

|              |                                                                                                                                                                                                                                                                         |                                                                                                                                                                                      |        |                                                                                                                                                                                      |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <table><tr><td>&lt;DataFormat&gt;</td><td>String</td><td>Optional. Format of output data.<br/><br/>.csv - Comma Delimited Data.<br/><br/>.tab - Tab Delimited Data files.<br/><br/>.dat - Ansys Plot Data files<br/><br/>.txt - Post Processing Files</td></tr></table> | <DataFormat>                                                                                                                                                                         | String | Optional. Format of output data.<br><br>.csv - Comma Delimited Data.<br><br>.tab - Tab Delimited Data files.<br><br>.dat - Ansys Plot Data files<br><br>.txt - Post Processing Files |
| <DataFormat> | String                                                                                                                                                                                                                                                                  | Optional. Format of output data.<br><br>.csv - Comma Delimited Data.<br><br>.tab - Tab Delimited Data files.<br><br>.dat - Ansys Plot Data files<br><br>.txt - Post Processing Files |        |                                                                                                                                                                                      |
| Return Value | None.                                                                                                                                                                                                                                                                   |                                                                                                                                                                                      |        |                                                                                                                                                                                      |

|                |                                                                                            |
|----------------|--------------------------------------------------------------------------------------------|
| Python Syntax  | <code>ExportTransientData(&lt;SoluName&gt;, &lt;Variation&gt;, &lt;FileName&gt;)</code>    |
| Python Example | <code>oModule.ExportTransientData("Setup1:Transient", "", "C:/transient_solu1.csv")</code> |

|            |                                                                                           |
|------------|-------------------------------------------------------------------------------------------|
| VB Syntax  | <code>ExportTransientData &lt;SoluName&gt;, &lt;Variation&gt;, &lt;FileName&gt;</code>    |
| VB Example | <code>oModule.ExportTransientData "Setup1:Transient", "", "C:/transient_solu1.csv"</code> |

## FFTONReport

Performs an FFT on a selected report.

|            |                                                                                           |        |                                                                                                                                            |
|------------|-------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | Right-click on <b>Results</b> in the project tree, select <b>Perform FFT On Report...</b> |        |                                                                                                                                            |
| Parameters | Name                                                                                      | Type   | Description                                                                                                                                |
|            | <code>&lt;ReportName&gt;</code>                                                           | String | Name of specified report.                                                                                                                  |
|            | <code>&lt;WindowName&gt;</code>                                                           | String | Name of window to apply for FFT. Possible values are "Rectangular", "Tri", "Van Hann", "Hamming", "Blackman", "Lanczos", "Weber", "Welch". |

|              |                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                             |        |                                                                                                                                                                                                             |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <table><tr><td>&lt;Function&gt;</td><td>String</td><td>Function to apply on transformation values. Possible values are "none", "ang_deg", "ang_rad", "arg", "cang_deg", "cang_rad", "dB", "dB1 normalize", "dB20normalize", "dBc", "im", "mag", "normalize", "re".</td></tr></table> | <Function>                                                                                                                                                                                                  | String | Function to apply on transformation values. Possible values are "none", "ang_deg", "ang_rad", "arg", "cang_deg", "cang_rad", "dB", "dB1 normalize", "dB20normalize", "dBc", "im", "mag", "normalize", "re". |
| <Function>   | String                                                                                                                                                                                                                                                                               | Function to apply on transformation values. Possible values are "none", "ang_deg", "ang_rad", "arg", "cang_deg", "cang_rad", "dB", "dB1 normalize", "dB20normalize", "dBc", "im", "mag", "normalize", "re". |        |                                                                                                                                                                                                             |
| Return Value | None.                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                             |        |                                                                                                                                                                                                             |

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | FFTONReport <ReportName>, <WindowName>, <Function>    |
| <b>Python Example</b> | oModule.FFTONReport("XY Plot 1", "Rectangular", "dB") |

|                   |                                                      |
|-------------------|------------------------------------------------------|
| <b>VB Syntax</b>  | FFTONReport(<ReportName>, <WindowName>, <Function>)  |
| <b>VB Example</b> | oModule.FFTONReport "XY Plot 1", "Rectangular", "dB" |

## GetAdaptiveSettings

Obtains the adaptive frequency settings of a specified setup.

|                     |                                                         |        |                          |
|---------------------|---------------------------------------------------------|--------|--------------------------|
| <b>UI Access</b>    | N/A                                                     |        |                          |
| <b>Parameters</b>   | Name                                                    | Type   | Description              |
|                     | <SetupName>                                             | String | Name of specified setup. |
| <b>Return Value</b> | Array of stings represents adaptive frequency settings. |        |                          |

|                       |                                       |
|-----------------------|---------------------------------------|
| <b>Python Syntax</b>  | GetAdaptiveSettings(<SetupName>)      |
| <b>Python Example</b> | oModule.GetAdaptiveSettings("Setup1") |

|                   |                                                   |
|-------------------|---------------------------------------------------|
| <b>VB Syntax</b>  | GetAdaptiveSettings <SetupName>                   |
| <b>VB Example</b> | <code>oModule.GetAdaptiveSettings "Setup1"</code> |

## GetAllSourceMagnitudes

Obtains the magnitude values of all defined sources.

|                     |                                               |
|---------------------|-----------------------------------------------|
| <b>UI Access</b>    | N/A                                           |
| <b>Parameters</b>   | None.                                         |
| <b>Return Value</b> | Array of strings containing magnitude values. |

|                       |                                               |
|-----------------------|-----------------------------------------------|
| <b>Python Syntax</b>  | GetAllSourceMagnitudes()                      |
| <b>Python Example</b> | <code>oModule.GetAllSourceMagnitudes()</code> |

|                   |                                             |
|-------------------|---------------------------------------------|
| <b>VB Syntax</b>  | GetAllSourceMagnitudes                      |
| <b>VB Example</b> | <code>oModule.GetAllSourceMagnitudes</code> |

## GetAllSourceModes

Obtains mode information of all defined sources.

|                     |                                               |
|---------------------|-----------------------------------------------|
| <b>UI Access</b>    | N/A                                           |
| <b>Parameters</b>   | None.                                         |
| <b>Return Value</b> | Array of strings containing mode information. |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAllSourceModes()</code>         |
| <b>Python Example</b> | <code>oModule.GetAllSourceModes()</code> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | <code>GetAllSourceModes</code>         |
| <b>VB Example</b> | <code>oModule.GetAllSourceModes</code> |

## GetAllSourcePhases

Obtains the phase values of all defined sources.

|                     |                                           |
|---------------------|-------------------------------------------|
| <b>UI Access</b>    | N/A                                       |
| <b>Parameters</b>   | None.                                     |
| <b>Return Value</b> | Array of strings containing phase values. |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | <code>GetAllSourcePhases()</code>         |
| <b>Python Example</b> | <code>oModule.GetAllSourcePhases()</code> |

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>VB Syntax</b>  | GetAllSourcePhases                      |
| <b>VB Example</b> | <code>oModule.GetAllSourcePhases</code> |

## GetAllSources

Retrieves all sources defined in the current solution setup.

|                     |                                           |
|---------------------|-------------------------------------------|
| <b>UI Access</b>    | N/A                                       |
| <b>Parameters</b>   | None.                                     |
| <b>Return Value</b> | Array of strings containing source names. |

|                       |                                      |
|-----------------------|--------------------------------------|
| <b>Python Syntax</b>  | GetAllSources()                      |
| <b>Python Example</b> | <code>oModule.GetAllSources()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | GetAllSources                      |
| <b>VB Example</b> | <code>oModule.GetAllSources</code> |

## GetAntennaParameters

Retrieves antenna parameters defined in a specified setup.

|                  |     |
|------------------|-----|
| <b>UI Access</b> | N/A |
|------------------|-----|



|              |                                                 |        |                          |
|--------------|-------------------------------------------------|--------|--------------------------|
| Parameters   | Name                                            | Type   | Description              |
|              | <SetupName>                                     | String | Name of specified setup. |
| Return Value | Array of strings containing antenna parameters. |        |                          |

|                |                                                     |
|----------------|-----------------------------------------------------|
| Python Syntax  | GetAntennaParameters(<SetupName>)                   |
| Python Example | <code>oModule.GetAntennaParameters("Setup1")</code> |

|            |                                                    |
|------------|----------------------------------------------------|
| VB Syntax  | GetAntennaParameters <SetupName>                   |
| VB Example | <code>oModule.GetAntennaParameters "Setup1"</code> |

## GetFieldType

Gets the field type of a driven modal design solution.

|              |                               |
|--------------|-------------------------------|
| UI Access    | N/A                           |
| Parameters   | None.                         |
| Return Value | String containing field type. |

|                |                                     |
|----------------|-------------------------------------|
| Python Syntax  | GetFieldType()                      |
| Python Example | <code>oModule.GetFieldType()</code> |

|                   |                                   |
|-------------------|-----------------------------------|
| <b>VB Syntax</b>  | GetFieldType                      |
| <b>VB Example</b> | <code>oModule.GetFieldType</code> |

## GetIncludePortPostProcessing

Determines whether the Include Port Post Processing Effects option is enabled.

|                     |                                                                                                                                                                                        |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                                                                                                                                                                    |
| <b>Parameters</b>   | None.                                                                                                                                                                                  |
| <b>Return Value</b> | <p>Integer.</p> <ul style="list-style-type: none"><li>• 1 - Include Port Post Processing Effects is enabled.</li><li>• 0 - Include Port Post Processing Effects is disabled.</li></ul> |

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>Python Syntax</b>  | GetIncludePortPostProcessing()                      |
| <b>Python Example</b> | <code>oModule.GetIncludePortPostProcessing()</code> |

|                   |                                                   |
|-------------------|---------------------------------------------------|
| <b>VB Syntax</b>  | GetIncludePortPostProcessing                      |
| <b>VB Example</b> | <code>oModule.GetIncludePortPostProcessing</code> |

## GetLayerAndNetOfMeshBody

For a given mesh body name, returns a pair of layer and net name to which this body belongs.

|                     |                                                                  |
|---------------------|------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                                              |
| <b>Parameters</b>   | The input is a mesh body name.                                   |
| <b>Return Value</b> | Returns a pair of layer and net name to which this body belongs. |

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>GetLayerAndNetOfMeshBody()</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>Python Example</b> | <pre>LayerAndNetOfMeshBody("LC1_1:METAL-1") print layerNetInfo #prints ["METAL-1", "trace_P"]</pre> <p>Example usage to plot on the objects on certain layers and nets using GetLayerAndNetOfMeshBody with GetMeshBodiesOnLayerAndNet.</p> <pre>meshBodies = oModule.GetMeshBodiesOnLayerAndNet(<br/>    [<br/>        "NAME:LayerNetInfo",<br/>        "METAL-1:=" , ["trace_P", "ground"]<br/>    ]<br/>)<br/><br/>plotGeomInfo = [1, "Volume", "ObjList", ]<br/>plotGeomInfo.append(len(meshBodies))<br/>plotGeomInfo += meshBodies<br/><br/>oFieldsReporterModule = oDesign.GetModule("FieldsReporter")<br/>oFieldsReporterModule.CreateFieldPlot(<br/>    [<br/>        "NAME:Mesh1",</pre> |

```
"SolutionName:=" , "Setup1 : LastAdaptive",
"UserSpecifyName:=" , 0,
"UserSpecifyFolder:=" , 0,
"QuantityName:=" , "Mesh",
"PlotFolder:=" , "MeshPlots",
"FieldType:=" , "Fields",
"StreamlinePlot:=" , False,
"AdjacentSidePlot:=" , False,
"FullModelPlot:=" , False,
"IntrinsicVar:=" , "Freq=\'5GHz\' Phase=\'0deg\'",
"PlotGeomInfo:=" , plotGeomInfo,
"FilterBoxes:=" , [0],
"Real time mode:=" , True,
[
 "NAME:MeshSettings",
 "Scale factor:=" , 100,
 "Transparency:=" , 0,
 "Mesh type:=" , "Shaded",
 "Surface only:=" , True,
 "Add grid:=" , True,
```

```
"Refinement:=" , 0,
"Use geometry color:=" , True,
"Mesh line color:=" , [0,0,255],
"Filled color:=" , [255,255,255]
],
"EnableGaussianSmoothing:=", False,
"SurfaceOnly:=" , False
], "Field")
```

## GetMeshBodiesOnLayerAndNet

Gets the field type of a driven modal design solution.

|                     |                                                                                                            |
|---------------------|------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                                                                                        |
| <b>Parameters</b>   | The inputs are a layer net block and a vector of component names (optional).                               |
| <b>Return Value</b> | The output is a vector of mesh body names that belong to the specified nets and layers and the component.. |

|                       |                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetMeshBodiesOnLayerAndNet()                                                                                   |
| <b>Python Example</b> | <pre>import ScriptEnv<br/>ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")<br/>oDesktop.RestoreWindow()</pre> |

```

oProject = oDesktop.SetActiveProject("Tests")
oDesign = oProject.SetActiveDesign("Reference_multiple")
oModule = oDesign.GetModule("Solutions")
meshBodies = oModule.GetMeshBodiesOnLayerAndNet(
 [
 "NAME:LayerNetInfo",
 "METAL-1:=" , ["trace_P","ground"]
])
print meshBodies #prints["LC1_1:METAL-1", "LC1_2:METAL-1"]
meshBodiesForLC1_1 = oModule.GetMeshBodiesOnLayerAndNet(
 [
 "NAME:LayerNetInfo",
 "METAL-1:=" , ["trace_P","ground"]
], ["LC1_1"])
print meshBodiesForLC1_1 #prints ["LC1_1:METAL-1"]
moreMeshBodies = oModule.GetMeshBodiesOnLayerAndNet(
 [
 "NAME:LayerNetInfo",
 "METAL-1:=" , ["trace_P","Ground"],
 "METAL-2:=" , ["TRACE_p","GROUND"] #case insensitive
])

```

```

])

print moreMeshBodies #prints ['LC1_1:METAL-1', 'LC1_2:METAL-1', 'LC1_1:METAL-2',
 'LC1_2:METAL-2']

moreMeshBodiesForLC1_1 = oModule.GetMeshBodiesOnLayerAndNet(

 [
 "NAME:LayerNetInfo",
 "METAL-1:=" , ["trace_P","ground"],
 "METAL-2:=" , ["trace_P","ground"]
], ["LC1_1"])

print moreMeshBodiesForLC1_1 #prints['LC1_1:METAL-1', 'LC1_1:METAL-2']

```

## GetMultipactionBreakdown

Gets multipaction breakdown from the specified setup.

| UI Access    | N/A                                                    |        |                                                                        |
|--------------|--------------------------------------------------------|--------|------------------------------------------------------------------------|
| Parameters   | Name                                                   | Type   | Description                                                            |
|              | <SetupName>                                            | String | Name of specified setup.                                               |
|              | <Variation>                                            | String | Design variation. Pass empty string for the current nominal variation. |
| Return Value | Array of string containing the multipaction breakdown. |        |                                                                        |

| Python Syntax | GetMultipactionBreakdown(<SetupName>, <Variation>) |
|---------------|----------------------------------------------------|
|---------------|----------------------------------------------------|

|                       |                                                             |
|-----------------------|-------------------------------------------------------------|
| <b>Python Example</b> | <code>oModule.GetMultipactionBreakdown("Setup1", "")</code> |
|-----------------------|-------------------------------------------------------------|

|                   |                                                                            |
|-------------------|----------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetMultipactionBreakdown &lt;SetupName&gt;, &lt;Variation&gt;</code> |
| <b>VB Example</b> | <code>oModule.GetMultipactionBreakdown "Setup1", ""</code>                 |

## GetNetworkDataSolution

Gets matrix data solution from a specified setup.

|                     |                                |             |                                                                            |
|---------------------|--------------------------------|-------------|----------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                            |             |                                                                            |
| <b>Parameters</b>   | <b>Name</b>                    | <b>Type</b> | <b>Description</b>                                                         |
|                     | <SoluName>                     | String      | Name of solution.                                                          |
|                     | <VariationKey>                 | String      | Design variation key. Pass empty string for the current nominal variation. |
| <b>Return Value</b> | String containing matrix data. |             |                                                                            |

|                       |                                                                             |
|-----------------------|-----------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetNetworkDataSolution(&lt;SoluName&gt;, &lt;VariationKey&gt;)</code> |
| <b>Python Example</b> | <code>oModule.GetNetworkDataSolution("Setup1:Sweep1", "")</code>            |

|                   |                                                                            |
|-------------------|----------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetNetworkDataSolution &lt;SoluName&gt;, &lt;VariationKey&gt;</code> |
| <b>VB Example</b> | <code>oModule.GetNetworkDataSolution "Setup1:Sweep1", ""</code>            |



## GetNetworkDataSolutionDefinition

Gets definition of network data solution from a specified setup.

|                     |                                            |        |                             |
|---------------------|--------------------------------------------|--------|-----------------------------|
| <b>UI Access</b>    | N/A                                        |        |                             |
| <b>Parameters</b>   | Name                                       | Type   | Description                 |
|                     | <SoluName>                                 | String | Name of specified solution. |
| <b>Return Value</b> | String containing network data definition. |        |                             |

|                       |                                                                        |
|-----------------------|------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetNetworkDataSolutionDefinition(<SoluName>)                           |
| <b>Python Example</b> | <code>oModule.GetNetworkDataSolutionDefinition("Setup1:Sweep1")</code> |

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetNetworkDataSolutionDefinition <SoluName>                           |
| <b>VB Example</b> | <code>oModule.GetNetworkDataSolutionDefinition "Setup1:Sweep1"</code> |

## GetNetworkPostprocSetup

Obtains network postprocessing setup.

|                     |                                         |
|---------------------|-----------------------------------------|
| <b>UI Access</b>    | N/A                                     |
| <b>Parameters</b>   | None.                                   |
| <b>Return Value</b> | String containing postprocessing setup. |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | GetNetworkPostprocSetup()                      |
| <b>Python Example</b> | <code>oModule.GetNetworkPostprocSetup()</code> |

|                   |                                              |
|-------------------|----------------------------------------------|
| <b>VB Syntax</b>  | GetNetworkPostprocSetup                      |
| <b>VB Example</b> | <code>oModule.GetNetworkPostprocSetup</code> |

## GetTerminalExcitationType

Obtains the type for terminal excitation in a driven terminal design.

|                     |                                    |
|---------------------|------------------------------------|
| <b>UI Access</b>    | N/A                                |
| <b>Parameters</b>   | None.                              |
| <b>Return Value</b> | String containing excitation type. |

|                       |                                                  |
|-----------------------|--------------------------------------------------|
| <b>Python Syntax</b>  | GetTerminalExcitationType()                      |
| <b>Python Example</b> | <code>oModule.GetTerminalExcitationType()</code> |

|                   |                                                |
|-------------------|------------------------------------------------|
| <b>VB Syntax</b>  | GetTerminalExcitationType                      |
| <b>VB Example</b> | <code>oModule.GetTerminalExcitationType</code> |

## GetTransientSolveTimes

Gets the transient solution solve time.

|              |                                         |        |                                                                        |
|--------------|-----------------------------------------|--------|------------------------------------------------------------------------|
| UI Access    | N/A                                     |        |                                                                        |
| Parameters   | Name                                    | Type   | Description                                                            |
|              | <SoluName>                              | String | Name of specified solution.                                            |
|              | <Variation>                             | String | Design variation. Pass empty string for the current nominal variation. |
| Return Value | Array of strings containing solve time. |        |                                                                        |

|                |                                                                     |  |  |
|----------------|---------------------------------------------------------------------|--|--|
| Python Syntax  | GetTransientSolveTimes(<SoluName>, <Variation>)                     |  |  |
| Python Example | <code>oModule.GetTransientSolveTimes("Setup1:Transient", "")</code> |  |  |

|            |                                                                    |  |  |
|------------|--------------------------------------------------------------------|--|--|
| VB Syntax  | GetTransientSolveTimes <SoluName>, <Variation>                     |  |  |
| VB Example | <code>oModule.GetTransientSolveTimes "Setup1:Transient", ""</code> |  |  |

## GetValidSolutionList

Gets all available solution names that exist in a design.

|            |                            |         |                                                                              |
|------------|----------------------------|---------|------------------------------------------------------------------------------|
| UI Access  | N/A                        |         |                                                                              |
| Parameters | Name                       | Type    | Description                                                                  |
|            | <IncludeImportedSolutions> | Boolean | Optional. Specifies whether to include imported solutions. Default is false. |

|                     |                                             |
|---------------------|---------------------------------------------|
| <b>Return Value</b> | Array of strings containing solution names. |
|---------------------|---------------------------------------------|

|                       |                                                                                 |
|-----------------------|---------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetValidISolutionList([Optional &lt;IncludeImportedSolutions&gt;])</code> |
| <b>Python Example</b> | <code>oModule.GetValidISolutionList()</code>                                    |

|                   |                                                                                |
|-------------------|--------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetValidISolutionList [Optional &lt;IncludeImportedSolutions&gt;]</code> |
| <b>VB Example</b> | <code>oModule.GetValidISolutionList</code>                                     |

## TDROnReport

Performs a time-domain reflectometry (TDR) analysis on a specified report.

|                     |                                                             |         |                                                                                                                              |
|---------------------|-------------------------------------------------------------|---------|------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click on <b>Results &gt; Perform TDR On Report...</b> |         |                                                                                                                              |
| <b>Parameters</b>   | Name                                                        | Type    | Description                                                                                                                  |
|                     | <ReportName>                                                | String  | Name of specified report.                                                                                                    |
|                     | <IsStep>                                                    | Boolean | Specifies the input signal as Step or Impulse.                                                                               |
|                     | <RiseTime>                                                  | Double  | Specifies rise time with unit in 'ps'.                                                                                       |
|                     | <WindowName>                                                | String  | Name of window to apply. Possible values are "Rectangular", "Bartlett", "Blackman", "Hamming", "Hanning", "Kaiser", "Welch". |
|                     | <WindowWidth>                                               | Integer | Width of window in percentage.                                                                                               |
|                     | <KaiserParams>                                              | Double  | Optional. If Kaiser window is specified, set corresponding parameters.                                                       |
| <b>Return Value</b> | None.                                                       |         |                                                                                                                              |

|                       |                                                                                                                                                           |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>TDROnReport(&lt;ReportName&gt;, &lt;IsStep&gt;, &lt;RiseTime&gt;, &lt;WindowName&gt;, &lt;WindowWidth&gt;, [Optional &lt;Kaiser-Params&gt;])</code> |
| <b>Python Example</b> | <code>oModule.TDROnReport("Rept2DRectFreq", True, 1E-12, "Hanning", 100, 1)</code>                                                                        |

|                   |                                                                                                                                                          |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>TDROnReport &lt;ReportName&gt;, &lt;IsStep&gt;, &lt;RiseTime&gt;, &lt;WindowName&gt;, &lt;WindowWidth&gt;, [Optional &lt;Kaiser-Params&gt;]</code> |
| <b>VB Example</b> | <code>oModule.TDROnReport "Rept2DRectFreq", true, 1E-12, "Hanning", 100, 1</code>                                                                        |

This page intentionally  
left blank.

## 18 - Field Overlays Module Script Commands

Field overlay commands should be executed by the Field Overlays module, which is called "FieldsReporter" in scripts.

Set oModule = oDesign.GetModule("FieldsReporter")

oModule.CommandName <args>

AddAntennaOverlay

[AddMarkerToPlot](#)

[CreateFieldPlot](#)

[DeleteFieldPlot](#)

[EditSurfaceMeshSummaryData](#)

[ExportMarkerTable](#)

[ExportPlotImageWithViewToFile](#)

[ExportSurfaceMeshSummary](#)

[GetFieldPlotName](#)

HideAntennaParametersOverlay

[ModifyFieldPlot](#)

[ModifyInceptionParameters](#)

[RenameFieldPlot](#)

[RenamePlotFolder](#)

[SetFieldPlotSettings](#)

[SetPlotFolderSettings](#)

SetPlotsSolutionsContext

[SetPlotsViewSolutionContext](#)

[UpdateAllFieldsPlots](#)

[UpdateQuantityFieldsPlots](#)

# AddMarkerToPlot

Adds a marker to a trace on a named field plot.

|              |            |        |                                                                    |
|--------------|------------|--------|--------------------------------------------------------------------|
| UI Access    | N/A        |        |                                                                    |
| Parameters   | Name       | Type   | Description                                                        |
|              | <Location> | Array  | Array of strings containing X.Y, and Z coordinates for the marker. |
|              | <PlotName> | String | Name of the field plot.                                            |
| Return Value | None.      |        |                                                                    |

|                |                                                                                                                                                                             |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AddMarkerToPlot( <Location>, <PlotName>)                                                                                                                                    |
| Python Example | <pre>oModule.AddMarkerToPlot (["0.290455877780914in", "-0.616900205612183in", "1.77635683940025e-015in"], "Mag_H1") oModule.AddMarkerToPlot (["-0.317279517650604in",</pre> |



|  |                                                          |
|--|----------------------------------------------------------|
|  | <code>"1.22481322288513in", "0in"],<br/>"Mag_H1")</code> |
|--|----------------------------------------------------------|

| VB Syntax  | AddMarkerToPlot <Location>, <PlotName>                                                                                                                                                                                                                            |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AddMarkerToPlot _<br/>Array("0.290455877780914in", _<br/>"-0.616900205612183in", "1.77635683940025e-015in"), _<br/>"Mag_E1"<br/>oModule.AddMarkerToPlot _<br/>Array("-0.317279517650604in", _<br/>"1.22481322288513in", "0in"), _<br/>"Mag_E1"</pre> |

## AddMarker[Fields Reporter]

Adds a marker to the current fields plot.

| UI Access    | Right-click <b>Field Overlays &gt; Plot Fields &gt; Marker &gt; Add Marker.</b> |       |                                          |
|--------------|---------------------------------------------------------------------------------|-------|------------------------------------------|
| Parameters   | Name                                                                            | Type  | Description                              |
|              | <Position>                                                                      | Array | Array containing X, Y and Z coordinates. |
| Return Value | None.                                                                           |       |                                          |

|                       |                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddMarker(<Position>)                                                                                                  |
| <b>Python Example</b> | <pre>oModule.AddMarker([     "-267.007756778494mil",     "-640.898759461403mil",     "9.09494701772928e-13mil"])</pre> |

|                   |                                                                                                                             |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AddMarker <Position>                                                                                                        |
| <b>VB Example</b> | <pre>oModule.AddMarker _ Array("-267.007756778494mil", _     "-640.898759461403mil", _     "9.09494701772928e-13mil")</pre> |

AddNamedExpression

Creates a named expression using the expression at the top of the stack.

|                   |                                                          |        |                                                                                                                                                                                          |
|-------------------|----------------------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Click <b>Add</b> in the <b>Fields Calculator</b> window. |        |                                                                                                                                                                                          |
| <b>Parameters</b> | Name                                                     | Type   | Description                                                                                                                                                                              |
|                   | <ExprName>                                               | String | Name for the new named expression.                                                                                                                                                       |
|                   | <FieldType>                                              | String | Type of field.                                                                                                                                                                           |
|                   | <Expression>                                             | Array  | Defines an expression to be solved in the calculator. It can be an Algebraic or Stacked expression; the Algebraic option requires a beta feature; see <a href="#">Algebraic option</a> . |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                                           |
|-----------------------|-----------------------------------------------------------|
| <b>Python Syntax</b>  | AddNamedExpression(<ExprName>, <FieldType>, <Expression>) |
| <b>Python Example</b> | oModule.AddNamedExpression("Mag_JxE", "Fields")           |

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| <b>VB Syntax</b>  | AddNamedExpression <ExprName>, <FieldType>, <Expression> |
| <b>VB Example</b> | oModule.AddNamedExpression "Mag_JxE", "Fields"           |

Maxwell users can create an expression that use the [Algebraic option](#). In this case, no Stack commands are necessary; the user can define the expression within AddNamedExpression command:

```
oModule.AddNamedExpression("new2", "Fields",
[
 "Algebraic:=" , "Mean(VolumeValue(Box1), Mag_B)"
])
```

The same command in Stack form would look like this:

```
oModule.AddNamedExpression("new2", "Fields",
[
 "NameOfExpression:=" , ["Mag_B"],
 "EnterVolume:=" , ["Box1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
```

])

For information, see [Syntax for Algebraic Option](#).

## CalcOp

Performs a calculator operation.

|              |                                                      |        |                                                  |
|--------------|------------------------------------------------------|--------|--------------------------------------------------|
| UI Access    | Operation commands like <b>Mag</b> , <b>+</b> , etc. |        |                                                  |
| Parameters   | Name                                                 | Type   | Description                                      |
|              | <OpString>                                           | String | The text on the corresponding calculator button. |
| Return Value | None.                                                |        |                                                  |

|                |                      |
|----------------|----------------------|
| Python Syntax  | CalcOp(<OpString>)   |
| Python Example | oModule.CalcOp ("+") |

|            |                    |
|------------|--------------------|
| VB Syntax  | CalcOp <OpString>  |
| VB Example | oModule.CalcOp "+" |

## CalcStack

Performs an operation on the stack.

|           |                                                              |
|-----------|--------------------------------------------------------------|
| UI Access | Stack operation buttons such as <b>Push</b> and <b>Pop</b> . |
|-----------|--------------------------------------------------------------|

|              |            |        |                                                  |
|--------------|------------|--------|--------------------------------------------------|
| Parameters   | Name       | Type   | Description                                      |
|              | <OpString> | String | The text on the corresponding calculator button. |
| Return Value | None.      |        |                                                  |

|                |                           |
|----------------|---------------------------|
| Python Syntax  | CalcStack(<OpString>)     |
| Python Example | oModule.CalcStack("push") |

|            |                          |
|------------|--------------------------|
| VB Syntax  | CalcStack <OpString>     |
| VB Example | oModule.CalcStack "push" |

## CalculatorRead

Gets a register file and applies it to the calculator stack.

|              |                                                              |        |                                                    |
|--------------|--------------------------------------------------------------|--------|----------------------------------------------------|
| UI Access    | Click <b>Read...</b> in the <b>Fields Calculator</b> dialog. |        |                                                    |
| Parameters   | Name                                                         | Type   | Description                                        |
|              | <FileName>                                                   | String | Path to and including name of input register file. |
|              | <SoluName>                                                   | String | Specified solution to read in.                     |
|              | <FieldType>                                                  | String | Type of specified field.                           |
|              | <VarArray>                                                   | Array  | Array of variable names, value pairs.              |
| Return Value | None.                                                        |        |                                                    |

|                       |                                                                                                                                                        |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | CalculatorRead(<FileName>, <SoluName>, <FieldType>, <VarArray>)                                                                                        |
| <b>Python Example</b> | <pre>oModule.CalculatorRead(<br/>    "c:\example.reg",<br/>    "Setup1: LastAdaptive", "Fields",<br/>    ["Freq:=", "10GHz", "Phase:=", "0deg"])</pre> |

|                   |                                                                                                                                                                 |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | CalculatorRead <FileName>, <SoluName>, <FieldType>, <VarArray>                                                                                                  |
| <b>VB Example</b> | <pre>oModule.CalculatorRead _<br/>    "c:\example.reg", _<br/>    "Setup1: LastAdaptive", "Fields", _<br/>    Array("Freq:=", "10GHz", "Phase:=", "0deg")</pre> |

## CalculatorWrite

Writes contents of top register to file.

|                   |                                                               |        |                                                                                 |
|-------------------|---------------------------------------------------------------|--------|---------------------------------------------------------------------------------|
| <b>UI Access</b>  | Click <b>Write...</b> in the <b>Fields Calculator</b> window. |        |                                                                                 |
| <b>Parameters</b> | Name                                                          | Type   | Description                                                                     |
|                   | <OutputFilePath>                                              | String | Path to and including name of output register file.                             |
|                   | <SoluNameArray>                                               | Array  | Array of strings containing solution names.                                     |
|                   | <Expression>                                                  | Array  | Defines an expression to be solved in the calculator. It can be an algebraic or |

|                     |                               |       |                                                                                                                                                                                                                                   |
|---------------------|-------------------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                               |       | stacked(default) expression; the <code>Algebraic</code> option requires a beta feature; see <a href="#">Algebraic option</a> . Note that for legacy scripts, this parameter is optional because the expression may already exist. |
|                     | <code>&lt;VarArray&gt;</code> | Array | Array of variable names, value pairs.                                                                                                                                                                                             |
| <b>Return Value</b> | None                          |       |                                                                                                                                                                                                                                   |

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                     |  |  |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | <code>CalculatorWrite(&lt;OutputFilePath&gt;, &lt;SoluNameArray&gt;, &lt;Expression&gt;, &lt;VarArray&gt;)</code>                                                                                                                                                                                                                                                                                                                   |  |  |
| <b>Python Example</b> | <p>Legacy script with no expression:</p> <pre>oModule.CalculatorWrite("C:\test.reg",     ["Solution:=", "Setup1 : LastAdaptive"],     ["Freq:=", "1GHz", "Phase:=", "0deg"] )</pre> <p>Example with algebraic expression:</p> <pre>oModule.CalculatorWrite("C:\test.reg",     [         "NAME:Write",         [             "NAME:Setup",             "Solution:="         ],         , "setup1 : LastAdaptive"     ],     [ </pre> |  |  |

```
"NAME:Algebraic",
 "Name:=" , [""],
 "Algebraic:=" , "Mean(Volume(pri), Mag_H)"
]
],
[
 "Freq:=" , "60Hz",
 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
])
```

Example with stack expression:

```
oModule.CalculatorWrite("C:\test.reg",
[
 "NAME:Write",
 [
 "NAME:Setup",
 "Solution:=" , "setup1 : LastAdaptive"
],
],
```



```
[
 "NAME:Expression",
 "Name:=" , [""],
 "Expression:=" , ["Mean(Volume(sec_1_1), Mag_H)"],
 "NameOfExpression:=" , ["Mag_H"],
 "EnterVolume:=" , ["sec_1_1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
],
[
 "NAME:Context",
 "Freq:=" , "60Hz",
 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
]
], "Fields")
```

**VB Syn-  
tax****CalculatorWrite <OutputFilePath>, <SoluNameArray>, <Expression>, <VarArray>**

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB<br/>Example</b> | <p>Legacy script with no expression:</p> <pre>oModule.CalculatorWrite "C:\test.reg", _<br/>    Array("Solution:=", "Setup1 : LastAdaptive"),_<br/>    Array("Freq:=", "1GHz", "Phase:=", "0deg")</pre> <p>Example with stack expression:</p> <pre>oModule.CalculatorWrite "C:\test.reg",_<br/>    Array("Solution:=", "setup1 : LastAdaptive")_<br/>    Array("Expression:=", Array("Mean(Volume(sec_1_1), Mag_H)"), "NameOfExpression:="_<br/>    Array("Mag_H"), "EnterVolume:=", Array("sec_1_1"), "Operation:=", Array<br/>    ("VolumeValue"),_<br/>    "Operation:=", Array("Mean"))_<br/>    Array(<br/>        "Freq:=", "60Hz",<br/>        "Ihv:=", "(833.3*1.414*232) A",<br/>        "Ilv:=", "-Ihv/1404",<br/>        "Phase:=", "0deg",<br/>        "height:=", "1mm"<br/>    ), "Fields")</pre> |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## ChangeGeomSettings

Changes the line discretization setting.

|                     |                                                                              |         |                            |
|---------------------|------------------------------------------------------------------------------|---------|----------------------------|
| <b>UI Access</b>    | In the <b>Fields Calculator</b> dialog box, click on <b>Geom Settings...</b> |         |                            |
| <b>Parameters</b>   | Name                                                                         | Type    | Description                |
|                     | <LineDiscr>                                                                  | Integer | Line discretization value. |
| <b>Return Value</b> | None.                                                                        |         |                            |

|                       |                                 |  |  |
|-----------------------|---------------------------------|--|--|
| <b>Python Syntax</b>  | ChangeGeomSettings(<LineDiscr>) |  |  |
| <b>Python Example</b> | oModule.ChangeGeomSettings(500) |  |  |

|                   |                                |  |  |
|-------------------|--------------------------------|--|--|
| <b>VB Syntax</b>  | ChangeGeomSettings <LineDiscr> |  |  |
| <b>VB Example</b> | oModule.ChangeGeomSettings 500 |  |  |

## ClcEval

Evaluates the expression at the top of the stack using the provided solution name and variable values.

|                   |                                                           |        |                                                                                                                                                                                                                                                                                                                                             |
|-------------------|-----------------------------------------------------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Click <b>Eval</b> in the <b>Fields Calculator</b> dialog. |        |                                                                                                                                                                                                                                                                                                                                             |
| <b>Parameters</b> | Name                                                      | Type   | Description                                                                                                                                                                                                                                                                                                                                 |
|                   | <SoluName>                                                | String | Name of specified solution.                                                                                                                                                                                                                                                                                                                 |
|                   | <VarArray>                                                | Array  | Array of variable name, value pairs.                                                                                                                                                                                                                                                                                                        |
|                   | <Expression>                                              | Array  | Defines an expression to be solved in the calculator. It can be an algebraic or stacked (default) expression; the <code>Algebraic</code> option requires a beta feature; see <a href="#">Algebraic option</a> . Note that for legacy scripts, this parameter is optional because the expression may have been created with another command. |
|                   | <FieldType>                                               | String | Optional. Type of specified field.                                                                                                                                                                                                                                                                                                          |

|              |      |
|--------------|------|
| Return Value | None |
|--------------|------|

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ClcEval(<SoluName>, <Expression>, <VarArray>, [Optional <FieldType>])                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| Python Example | <p>Legacy script with no expression:</p> <pre>oModule.ClcEval("Setup1: LastAdaptive",<br/>    ["Freq:=", "10GHz",<br/>    "Phase:=", "0deg"])</pre> <p>Example with algebraic expression:</p> <pre>oModule.ClcEval("setup1 : LastAdaptive",<br/>    [<br/>        "NAME:Evaluate",<br/>        [<br/>            "NAME:Algebraic",<br/>            "Name:=" , [""] ,<br/>            "Algebraic:=" , "Mean( Volume(sec_1_1), Mag_H )" ,<br/>        ],<br/>        [<br/>            "NAME:Context",<br/>            "Freq:=" , "60Hz",</pre> |

```

 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
]
], "Fields")

```

Example with stack expression:

```

oModule.ClcEval("setup1 : LastAdaptive",
[
 "NAME:Evaluate",
 [
 "NAME:Expression",
 "Name:=" , [""],
 "Expression:=" , ["Mean(Volume(sec_1_1), Mag_H)"],
 "NameOfExpression:=" , ["Mag_H"],
 "EnterVolume:=" , ["sec_1_1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
],
 [
 "NAME:Context",

```

|  |                                                                                                                                                                          |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Freq:=" , "60Hz",<br/>"Ihv:=" , "(833.3*1.414*232) A",<br/>"Ilv:=" , "-Ihv/1404",<br/>"Phase:=" , "0deg",<br/>"height:=" , "1mm"<br/><br/>]<br/>], "Fields")</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syn-<br>tax | ClcEval <SoluName>, <Expression>, <VarArray>, [Optional <FieldType>]                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB<br>Example  | <p>Legacy script with no expression:</p> <pre>oModule.ClcEval "Setup1: LastAdaptive", _<br/>    Array ("Freq:=", "10GHz", _<br/>        "Phase:=", "0deg")</pre> <p>Example with stack expression:</p> <pre>oModule.ClcEval "Setup1 : LastAdaptive",_<br/>    Array("Expression:=", Array("Mean(Volume(sec_1_1), Mag_H)"), "NameOfExpression:="_<br/>    Array("Mag_H"), "EnterVolume:=", Array("sec_1_1"), "Operation:=", Array<br/>    ("VolumeValue"),_<br/>    "Operation:=", Array("Mean"))_<br/><br/>    Array(</pre> |

|  |                                                                                                                                                                                          |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "Freq:="          , "60Hz", "Ihv:="           , " (833.3*1.414*232) A", "Ilv:="           , "-Ihv/1404", "Phase:="         , "0deg", "height:="        , "1mm" ), "Fields") </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## ClcMaterial

Performs a material operation on the top stack element.

|                     |                                                              |        |                                                          |
|---------------------|--------------------------------------------------------------|--------|----------------------------------------------------------|
| <b>UI Access</b>    | Click <b>Matl...</b> in the <b>Fields Calculator</b> dialog. |        |                                                          |
| <b>Parameters</b>   | Name                                                         | Type   | Description                                              |
|                     | <MatString>                                                  | String | The material property to apply.                          |
|                     | <OpString>                                                   | String | Name of operation. Possible values are "mult", or "div". |
| <b>Return Value</b> | None.                                                        |        |                                                          |

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| <b>Python Syntax</b>  | ClcMaterial(<MatString>, <OpString>)            |
| <b>Python Example</b> | oModule.ClcMaterial("Permeability (mu)" "mult") |

|                   |                                                |
|-------------------|------------------------------------------------|
| <b>VB Syntax</b>  | ClcMaterial <MatString>, <OpString>            |
| <b>VB Example</b> | oModule.ClcMaterial "Permeability (mu)" "mult" |

## ClcMaterialValue

Shows the value of the material property without performing any operation.

|                     |                                                             |        |                                      |
|---------------------|-------------------------------------------------------------|--------|--------------------------------------|
| <b>UI Access</b>    | Select <b>None</b> in the <b>Material Operation</b> dialog. |        |                                      |
| <b>Parameters</b>   | Name                                                        | Type   | Description                          |
|                     | <MaterialName>                                              | String | Name of specified material property. |
| <b>Return Value</b> | None.                                                       |        |                                      |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | ClcMaterialValue(<MaterialName>)         |
| <b>Python Example</b> | oModule.ClcMaterialValue("Mass Density") |

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>VB Syntax</b>  | ClcMaterialValue <MaterialName>         |
| <b>VB Example</b> | oModule.ClcMaterialValue "Mass Density" |

## ClearAllMarkers[Fields Reporter]

Clears all markers in the current fields overlay plot.

|                     |                                                                                 |
|---------------------|---------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right-click <b>Field Overlays &gt; Plot Fields &gt; Marker &gt; Clear All</b> . |
| <b>Parameters</b>   | None.                                                                           |
| <b>Return Value</b> | None.                                                                           |



|                       |                                        |
|-----------------------|----------------------------------------|
| <b>Python Syntax</b>  | <code>ClearAllMarkers()</code>         |
| <b>Python Example</b> | <code>oModule.ClearAllMarkers()</code> |

|                   |                                      |
|-------------------|--------------------------------------|
| <b>VB Syntax</b>  | <code>ClearAllMarkers</code>         |
| <b>VB Example</b> | <code>oModule.ClearAllMarkers</code> |

## ClearAllNamedExpr

Clears all user-defined named expressions from the list.

|                     |                                                               |
|---------------------|---------------------------------------------------------------|
| <b>UI Access</b>    | Click <b>ClearAll</b> in the <b>Fields Calculator</b> dialog. |
| <b>Parameters</b>   | None.                                                         |
| <b>Return Value</b> | None.                                                         |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | <code>ClearAllNamedExpr()</code>         |
| <b>Python Example</b> | <code>oModule.ClearAllNamedExpr()</code> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | <code>ClearAllNamedExpr</code>         |
| <b>VB Example</b> | <code>oModule.ClearAllNamedExpr</code> |

## CopyNamedExprToStack

Copies the named expression selected to the calculator stack.

|                     |                                                                 |        |                                                              |
|---------------------|-----------------------------------------------------------------|--------|--------------------------------------------------------------|
| <b>UI Access</b>    | Select a named expression and then click <b>Copy to stack</b> . |        |                                                              |
| <b>Parameters</b>   | Name                                                            | Type   | Description                                                  |
|                     | <ExprName>                                                      | String | Name of the expression to be copied to the top of the stack. |
| <b>Return Value</b> | None.                                                           |        |                                                              |

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | CopyNamedExprToStack(<ExprName>)                     |
| <b>Python Example</b> | <code>oModule.CopyNamedExprToStack("Mag_JxE")</code> |

|                   |                                                     |
|-------------------|-----------------------------------------------------|
| <b>VB Syntax</b>  | CopyNamedExprToStack <ExprName>                     |
| <b>VB Example</b> | <code>oModule.CopyNamedExprToStack "Mag_JxE"</code> |

## CreateFieldPlot

### Note:

Use in conjunction with [GetGeometryIdsForNetLayerCombinations](#) and `GetGeometryIdsForAllNetLayerCombinations`.

- `GetGeometryIdsForNetLayerCombinations` returns ID numbers of all faces and edges of the active design related to the current target combination of nets and layers.
- `GetGeometryIdsForAllNetLayerCombinations` returns ID numbers for all possible combinations of nets and layers it is possible to choose.

*Use:* Creates a field/mesh plot "Field" or a visual ray trace ("VRT") plot.

*Command:* **HFSS > Fields > Plot Fields > <field\_quantity>**

*Command:* **Icepak > Fields > Plot Fields > <field\_quantity>**

*Command:* **Maxwell3D or Maxwell2D > Fields > Fields > <field\_quantity>**

*Command:* **Maxwell3D > Fields > Fields > Field Line Trace** (for 3D Electrostatic designs - see [Examples](#) below)

*Command:* **Mechanical > Fields > Plot Fields > <field\_quantity>**

*Syntax:* `CreateFieldPlot <PlotParameterArray> ["Field" | "VRT"]`

*Return Value:* None

*Parameters:* `<PlotParameterArray> "Field"`

```
Array("NAME:<PlotName>",
 "SolutionName:=", <string>,
 "QuantityName:=", <string>,
 "PlotFolder:=", <string>,
 "UserSpecifyName:=", <int>,
 "UserSpecifyFolder:=", <int>,
 "IntrinsicVar:=", <string>,
```

```
"PlotGeomInfo:=", <PlotGeomArray>,
"FilterBoxes:=", <FilterBoxArray>,
<PlotOnPointsSettings>,
<PlotOnLineSettings>,
<PlotOnSurfaceSettings>,
<PlotOnVolumeSettings>)
```

*SolutionName:*

Name of the solution setup and solution formatted as:

```
"<SolveSetupName>: <WhichSolution>",
```

where <WhichSolution> can be "Adaptive\_<n>", "LastAdaptive", or "PortOnly".

For example: "Setup1 : Adaptive\_2"

A space is required on either side of the colon ( : ) character. If omitted, the plot will not be created.

*QuantityName:*

Type of plot to create. Possible values include:

**Mesh plots:** "Mesh"

**HFSS Field plots include:**

```
"Mag_E", "Mag_H", "Mag_Jvol", "Mag_Jsurf", "ComplexMag_E",
"ComplexMag_H", "ComplexMag_Jvol", "ComplexMag_Jsurf",
"Vector_E", "Vector_H", "Vector_Jvol", "Vector_Jsurf",
"Vector_RealPoynting", "Local_SAR", "Average_SAR"
```

**Maxwell Field plots include:**

```
"Mag_E", "Mag_H", "E_Vector", "H_Vector", "Mag_J", "J_Vector",
"ComplexMag_J", "Mag_Jc", "Jc_Vector", "ComplexMag_Jc",
"Mag_B_DC", "Vector_B_DC", "Mag_B_AC_DC", "Vector_B_AC_DC",
"Mag_B_AC_DC_Peak", "Vector_B_AC_DC_Peak", "Energy",
```

```
"coEnergy", "appEnergy", "Ohmic_Loss", "Total_Loss",
"Hysteresis_Loss", "Dielectric_Loss", "Temperature",
"Volume_Force_Density", "Average_Surface_Loss_Density",
"Surface_Force_Density", "Demag_Coef",
"QuantityName_FieldLineTrace", "Surface_Loss_Density",
"QSurf"
```

***Mechanical* Field plots include:**

(dependent on the solution type and whether there is trace mapping):

```
"Mag_Displacement", "Displacement_Vector", "Temperature",
"HeatFlux", "Mag_HeatFlux", "Surface Loss Density",
"Volume Loss Density", "Linked Heat Transfer Coefficient",
"Equivalent Stress", "Equivalent Strain",
"Thermal Conductivity X", "Thermal Conductivity Y",
"Thermal Conductivity Z", "Youngs Modulus X",
"Youngs Modulus Y", "Youngs Modulus Z", "Poisson Ratio XY",
"Poisson Ratio YZ", "Poisson Ratio XZ", "Thermal Expansion X",
"Thermal Expansion Y", "Thermal Expansion Z",
"Shear Modulus X", "Shear Modulus Y", "Shear Modulus Z"
```

*PlotFolder:*

Name of the folder to which the plot should be added. (Values vary with the design type.) Some possible values include:

```
"E Field", "H Field", "Jvol", "Jsurf", "SAR Field",
"Jc", "Surface-Loss", "QSurf", "Temperature", "Energy",
"Average-Surface-Loss-Density", "Dielectric_Loss",
"MeshPlots", "Heat Flux", and "Displacement"
```

*UserSpecifyName*

0 if default name for plot is used, 1 otherwise.

Not needed. <PlotName> will be respected regardless of whether thisflag is set.

*UserSpecifyFolder*

0 if default folder for plot is used, 1 otherwise.

Not needed. The specified PlotFolder will be respected regardless of whether this flag is set.

*IntrinsicVar:*

Formatted string that specifies the frequency and phase at which to make the plot.

For example: "Freq='1GHz' Phase='30deg' "

*IntrinsicVar (Maxwell):*

Formatted string that specifies the skew slice number at which to make the plot.

For example: "IntrinsicVar:=", "Slice='\2  
\'Time='\0.00020000000000000001s\' "

*PlotGeomInfo:*

Creating field plot on selected layer-net pairs:

For example, Maxwell: "PlotGeomInfo:=", [1,"Volume","ObjList",  
2,"LC1\_1:Top","LC1\_1:Top#L1"]

HFSS example with "Plot on surface" option is not checked:

```
"PlotGeomInfo:=", [1,"Volume","LayerNets",2, "Top",2,
"net1","net2","Ground",1,"GND"]
```

The command creates field plot on 2 layer-nets combinations. It starts with type "Volume", and subtype "LayerNets", followed by the number of layer-nets combinations which is 2 in this example.

The two layer-nets combinations are (1) "Top" layer, 2 nets, net names are "net1" and "net2", i.e., [Top, net1] pair, and [Top, net2] pair. (2) "Ground" layer and 1 net, net name is "GND", i.e., [Ground, GND] pair.

HFSS example with "Plot on surface" option checked:

```
"PlotGeomInfo:=", [1,"Surface","LayerNetsExtFace",2,
"Top",2,"net1","net2",_"Ground",1,"GND"]
```

The command creates field plot on 2 layer-nets combinations. It starts with type "Surface", and subtype "LayerNetsExtFace", followed by the number of layer-nets combinations which is 2 in this example.

The two layer-nets combinations are (1) "Top" layer, 2 nets, net names are "net1" and "net2", i.e., [Top, net1] pair, and [Top, net2] pair. (2) "Ground" layer and 1 net, net name is "GND", i.e., [Ground, GND] pair.

Limitations:

- No Layer/Nets name is supported for field plot command recording should be able to add it based on this change.
- No Layer/Nets name support for pure Layout design field plot command.

=====

PlotGeomInfo (*Maxwell*)

**For example:** "PlotGeomInfo :=" , [1,"Volume","ObjList",2,  
"LC1\_1:Top","LC1\_1:Top#L1"]

PlotGeomInfo (*Maxwell*)

**For example:** "PlotGeomInfo :=" , [1,"Volume","ObjList",2,  
"LC1\_1:Top","LC1\_1:Top#L1"]

<PlotGeomArray>

```
Array(<NumGeomTypes>, <GeomTypeData>, <GeomTypeData>, ...)
```

**For example:** Array(4, "Volume", "ObjList", 1, "Box1",  
"Surface", "FacesList", 1, "12", "Line", 1,  
"Polyline1", "Point", 2, "Point1", "Point2")

<NumGeomTypes>

Type: <int>

Number of different geometry types (volume, surface, line, point) plotted on at the same time.

<GeomTypeData>

<GeomType>, <ListType>, <NumIDs>, <ID>, <ID>, ...)

<GeomType>

Type: <string>

Possible values are "Volume", "Surface", "Line", "Point".

<ListType>

Type: <string>

Possible values are "ObjList", or "FacesList".

These are used for the GeomType of "Line" or "Point".

<NumIDs>

Type: <int>

Number of IDs or object names that will follow.

<ID>

Type: <int> or <string>

ID of a face or name of an object, line, or point on which to plot.

<FilterBoxArray>

Array of names of objects to use to restrict the plot range.

Array(<NumFilters>, <ObjName>, <ObjName>, ...)

Example: Array(1, "Box1")

Example: Array(0) *no filtering*

<PlotOnPointSettings>



```
Array("NAME:PlotOnPointSettings",
 "PlotMarker:=", <bool>,
 "PlotArrow:=", <bool>)
```

<PlotOnLineSettings>

```
Array("NAME:PlotOnLineSettings",
 Array("NAME:LineSettingsID",
 "Width:=", <int>,
 "Style:=", <string>),
 "IsoValType:=", <string>,
 "ArrowUniform:=", <bool>,
 "NumofArrow:=", <int>)
```

Style

**Possible values are:** "Cylinder", "Solid", "Dashdash", "Dotdot", "Dotdash"

IsoValType

**Possible values are:** "Tone", "Fringe", "Gourard"

<PlotOnSurfaceSettings>

```
Array("NAME:PlotOnSurfaceSettings",
 "Filled:=", <bool>,
 "IsoValType:=", <string>,
 "SmoothShade:=", <bool>,
 "AddGrid:=", <bool>,
 "MapTransparency:=", <bool>,
 "Transparency:=", <double>,
 "ArrowUniform:=", <bool>,
 "ArrowSpacing:=", <double>,
 "GridColor:=", Array(<int>, <int>, <int>))
```

IsoValType

**Possible values are:** "Tone", "Line", "Fringe", "Gourard"

### GridColor

Array containing the R, G, B components of the color. Components should be in the range 0 to 255.

<PlotOnVolumeSettings>

```
Array("NAME:PlotOnVolumeSettings",
 "PlotIsoSurface:=", <bool>,
 "CloudDensity:=", <double>,
 "PointSize:=", <int>,
 "ArrowUniform:=", <bool>,
 "ArrowSpacing:=", <double>)
```

### Python Example – E Magnitude Field Plot:

```
oModule.CreateFieldPlot(
[
 "NAME:Mag_E1",
 "SolutionName:=", "Setup1 : LastAdaptive",
 "QuantityName:=", "Mag_E",
 "PlotFolder:=", "E Field1",
 "UserSpecifyName:=", 0,
 "UserSpecifyFolder:=", 0,
 "IntrinsicVar:=", "Freq='1GHz' Phase='0deg'",
 "PlotGeomInfo:=", [1, "Surface", "FacesList", 1, "7"],
 "FilterBoxes:=", [0],
 [
 "NAME:PlotOnSurfaceSettings",
 "Filled:=", False,
 "IsoValType:=", "Fringe",
 "SmoothShade:=", True,
 "AddGrid:=", False,
```

```

 "MapTransparency:=", True,
 "Transparency:=" , 0,
 "ArrowUniform:=" , True,
 "ArrowSpacing:=" , 0.1000000001490116,
 "GridColor:=" , [255, 255, 255]
]
], "Field")

```

### VB Example – Demag\_Coef Field Plot:

```

oModule.CreateFieldPlot Array("NAME:Demag_Coef2", _
 "SolutionName:=", "Setup1 : Transient", _
 "UserSpecifyName:=", 0, "UserSpecifyFolder:=", 0, _
 "QuantityName:=", "Demag_Coef", "PlotFolder:=", "Demag-Coef", _
 "StreamlinePlot:=", false, "AdjacentSidePlot:=", false, _
 "FullModelPlot:=", false, "IntrinsicVar:=", _
 "Time=" & Chr(39) & "0s" & Chr(39) & "", "PlotGeomInfo:=", _
 Array(1, "Volume", "ObjList", 1, "Mag2_0"), "FilterBoxes:=", _
 Array(1, "Mag1_0"), _
 Array("NAME:PlotOnVolumeSettings", "PlotIsoSurface:=", _
 true, "PointSize:=", 1, "Refinement:=", 0, _
 "CloudSpacing:=", 0.5, "CloudMinSpacing:=", -1, _
 "CloudMaxSpacing:=", -1, _
 Array("NAME:Arrow3DSpacingSettings", _
 "ArrowUniform:=", true, "ArrowSpacing:=", 0, _
 "MinArrowSpacing:=", 0, "MaxArrowSpacing:=", 0)), _
 "EnableGaussianSmoothing:=", false), "Field"

```

*Parameters:* <PlotParameterArray> "VRT" (VB)

```

Array("NAME:<PlotName>", _
 "SolutionName:=", <string>, _

```

```
"QuantityName:=", <string>, _
"PlotFolder:=", <string>, _
"UserSpecifyName:=", <int>, _
"UserSpecifyFolder:=", <int>, _
"IntrinsicVar:=", <string>, _
"MaxFrequency:=", "<Number><FreqUnits>", _
"LaunchFrom:=", ["Launch from custom", | _
"LaunchFromPointID:=",] 18007, "RayDensity:=", <int>, _
"NumberBounces:=", <int>, _
"Multi-Bounce Ray Density Control:=", <Boolean>, _
"MBRD Max sub divisions:=", <int>, _
"ShootFilterType:=", ["All Rays" | "Rays by index", _
"start index:=", <int>, "stop index:=", <int> | _
"index step:=", <int> "Rays in box", "FilterBoxID:=", _
<objID> | "Single ray", "Ray elevation:=", _
"<real> deg", "Ray azimuth:=", "<real>deg"])), "VRT"
```

### VB Example – VRT Plot:

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("abrams_1p5")
Set oDesign = oProject.SetActiveDesign("UseSbr")
```

```

Set oModule = oDesign.GetModule("FieldsReporter")
oModule.CreateFieldPlot _
 Array("NAME:VRT_Plot1", "SolutionName:=", _
 "Setup1 : LastAdaptive", "QuantityName:=", _
 "QuantityName_SBR", "PlotFolder:=", _
 "Visual Ray Trace SBR", "UserSpecifyName:=", 0, _
 "UserSpecifyFolder:=", 0, "IntrinsicVar:=", "", _
 "MaxFrequency:=", "1GHz", "LaunchFrom:=", _
 "Launch from custom", "LaunchFromPointID:=", _
 18007, "RayDensity:=", 2, "NumberBounces:=", _
 5, "Multi-Bounce Ray Density Control:=", _
 false, "MBRD Max sub divisions:=", 2, _
 "ShootFilterType:=", "All Rays"), "VRT"

```

### VB Example – E Magnitude Field Plot:

```

oModule.CreateFieldPlot Array("NAME:Mag_E1", _
 "SolutionName:=", "Setup1 : LastAdaptive", _
 "QuantityName:=", "Mag_E", _
 "PlotFolder:=", "E Field1", _
 "UserSpecifyName:=", 0, _
 "UserSpecifyFolder:=", 0, _
 "IntrinsicVar:=", "Freq='1GHz' Phase='0deg'", _
 "PlotGeomInfo:=", Array(1, "Surface", "FacesList", 1, "7"), _
 "FilterBoxes:=", Array(0), _
 Array("NAME:PlotOnSurfaceSettings", _
 "Filled:=", false, "IsoValType:=", "Fringe", _
 "SmoothShade:=", true, _
 "AddGrid:=", false, _
 "MapTransparency:=", true, _
 "Transparency:=", 0, _
 "ArrowUniform:=", true, _

```

```
"ArrowSpacing:=", 0.100000001490116, _
"GridColor:=", Array(255, 255, 255))), "Field"
```

### Python Example – Electron Density Field Plot:

```
oModule = oDesign.GetModule("FieldsReporter")
oModule.CreateFieldPlot(
[
 "NAME:Electron_Density3",
 "SolutionName:=", "AIR : RFDIScharge",
 "UserSpecifyName:=", 0,
 "UserSpecifyFolder:=", 0,
 "QuantityName:=", "Electron_Density",
 "PlotFolder:=", "RF Discharge Fields",
 "StreamlinePlot:=", False,
 "AdjacentSidePlot:=", False,
 "FullModelPlot:=", False,
 "IntrinsicVar:=", "Freq='\0.200000000000000001GHz\
 GasPressure='\0.02kPascal\
 "PlotGeomInfo:=", [1,"Surface","FacesList",1,"48"],
 "FilterBoxes:=", [0],
 [
 "NAME:PlotOnSurfaceSettings",
 "Filled:=", False,
 "IsoValType:=", "Tone",
 "AddGrid:=", False,
 "MapTransparency:=", True,
 "Refinement:=", 0,
 "Transparency:=", 0,
 "SmoothingLevel:=", 0,
]
]
```

```

 "ShadingType:=" , 0,
 [
 "NAME:Arrow3DSpacingSettings",
 "ArrowUniform:=" , True,
 "ArrowSpacing:=" , 0,
 "MinArrowSpacing:=" , 0,
 "MaxArrowSpacing:=" , 0
 "GridColor:=" , [255,255,255]
],
],
"EnableGaussianSmoothing:=" , False,
"SurfaceOnly:=" , False
], "Field")

```

### VB Example – Mesh Plot:

```

Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("abrams_1p5")
Set oDesign = oProject.SetActiveDesign("UseSbr")
Set oModule = oDesign.GetModule("FieldsReporter")
oModule.CreateFieldPlot Array("NAME:Mesh1", "SolutionName:=", _
 "Setup1 : LastAdaptive", "QuantityName:=", "Mesh", _
 "PlotFolder:=", "MeshPlots", "FieldType:=", "Fields", _
 "UserSpecifyName:=", 0, "UserSpecifyFolder:=", 0, _

```

```
"StreamlinePlot:=", false, "IntrinsicVar:=", "Freq=" _
& Chr(39) & "2GHz" & Chr(39) & " Phase=" _
& Chr(39) & "0deg" & Chr(39) & "" & "", _
"PlotGeomInfo:=", Array(1, "Surface", "FacesList", 1, _
"10568"), "FilterBoxes:=", Array(0), "Real time mode:=", _
true, Array("NAME:MeshSettings", "Scale factor:=", 100, _
 "Transparency:=", 0, "Mesh type:=", "Shaded", _
 "Surface only:=", true, "Add grid:=", true, _
 "Refinement:=", 0, "Use geometry color:=", true, _
 "Mesh line color:=", Array(0, 0, 255), "Filled color:=", _
 Array(255, 255, 255))), _
"Field"
```

### VB Example – Maxwell Plot with Skew Slice:

```
oModule.CreateFieldPlot
 Array("NAME:energy2", "SolutionName:=", "Setup1 : Transient", _
 "UserSpecifyName:=", 0, "UserSpecifyFolder:=", 0, _
 "QuantityName:=", "energy", "PlotFolder:=", "Energy", _
 "StreamlinePlot:=", false, "AdjacentSidePlot:=", false, _
 "FullModelPlot:=", false, "IntrinsicVar:=", "Slice=" & _
 Chr(39) & "2" & Chr(39) & " Time=" & Chr(39) _
 & "0.0002000000000000000001s" & Chr(39) & "",...)
```

### Python Example: Maxwell Plot with Skew Slice

```
oModule.CreateFieldPlot(
[
 "NAME:Mesh1",
 "SolutionName:=" , "Setup1 : Transient",
```



```

 "UserSpecifyName:=" , 0,
 "UserSpecifyFolder:=" , 0,
 "QuantityName:=" , "Mesh",
 "PlotFolder:=" , "MeshPlots",
 "FieldType:=" , "Fields",
 "StreamlinePlot:=" , False,
 "AdjacentSidePlot:=" , False,
 "FullModelPlot:=" , False,
 "IntrinsicVar:=" , "Slice=\2\'
 Time=\0.000200000000000000001s\'",
 "PlotGeomInfo:=" , [1,"Surface","FacesList",1,"2955"],
 "FilterBoxes:=" , [0],
 ...
])

```

### Python Example – Maxwell Average\_Surface\_Loss\_Density Plot:

```

oModule.CreateFieldPlot(
[
 "NAME:Average_Surface_Loss_Density1",
 "SolutionName:=" , "Setup1 : Transient",
 "UserSpecifyName:=" , 0,
 "UserSpecifyFolder:=" , 0,
 "QuantityName:=" , "Average_Surface_Loss_Density",
 "PlotFolder:=" , "Average-Surface-Loss-Density",
 "StreamlinePlot:=" , False,
 "AdjacentSidePlot:=" , False,
 "FullModelPlot:=" , False,
 "IntrinsicVar:=" , "Time=\4us\' Time0=\0s\'",
 "PlotGeomInfo:=" , [1,"Surface","FacesList",1,"6"],
 ...
])

```

## Python Examples – Maxwell ACConduction Design Plot:

```
oModule.CreateFieldPlot(
[
 "NAME:Mag_Jc1",
 "SolutionName:=" , "Setup1 : LastAdaptive",
 "UserSpecifyName:=" , 0,
 "UserSpecifyFolder:=" , 0,
 "QuantityName:=" , "Mag_Jc",
 "PlotFolder:=" , "Jc",
 "StreamlinePlot:=" , False,
 "AdjacentSidePlot:=" , False,
 "FullModelPlot:=" , False,
 "IntrinsicVar:=" , "Freq='\1000000Hz\' Phase='\0deg\'",
 "PlotGeomInfo:=" , [1,"Volume","ObjList",1,"Box1"],
 "FilterBoxes:=" , [1,""],
 [
 "NAME:PlotOnVolumeSettings",
 "PlotIsoSurface:=" , True,
 "PointSize:=" , 1,
 "Refinement:=" , 0,
 "CloudSpacing:=" , 0.5,
 "CloudMinSpacing:=" , -1,
 "CloudMaxSpacing:=" , -1,
 "ShadingType:=" , 0,
 [
 "NAME:Arrow3DSpacingSettings",
 "ArrowUniform:=" , True,
 "ArrowSpacing:=" , 0,
 "MinArrowSpacing:=" , 0,
 "MaxArrowSpacing:=" , 0
```

```
],
],
 "EnableGaussianSmoothing:=", False
], "Field")

oModule.CreateFieldPlot(
[
 "NAME:ComplexMag_Jc1",
 "SolutionName:=", "Setup1 : LastAdaptive",
 "UserSpecifyName:=", 0,
 "UserSpecifyFolder:=", 0,
 "QuantityName:=", "ComplexMag_Jc",
 "PlotFolder:=", "Jc",
 ...
], "Field")

oModule.CreateFieldPlot(
[
 "NAME:Jc_Vector2",
 "SolutionName:=", "Setup1 : LastAdaptive",
 "UserSpecifyName:=", 0,
 "UserSpecifyFolder:=", 0,
 "QuantityName:=", "Jc_Vector",
 "PlotFolder:=", "Jc",
 ...
], "Field")

oModule.CreateFieldPlot(
[
 "NAME:Dielectric_Loss1",
 "SolutionName:=", "Setup1 : LastAdaptive",
 "UserSpecifyName:=", 0,
 "UserSpecifyFolder:=", 0,
```

```
"QuantityName:=" , "Dielectric_Loss",
"PlotFolder:=" , "Dielectric-Loss",
...
], "Field")
```

## Maxwell Field Line Trace Plot Examples

### VB Example – Maxwell Field Line Trace Plot:

```
oModule.CreateFieldPlot
Array("NAME:FieldLineTrace_Plot3", _
 "SolutionName:=" , "Setup1 : LastAdaptive", _
 "UserSpecifyName:=" , 0, _
 "UserSpecifyFolder:=" , 0, _
 "QuantityName:=" , "QuantityName_FieldLineTrace", _
 "PlotFolder:=" , "Field line trace plot", _
 "IntrinsicVar:=" , "", _
 "Trace Step Length:=" , "0.001mm", _
 "Use Adaptive Step:=" , true, _
 "Seeding Faces:=" , Array(1, 15091), _
 "Seeding Markers:=" , Array(0), _
 "Surface Tracing Objects:=", Array(1, 15), _
 "Volume Tracing Objects:=", Array(1, 36), _
 "Seeding Sampling Option:=", true, _
 "Seeding Points Number:=" , 15, _
 "Fractional of Maximal:=" , 0.8, _
 "Discrete Seeds Option:=" , "Marker Point", _
 Array("NAME:InceptionEvaluationSettings", _
 "Gas Type:=" , 0, _
 "Gas Pressure:=" , 1), _
```

```

 Array("NAME:FieldLineTracePlotSettings", _
 Array("NAME:LineSettingsID", _
 "Width:=" , 1, _
 "Style:=" , "Solid"), _
 "IsoValType:=" , "Tone")) _
 "FieldLineTrace")

oModule.ModifyInceptionParameters "FieldLineTrace_Plot3", _
Array("NAME:InceptionEvaluationSettings", _
 "Gas Type:=" , 2, _
 "Gas Pressure:=" , 1, _
 "Use Inception:=" , true, _
 "Potential U0:=" , 0, _
 "Potential K:=" , 1, _
 "Potential A:=" , 1, _
 "Critical Value:=" , 5.9426, _
 "Streamer Constant:=" , 6.5, _
 "Ionization Coefficient Dataset:=", Array(0))

```

### Python Example – Maxwell Field Line Trace Plot:

```

oModule.CreateFieldPlot(
[
 "NAME:FieldLineTrace_Plot3",
 "SolutionName:=" , "Setup1 : LastAdaptive",
 "UserSpecifyName:=" , 0,
 "UserSpecifyFolder:=" , 0,
 "QuantityName:=" , "QuantityName_FieldLineTrace",
 "PlotFolder:=" , "Field line trace plot",
 "IntrinsicVar:=" , "",
 "Trace Step Length:=" , "0.001mm",
 "Use Adaptive Step:=" , True,

```

```
"Seeding Faces:=" , [1,15091],
"Seeding Markers:=", [0],
"Surface Tracing Objects:=", [1,15],
"Volume Tracing Objects:=" , [1,36],
"Seeding Sampling Option:=", True,
"Seeding Points Number:=" , 15,
"Fractional of Maximal:=" , 0.8,
"Discrete Seeds Option:=" , "Marker Point",
[
 "NAME:InceptionEvaluationSettings",
 "Gas Type:=" , 0,
 "Gas Pressure:=" , 1,
 "Use Inception:=" , True,
 "Potential U0:=" , 0,
 "Potential K:=" , 1,
 "Potential A:=" , 1
],
[
 "NAME:FieldLineTracePlotSettings",
 [
 "NAME:LineSettingsID",
 "Width:=" , 1,
 "Style:=" , "Solid"
],
 "IsoValType:=" , "Tone"
]
], "FieldLineTrace")
```

**For Q3D Extractor and 2D Extractor, the command details are as follows:**

*Use:* Creates a field/mesh plot.

*Command:* **Q3D Extractor** or **2D Extractor** > **Fields**

**Syntax:** CreateFieldPlot <PlotParameterArray>

**Return Value:** None

**Parameters:** <PlotParameterArray>

```
Array("NAME:<PlotName>",
 "SolutionName:=" , <string>, _
 "QuantityName:=" , <string>, _
 "PlotFolder:=" , <string>, _
 "UserSpecifyName:=" , <int>, _
 "UserSpecifyFolder:=" , <int>, _
 "IntrinsicVar:=" , <string>, _
 "PlotGeomInfo:=" , <PlotGeomArray>, _
 "FilterBoxes:=" , <FilterBoxArray>, _
 <PlotOnPointSettings> , <PlotOnLineSettings>, _
 <PlotOnSurfaceSettings>, <PlotOnVolumeSettings>)
```

**SolutionName**

Name of the solution setup and solution formatted as

```
"<SolveSetupName> : <WhichSolution>",
```

where <WhichSolution> can be "Adaptive\_<n>", "LastAdaptive", or "PortOnly".

For example: "Setup1 : Adaptive\_2"

HFSS requires a space on both sides of the ':' character. Otherwise, the plot is not be created.

**QuantityName**

Type of plot to create. Possible values are:

Mesh plots: "Mesh"

**Q3D Field Plots:**

| Field type        | Plot quantity names                   |
|-------------------|---------------------------------------|
| AC R/L Fields     | "SurfaceJac", "Mag_SurfaceJac"        |
| DC R/L PEC Fields | "SurfaceJdc", "Mag_SurfaceJdc"        |
| DC R/L Fields     | "VolumeJdc", "Mag_VolumeJdc", "Phidc" |
| C Fields          | "SmoothQ", "ABS_Q"                    |

**2D Extractor Field Plots:**

| Field type | Plot quantity names                                                                                                                    |
|------------|----------------------------------------------------------------------------------------------------------------------------------------|
| CG Fields  | "Mag_Phi", "PhiAtPhase", "Mag_E", "VectorE", "Mag_Jcg", "VectorJcg", and "energyCG"                                                    |
| RL Fields  | "Flux Lines", "VectorA", "Mag_B", "VectorB", "Mag_H", "VectorH", "Jrl", "VectorJrl", "energyRL", "coenergy", "appenergy", and "emloss" |

PlotFolder

Name of the folder to which the plot should be added. Possible values are:

"Q", "ABS\_Q", "JDC Vol", "Phi", "JDC Surf", and "JAC".

UserSpecifyName

0 if default name for plot is used, 1 otherwise.

This parameter is not essential. <PlotName> is respected regardless of whether this flag is set.

UserSpecifyFolder

0 if the default folder for plot is used, 1 otherwise.

This parameter is not essential. The specified PlotFolder is respected regardless of whether this flag is set.

IntrinsicVar



Formatted string that specifies the frequency and phase at which to create the plot.

For example: "Freq='1GHz' Phase='30deg' "

<PlotGeomArray>

Array(<NumGeomTypes>, <GeomTypeData>, <GeomTypeData>, ...)

For example:

```
Array(4, "Volume", "ObjList", 1, "Box1",
 "Surface", "FacesList", 1, "12", "Line", 1, "
 "Polyline1", "Point", 2, "Point1", "Point2")
```

<NumGeomTypes>

Type: <int>

Number of different geometry types (volume, surface, line, point) plotted at the same time.

<GeomTypeData>

<GeomType>, <ListType>, <NumIDs>, <ID>, <ID>, ...)

<GeomType>

Type: <string>

Possible values are:

"Volume", "Surface", "Line", "Point"

<ListType>

Type: <string>

Possible values are:

"ObjList", "FacesList".

These are used for GeomType values "Line" or "Point".

<NumIDs>

Type: <int>

Number of IDs or object names that will follow.

<ID>

Type: <int> or <string>

ID of a face or name of an object, line, or point on which to plot.

<FilterBoxArray>

Array of object names used to restrict the plot range.

Array(<NumFilters>, <ObjName>, <ObjName>, ...)

Examples:

```
Array(1, "Box1")
```

```
Array(0) no filtering
```

<PlotOnPointSettings>

```
Array("NAME:PlotOnPointSettings", _
 "PlotMarker:=", <bool>, _
 "PlotArrow:=" , <bool>)
```

<PlotOnLineSettings>

```
Array("NAME:PlotOnLineSettings",
 Array("NAME:LineSettingsID", _
 "Width:=" , <int>, _
 "Style:=" , <string>), _
```

```

 "IsoValType:=" , <string>, _
 "ArrowUniform:=" , <bool>, _
 "NumofArrow:=" , <int>)

```

Style

Possible values are:

"Cylinder", "Solid", "Dashdash", "Dotdot", "Dotdash".

IsoValType

Possible values are:

"Tone", "Fringe", "Gourard"

<PlotOnSurfaceSettings>

```

Array("NAME:PlotOnSurfaceSettings", _
 "Filled:=" , <bool>, _
 "IsoValType:=" , <string>, _
 "SmoothShade:=" , <bool>, _
 "AddGrid:=" , <bool>, _
 "MapTransparency:=" , <bool>, _
 "Transparency:=" , <double>, _
 "ArrowUniform:=" , <bool>, _
 "ArrowSpacing:=" , <double>, _
 "GridColor:=" , Array(<int>, <int>, <int>))

```

IsoValType

Possible values are:

"Tone", "Line", "Fringe", "Gourard"

GridColor

Array containing the R, G, B components of the color. Components should be in the range 0 to 255.

```
<PlotOnVolumeSettings>
 Array("NAME:PlotOnVolumeSettings", _
 "PlotIsoSurface:=", <bool>, _
 "CloudDensity:=" , <double>, _
 "PointSize:=" , <int>, _
 "ArrowUniform:=" , <bool>, _
 "ArrowSpacing:=" , <double>)
```

DeleteFieldPlot

Deletes one or more field plots.

|              |                                                       |       |                                                               |
|--------------|-------------------------------------------------------|-------|---------------------------------------------------------------|
| UI Access    | Right-click on one filed plot, select <b>Delete</b> . |       |                                                               |
| Parameters   | Name                                                  | Type  | Description                                                   |
|              | <NameArray>                                           | Array | Array of strings containing the names of the plots to delete. |
| Return Value | None.                                                 |       |                                                               |

|                |                                                  |
|----------------|--------------------------------------------------|
| Python Syntax  | DeleteFieldPlot(<NameArray>)                     |
| Python Example | oModule.DeleteFieldPlot(["Mag_E1", "Vector_E1"]) |

|            |                                                      |
|------------|------------------------------------------------------|
| VB Syntax  | DeleteFieldPlot <NameArray>                          |
| VB Example | oModule.DeleteFieldPlot Array("Mag_E1", "Vector_E1") |

## DeleteMarker[Fields Reporter]

Deletes one or more marker in the current field plot.

|              |                                                                                     |       |                                           |
|--------------|-------------------------------------------------------------------------------------|-------|-------------------------------------------|
| UI Access    | Right-click <b>Field Overlays &gt; Plot Fields &gt; Marker &gt; Delete Marker</b> . |       |                                           |
| Parameters   | Name                                                                                | Type  | Description                               |
|              | <MarkerNames>                                                                       | Array | Array of strings containing marker names. |
| Return Value | None.                                                                               |       |                                           |

|                |                                                 |
|----------------|-------------------------------------------------|
| Python Syntax  | DeleteMarker(<MarkerNames>)                     |
| Python Example | <code>oModule.DeleteMarker(["m1", "m2"])</code> |

|            |                                                     |
|------------|-----------------------------------------------------|
| VB Syntax  | DeleteMarker <MarkerNames>                          |
| VB Example | <code>oModule.DeleteMarker Array("m1", "m2")</code> |

## DeleteNamedExpr

Deletes the selected named expression from the list.

|              |                                                          |        |                                     |
|--------------|----------------------------------------------------------|--------|-------------------------------------|
| UI Access    | Select a named expression and then click <b>Delete</b> . |        |                                     |
| Parameters   | Name                                                     | Type   | Description                         |
|              | <ExprName>                                               | String | Name of specified named expression. |
| Return Value | None.                                                    |        |                                     |

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| <b>Python Syntax</b>  | DeleteNamedExpr(<ExprName>)                     |
| <b>Python Example</b> | <code>oModule.DeleteNamedExpr("Mag_JxE")</code> |

|                   |                                                |
|-------------------|------------------------------------------------|
| <b>VB Syntax</b>  | DeleteNamedExpr <ExprName>                     |
| <b>VB Example</b> | <code>oModule.DeleteNamedExpr "Mag_JxE"</code> |

## DeleteUneditablePlot

Deletes one or more uneditable plots.

|                     |             |       |                                    |
|---------------------|-------------|-------|------------------------------------|
| <b>UI Access</b>    | N/A         |       |                                    |
| <b>Parameters</b>   | Name        | Type  | Description                        |
|                     | <NameArray> | Array | Array of the plot names to delete. |
| <b>Return Value</b> | None.       |       |                                    |

|                       |                                                                    |
|-----------------------|--------------------------------------------------------------------|
| <b>Python Syntax</b>  | DeleteUneditablePlot(<NameArray>)                                  |
| <b>Python Example</b> | <code>oModule.DeleteUneditablePlot(["Mag_E1", "Vector_E1"])</code> |

|                  |                                  |
|------------------|----------------------------------|
| <b>VB Syntax</b> | DeleteUneditablePlot <NameArray> |
|------------------|----------------------------------|

|                   |                                                                        |
|-------------------|------------------------------------------------------------------------|
| <b>VB Example</b> | <code>oModule.DeleteUneditablePlot Array("Mag_E1", "Vector_E1")</code> |
|-------------------|------------------------------------------------------------------------|

## DoesNamedExpressionExists

Determines whether specified named expression exists.

|                     |                                                                                                                                                         |        |                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|--------|-----------------------------------------|
| <b>UI Access</b>    | N/A                                                                                                                                                     |        |                                         |
| <b>Parameters</b>   | Name                                                                                                                                                    | Type   | Description                             |
|                     | <ExpName>                                                                                                                                               | String | Name of the specified named expression. |
| <b>Return Value</b> | Boolean: <ul style="list-style-type: none"> <li>• <b>1</b> - named expression exists.</li> <li>• <b>0</b> - named expression does not exist.</li> </ul> |        |                                         |

|                       |                                                           |
|-----------------------|-----------------------------------------------------------|
| <b>Python Syntax</b>  | <code>DoesNamedExpressionExists(&lt;ExpName&gt;)</code>   |
| <b>Python Example</b> | <code>oModule.DoesNamedExpressionExists("Mag_JxE")</code> |

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| <b>VB Syntax</b>  | <code>DoesNamedExpressionExists &lt;ExpName&gt;</code>   |
| <b>VB Example</b> | <code>oModule.DoesNamedExpressionExists "Mag_JxE"</code> |

## EditSurfaceMeshSummaryData

*Use:* Defines or Edits Surface Mesh Summary Data.

*Command:* Create Surface Mesh Summary, **Setup...**

*Syntax:* EditSurfaceMeshSummaryData Array(Array(<Parameters Array>))

*Return Value:* None

*Parameters:*

"NAME:SurfaceMeshSummary"

Type: <string>

"NAME:SurfaceMeshSummary"

"SolutionName:=" <string>

Type: <string>

Solution Name.

"Variation Name:=" <string>

Type: <string>

Variation name.

<RowItemsArray>

Array<MeshRowItems>

Row data.

<ItemPerRow>

Array<EntityPerRow>

Entity ID.

<MeshDataPer Item>

Array<Mesh Data Category and Stats>



*VB Example:*

```
Set oModule = oDesign.GetModule("FieldsReporter")

oModule.EditSurfaceMeshSummaryData Array(Array("NAME:SurfaceMeshSummary", "SolutionName:=", _
"Setup1 : LastAdaptive", "Variation:=", "Nominal", Array("NAME:MeshRowItems", Array
("NAME:ItemPerRow", "EntityPerRow:=", _
"annular_rng", Array("NAME:MeshDataPerItem", "Min Edge Len:=", 0.000166961133900917, "Max Edge
Len:=", _
0.00145730991671888, "Mean Edge Len:=", 0.000524160923260581, "Std Devn Edge Len:=", _
0.000217758706109324, "Min Tri Area:=", 3.4982570283924E-09, "Max Tri Area:=", _
3.73116346661249E-07, "Mean Tri Area:=", 8.91526236529066E-08, "Std Devn Tri Area:=", _
6.89482807990471E-08, "DataState:=", 2)), Array("NAME:ItemPerRow", "EntityPerRow:=", _
"gap", Array("NAME:MeshDataPerItem", "Min Edge Len:=", 0.000105681286042456, "Max Edge Len:=", _
0.000362084671538928, "Mean Edge Len:=", 0.000233090954707978, "Std Devn Edge Len:=", _
7.55387369793154E-05, "Min Tri Area:=", 1.74640378345369E-09, "Max Tri Area:=", _
3.78166789228631E-08, "Mean Tri Area:=", 1.56497981245286E-08, "Std Devn Tri Area:=", _
1.04691637802914E-08, "DataState:=", 2))))))
```

| Python Syntax  | RenameReport (<OldReportName>, <NewReportName>)                                                |
|----------------|------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EditSurfaceMeshSummaryData (     [     [         "NAME:SurfaceMeshSummary",</pre> |

```
"SolutionName:=" , "Setup1 : LastAdaptive",
"Variation:=" , "Nominal",
[
 "NAME:MeshRowItems",
 [
 "NAME:ItemPerRow",
 "EntityPerRow:=" , "annular_rng",
 [
 "NAME:MeshDataPerItem",
 "Min Edge Len:=" , 0.000166961133900917,
 "Max Edge Len:=" , 0.00145730991671888,
 "Mean Edge Len:=" , 0.000524160923260581,
 "Std Devn Edge Len:=" , 0.000217758706109324,
 "Min Tri Area:=" , 3.4982570283924E-09,
 "Max Tri Area:=" , 3.73116346661249E-07,
 "Mean Tri Area:=" , 8.91526236529066E-08,
 "Std Devn Tri Area:=" , 6.89482807990471E-08,
 "DataState:=" , 2
]
]
],
```

```
[
 "NAME:ItemPerRow",
 "EntityPerRow:=" , "gap",
 [
 "NAME:MeshDataPerItem",
 "Min Edge Len:=" , 0.000105681286042456,
 "Max Edge Len:=" , 0.000362084671538928,
 "Mean Edge Len:=" , 0.000233090954707978,
 "Std Devn Edge Len:=" , 7.55387369793154E-05,
 "Min Tri Area:=" , 1.74640378345369E-09,
 "Max Tri Area:=" , 3.78166789228631E-08,
 "Mean Tri Area:=" , 1.56497981245286E-08,
 "Std Devn Tri Area:=" , 1.04691637802914E-08,
 "DataState:=" , 2
]
]
]
])
```

## EnterComplex

Enters a complex number onto the stack.

|                     |                                                                                 |        |                                        |
|---------------------|---------------------------------------------------------------------------------|--------|----------------------------------------|
| <b>UI Access</b>    | Click <b>Number</b> , and then click <b>Scalar. Complex</b> option is selected. |        |                                        |
| <b>Parameters</b>   | Name                                                                            | Type   | Description                            |
|                     | <ComplexNum>                                                                    | String | String of complex value. Ex. "1 + 2j". |
| <b>Return Value</b> | None.                                                                           |        |                                        |

|                       |                                 |  |  |
|-----------------------|---------------------------------|--|--|
| <b>Python Syntax</b>  | EnterComplex(<ComplexNum>)      |  |  |
| <b>Python Example</b> | oModule.EnterComplex("1 + 2 j") |  |  |

|                   |                                |  |  |
|-------------------|--------------------------------|--|--|
| <b>VB Syntax</b>  | EnterComplex <ComplexNum>      |  |  |
| <b>VB Example</b> | oModule.EnterComplex "1 + 2 j" |  |  |

## EnterComplexVector

Enters a complex vector onto the stack.

|                     |                                                                                 |       |                                                        |
|---------------------|---------------------------------------------------------------------------------|-------|--------------------------------------------------------|
| <b>UI Access</b>    | Click <b>Number</b> , and then click <b>Vector. Complex</b> option is selected. |       |                                                        |
| <b>Parameters</b>   | Name                                                                            | Type  | Description                                            |
|                     | <ComplexVector>                                                                 | Array | Array of strings containing X, Y and Z complex values. |
| <b>Return Value</b> | None.                                                                           |       |                                                        |

| Python Syntax  | EnterComplexVector(<ComplexVector>)                                                                               |
|----------------|-------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.EnterComplexVector<br/><br/>( [<br/>    "1 + 2 j",<br/>    "1 + 2 j",<br/>    "1 + 2 j"<br/>])</pre> |

| VB Syntax  | EnterComplexVector <ComplexVector>                                                                   |
|------------|------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.EnterComplexVector _<br/>Array("1 + 2 j",_<br/>    "1 + 2 j",_<br/>    "1 + 2 j")</pre> |

## EnterCoord

Enters a coordinate system defined in the 3D Modeler editor.

| UI Access    | Click <b>Geometry</b> and then select <b>Coord</b> . |        |                                                               |
|--------------|------------------------------------------------------|--------|---------------------------------------------------------------|
| Parameters   | Name                                                 | Type   | Description                                                   |
|              | <CoordName>                                          | String | Name of a coordinate system defined in the 3D Modeler editor. |
| Return Value | None.                                                |        |                                                               |

|                       |                              |
|-----------------------|------------------------------|
| <b>Python Syntax</b>  | EnterCoord(<CoordName>)      |
| <b>Python Example</b> | oModule.EnterCoord("Global") |

|                   |                             |
|-------------------|-----------------------------|
| <b>VB Syntax</b>  | EnterCoord <CoordName>      |
| <b>VB Example</b> | oModule.EnterCoord "Global" |

## EnterEdge

Enters an edge defined in the 3D Modeler editor.

|                     |            |        |                                                   |
|---------------------|------------|--------|---------------------------------------------------|
| <b>UI Access</b>    | N/A        |        |                                                   |
| <b>Parameters</b>   | Name       | Type   | Description                                       |
|                     | <EdgeName> | String | Name of an edge defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.      |        |                                                   |

|                       |                              |
|-----------------------|------------------------------|
| <b>Python Syntax</b>  | EnterEdge(<EdgeName>)        |
| <b>Python Example</b> | oModule.EnterPoint("Edge_1") |

|                   |                             |
|-------------------|-----------------------------|
| <b>VB Syntax</b>  | EnterEdge <EdgeName>        |
| <b>VB Example</b> | oModule.EnterPoint "Edge_1" |

## EnterLine

Enters a line defined in the 3D Modeler editor.

|              |                                                   |        |                                                  |
|--------------|---------------------------------------------------|--------|--------------------------------------------------|
| UI Access    | Click <b>Geometry</b> and then select <b>Line</b> |        |                                                  |
| Parameters   | Name                                              | Type   | Description                                      |
|              | <LineName>                                        | String | Name of a line defined in the 3D Modeler editor. |
| Return Value | None.                                             |        |                                                  |

|                |                                         |
|----------------|-----------------------------------------|
| Python Syntax  | EnterLine(<LineName>)                   |
| Python Example | <code>oModule.EnterLine("Line1")</code> |

|            |                                        |
|------------|----------------------------------------|
| VB Syntax  | EnterLine <LineName>                   |
| VB Example | <code>oModule.EnterLine "Line1"</code> |

## EnterOutputVar

Enters Output Vars, only valid for Eigenmode problems.

|              |                                                            |        |                                                    |
|--------------|------------------------------------------------------------|--------|----------------------------------------------------|
| UI Access    | Click <b>Geometry</b> and then select <b>Output Vars</b> . |        |                                                    |
| Parameters   | Name                                                       | Type   | Description                                        |
|              | <VarName>                                                  | String | Name of the output vars. Only 'freq' is supported. |
|              | <VarType>                                                  | String | Type of the output vars. 'Complex' type.           |
| Return Value | None.                                                      |        |                                                    |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | EnterOutputVar(<VarName>, <VarType>)      |
| <b>Python Example</b> | oModule.EnterOutputVar("Freq", "Complex") |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | EnterOutputVar <VarName>, <VarType>      |
| <b>VB Example</b> | oModule.EnterOutputVar "Freq", "Complex" |

## EnterPoint

Enters a point defined in the 3D Modeler editor.

|                     |                                                      |        |                                                   |
|---------------------|------------------------------------------------------|--------|---------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Point</b> . |        |                                                   |
| <b>Parameters</b>   | Name                                                 | Type   | Description                                       |
|                     | <PointName>                                          | String | Name of a point defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                                |        |                                                   |

|                       |                              |
|-----------------------|------------------------------|
| <b>Python Syntax</b>  | EnterPoint(<PointName>)      |
| <b>Python Example</b> | oModule.EnterPoint("Point1") |

|                  |                        |
|------------------|------------------------|
| <b>VB Syntax</b> | EnterPoint <PointName> |
|------------------|------------------------|



|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Example</b> | <code>oModule.EnterPoint "Point1"</code> |
|-------------------|------------------------------------------|

## EnterQty

Enters a field quantity.

|                     |                                                        |        |                                                  |
|---------------------|--------------------------------------------------------|--------|--------------------------------------------------|
| <b>UI Access</b>    | Click <b>Quantity</b> , and then select from the list. |        |                                                  |
| <b>Parameters</b>   | Name                                                   | Type   | Description                                      |
|                     | <FieldQty>                                             | String | The field quantity to be entered onto the stack. |
| <b>Return Value</b> | None.                                                  |        |                                                  |

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | <code>EnterQty(&lt;FieldQty&gt;)</code> |
| <b>Python Example</b> | <code>oModule.EnterQty("E")</code>      |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | <code>EnterQty &lt;FieldQty&gt;</code> |
| <b>VB Example</b> | <code>oModule.EnterQty "E"</code>      |

## EnterScalar

Enters a scalar onto the stack.

|                   |                                                                                        |        |                                          |
|-------------------|----------------------------------------------------------------------------------------|--------|------------------------------------------|
| <b>UI Access</b>  | Click <b>Number</b> and then click <b>Scalar</b> . <b>Complex</b> option not selected. |        |                                          |
| <b>Parameters</b> | Name                                                                                   | Type   | Description                              |
|                   | <Scalar>                                                                               | Double | The real number to enter onto the stack. |

|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                                    |
|-----------------------|----------------------------------------------------|
| <b>Python Syntax</b>  | EnterScalar(<Scalar>)                              |
| <b>Python Example</b> | <code>oModule.EnterScalar(3.14159265358979)</code> |

|                   |                                                   |
|-------------------|---------------------------------------------------|
| <b>VB Syntax</b>  | EnterScalar <Scalar>                              |
| <b>VB Example</b> | <code>oModule.EnterScalar 3.14159265358979</code> |

## EnterScalarFunc

Enters a scalar function.

|                     |                                                       |        |                                                                  |
|---------------------|-------------------------------------------------------|--------|------------------------------------------------------------------|
| <b>UI Access</b>    | Click <b>Function</b> and then select <b>Scalar</b> . |        |                                                                  |
| <b>Parameters</b>   | Name                                                  | Type   | Description                                                      |
|                     | <VarName>                                             | String | Name of a variable to enter as a scalar function onto the stack. |
| <b>Return Value</b> | None.                                                 |        |                                                                  |

|                       |                                               |
|-----------------------|-----------------------------------------------|
| <b>Python Syntax</b>  | EnterScalarFunc(<VarName>)                    |
| <b>Python Example</b> | <code>oModule.EnterScalarFunc("Phase")</code> |

|                   |                                 |
|-------------------|---------------------------------|
| <b>VB Syntax</b>  | EnterScalarFunc <VarName>       |
| <b>VB Example</b> | oModule.EnterScalarFunc "Phase" |

## EnterSurf

Enters a surface defined in the 3D Modeler editor.

|                     |                                                        |        |                                                     |
|---------------------|--------------------------------------------------------|--------|-----------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Surface</b> . |        |                                                     |
| <b>Parameters</b>   | Name                                                   | Type   | Description                                         |
|                     | <SurfName>                                             | String | Name of a surface defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                                  |        |                                                     |

|                       |                                 |
|-----------------------|---------------------------------|
| <b>Python Syntax</b>  | EnterSurf(<SurfName>)           |
| <b>Python Example</b> | oModule.EnterSurf("Rectangle1") |

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Syntax</b>  | EnterSurf <SurfName>           |
| <b>VB Example</b> | oModule.EnterSurf "Rectangle1" |

## EnterVector

Enters a vector onto the stack.

|                   |                                                                                          |      |             |
|-------------------|------------------------------------------------------------------------------------------|------|-------------|
| <b>UI Access</b>  | Click <b>Number</b> , and then click <b>Vector</b> . <b>Complex</b> option not selected. |      |             |
| <b>Parameters</b> | Name                                                                                     | Type | Description |
|                   |                                                                                          |      |             |

|               |                                                                                                                                             |                                                                  |       |                                                                  |
|---------------|---------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------|-------|------------------------------------------------------------------|
|               | <table><tr><td>&lt;VectorArray&gt;</td><td>Array</td><td>Array of strings containing X, Y and Z components of the vector.</td></tr></table> | <VectorArray>                                                    | Array | Array of strings containing X, Y and Z components of the vector. |
| <VectorArray> | Array                                                                                                                                       | Array of strings containing X, Y and Z components of the vector. |       |                                                                  |
| Return Value  | None.                                                                                                                                       |                                                                  |       |                                                                  |

|                |                                                   |
|----------------|---------------------------------------------------|
| Python Syntax  | EnterVector(<VectorArray>)                        |
| Python Example | <code>oModule.EnterVector([1.0, 1.0, 1.0])</code> |

|            |                                                       |
|------------|-------------------------------------------------------|
| VB Syntax  | EnterVector <VectorArray>                             |
| VB Example | <code>oModule.EnterVector Array(1.0, 1.0, 1.0)</code> |

## EnterVectorFunc

Enters a vector function.

|              |                                                       |       |                                                                                                                                            |
|--------------|-------------------------------------------------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Click <b>Function</b> and then select <b>Vector</b> . |       |                                                                                                                                            |
| Parameters   | Name                                                  | Type  | Description                                                                                                                                |
|              | <VarNames>                                            | Array | Array of strings containing names of a variable for the X, Y, and Z coordinates, respectively, to enter as a vector function on the stack. |
| Return Value | None.                                                 |       |                                                                                                                                            |

|               |                             |
|---------------|-----------------------------|
| Python Syntax | EnterVectorFunc(<VarNames>) |
|---------------|-----------------------------|

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Example</b> | <code>oModuleEnterVectorFunc(["X", "Y", "Z"])</code> |
|-----------------------|------------------------------------------------------|

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EnterVectorFunc &lt;VarNames&gt;</code>            |
| <b>VB Example</b> | <code>oModuleEnterVectorFunc Array("X", "Y", "Z")</code> |

## EnterVol

Enters a volume defined in the 3D Modeler editor.

|                     |                                                       |             |                                                    |
|---------------------|-------------------------------------------------------|-------------|----------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Volume</b> . |             |                                                    |
| <b>Parameters</b>   | <b>Name</b>                                           | <b>Type</b> | <b>Description</b>                                 |
|                     | <VolumeName>                                          | String      | Name of a volume defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                                 |             |                                                    |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | <code>EnterVol(&lt;VolumeName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.EnterVol("Box1")</code>     |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | <code>EnterVol &lt;VolumeName&gt;</code> |
| <b>VB Example</b> | <code>oModule.EnterVol "Box1"</code>     |

## ExportFieldPlot

Exports a field plot to a file.

|                     |              |             |                                                                                        |
|---------------------|--------------|-------------|----------------------------------------------------------------------------------------|
| <b>UI Access</b>    | N/A          |             |                                                                                        |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>                                                                     |
|                     | <PlotName>   | String      | Name of the field plot to export.                                                      |
|                     | <ShowHeader> | Boolean     | <b>True</b> - export field plot header to a file.<br><b>False</b> - export field plot. |
|                     | <FileName>   | String      | Name of file to save as, including the file path.                                      |
| <b>Return Value</b> | None.        |             |                                                                                        |

|                       |                                                                               |
|-----------------------|-------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportFieldPlot(<PlotName>, <ShowHeader>, <FileName>)                         |
| <b>Python Example</b> | <pre>oModule.ExportFieldPlot("Mag_E1",<br/>True, "C:/field_report.dsp")</pre> |

|                   |                                                                                |
|-------------------|--------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportFieldPlot <PlotName>, <ShowHeader>, <FileName>                           |
| <b>VB Example</b> | <pre>oModule.ExportFieldPlot "Mag_E1", _<br/>true, "C:/field_report.dsp"</pre> |

## ExportMarkerTable

Exports the marker table to a .csv or .tab file.

|                  |                                                                                  |
|------------------|----------------------------------------------------------------------------------|
| <b>UI Access</b> | [product] > <b>Fields &gt; Plot Fields &gt; Marker &gt; Export Marker Table.</b> |
|------------------|----------------------------------------------------------------------------------|

|              |            |        |                                   |
|--------------|------------|--------|-----------------------------------|
| Parameters   | Name       | Type   | Description                       |
|              | <FileName> | String | Name of export file include path. |
| Return Value | None.      |        |                                   |

|                |                                                                        |
|----------------|------------------------------------------------------------------------|
| Python Syntax  | ExportMarkerTable(<FileName>)                                          |
| Python Example | <code>oModule.ExportMarkerTable("C:/work/FieldMarkerTable.csv")</code> |

|            |                                                                       |
|------------|-----------------------------------------------------------------------|
| VB Syntax  | ExportMarkerTable <FileName>                                          |
| VB Example | <code>oModule.ExportMarkerTable "C:/work/FieldMarkerTable.csv"</code> |

## ExportOnGrid [Field Overlays]

Evaluates the top stack element at a set of points specified by a grid and exports the data to a file.

|            |                                                       |         |                                                                                          |
|------------|-------------------------------------------------------|---------|------------------------------------------------------------------------------------------|
| UI Access  | Click <b>Export</b> , and then click <b>On Grid</b> . |         |                                                                                          |
| Parameters | Name                                                  | Type    | Description                                                                              |
|            | <OutputFile>                                          | String  | Name of the output file.                                                                 |
|            | <Min>                                                 | Array   | Minimum values for the coordinate components of the grid system                          |
|            | <Max>                                                 | Array   | Maximum values for the coordinate components of the grid system                          |
|            | <Spacing>                                             | Array   | Spacing values for the coordinate components of the grid system                          |
|            | <SolnName>                                            | String  | Name of the simulation setup                                                             |
|            | <VarVals>                                             | Array   | Array of strings containing setup definitions.                                           |
|            | <IncludePoints>                                       | Boolean | Optional. Specifies whether include points in the output file.                           |
|            | <CSType>                                              | String  | Optional. Type of coordinate system. "Cartesian" (default)   "Cylindrical"   "Spherical" |

|                     |                        |         |                                                                                                                                        |
|---------------------|------------------------|---------|----------------------------------------------------------------------------------------------------------------------------------------|
|                     | <b>&lt;Offsets&gt;</b> | Array   | Optional. Origin for the offset coordinate system. For Cartesian, x, y, z, for Cylindrical, R, Phi, Z, for Spherical, Rho, Theta, Phi. |
|                     | <b>&lt;ByCount&gt;</b> | Boolean | Optional.                                                                                                                              |
| <b>Return Value</b> | None.                  |         |                                                                                                                                        |

**Note:**Regarding the **ExportOnGrid** legacy script which only has “IncludePtInOutput” argument (the last Boolean one), AEDT can still read it and assign other new arguments as default values. Those default values are RefCSName = “global”, PtInSI = “True”, FieldInReCS = “False”

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                     |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportOnGrid(<OutputFile>, <Min>, <Max>, <Spacing>, <SolnName>, <SolnParameters>, [<ExportOption>, "IncludePointsInOutput:=", <boolean>], " RefCSName:=", <CSName>, "PtsInSI:=", <boolean>, "FieldRefCS:=", <boolean>], "<CSType>", [ <Offsets>, <ByCount>)                                                                                                                                                         |
| <b>Python Example</b> | <pre>oModule.ExportOnGrid("C:\offset_grid_model_unit_ref.fld",     ["-1mm", "16mm", "0mm"],     ["1mm", "18mm", "1mm"],     ["2mm", "2mm", "1mm"],     "4500MHz : LastAdaptive",     ["Freq:=", "4.5GHz", "Phase:="      , "0deg"],     [         "NAME:ExportOption",         "IncludePtInOutput:="      , True,         "RefCSName:="              , "offset",         "PtInSI:="                  , False,</pre> |



|  |                                                                                                       |
|--|-------------------------------------------------------------------------------------------------------|
|  | <pre> "FieldInRefCS:="          , True      ],     "Cartesian", ["0mm", "0mm", "0mm"], False ) </pre> |
|--|-------------------------------------------------------------------------------------------------------|

| VB Syntax  | ExportOnGrid(<OutputFile>, <Min>, <Max>, <Spacing>, <SolnName>, <SolnParameters>, [<ExportOption>, "IncludePointsInOutput:=", <boolean>], "RefCSName:=", <CSName>, "PtsInSl:=", <boolean>, "FieldRefCS:=", <boolean>], "<CSType>", [<Offsets>, <ByCount>])                                                                                                                                                                                            |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.ExportOnGrid "C:/Grid.fld", _ Array( "0mm", "0deg", "-25mm"), _ Array("20mm", "90deg", "125mm"), _ Array("10mm", "45deg", "50mm"), _ "Setup1 : LastAdaptive", _ Array("Freq:=", "10000Hz", "Phase:=", "0deg"), _ true, "Cylindrical", _ Array("0mm", "0mm", "0mm")  oModule.ExportOnGrid("C:\offset_grid_model_unit_ref.fld",     Array("-1mm", "16mm", "0mm"),     Array("1mm", "18mm", "1mm"),     Array("2mm", "2mm", "1mm"), </pre> |

|  |                                                                                                                                                                                                                                                                                                                                         |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"4500MHz : LastAdaptive", Array("Freq:=", "4.5GHz","Phase:="      , "0deg"), [     "NAME:ExportOption",     "IncludePtInOutput:="      , True,     "RefCSName:="              , "offset",     "PtInSI:="                  , False,     "FieldInRefCS:="           , True ], "Cartesian", Array("0mm", "0mm", "0mm"), False )</pre> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

ExportPlotImageToFile

Deprecated. Use [ExportPlotImageWithViewToFile](#).

Creates field plot exports of existing field plots from a given view points, and with the model being auto-sized automatically for each view.

|            |                                |        |                                                             |
|------------|--------------------------------|--------|-------------------------------------------------------------|
| UI Access  | N/A                            |        |                                                             |
| Parameters | Name                           | Type   | Description                                                 |
|            | <FileName>                     | String | Full path plus file name.                                   |
|            | <FolderName>                   | String | Plot folder name.                                           |
|            | <ItemName>                     | String | Name of fields to plot.                                     |
|            | <SetViewTopDownDirectionByRCS> | String | Optional. Name of relative coordinate system to use for the |

|                     |                                  |
|---------------------|----------------------------------|
|                     | <input type="text"/> field plot. |
| <b>Return Value</b> | None.                            |

|                       |                                                                                                                                  |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ExportPlotImageToFile(&lt;FileName&gt;, &lt;FolderName&gt;, &lt;ItemName&gt;, &lt;SetViewTopDownDirectionByRCS&gt;)</code> |
| <b>Python Example</b> | <pre>oModule.ExportPlotImageToFile(<br/>    "C:\TestEPITF2.jpg", "",<br/>    "Mag_E2", "RelativeCS1")</pre>                      |

|                   |                                                                                                                                 |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>ExportPlotImageToFile &lt;FileName&gt;, &lt;FolderName&gt;, &lt;ItemName&gt;, &lt;SetViewTopDownDirectionByRCS&gt;</code> |
| <b>VB Example</b> | <pre>oModule.ExportPlotImageToFile _<br/>    "C:\TestEPITF2.jpg", "", _<br/>    "Mag_E2", "RelativeCS1"</pre>                   |

## ExportPlotImageWithViewToFile [Reporter]

Exports a field plot image to a specified file.

**Note:**

This script replaces ExportPlotImageToFile, which is deprecated.

|                  |     |
|------------------|-----|
| <b>UI Access</b> | N/A |
|------------------|-----|

|              |                                               |         |                                                       |
|--------------|-----------------------------------------------|---------|-------------------------------------------------------|
| Parameters   | Name                                          | Type    | Description                                           |
|              | <FileName>                                    | String  | Full path of file to export.                          |
|              | <PlotQuantityName>                            | String  | Name of quantity to plot.                             |
|              | <PlotItemName>                                | String  | Name of fields to plot.                               |
|              | <PixelSizeX>                                  | Integer | Desired image width, in pixels.                       |
|              | <PixelSizeY>                                  | Integer | Desired image height, in pixels.                      |
|              | <ViewOrientation>                             | String  | <i>Optional.</i> Name of orientation to use for plot. |
| Return Value | Image file is exported to the specified path. |         |                                                       |

|                |                                                                                                                                    |
|----------------|------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ExportPlotImageWithViewToFile(<FileName>, <PlotQuantityName>, <PlotItemName>, <PixelSizeX>, <PixelSizeY>, <ViewOrientation>)       |
| Python Example | <pre>oModule.ExportPlotImageWithViewToFile("E:/MyDir/OptimToutput/magE2.gif", "E Field",<br/>"Mag_E2", 1920, 1080, "newot2")</pre> |

|            |                                                                                                                                     |
|------------|-------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | ExportPlotImageWithViewToFile <FileName>, <PlotQuantityName>, <PlotItemName>, <PixelSizeX>, <PixelSizeY>, <ViewOrientation>         |
| VB Example | <pre>oModule.ExportPlotImageWithViewToFile "E:/MyDir/OptimToutput/magE2.gif", "E Field", _<br/>"Mag_E2", 1920, 1080, "newot2"</pre> |

## ExportSurfaceMeshSummary

*Use:* Exports a Surface Mesh Summary.

*Command:* **Create Surface Mesh Summary.**

**Syntax:** ExportSurfaceMeshSummary Array(<SolutionName>, <DesignVariationKey> <ExportFileName>)

**Return Value:** None

**Parameters:** <SolutionName>

Type: <string>

*Original name of the plot.*

<DesignVariationKey>

Type: <string>

*New name of the plot.*

*VB Example:*

```
oModule.ExportSurfaceMeshSummary Array("SolutionName:=", "Setup1 : LastAdaptive", "DesignVari-
ationKey:=", _
"Nominal", "ExportFileName:=", "D:\documents\surfaceMeshSummaryReport.csv")
```

| Python Syntax  | ExportSurfaceMeshSummary (<SolutionName>, <DesignVariationKey>, <ExportFileName>)                                                                                                                                                                                                  |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule.ExportSurfaceMeshSummary(<br/>    [<br/>        "SolutionName:=" , "Setup1 : LastAdaptive",<br/>        "DesignVariationKey:=" , "Nominal",<br/>        "ExportFileName:=" , "D:\\Program Files\\Docu-<br/>ments\\surfaceMeshSummaryReport.csv"<br/>    ]<br/>)</pre> |

# ExportToFile

**Note:**

The ExportToFile script command has replaced the script command [ExportReport](#). ExportReport remains in order to retain backward compatibility for existing scripts, but it is strongly recommended that you now use ExportToFile.

From a data table or plot, generates text format, comma delimited, tab delimited, or .dat type output files.

|              |                                                                                |         |                                                                                                                                                                                                   |
|--------------|--------------------------------------------------------------------------------|---------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Right-click on report name in the project tree and select <b>Export Data</b> . |         |                                                                                                                                                                                                   |
| Parameters   | Name                                                                           | Type    | Description                                                                                                                                                                                       |
|              | <ReportName>                                                                   | String  | Name of report to be exported.                                                                                                                                                                    |
|              | <FileName>                                                                     | String  | Full path of the exported image file name; with extension of<br>.txt - Post processor format file<br>.csv - Comma-delimited data file<br>.tab - Tab-separated file<br>.dat - Ansys plot data file |
|              | <UnitSpec>                                                                     | String  | For example, "kV, Mhz, yd"                                                                                                                                                                        |
|              | <UseTraceNumberFormat>                                                         | Boolean | "True", "False"                                                                                                                                                                                   |
| Return Value | None.                                                                          |         |                                                                                                                                                                                                   |

|                       |                                                                                                                                                                                           |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportToFile(<ReportName>, <FileName>)                                                                                                                                                    |
| <b>Python Example</b> | <pre>oModule.ExportToFile("rETotal", "C:/Users/Documents/rETotal.csv")<br/>oModule.ExportToFile("S Parameter Table 1", "D:/Users/Documents/cfft.csv", False, " kV, MHz, yd ", True)</pre> |

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportToFile <ReportName>, <FileName>                                 |
| <b>VB Example</b> | <pre>oModule.ExportToFile "rETotal", "C:/Documents/rETotal.csv"</pre> |

## GetFieldFolderNames

Gets the folder names of the field plots.

|                     |                        |
|---------------------|------------------------|
| <b>UI Access</b>    | N/A                    |
| <b>Parameters</b>   | None.                  |
| <b>Return Value</b> | Array of folder names. |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | GetFieldFolderNames()                    |
| <b>Python Example</b> | <pre>oModule.GetFieldFolderNames()</pre> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | GetFieldFolderNames                    |
| <b>VB Example</b> | <pre>oModule.GetFieldFolderNames</pre> |

## GetFieldPlotNames

Gets the names of field overlay plots defined in a design.

|                     |                                                |
|---------------------|------------------------------------------------|
| <b>UI Access</b>    | N/A                                            |
| <b>Parameters</b>   | None.                                          |
| <b>Return Value</b> | Array of strings containing field plots names. |

|                       |                                          |
|-----------------------|------------------------------------------|
| <b>Python Syntax</b>  | GetFieldPlotNames()                      |
| <b>Python Example</b> | <code>oModule.GetFieldPlotNames()</code> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | GetFieldPlotNames                      |
| <b>VB Example</b> | <code>oModule.GetFieldPlotNames</code> |

## GetFieldPlotQuantityName

Gets the quantity name of a specified field plot.

|                     |                            |        |                         |
|---------------------|----------------------------|--------|-------------------------|
| <b>UI Access</b>    | N/A                        |        |                         |
| <b>Parameters</b>   | Name                       | Type   | Description             |
|                     | <PlotName>                 | String | Name of specified plot. |
| <b>Return Value</b> | String plot quantity name. |        |                         |



|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <b>Python Syntax</b>  | GetFieldPlotQuantityName(<PlotName>)                    |
| <b>Python Example</b> | <code>oModule.GetFieldPlotQuantityName("Mag_H1")</code> |

|                   |                                                        |
|-------------------|--------------------------------------------------------|
| <b>VB Syntax</b>  | GetFieldPlotQuantityName <PlotName>                    |
| <b>VB Example</b> | <code>oModule.GetFieldPlotQuantityName "Mag_H1"</code> |

## GetMeshPlotNames

Gets the names of mesh plots defined in a design.

|                     |                     |
|---------------------|---------------------|
| <b>UI Access</b>    | N/A                 |
| <b>Parameters</b>   | None..              |
| <b>Return Value</b> | Array of plot name. |

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | GetMeshPlotNames                        |
| <b>Python Example</b> | <code>oModule.GetMeshPlotNames()</code> |

|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | GetMeshPlotNames                      |
| <b>VB Example</b> | <code>oModule.GetMeshPlotNames</code> |

# ModifyFieldPlot

Modifies a plot definition.

|              |                                            |        |                                     |
|--------------|--------------------------------------------|--------|-------------------------------------|
| UI Access    | [product] > <b>Fields &gt; Modify Plot</b> |        |                                     |
| Parameters   | Name                                       | Type   | Description                         |
|              | <OriginalName>                             | String | Name of the plot to be modified.    |
|              | <PlotParams>                               | Array  | Array containing modified settings. |
| Return Value | None.                                      |        |                                     |

|                |                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ModifyFieldPlot(<OriginalName>, <PlotParams>)                                                                                                                                                                                                                                                                                                                                               |
| Python Example | <pre>oModule.ModifyFieldPlot("Vector_E1" ,     ["NAME:Vector_E2",         "SolutionName:=", "Setup1 : LastAdaptive",         "QuantityName:=", "Vector_E",         "PlotFolder:=", "E Field1",         "UserSpecifyName:=", 0,         "UserSpecifyFolder:=", 0,         "IntrinsicVar:=", "Freq='1GHz' Phase='30deg'",         "PlotGeomInfo:=", [1, "Surface","FacesList", 1, "7"],</pre> |

|  |                                                                                                                                                                                                                                                                                                            |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "FilterBoxes:=", [0], ["NAME:PlotOnSurfaceSettings", "Filled:=", False, "IsoValType:=", "Fringe", "SmoothShade:=", True, "AddGrid:=", False, "MapTransparency:=", True, "Transparency:=", 0, _ "ArrowUniform:=", True, "ArrowSpacing:=", 0.100000001490116, "GridColor:=", [255, 255, 255]] ] </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | ModifyFieldPlot <OriginalName>, <PlotParams>                                                                                                                                                                                                                                               |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.ModifyFieldPlot "Vector_E1" ,_     Array("NAME:Vector_E2", _         "SolutionName:=", "Setup1 : LastAdaptive", _         "QuantityName:=", "Vector_E", _         "PlotFolder:=", "E Field1", _         "UserSpecifyName:=", 0, _         "UserSpecifyFolder:=", 0, _ </pre> |

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"IntrinsicVar:=", "Freq='1GHz' Phase='30deg'", _<br/>"PlotGeomInfo:=", Array(1, "Surface", "FacesList", 1, "7"), _<br/>"FilterBoxes:=", Array(0), _<br/>Array("NAME:PlotOnSurfaceSettings", _<br/>"Filled:=", false, _<br/>"IsoValType:=", "Fringe", _<br/>"SmoothShade:=", true, _<br/>"AddGrid:=", false, _<br/>"MapTransparency:=", true, _<br/>"Transparency:=", 0, _<br/>"ArrowUniform:=", true, _<br/>"ArrowSpacing:=", 0.100000001490116, _<br/>"GridColor:=", Array(255, 255, 255)))</pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

GetTopEntryValue

Gets the value of the top entry of the calculator stack.

|            |            |        |                                      |
|------------|------------|--------|--------------------------------------|
| UI Access  | N/A        |        |                                      |
| Parameters | Name       | Type   | Description                          |
|            | <SolnName> | String | Name of specified solution.          |
|            | <VarVals>  | Array  | Array of variable name, value pairs. |

|                     |                                               |
|---------------------|-----------------------------------------------|
| <b>Return Value</b> | Array of strings containing top entry values. |
|---------------------|-----------------------------------------------|

|                       |                                                                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetTopEntryValue(<SolnName>, <VarVals>)                                                                                                    |
| <b>Python Example</b> | <pre>oModule.GetTopEntryValue(<br/>    "Setup1:LastAdaptive",<br/>    ["Freq:=", "1GHz", "Phase:=", "0deg", "x_size:=", "2mm"]<br/>)</pre> |

|                   |                                                                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetTopEntryValue <SolnName>, <VarVals>                                                                                                       |
| <b>VB Example</b> | <pre>oModule.GetTopEntryValue "Setup1:LastAdaptive", _<br/>    Array("Freq:=", "1GHz", "Phase:=", "0deg", _<br/>    "x_size:=", "2mm")</pre> |

## HidePolarPlot

Hides a currently visible polar plot.

|                     |            |        |                         |
|---------------------|------------|--------|-------------------------|
| <b>UI Access</b>    | N/A        |        |                         |
| <b>Parameters</b>   | Name       | Type   | Description             |
|                     | <PlotName> | String | Name of the polar plot. |
| <b>Return Value</b> | None.      |        |                         |

|                       |                                    |
|-----------------------|------------------------------------|
| <b>Python Syntax</b>  | HidePolarPlot(<PlotName>)          |
| <b>Python Example</b> | oModule.HidePolarPlot("rE Plot 1") |

|                   |                                   |
|-------------------|-----------------------------------|
| <b>VB Syntax</b>  | HidePolarPlot <PlotName>          |
| <b>VB Example</b> | oModule.HidePolarPlot "rE Plot 1" |

## HideRadiatedPlotOverlay

Hides overlay radiation fields plot.

|                     |            |        |                         |
|---------------------|------------|--------|-------------------------|
| <b>UI Access</b>    | N/A        |        |                         |
| <b>Parameters</b>   | Name       | Type   | Description             |
|                     | <PlotName> | String | Name of specified plot. |
| <b>Return Value</b> | None.      |        |                         |

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Syntax</b>  | HideRadiatedPlotOverlay(<PlotName>)            |
| <b>Python Example</b> | oModule.HideRadiatedPlotOverlay("Gain Plot 1") |

|                   |                                               |
|-------------------|-----------------------------------------------|
| <b>VB Syntax</b>  | HideRadiatedPlotOverlay <PlotName>            |
| <b>VB Example</b> | oModule.HideRadiatedPlotOverlay "Gain Plot 1" |

## LoadNamedExpressions

Loads a named expression definition from a saved file.

|                     |                                                                          |        |                                                                                       |
|---------------------|--------------------------------------------------------------------------|--------|---------------------------------------------------------------------------------------|
| <b>UI Access</b>    | In the Fields Calculator, click <b>Load From...</b> in the Library area. |        |                                                                                       |
| <b>Parameters</b>   | Name                                                                     | Type   | Description                                                                           |
|                     | <FileName>                                                               | String | Filename and full path to the file to hold the named expression definition.           |
|                     | <FieldType>                                                              | String | For products with just one filed type, it is set to "Fields".                         |
|                     | <NamedExpr>                                                              | Array  | rray of strings containing the names of expression definitions to load from the file. |
| <b>Return Value</b> | None.                                                                    |        |                                                                                       |

|                       |                                                                                                                 |  |  |
|-----------------------|-----------------------------------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | LoadNamedExpressions(<FileName>, <FieldType>, <NamedExpr>)                                                      |  |  |
| <b>Python Example</b> | <pre>oModule.LoadNamedExpressions (<br/>"C:\Ansoft\PersonalLib\smth.clc",<br/>"Fields", ["smoothedtemp"])</pre> |  |  |

|                   |                                                                                                                       |  |  |
|-------------------|-----------------------------------------------------------------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | LoadNamedExpressions <FileName>, <FieldType>, <NamedExpr>                                                             |  |  |
| <b>VB Example</b> | <pre>oModule.LoadNamedExpressions _<br/>"C:\Ansoft\PersonalLib\smth.clc", _<br/>"Fields", Array("smoothedtemp")</pre> |  |  |

# ModifyFieldPlot

Modifies a plot definition.

|              |                                                |        |                                     |
|--------------|------------------------------------------------|--------|-------------------------------------|
| UI Access    | [product] > <b>Fields</b> > <b>Modify Plot</b> |        |                                     |
| Parameters   | Name                                           | Type   | Description                         |
|              | <OriginalName>                                 | String | Name of the plot to be modified.    |
|              | <PlotParams>                                   | Array  | Array containing modified settings. |
| Return Value | None.                                          |        |                                     |

|                |                                                                                                                                                                                                                                                                                                                                                                                              |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ModifyFieldPlot(<OriginalName>, <PlotParams>)                                                                                                                                                                                                                                                                                                                                                |
| Python Example | <pre>oModule.ModifyFieldPlot("Vector_E1" ,     ["NAME:Vector_E2",         "SolutionName:=", "Setup1 : LastAdaptive",         "QuantityName:=", "Vector_E",         "PlotFolder:=", "E Field1",         "UserSpecifyName:=", 0,         "UserSpecifyFolder:=", 0,         "IntrinsicVar:=", "Freq='1GHz' Phase='30deg'",         "PlotGeomInfo:=", [1, "Surface","FacesList", 1, "7"]],</pre> |



|  |                                                                                                                                                                                                                                                                                                            |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "FilterBoxes:=", [0], ["NAME:PlotOnSurfaceSettings", "Filled:=", False, "IsoValType:=", "Fringe", "SmoothShade:=", True, "AddGrid:=", False, "MapTransparency:=", True, "Transparency:=", 0, _ "ArrowUniform:=", True, "ArrowSpacing:=", 0.100000001490116, "GridColor:=", [255, 255, 255]] ] </pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | ModifyFieldPlot <OriginalName>, <PlotParams>                                                                                                                                                                                                                                               |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.ModifyFieldPlot "Vector_E1" ,_     Array("NAME:Vector_E2", _         "SolutionName:=", "Setup1 : LastAdaptive", _         "QuantityName:=", "Vector_E", _         "PlotFolder:=", "E Field1", _         "UserSpecifyName:=", 0, _         "UserSpecifyFolder:=", 0, _ </pre> |

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"IntrinsicVar:=", "Freq='1GHz' Phase='30deg'", _<br/>"PlotGeomInfo:=", Array(1, "Surface", "FacesList", 1, "7"), _<br/>"FilterBoxes:=", Array(0), _<br/>Array("NAME:PlotOnSurfaceSettings", _<br/>"Filled:=", false, _<br/>"IsoValType:=", "Fringe", _<br/>"SmoothShade:=", true, _<br/>"AddGrid:=", false, _<br/>"MapTransparency:=", true, _<br/>"Transparency:=", 0, _<br/>"ArrowUniform:=", true, _<br/>"ArrowSpacing:=", 0.100000001490116, _<br/>"GridColor:=", Array(255, 255, 255)))</pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

ModifyInceptionParameters

**Use:** For field line trace plots, allows you to set inception voltage evaluation parameters.

|            |                                                                                                                  |        |                                              |
|------------|------------------------------------------------------------------------------------------------------------------|--------|----------------------------------------------|
| UI Access  | Right-click on a <b>Field Line Trace Plot</b> in the project tree and select <b>Inception Voltage Evaluation</b> |        |                                              |
| Parameters | Name                                                                                                             | Type   | Description                                  |
|            | <NAME>                                                                                                           | string | The name of the inception value settings     |
|            | <Gas Type>                                                                                                       | int    | 0 for Dry Air, 1 for SF6, 2 for User Defined |
|            | <Gas Pressure>                                                                                                   | int    | Gas pressure in Bar                          |

|                     |                 |      |                                                      |
|---------------------|-----------------|------|------------------------------------------------------|
|                     | <Use Inception> | bool | <b>True</b> to use the inception parameters U0, K, A |
|                     | <Potential U0>  | int  | U0 parameter                                         |
|                     | <Potential K>   | int  | Streamer constant (empirical value)                  |
|                     | <Potential A>   | int  | A parameter                                          |
| <b>Return Value</b> | None            |      |                                                      |

|                       |                                                                                                                                                                                                                                                                                                                                                                                     |  |  |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | ModifyInceptionParameters (<NAME>, <Gas Type>, <Gas Pressure>, <Use Inception>, <Potential U0>, <Potential K>, <Potential A>)                                                                                                                                                                                                                                                       |  |  |
| <b>Python Example</b> | <pre>oModule.ModifyInceptionParameters("FieldLineTrace_Plot1",     [         "NAME:InceptionEvaluationSettings",         "Gas Type:="                , 1,         "Gas Pressure:="           , 1,         "Use Inception:="          , True,         "Potential U0:="            , 0,         "Potential K:="             , 1,         "Potential A:="            , 1     ]) </pre> |  |  |

|                   |                                                                                                                                                                                                                                              |  |  |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | ModifyInceptionParameters (<NAME>, <Gas Type>, <Gas Pressure>, <Use Inception>, <Potential U0>, <Potential K>, <Potential A>)                                                                                                                |  |  |
| <b>VB Example</b> | <pre>oModule.ModifyInceptionParameters "FieldLineTrace_Plot1", Array("NAME:InceptionEvaluationSettings", "Gas Type:=", 1, "Gas Pressure:=", 1, "Use Inception:=", false, "Potential U0:=", 0, "Potential K:=", 1, "Potential A:=", 1) </pre> |  |  |

# ReassignFieldPlot

Reassigns a field plot.

|              |                                                 |        |                                     |
|--------------|-------------------------------------------------|--------|-------------------------------------|
| UI Access    | Right-click on a field plot > <b>Reassign</b> . |        |                                     |
| Parameters   | Name                                            | Type   | Description                         |
|              | <PlotName>                                      | String | Name of specified plot to reassign. |
|              | <PlotParameterArray>                            | Array  | See <a href="#">CreateFieldPlot</a> |
| Return Value | None.                                           |        |                                     |

|                |                                                                                                                                                                                                                                                                                                                      |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ReassignFieldPlot(<PlotName>, <PlotParameterArray>)                                                                                                                                                                                                                                                                  |
| Python Example | <pre>oModule.ReassignFieldPlot "Mag_H1", Array("NAME:Mag_H1", _ "SolutionName:=", "Setup1 : LastAdaptive", _ "UserSpecifyName:=", 0, _ "UserSpecifyFolder:=", 0, _ "QuantityName:=", "Mag_H", _ "PlotFolder:=", "H Field", _ "StreamlinePlot:=", False, "AdjacentSidePlot:=", False, "FullModelPlot:=", False,</pre> |

```
"IntrinsicVar:=", "Freq=\'20GHz\' Phase=\'0deg\'",
"PlotGeomInfo:=", [1,"Surface","FacesList",1,"117"],
"FilterBoxes:=", [0],
["NAME:PlotOnSurfaceSettings",
 "Filled:=", False,
 "IsoValType:=", "Fringe",
 "AddGrid:=", False,
 "MapTransparency:=", True,
 "Refinement:=", 0,
 "Transparency:=", 0,
 "SmoothingLevel:=", 0,
 "ShadingType:=", 0,
 ["NAME:Arrow3DSpacingSettings",
 "ArrowUniform:=", True,
 "ArrowSpacing:=", 0,
 "MinArrowSpacing:=", 0,
 "MaxArrowSpacing:=", 0
],
 "GridColor:=", [255,255,255]
],
"EnableGaussianSmoothing:=", False
```

|  |    |
|--|----|
|  | l) |
|--|----|

| VB Syntax  | ReassignFieldPlot <PlotName>, <PlotParameterArray>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.ReassignFieldPlot "Mag_E1", _ Array("NAME:Mag_E1", "SolutionName:=", _ "Setup1 : LastAdaptive", "UserSpecifyName:=", 0, _ "UserSpecifyFolder:=", 0, "QuantityName:=", _ "Mag_E", "PlotFolder:=", "E Field", _ "StreamlinePlot:=", false, "AdjacentSidePlot:=", _ false, "FullModelPlot:=", false, "IntrinsicVar:=", _ "Freq=" &amp; Chr(39) &amp; "20GHz" &amp; Chr(39) &amp; " _ Phase=" &amp; Chr(39) &amp; "0deg" &amp; Chr(39) &amp; "", "PlotGeomInfo:=", Array( _ 1, "Surface", "FacesList", 1, "149"), _ "FilterBoxes:=", Array(0), _ Array("NAME:PlotOnSurfaceSettings", "Filled:=", _ false, "IsoValType:=", "Fringe", _ "AddGrid:=", false, "MapTransparency:=", true, "Refinement:=", _ 0, "Transparency:=", 0, "SmoothingLevel:=", 0, _ "ShadingType:=", 0, Array("NAME:Arrow3DSpacingSettings", "ArrowUniform:=", _</pre> |

```
true, "ArrowSpacing:=", 0, _
"MinArrowSpacing:=", 0, "MaxArrowSpacing:=", 0), "GridColor:=", Array(_
255, 255, 255)), "EnableGaussianSmoothing:=", false)
```

## RenameFieldPlot

Renames a plot.

|                     |                                                                                                                 |             |                            |
|---------------------|-----------------------------------------------------------------------------------------------------------------|-------------|----------------------------|
| <b>UI Access</b>    | Right-click the plot you want to rename in the project tree, and then click <b>Rename</b> on the shortcut menu. |             |                            |
| <b>Parameters</b>   | <b>Name</b>                                                                                                     | <b>Type</b> | <b>Description</b>         |
|                     | <OldName>                                                                                                       | String      | Original name of the plot. |
|                     | <NewName>                                                                                                       | String      | New name of the plot.      |
| <b>Return Value</b> | None.                                                                                                           |             |                            |

|                       |                                                                   |
|-----------------------|-------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameFieldPlot(&lt;OldName&gt;, &lt;NewName&gt;)</code>    |
| <b>Python Example</b> | <pre>oModule.RenameFieldPlot(<br/>"Vector_E1", "Vector_E2")</pre> |

|                   |                                                                   |
|-------------------|-------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameFieldPlot &lt;OldName&gt;, &lt;NewName&gt;</code>     |
| <b>VB Example</b> | <pre>oModule.RenameFieldPlot _<br/>"Vector_E1", "Vector_E2"</pre> |

# RenamePlotFolder

Renames a plot folder.

|              |                                                                                                   |        |                              |
|--------------|---------------------------------------------------------------------------------------------------|--------|------------------------------|
| UI Access    | Right-click a plot folder in the project tree, and then click <b>Rename</b> on the shortcut menu. |        |                              |
| Parameters   | Name                                                                                              | Type   | Description                  |
|              | <OldName>                                                                                         | String | Original name of the folder. |
|              | <NewName>                                                                                         | String | New name of the folder.      |
| Return Value | None.                                                                                             |        |                              |

|                |                                                                          |  |  |
|----------------|--------------------------------------------------------------------------|--|--|
| Python Syntax  | RenamePlotFolder(<OldName>, <NewName>)                                   |  |  |
| Python Example | <pre>oModule.RenamePlotFolder(<br/>    "E Field", "Surface Plots")</pre> |  |  |

|            |                                                                          |  |  |
|------------|--------------------------------------------------------------------------|--|--|
| VB Syntax  | RenamePlotFolder <OldName>, <NewName>                                    |  |  |
| VB Example | <pre>oModule.RenamePlotFolder _<br/>    "E Field", "Surface Plots"</pre> |  |  |

# SaveFieldsPlots

Saves field plot(s) to a file.



|                     |                                         |        |                                                       |
|---------------------|-----------------------------------------|--------|-------------------------------------------------------|
| <b>UI Access</b>    | <b>HFSS &gt; Fields &gt; Save As...</b> |        |                                                       |
| <b>Parameters</b>   | Name                                    | Type   | Description                                           |
|                     | <PlotNames>                             | Array  | Array of plot names.                                  |
|                     | <FileName>                              | String | Name of the file to save as, including the file path. |
| <b>Return Value</b> | None.                                   |        |                                                       |

|                       |                                                                                     |  |  |
|-----------------------|-------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | SaveFieldsPlots(<PlotNames>, <FileName>)                                            |  |  |
| <b>Python Example</b> | <pre>oModule.SaveFieldsPlots(["Mag_E1", "Mag_E2"],<br/>"C:/field_report.dsp")</pre> |  |  |

|                   |                                                                                           |  |  |
|-------------------|-------------------------------------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | SaveFieldsPlots <PlotNames>, <FileName>                                                   |  |  |
| <b>VB Example</b> | <pre>oModule.SaveFieldsPlots Array("Mag_E1", "Mag_E2"), _<br/>"C:/field_report.dsp"</pre> |  |  |

## SetFieldPlotSettings

Sets plot attributes.

|                   |                                                          |        |                                                               |
|-------------------|----------------------------------------------------------|--------|---------------------------------------------------------------|
| <b>UI Access</b>  | <b>[product] &gt; Fields &gt; Modify Plot Attributes</b> |        |                                                               |
| <b>Parameters</b> | Name                                                     | Type   | Description                                                   |
|                   | <PlotName>                                               | String | Name of the plot to modify.                                   |
|                   | <PlotItemAttributes>                                     | Array  | Structured array.<br><br>Array("NAME:FieldsPlotItemSettings", |

|              |                                                                                                                                                                                                                                                                  |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <div><div></div><div></div><div><div>&lt;PlotOnPointsSettings&gt;,<br/>&lt;PlotOnLineSettings&gt;,<br/>&lt;PlotOnSurfaceSettings&gt;,<br/>&lt;PlotOnVolumeSettings&gt;<br/><br/><i>See description of CreateFieldPlot command for details.</i></div></div></div> |
| Return Value | None.                                                                                                                                                                                                                                                            |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | SetFieldPlotSettings(<PlotName> <PlotItemAttributes>)                                                                                                                                                                                                                                                                                                                                                                                                       |
| Python Example | <pre>oModule.SetFieldPlotSettings("Mag_E2",<br/>    ["NAME:FieldsPlotItemSettings",<br/>        ["NAME:PlotOnLineSettings",<br/>            ["NAME:LineSettingsID",<br/>                "Width:=", 4,<br/>                "Style:=", "Cylinder"),<br/>            "IsoValType:=", "Tone",<br/>            "ArrowUniform:=", True,<br/>            "NumofArrow:=", 100),<br/>        ["NAME:PlotOnSurfaceSettings",<br/>            "Filled:=", False,</pre> |

|  |                                                                                                                                                                                                                                                              |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"IsoValType:=", "Tone",<br/>"SmoothShade:=", True,<br/>"AddGrid:=", False,<br/>"MapTransparency:=", True,<br/>"Transparency:=", 0,<br/>"ArrowUniform:=", True,<br/>"ArrowSpacing:=", 0.100000001490116,<br/>"GridColor:=", [255, 255, 255]]<br/>)</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | SetFieldPlotSettings <PlotName> <PlotItemAttributes>                                                                                                                                                                                                                                                                                                                                                                                                             |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.SetFieldPlotSettings "Mag_E2", _<br/>    Array("NAME:FieldsPlotItemSettings", _<br/>        Array("NAME:PlotOnLineSettings", _<br/>            Array("NAME:LineSettingsID", _<br/>                "Width:=", 4, _<br/>                "Style:=", "Cylinder"), _<br/>            "IsoValType:=", "Tone", _<br/>            "ArrowUniform:=", true, _<br/>            "NumofArrow:=", 100), _<br/>        Array("NAME:PlotOnSurfaceSettings", _</pre> |

|  |                                                                                                                                                                                                                                                                                                     |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Filled:=", false, _<br/>"IsoValType:=", "Tone", _<br/>"SmoothShade:=", true, _<br/>"AddGrid:=", false, _<br/>"MapTransparency:=", true, _<br/>"Transparency:=", 0, _<br/>"ArrowUniform:=", true, _<br/>"ArrowSpacing:=", 0.100000001490116, _<br/>"GridColor:=", Array(255, 255, 255)))</pre> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

SaveNamedExpressions

Saves a named expression definition to a file.

|              |                                                                        |         |                                                                                        |
|--------------|------------------------------------------------------------------------|---------|----------------------------------------------------------------------------------------|
| UI Access    | In the Fields Calculator, click <b>Save To...</b> in the Library area. |         |                                                                                        |
| Parameters   | Name                                                                   | Type    | Description                                                                            |
|              | <FileName>                                                             | String  | Filename and full path to the file to hold the named expression definition.            |
|              | <NamedExprs>                                                           | Array   | Array of strings containing the names of expression definitions to load from the file. |
|              | <Overwrite>                                                            | Boolean | Specifies whether to overwrite the file.                                               |
| Return Value | None.                                                                  |         |                                                                                        |

|                       |                                                                                                    |
|-----------------------|----------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>SaveNamedExpressions(&lt;FileName&gt;, &lt;NamedExprs&gt;, &lt;Overwrite&gt;)</code>         |
| <b>Python Example</b> | <pre>oModule.SaveNamedExpressions( "C:\Ansoft\PersonalLib\smth.clc", ["smoothedtemp"], True)</pre> |

|                   |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>SaveNamedExpressions &lt;FileName&gt;, &lt;NamedExprs&gt;, &lt;Overwrite&gt;</code>                 |
| <b>VB Example</b> | <pre>oModule.SaveNamedExpressions _ "C:\Ansoft\PersonalLib\smth.clc", _ Array("smoothedtemp"), true</pre> |

## SetFieldPlotSettings

Sets plot attributes.

|                   |                                                           |        |                                                                                                                                                                                                |
|-------------------|-----------------------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | [product] > <b>Fields</b> > <b>Modify Plot Attributes</b> |        |                                                                                                                                                                                                |
| <b>Parameters</b> | Name                                                      | Type   | Description                                                                                                                                                                                    |
|                   | <PlotName>                                                | String | Name of the plot to modify.                                                                                                                                                                    |
|                   | <PlotItemAttributes>                                      | Array  | Structured array.                                                                                                                                                                              |
|                   |                                                           |        | <pre>Array("NAME:FieldsPlotItemSettings",       &lt;PlotOnPointsSettings&gt;,       &lt;PlotOnLineSettings&gt;,       &lt;PlotOnSurfaceSettings&gt;,       &lt;PlotOnVolumeSettings&gt;)</pre> |

|              |                                                                                            |
|--------------|--------------------------------------------------------------------------------------------|
|              | <div></div> <div></div> <div>See description of CreateFieldPlot command for details.</div> |
| Return Value | None.                                                                                      |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | SetFieldPlotSettings(<PlotName> <PlotItemAttributes>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
| Python Example | <pre>oModule.SetFieldPlotSettings("Mag_E2",     ["NAME:FieldsPlotItemSettings",         ["NAME:PlotOnLineSettings",             ["NAME:LineSettingsID",                 "Width:=", 4,                 "Style:=", "Cylinder"),             "IsoValType:=", "Tone",             "ArrowUniform:=", True,             "NumofArrow:=", 100),         ["NAME:PlotOnSurfaceSettings",             "Filled:=", False,             "IsoValType:=", "Tone",             "SmoothShade:=", True,             "AddGrid:=", False,             "MapTransparency:=", True,</pre> |

|  |                                                                                                                                  |
|--|----------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> "Transparency:=", 0, "ArrowUniform:=", True, "ArrowSpacing:=", 0.100000001490116, "GridColor:=", [255, 255, 255]] ) </pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | SetFieldPlotSettings <PlotName> <PlotItemAttributes>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> oModule.SetFieldPlotSettings "Mag_E2", _     Array("NAME:FieldsPlotItemSettings", _         Array("NAME:PlotOnLineSettings", _             Array("NAME:LineSettingsID", _                 "Width:=", 4, _                 "Style:=", "Cylinder"), _             "IsoValType:=", "Tone", _             "ArrowUniform:=", true, _             "NumofArrow:=", 100), _         Array("NAME:PlotOnSurfaceSettings", _             "Filled:=", false, _             "IsoValType:=", "Tone", _             "SmoothShade:=", true, _             "AddGrid:=", false, _ </pre> |

|  |                                                                                                                                                                                      |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"MapTransparency:=", true, _<br/>"Transparency:=", 0, _<br/>"ArrowUniform:=", true, _<br/>"ArrowSpacing:=", 0.100000001490116, _<br/>"GridColor:=", Array(255, 255, 255))</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

SetPlotFolderSettings

Sets the attributes of all plots in the specified folder.

| UI Access  | [product] > Fields > Modify Plot Attributes |        |                                                                                                                                                                                    |
|------------|---------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                        | Type   | Description                                                                                                                                                                        |
|            | <PlotFolderName>                            | String | Name of the folder with the attributes to modify.                                                                                                                                  |
|            | <PlotFolderAttributes>                      | Array  | Structured array.<br><br>Array("NAME:FieldsPlotSettings",<br>"Real time mode:=", <bool>,<br><ColorMapSettings>,<br><Scale3DSettings>,<br><Marker3DSettings>,<br><Arrow3DSettings>) |
|            | <ColorMapSettings>                          | Array  | Structured array.<br><br>Array("NAME:ColorMapSettings",<br>"ColorMapType:=", <string Possible values are                                                                           |



|  |                    |       |                                                                                                                                                                                                                                                                                                                                                                                                               |
|--|--------------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  |                    |       | <pre> "Uniform", "Ramp", "Spectrum"&gt;,  "SpectrumType:=", &lt;string Possible values are "Rainbow", "Temperature", "Magenta", "Gray"&gt;,  "UniformColor:=", &lt;array containing the R, G, B components of the color. Components should be in the range 0 to 255.&gt;,  "RampColor:=", &lt;array containing the R, G, B com- ponents of the color. Components should be in the range 0 to 255.&gt;) </pre> |
|  | <Scale3DSettings>  | Array | <p>Structured array.</p> <pre> Array("NAME:Scale3DSettings",    "m_nLevels:=", &lt;int&gt;,    "m_autoScale:=", &lt;bool&gt;,    "minvalue:=", &lt;double&gt;,    "maxvalue:=", &lt;double&gt;,    "log:=", &lt;bool&gt;,    "IntrinsicMin:=", &lt;double&gt;,    "IntrinsicMax:=", &lt;double&gt;) </pre>                                                                                                    |
|  | <Marker3DSettings> | Array | <p>Structured array.</p> <pre> Array("NAME:Marker3DSettings",    "MarkerType:=", &lt;integer  9: Sphere  10: Box  11: Tetrahedron </pre>                                                                                                                                                                                                                                                                      |

|              |                   |       |                                                                                                                                                                                                                                                                                                                                                                                    |
|--------------|-------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              |                   |       | <div>12: Octahedron</div> <div>default: Sphere&gt;</div> <div>"MarkerMapSize:=", &lt;bool&gt;</div> <div>"MarkerMapColor:=", &lt;bool&gt;</div> <div>"MarkerSize:=", &lt;double&gt;</div>                                                                                                                                                                                          |
|              | <Arrow3DSettings> | Array | <div>Structured array.</div> <div>Array("NAME:Arrow3DSettings",</div> <div>"ArrowType:=", &lt;integer</div> <div>0: Line</div> <div>1: Cylinder</div> <div>2: Umbrella</div> <div>default: Line&gt;</div> <div>"ArrowMapSize:=", &lt;bool&gt;</div> <div>"ArrowMapColor:=", &lt;bool&gt;</div> <div>"ShowArrowTail:=", &lt;bool&gt;</div> <div>"ArrowSize:=", &lt;double&gt;</div> |
| Return Value | None.             |       |                                                                                                                                                                                                                                                                                                                                                                                    |

|               |                                                                 |
|---------------|-----------------------------------------------------------------|
| Python Syntax | SetPlotFolderSettings(<PlotFolderName>, <PlotFolderAttributes>) |
|---------------|-----------------------------------------------------------------|

**Python Example**

```
oModule.SetPlotFolderSettings("E Field1",
 ["NAME:FieldsPlotSettings",
 "Real time mode:=", True,
 ["NAME:ColorMapSettings",
 "ColorMapType:=", "Spectrum",
 "SpectrumType:=", "Rainbow",
 "UniformColor:=", [127, 255, 255],
 "RampColor:=", [255, 127, 127]],
 ["NAME:Scale3DSettings",
 "m_nLevels:=", 27,
 "m_autoScale:=", True,
 "minvalue:=", 9.34379863739014,
 "maxvalue:=", 13683.755859375,
 "log:=", False,
 "IntrinsicMin:=", 9.34379863739014,
 "IntrinsicMax:=", 13683.755859375),
 ["NAME:Marker3DSettings",
 "MarkerType:=", 0,
 "MarkerMapSize:=", True,
 "MarkerMapColor:=", False,
 "MarkerSize:=", 0.25],
```

|  |                                                                                                                                                                                            |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>[ "NAME:Arrow3DSettings",<br/>  "ArrowType:=", 1,<br/>  "ArrowMapSize:=", True,<br/>  "ArrowMapColor:=", True,<br/>  "ShowArrowTail:=", True,<br/>  "ArrowSize:=", 0.25]]<br/>)</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | SetPlotFolderSettings <PlotFolderName>, <PlotFolderAttributes>                                                                                                                                                                                                                                                                                                                                                                            |
|------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.SetPlotFolderSettings "E Field1", _<br/>  Array("NAME:FieldsPlotSettings", _<br/>    "Real time mode:=", true, _<br/>    Array("NAME:ColorMapSettings", _<br/>      "ColorMapType:=", "Spectrum", _<br/>      "SpectrumType:=", "Rainbow", _<br/>      "UniformColor:=", Array(127, 255, 255), _<br/>      "RampColor:=", Array(255, 127, 127))), _<br/>  Array("NAME:Scale3DSettings", _<br/>    "m_nLevels:=", 27, _</pre> |

```

"m_autoScale=", true, _
"minvalue=", 9.34379863739014, _
"maxvalue=", 13683.755859375, _
"log=", false, _
"IntrinsicMin=", 9.34379863739014, _
"IntrinsicMax=", 13683.755859375), _
Array("NAME:Marker3DSettings", _
"MarkerType=", 0, _
"MarkerMapSize=", true, _
"MarkerMapColor=", false, _
"MarkerSize=", 0.25), _
Array("NAME:Arrow3DSettings", _
"ArrowType=", 1, _
"ArrowMapSize=", true, _
"ArrowMapColor=", true, _
"ShowArrowTail=", true, _
"ArrowSize=", 0.25))

```

## SetPlotsViewSolutionContext

Sets the context of named field overlay plots to the specified time using the intrinsic Time value of the active window. Also updates the associated view solution context for each plot.

|                     |                                                                 |        |                                                     |
|---------------------|-----------------------------------------------------------------|--------|-----------------------------------------------------|
| <b>UI Access</b>    | [product] > <b>Fields</b> > <b>Set Context to Active Window</b> |        |                                                     |
| <b>Parameters</b>   | Name                                                            | Type   | Description                                         |
|                     | <Plots>                                                         | Array  | Names of the plots to modify.                       |
|                     | <SolName>                                                       | String | Name of a transient analysis solution.              |
|                     | <SolTime>                                                       | String | Intrinsic variable name (must be "Time") and value. |
| <b>Return Value</b> | None.                                                           |        |                                                     |

|                       |                                                                                                                     |
|-----------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SetPlotsViewSolutionContext(<Plots>, <SolName>, <SolTime>)                                                          |
| <b>Python Example</b> | <pre>oModule.SetPlotsViewSolutionContext(<br/>["Flux_Lines1", "Mag_H1"],<br/>"Setup1 : Transient", "Time=-1")</pre> |

|                   |                                                                                                                            |
|-------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetPlotsViewSolutionContext <Plots>, <SolName>, <SolTime>                                                                  |
| <b>VB Example</b> | <pre>oModule.SetPlotsViewSolutionContext _<br/>Array("Flux_Lines1", "Mag_H1"), _<br/>"Setup1 : Transient", "Time=-1"</pre> |

## UpdateAllFieldsPlots

Updates the All Fields Plots.

|                  |     |
|------------------|-----|
| <b>UI Access</b> | N/A |
|------------------|-----|

|                     |       |
|---------------------|-------|
| <b>Parameters</b>   | None. |
| <b>Return Value</b> | None. |

|                       |                                             |
|-----------------------|---------------------------------------------|
| <b>Python Syntax</b>  | UpdateAllFieldsPlots()                      |
| <b>Python Example</b> | <code>oModule.UpdateAllFieldsPlots()</code> |

|                   |                                           |
|-------------------|-------------------------------------------|
| <b>VB Syntax</b>  | UpdateAllFieldsPlots                      |
| <b>VB Example</b> | <code>oModule.UpdateAllFieldsPlots</code> |

## UpdateQuantityFieldsPlots

Updates the Quantity Fields Plots.

|                     |              |             |                                           |
|---------------------|--------------|-------------|-------------------------------------------|
| <b>UI Access</b>    | N/A          |             |                                           |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>                        |
|                     | <FolderName> | String      | Folder containing the plotted quantities. |
| <b>Return Value</b> | None.        |             |                                           |

|                       |                                                          |
|-----------------------|----------------------------------------------------------|
| <b>Python Syntax</b>  | UpdateQuantityFieldsPlots(<FolderName>)                  |
| <b>Python Example</b> | <code>oModule.UpdateQuantityFieldsPlots("fldplt")</code> |

|                   |                                                 |
|-------------------|-------------------------------------------------|
| <b>VB Syntax</b>  | UpdateQuantityFieldsPlots < <i>FolderName</i> > |
| <b>VB Example</b> | oModule.UpdateQuantityFieldsPlots "fldplt"      |



## 19 - Fields Calculator Script Commands

Fields Calculator commands should be executed by the Field Overlays module, which is called "FieldsReporter" in Maxwell scripts.

```
Set oModule = oDesign.GetModule("FieldsReporter")
```

```
oModule.CommandName <args>
```

The command associated with each of the following scripting commands will be a button pressed in the Fields Calculator.

[AddNamedExpression](#)

[CalcOp](#)

[CalculatorRead](#)

[CalcStack](#)

[CalculatorWrite](#)

[ChangeGeomSettings](#)

[ClcEval](#)

[ClcMaterial](#)

[ClearAllNamedExpr](#)

[CopyNamedExprToStack](#)

[DeleteNamedExpr](#)

[EnterComplex](#)

[EnterComplexVector](#)

[EnterLine](#)

[EnterPoint](#)

[EnterQty](#)

[EnterScalar](#)

[EnterScalarFunc](#)

[EnterSurf](#)

[EnterVector](#)

[EnterVectorFunc](#)

[EnterVol](#)

[ExportOnGrid \[Fields Calculator\]](#)

[ExportToFile \[Fields Calculator\]](#)

[GetTopEntryValue](#)

[LoadNamedExpressions](#)

[SaveNamedExpressions](#)

# AddNamedExpression

Creates a named expression using the expression at the top of the stack.

| UI Access  | Click <b>Add</b> in the <b>Fields Calculator</b> window. |        |                                                                                                                                                                                          |
|------------|----------------------------------------------------------|--------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                     | Type   | Description                                                                                                                                                                              |
|            | <ExprName>                                               | String | Name for the new named expression.                                                                                                                                                       |
|            | <FieldType>                                              | String | Type of field.                                                                                                                                                                           |
|            | <Expression>                                             | Array  | Defines an expression to be solved in the calculator. It can be an Algebraic or Stacked expression; the Algebraic option requires a beta feature; see <a href="#">Algebraic option</a> . |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                                           |
|-----------------------|-----------------------------------------------------------|
| <b>Python Syntax</b>  | AddNamedExpression(<ExprName>, <FieldType>, <Expression>) |
| <b>Python Example</b> | oModule.AddNamedExpression("Mag_JxE", "Fields")           |

|                   |                                                          |
|-------------------|----------------------------------------------------------|
| <b>VB Syntax</b>  | AddNamedExpression <ExprName>, <FieldType>, <Expression> |
| <b>VB Example</b> | oModule.AddNamedExpression "Mag_JxE", "Fields"           |

Maxwell users can create an expression that use the [Algebraic option](#). In this case, no Stack commands are necessary; the user can define the expression within AddNamedExpression command:

```
oModule.AddNamedExpression("new2", "Fields",
[
 "Algebraic:=" , "Mean(VolumeValue(Box1), Mag_B)"
])
```

The same command in Stack form would look like this:

```
oModule.AddNamedExpression("new2", "Fields",
[
 "NameOfExpression:=" , ["Mag_B"],
 "EnterVolume:=" , ["Box1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
```

```
])
```

For information, see [Syntax for Algebraic Option](#).

## Syntax for Algebraic Option

When [adding a named expression](#), Maxwell users have the option of defining it using algebraic expressions. The syntax for an algebraic expression is listed below.

```
expression : namedExpression
 | constant
 | name
 | operator
 | expression binaryOperator expression
 | '(' expression ')'

operator : opName arguments

binaryOperator : + | - | * | /

arguments : <empty>
 | '(' expression ')'
 | '(' expression ',' expression ')'
```

A `namedExpression` is any name already defined in the calculator. The full list is in the upper left group box of the Fields Calculator window.

A `name` can be a design variable or a geometry or an intrinsic variable.

A `constant` can be a

- scalar literal; for example, “1.5e-10”
- complex number; for example, “<(3.5, -1.0)>”
- 3D vector; for example, “<(1.0, 2.0, 3.0)>”
- complex 3D vector; for example, “<(1.0, -2.0), (2.0, -3.0) ,(3.0, -4.0)>”

An `opName` is any of the functions in the General, Scalar, or Vector columns of the Fields Calculator window. Operators may have 0, 1, or 2 arguments. If arguments are present, they appear in parentheses.

`binaryOperator` is one of the four arithmetic operators. Parentheses are provided to allow the user to group as needed.

## CalcOp

Performs a calculator operation.

|              |                                                      |        |                                                  |
|--------------|------------------------------------------------------|--------|--------------------------------------------------|
| UI Access    | Operation commands like <b>Mag</b> , <b>+</b> , etc. |        |                                                  |
| Parameters   | Name                                                 | Type   | Description                                      |
|              | <OpString>                                           | String | The text on the corresponding calculator button. |
| Return Value | None.                                                |        |                                                  |

|                |                                   |
|----------------|-----------------------------------|
| Python Syntax  | CalcOp(<OpString>)                |
| Python Example | <code>oModule.CalcOp ("+")</code> |

|            |                                 |
|------------|---------------------------------|
| VB Syntax  | CalcOp <OpString>               |
| VB Example | <code>oModule.CalcOp "+"</code> |

# CalcRead(deprecated)

Reads a file that is written out by the CalcWrite command, and puts the result into a calculator numeric.

|              |                                                              |        |                                |
|--------------|--------------------------------------------------------------|--------|--------------------------------|
| UI Access    | Click <b>Read...</b> in the <b>Fields Calculator</b> dialog. |        |                                |
| Parameters   | Name                                                         | Type   | Description                    |
|              | <FileName>                                                   | String | Name of the file to be read.   |
|              | <SoluName>                                                   | String | Specified solution to read in. |
|              | <VarArray>                                                   | Array  | Array of variables to read.    |
| Return Value | None.                                                        |        |                                |

|                |                                                                                                                                        |  |  |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------|--|--|
| Python Syntax  | CalcRead(<FileName>, <SoluName>, <VarArray>)                                                                                           |  |  |
| Python Example | <pre>oModule.CalcRead(<br/>    "c:\example.reg",<br/>    "Setup1: LastAdaptive",<br/>    ["Freq:=", "10GHz", "Phase:=", "0deg"])</pre> |  |  |

|            |                                                                                             |  |  |
|------------|---------------------------------------------------------------------------------------------|--|--|
| VB Syntax  | CalcRead <FileName>, <SoluName>, <VarArray>                                                 |  |  |
| VB Example | <pre>oModule.CalcRead _<br/>    "c:\example.reg", _<br/>    "Setup1: LastAdaptive", _</pre> |  |  |

```
Array("Freq:=", "10GHz", "Phase:=", "0deg")
```

## CalculatorRead

Gets a register file and applies it to the calculator stack.

| UI Access    | Click <b>Read...</b> in the <b>Fields Calculator</b> dialog. |        |                                                    |
|--------------|--------------------------------------------------------------|--------|----------------------------------------------------|
| Parameters   | Name                                                         | Type   | Description                                        |
|              | <FileName>                                                   | String | Path to and including name of input register file. |
|              | <SoluName>                                                   | String | Specified solution to read in.                     |
|              | <FieldType>                                                  | String | Type of specified field.                           |
|              | <VarArray>                                                   | Array  | Array of variable names, value pairs.              |
| Return Value | None.                                                        |        |                                                    |

| Python Syntax  | CalculatorRead(<FileName>, <SoluName>, <FieldType>, <VarArray>)                                                                            |  |  |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| Python Example | <pre>oModule.CalculatorRead(     "c:\example.reg",     "Setup1: LastAdaptive", "Fields",     ["Freq:=", "10GHz", "Phase:=", "0deg"])</pre> |  |  |

| VB Syntax  | CalculatorRead <FileName>, <SoluName>, <FieldType>, <VarArray> |  |  |
|------------|----------------------------------------------------------------|--|--|
| VB Example | <pre>oModule.CalculatorRead _     "c:\example.reg", _</pre>    |  |  |

|  |                                                                                              |
|--|----------------------------------------------------------------------------------------------|
|  | <pre>"Setup1: LastAdaptive", "Fields", _<br/>Array("Freq=", "10GHz", "Phase=", "0deg")</pre> |
|--|----------------------------------------------------------------------------------------------|

## CalcStack

Performs an operation on the stack.

|                     |                                                              |        |                                                  |
|---------------------|--------------------------------------------------------------|--------|--------------------------------------------------|
| <b>UI Access</b>    | Stack operation buttons such as <b>Push</b> and <b>Pop</b> . |        |                                                  |
| <b>Parameters</b>   | Name                                                         | Type   | Description                                      |
|                     | <OpString>                                                   | String | The text on the corresponding calculator button. |
| <b>Return Value</b> | None.                                                        |        |                                                  |

|                       |                                        |
|-----------------------|----------------------------------------|
| <b>Python Syntax</b>  | CalcStack(<OpString>)                  |
| <b>Python Example</b> | <code>oModule.CalcStack("push")</code> |

|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | CalcStack <OpString>                  |
| <b>VB Example</b> | <code>oModule.CalcStack "push"</code> |

## CalculatorWrite

Writes contents of top register to file.

|                  |                                                               |
|------------------|---------------------------------------------------------------|
| <b>UI Access</b> | Click <b>Write...</b> in the <b>Fields Calculator</b> window. |
|------------------|---------------------------------------------------------------|



| Parameters   | Name             | Type   | Description                                                                                                                                                                                                                                                                                                       |
|--------------|------------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <OutputFilePath> | String | Path to and including name of output register file.                                                                                                                                                                                                                                                               |
|              | <SoluNameArray>  | Array  | Array of strings containing solution names.                                                                                                                                                                                                                                                                       |
|              | <Expression>     | Array  | Defines an expression to be solved in the calculator. It can be an algebraic or stacked(default) expression; the <code>Algebraic</code> option requires a beta feature; see <a href="#">Algebraic option</a> . Note that for legacy scripts, this parameter is optional because the expression may already exist. |
|              | <VarArray>       | Array  | Array of variable names, value pairs.                                                                                                                                                                                                                                                                             |
| Return Value | None             |        |                                                                                                                                                                                                                                                                                                                   |

| Python Syntax  | CalculatorWrite(<OutputFilePath>, <SoluNameArray>, <Expression>,<VarArray>)                                                                                                                                                                                                                                                                     |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <p>Legacy script with no expression:</p> <pre>oModule.CalculatorWrite("C:\test.reg",     ["Solution:=", "Setup1 : LastAdaptive"],     ["Freq:=", "1GHz", "Phase:=", "0deg"] )</pre> <p>Example with algebraic expression:</p> <pre>oModule.CalculatorWrite("C:\test.reg",     [         "NAME:Write",         [             "NAME:Setup",</pre> |

```
 "Solution:=" , "setup1 : LastAdaptive"
],
 [
 "NAME:Algebraic",
 "Name:=" , [""],
 "Algebraic:=" , "Mean(Volume(pri), Mag_H)"
]
],
[
 "Freq:=" , "60Hz",
 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
])
```

**Example with stack expression:**

```
oModule.CalculatorWrite("C:\test.reg",
 [
 "NAME:Write",
]
```

```

 "NAME:Setup",
 "Solution:=" , "setup1 : LastAdaptive"
],
[
 "NAME:Expression",
 "Name:=" , [""],
 "Expression:=" , ["Mean(Volume(sec_1_1), Mag_H)"],
 "NameOfExpression:=" , ["Mag_H"],
 "EnterVolume:=" , ["sec_1_1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
],
[
 "NAME:Context",
 "Freq:=" , "60Hz",
 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
]
], "Fields")

```

| VB Syntax  | CalculatorWrite <OutputFilePath>, <SoluNameArray>, <Expression>, <VarArray>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <p>Legacy script with no expression:</p> <pre>oModule.CalculatorWrite "C:\test.reg", _<br/>    Array("Solution:=", "Setup1 : LastAdaptive"),_<br/>    Array("Freq:=", "1GHz", "Phase:=", "0deg")</pre> <p>Example with stack expression:</p> <pre>oModule.CalculatorWrite "C:\test.reg",_<br/>    Array("Solution:=", "setup1 : LastAdaptive")_<br/>    Array("Expression:=", Array("Mean(Volume(sec_1_1), Mag_H)"), "NameOfExpression:="_<br/>    Array("Mag_H"), "EnterVolume:=", Array("sec_1_1"), "Operation:=", Array<br/>    ("VolumeValue"),_<br/>    "Operation:=", Array("Mean"))_<br/>    Array(<br/>        "Freq:="                , "60Hz",<br/>        "Ihv:="                 , " (833.3*1.414*232) A",<br/>        "Ilv:="                 , "-Ihv/1404",<br/>        "Phase:="               , "0deg",<br/>        "height:="              , "1mm"<br/>    ), "Fields")</pre> |

## ChangeGeomSettings

Changes the line discretization setting.

|                     |                                                                              |         |                            |
|---------------------|------------------------------------------------------------------------------|---------|----------------------------|
| <b>UI Access</b>    | In the <b>Fields Calculator</b> dialog box, click on <b>Geom Settings...</b> |         |                            |
| <b>Parameters</b>   | Name                                                                         | Type    | Description                |
|                     | <LineDiscr>                                                                  | Integer | Line discretization value. |
| <b>Return Value</b> | None.                                                                        |         |                            |

|                       |                                 |
|-----------------------|---------------------------------|
| <b>Python Syntax</b>  | ChangeGeomSettings(<LineDiscr>) |
| <b>Python Example</b> | oModule.ChangeGeomSettings(500) |

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Syntax</b>  | ChangeGeomSettings <LineDiscr> |
| <b>VB Example</b> | oModule.ChangeGeomSettings 500 |

## ClcEval

Evaluates the expression at the top of the stack using the provided solution name and variable values.

|                   |                                                           |        |                                                                                                                                                                          |
|-------------------|-----------------------------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Click <b>Eval</b> in the <b>Fields Calculator</b> dialog. |        |                                                                                                                                                                          |
| <b>Parameters</b> | Name                                                      | Type   | Description                                                                                                                                                              |
|                   | <SoluName>                                                | String | Name of specified solution.                                                                                                                                              |
|                   | <VarArray>                                                | Array  | Array of variable name, value pairs.                                                                                                                                     |
|                   | <Expression>                                              | Array  | Defines an expression to be solved in the calculator. It can be an algebraic or stacked (default) expression; the <code>Algebraic</code> option requires a beta feature; |

|              |             |        |                                                                                                                                                                    |
|--------------|-------------|--------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              |             |        | see <a href="#">Algebraic option</a> . Note that for legacy scripts, this parameter is optional because the expression may have been created with another command. |
|              | <FieldType> | String | Optional. Type of specified field.                                                                                                                                 |
| Return Value | None        |        |                                                                                                                                                                    |

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ClcEval(<SoluName>, <Expression>, <VarArray>, [Optional <FieldType>])                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| Python Example | <p>Legacy script with no expression:</p> <pre>oModule.ClcEval("Setup1: LastAdaptive",<br/>    ["Freq:=", "10GHz",<br/>    "Phase:=", "0deg"])</pre> <p>Example with algebraic expression:</p> <pre>oModule.ClcEval("setup1 : LastAdaptive",<br/>    [<br/>        "NAME:Evaluate",<br/>        [<br/>            "NAME:Algebraic",<br/>            "Name:="                                , [""] ,<br/>            "Algebraic:="                            , "Mean( Volume(sec_1_1), Mag_H )" ,<br/>        ],<br/>    ],<br/>    [<br/>        ]</pre> |

```

 "NAME:Context",
 "Freq:=" , "60Hz",
 "Ihv:=" , "(833.3*1.414*232) A",
 "Ilv:=" , "-Ihv/1404",
 "Phase:=" , "0deg",
 "height:=" , "1mm"
]
], "Fields")

```

Example with stack expression:

```

oModule.ClcEval("setup1 : LastAdaptive",
[
 "NAME:Evaluate",
 [
 "NAME:Expression",
 "Name:=" , [""],
 "Expression:=" , ["Mean(Volume(sec_1_1), Mag_H)"],
 "NameOfExpression:=" , ["Mag_H"],
 "EnterVolume:=" , ["sec_1_1"],
 "Operation:=" , ["VolumeValue"],
 "Operation:=" , ["Mean"]
],
],

```

|  |                                                                                                                                                                                                                                                                     |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>[     "NAME:Context",     "Freq:="                , "60Hz",     "Ihv:="                  , "(833.3*1.414*232) A",     "Ilv:="                  , "-Ihv/1404",     "Phase:="                , "0deg",     "height:="               , "1mm" ] ], "Fields")</pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | ClcEval <SoluName>, <Expression>, <VarArray>, [Optional <FieldType>]                                                                                                                                                                                                                                                                                                                                                   |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <p>Legacy script with no expression:</p> <pre>oModule.ClcEval "Setup1: LastAdaptive", _     Array ("Freq:=", "10GHz", _         "Phase:=", "0deg")</pre> <p>Example with stack expression:</p> <pre>oModule.ClcEval "Setup1 : LastAdaptive",_     Array("Expression:=", Array("Mean(Volume(sec_1_1), Mag_H)"), "NameOfExpression:="_     Array("Mag_H"), "EnterVolume:=", Array("sec_1_1"), "Operation:=", Array</pre> |



```
("VolumeValue"),_
"Operation:=", Array("Mean"))_
Array(
 "Freq:=", , "60Hz",
 "Ihv:=", , "(833.3*1.414*232) A",
 "Ilv:=", , "-Ihv/1404",
 "Phase:=", , "0deg",
 "height:=", , "1mm"
), "Fields")
```

## ClcMaterial

Performs a material operation on the top stack element.

| UI Access    | Click <b>Matl...</b> in the <b>Fields Calculator</b> dialog. |        |                                                          |
|--------------|--------------------------------------------------------------|--------|----------------------------------------------------------|
| Parameters   | Name                                                         | Type   | Description                                              |
|              | <MatString>                                                  | String | The material property to apply.                          |
|              | <OpString>                                                   | String | Name of operation. Possible values are "mult", or "div". |
| Return Value | None.                                                        |        |                                                          |

| Python Syntax  | ClcMaterial(<MatString>, <OpString>)            |
|----------------|-------------------------------------------------|
| Python Example | oModule.ClcMaterial("Permeability (mu)" "mult") |

|                   |                                                |
|-------------------|------------------------------------------------|
| <b>VB Syntax</b>  | ClcMaterial <MatString>, <OpString>            |
| <b>VB Example</b> | oModule.ClcMaterial "Permeability (mu)" "mult" |

## ClearAllNamedExpr

Clears all user-defined named expressions from the list.

|                     |                                                               |
|---------------------|---------------------------------------------------------------|
| <b>UI Access</b>    | Click <b>ClearAll</b> in the <b>Fields Calculator</b> dialog. |
| <b>Parameters</b>   | None.                                                         |
| <b>Return Value</b> | None.                                                         |

|                       |                             |
|-----------------------|-----------------------------|
| <b>Python Syntax</b>  | ClearAllNamedExpr()         |
| <b>Python Example</b> | oModule.ClearAllNamedExpr() |

|                   |                           |
|-------------------|---------------------------|
| <b>VB Syntax</b>  | ClearAllNamedExpr         |
| <b>VB Example</b> | oModule.ClearAllNamedExpr |

## CopyNamedExprToStack

Copies the named expression selected to the calculator stack.

|                  |                                                                 |
|------------------|-----------------------------------------------------------------|
| <b>UI Access</b> | Select a named expression and then click <b>Copy to stack</b> . |
|------------------|-----------------------------------------------------------------|

|              |            |        |                                                              |
|--------------|------------|--------|--------------------------------------------------------------|
| Parameters   | Name       | Type   | Description                                                  |
|              | <ExprName> | String | Name of the expression to be copied to the top of the stack. |
| Return Value | None.      |        |                                                              |

|                |                                                      |
|----------------|------------------------------------------------------|
| Python Syntax  | CopyNamedExprToStack(<ExprName>)                     |
| Python Example | <code>oModule.CopyNamedExprToStack("Mag_JxE")</code> |

|            |                                                     |
|------------|-----------------------------------------------------|
| VB Syntax  | CopyNamedExprToStack <ExprName>                     |
| VB Example | <code>oModule.CopyNamedExprToStack "Mag_JxE"</code> |

## DeleteNamedExpr

Deletes the selected named expression from the list.

|              |                                                          |        |                                     |
|--------------|----------------------------------------------------------|--------|-------------------------------------|
| UI Access    | Select a named expression and then click <b>Delete</b> . |        |                                     |
| Parameters   | Name                                                     | Type   | Description                         |
|              | <ExprName>                                               | String | Name of specified named expression. |
| Return Value | None.                                                    |        |                                     |

|                |                                                 |
|----------------|-------------------------------------------------|
| Python Syntax  | DeleteNamedExpr(<ExprName>)                     |
| Python Example | <code>oModule.DeleteNamedExpr("Mag_JxE")</code> |

|                   |                                   |
|-------------------|-----------------------------------|
| <b>VB Syntax</b>  | DeleteNamedExpr <ExprName>        |
| <b>VB Example</b> | oModule.DeleteNamedExpr "Mag_JxE" |

## EnterComplex

Enters a complex number onto the stack.

|                     |                                                                                 |        |                                        |
|---------------------|---------------------------------------------------------------------------------|--------|----------------------------------------|
| <b>UI Access</b>    | Click <b>Number</b> , and then click <b>Scalar. Complex</b> option is selected. |        |                                        |
| <b>Parameters</b>   | Name                                                                            | Type   | Description                            |
|                     | <ComplexNum>                                                                    | String | String of complex value. Ex. "1 + 2j". |
| <b>Return Value</b> | None.                                                                           |        |                                        |

|                       |                                 |
|-----------------------|---------------------------------|
| <b>Python Syntax</b>  | EnterComplex(<ComplexNum>)      |
| <b>Python Example</b> | oModule.EnterComplex("1 + 2 j") |

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Syntax</b>  | EnterComplex <ComplexNum>      |
| <b>VB Example</b> | oModule.EnterComplex "1 + 2 j" |

## EnterComplexVector

Enters a complex vector onto the stack.

|                  |                                                                                 |
|------------------|---------------------------------------------------------------------------------|
| <b>UI Access</b> | Click <b>Number</b> , and then click <b>Vector. Complex</b> option is selected. |
|------------------|---------------------------------------------------------------------------------|

|              |                 |       |                                                        |
|--------------|-----------------|-------|--------------------------------------------------------|
| Parameters   | Name            | Type  | Description                                            |
|              | <ComplexVector> | Array | Array of strings containing X, Y and Z complex values. |
| Return Value | None.           |       |                                                        |

|                |                                                                                                                  |
|----------------|------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | EnterComplexVector(<ComplexVector>)                                                                              |
| Python Example | <pre>oModule.EnterComplexVector<br/><br/>([<br/>    "1 + 2 j",<br/>    "1 + 2 j",<br/>    "1 + 2 j"<br/>])</pre> |

|            |                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------|
| VB Syntax  | EnterComplexVector <ComplexVector>                                                                   |
| VB Example | <pre>oModule.EnterComplexVector _<br/>Array("1 + 2 j",_<br/>    "1 + 2 j",_<br/>    "1 + 2 j")</pre> |

## EnterLine

Enters a line defined in the 3D Modeler editor.

|                     |                                                   |        |                                                  |
|---------------------|---------------------------------------------------|--------|--------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Line</b> |        |                                                  |
| <b>Parameters</b>   | Name                                              | Type   | Description                                      |
|                     | <LineName>                                        | String | Name of a line defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                             |        |                                                  |

|                       |                            |  |  |
|-----------------------|----------------------------|--|--|
| <b>Python Syntax</b>  | EnterLine(<LineName>)      |  |  |
| <b>Python Example</b> | oModule.EnterLine("Line1") |  |  |

|                   |                           |  |  |
|-------------------|---------------------------|--|--|
| <b>VB Syntax</b>  | EnterLine <LineName>      |  |  |
| <b>VB Example</b> | oModule.EnterLine "Line1" |  |  |

## EnterPoint

Enters a point defined in the 3D Modeler editor.

|                     |                                                      |        |                                                   |
|---------------------|------------------------------------------------------|--------|---------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Point</b> . |        |                                                   |
| <b>Parameters</b>   | Name                                                 | Type   | Description                                       |
|                     | <PointName>                                          | String | Name of a point defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                                |        |                                                   |

|                       |                              |
|-----------------------|------------------------------|
| <b>Python Syntax</b>  | EnterPoint(<PointName>)      |
| <b>Python Example</b> | oModule.EnterPoint("Point1") |

|                   |                             |
|-------------------|-----------------------------|
| <b>VB Syntax</b>  | EnterPoint <PointName>      |
| <b>VB Example</b> | oModule.EnterPoint "Point1" |

## EnterQty

Enters a field quantity.

|                     |                                                        |        |                                                  |
|---------------------|--------------------------------------------------------|--------|--------------------------------------------------|
| <b>UI Access</b>    | Click <b>Quantity</b> , and then select from the list. |        |                                                  |
| <b>Parameters</b>   | Name                                                   | Type   | Description                                      |
|                     | <FieldQty>                                             | String | The field quantity to be entered onto the stack. |
| <b>Return Value</b> | None.                                                  |        |                                                  |

|                       |                       |
|-----------------------|-----------------------|
| <b>Python Syntax</b>  | EnterQty(<FieldQty>)  |
| <b>Python Example</b> | oModule.EnterQty("E") |

|                   |                      |
|-------------------|----------------------|
| <b>VB Syntax</b>  | EnterQty <FieldQty>  |
| <b>VB Example</b> | oModule.EnterQty "E" |

## EnterScalar

Enters a scalar onto the stack.

|              |                                                                                        |        |                                          |
|--------------|----------------------------------------------------------------------------------------|--------|------------------------------------------|
| UI Access    | Click <b>Number</b> and then click <b>Scalar</b> . <b>Complex</b> option not selected. |        |                                          |
| Parameters   | Name                                                                                   | Type   | Description                              |
|              | <Scalar>                                                                               | Double | The real number to enter onto the stack. |
| Return Value | None.                                                                                  |        |                                          |

|                |                                                    |
|----------------|----------------------------------------------------|
| Python Syntax  | EnterScalar(<Scalar>)                              |
| Python Example | <code>oModule.EnterScalar(3.14159265358979)</code> |

|            |                                                   |
|------------|---------------------------------------------------|
| VB Syntax  | EnterScalar <Scalar>                              |
| VB Example | <code>oModule.EnterScalar 3.14159265358979</code> |

## EnterScalarFunc

Enters a scalar function.

|            |                                                       |        |                                                                  |
|------------|-------------------------------------------------------|--------|------------------------------------------------------------------|
| UI Access  | Click <b>Function</b> and then select <b>Scalar</b> . |        |                                                                  |
| Parameters | Name                                                  | Type   | Description                                                      |
|            | <VarName>                                             | String | Name of a variable to enter as a scalar function onto the stack. |



|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                               |
|-----------------------|-----------------------------------------------|
| <b>Python Syntax</b>  | EnterScalarFunc(<VarName>)                    |
| <b>Python Example</b> | <code>oModule.EnterScalarFunc("Phase")</code> |

|                   |                                              |
|-------------------|----------------------------------------------|
| <b>VB Syntax</b>  | EnterScalarFunc <VarName>                    |
| <b>VB Example</b> | <code>oModule.EnterScalarFunc "Phase"</code> |

## EnterSurf

Enters a surface defined in the 3D Modeler editor.

|                     |                                                        |        |                                                     |
|---------------------|--------------------------------------------------------|--------|-----------------------------------------------------|
| <b>UI Access</b>    | Click <b>Geometry</b> and then select <b>Surface</b> . |        |                                                     |
| <b>Parameters</b>   | Name                                                   | Type   | Description                                         |
|                     | <SurfName>                                             | String | Name of a surface defined in the 3D Modeler editor. |
| <b>Return Value</b> | None.                                                  |        |                                                     |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | EnterSurf(<SurfName>)                        |
| <b>Python Example</b> | <code>oModule.EnterSurf("Rectangle1")</code> |

|                  |                      |
|------------------|----------------------|
| <b>VB Syntax</b> | EnterSurf <SurfName> |
|------------------|----------------------|

|                   |                                             |
|-------------------|---------------------------------------------|
| <b>VB Example</b> | <code>oModule.EnterSurf "Rectangle1"</code> |
|-------------------|---------------------------------------------|

## EnterVector

Enters a vector onto the stack.

|                     |                                                                                          |       |                                                                  |
|---------------------|------------------------------------------------------------------------------------------|-------|------------------------------------------------------------------|
| <b>UI Access</b>    | Click <b>Number</b> , and then click <b>Vector</b> . <b>Complex</b> option not selected. |       |                                                                  |
| <b>Parameters</b>   | Name                                                                                     | Type  | Description                                                      |
|                     | <VectorArray>                                                                            | Array | Array of strings containing X, Y and Z components of the vector. |
| <b>Return Value</b> | None.                                                                                    |       |                                                                  |

|                       |                                                   |
|-----------------------|---------------------------------------------------|
| <b>Python Syntax</b>  | <code>EnterVector(&lt;VectorArray&gt;)</code>     |
| <b>Python Example</b> | <code>oModule.EnterVector([1.0, 1.0, 1.0])</code> |

|                   |                                                       |
|-------------------|-------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EnterVector &lt;VectorArray&gt;</code>          |
| <b>VB Example</b> | <code>oModule.EnterVector Array(1.0, 1.0, 1.0)</code> |

## EnterVectorFunc

Enters a vector function.

|                   |                                                       |      |             |
|-------------------|-------------------------------------------------------|------|-------------|
| <b>UI Access</b>  | Click <b>Function</b> and then select <b>Vector</b> . |      |             |
| <b>Parameters</b> | Name                                                  | Type | Description |
|                   |                                                       |      |             |

|              |                                                                                                                                                                                                                    |                                                                                                                                            |       |                                                                                                                                            |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------|
|              | <table><tr><td>&lt;VarNames&gt;</td><td>Array</td><td>Array of strings containing names of a variable for the X, Y, and Z coordinates, respectively, to enter as a vector function on the stack.</td></tr></table> | <VarNames>                                                                                                                                 | Array | Array of strings containing names of a variable for the X, Y, and Z coordinates, respectively, to enter as a vector function on the stack. |
| <VarNames>   | Array                                                                                                                                                                                                              | Array of strings containing names of a variable for the X, Y, and Z coordinates, respectively, to enter as a vector function on the stack. |       |                                                                                                                                            |
| Return Value | None.                                                                                                                                                                                                              |                                                                                                                                            |       |                                                                                                                                            |

|                |                                                      |
|----------------|------------------------------------------------------|
| Python Syntax  | EnterVectorFunc(<VarNames>)                          |
| Python Example | <code>oModuleEnterVectorFunc(["X", "Y", "Z"])</code> |

|            |                                                          |
|------------|----------------------------------------------------------|
| VB Syntax  | EnterVectorFunc <VarNames>                               |
| VB Example | <code>oModuleEnterVectorFunc Array("X", "Y", "Z")</code> |

## EnterVol

Enters a volume defined in the 3D Modeler editor.

|              |                                                       |        |                                                    |
|--------------|-------------------------------------------------------|--------|----------------------------------------------------|
| UI Access    | Click <b>Geometry</b> and then select <b>Volume</b> . |        |                                                    |
| Parameters   | Name                                                  | Type   | Description                                        |
|              | <VolumeName>                                          | String | Name of a volume defined in the 3D Modeler editor. |
| Return Value | None.                                                 |        |                                                    |

|                |                                       |
|----------------|---------------------------------------|
| Python Syntax  | EnterVol(<VolumeName>)                |
| Python Example | <code>oModule.EnterVol("Box1")</code> |

|                   |                         |
|-------------------|-------------------------|
| <b>VB Syntax</b>  | EnterVol <VolumeName>   |
| <b>VB Example</b> | oModule.EnterVol "Box1" |

## ExportOnGrid [Fields Calculator]

Evaluates the top stack element at a set of points specified by a grid and exports the data to a file.

|                     |                                                       |         |                                                                                                                                        |
|---------------------|-------------------------------------------------------|---------|----------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Click <b>Export</b> , and then click <b>On Grid</b> . |         |                                                                                                                                        |
| <b>Parameters</b>   | Name                                                  | Type    | Description                                                                                                                            |
|                     | <OutputFile>                                          | String  | Name of the output file.                                                                                                               |
|                     | <Min>                                                 | Array   | Minimum values for the coordinate components of the grid system                                                                        |
|                     | <Max>                                                 | Array   | Maximum values for the coordinate components of the grid system                                                                        |
|                     | <Spacing>                                             | Array   | Spacing values for the coordinate components of the grid system                                                                        |
|                     | <SolnName>                                            | String  | Name of the simulation setup                                                                                                           |
|                     | <VarVals>                                             | Array   | Array of strings containing setup definitions.                                                                                         |
|                     | <IncludePoints>                                       | Boolean | Optional. Specifies whether include points in the output file.                                                                         |
|                     | <CSType>                                              | String  | Optional. Type of coordinate system. "Cartesian" (default)   "Cylindrical"   "Spherical"                                               |
|                     | <Offsets>                                             | Array   | Optional. Origin for the offset coordinate system. For Cartesian, x, y, z, for Cylindrical, R, Phi, Z, for Spherical, Rho, Theta, Phi. |
|                     | <ByCount>                                             | Boolean | Optional.                                                                                                                              |
| <b>Return Value</b> | None.                                                 |         |                                                                                                                                        |

**Note:** Regarding the **ExportOnGrid** legacy script which only has "IncludePtInOutput" argument (the last Boolean one), AEDT can still read it and assign other new arguments as default values. Those default values are RefCSName = "global", PtInSI = "True", FieldInRefCS = "False"

|                      |                                                                                                     |
|----------------------|-----------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b> | ExportOnGrid(<OutputFile>, <Min>, <Max>, <Spacing>, <SolnName>, <SolnParameters>, [<ExportOption>], |
|----------------------|-----------------------------------------------------------------------------------------------------|

|                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                | "IncludePointsInOutput:=", <boolean>], "RefCSName:=", <CSName>, "PtsInSI:=", <boolean>, "FieldRefCS:=", <boolean>], "<CSType>", [ <Offsets>, <ByCount>)                                                                                                                                                                                                                                                                                                            |
| Python Example | <pre>oModule.ExportOnGrid("C:\offset_grid_model_unit_ref.fld",     ["-1mm", "16mm", "0mm"],     ["1mm", "18mm", "1mm"],     ["2mm", "2mm", "1mm"],     "4500MHz : LastAdaptive",     ["Freq:=", "4.5GHz", "Phase:=" , "0deg"],     [         "NAME:ExportOption",         "IncludePtInOutput:=" , True,         "RefCSName:=" , "offset",         "PtInSI:=" , False,         "FieldInRefCS:=" , True     ],     "Cartesian", ["0mm", "0mm", "0mm"], False )</pre> |

|            |                                                                                                                                                                                                                                                            |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | ExportOnGrid(<OutputFile>, <Min>, <Max>, <Spacing>, <SolnName>, <SolnParameters>, [<ExportOption>, "IncludePointsInOutput:=", <boolean>], "RefCSName:=", <CSName>, "PtsInSI:=", <boolean>, "FieldRefCS:=", <boolean>], "<CSType>", [ <Offsets>, <ByCount>) |
| VB Example | oModule.ExportOnGrid "C:/Grid.fld", _                                                                                                                                                                                                                      |

```

Array("0mm", "0deg", "-25mm"), _
Array("20mm", "90deg", "125mm"), _
Array("10mm", "45deg", "50mm"), _
"Setup1 : LastAdaptive", _
Array("Freq:=", "10000Hz", "Phase:=", "0deg"), _
true, "Cylindrical", _
Array("0mm", "0mm", "0mm")

oModule.ExportOnGrid("C:\offset_grid_model_unit_ref.fld",
 Array("-1mm", "16mm", "0mm"),
 Array("1mm", "18mm", "1mm"),
 Array("2mm", "2mm", "1mm"),
 "4500MHz : LastAdaptive",
 Array("Freq:=", "4.5GHz", "Phase:=", "0deg"),
 [
 "NAME:ExportOption",
 "IncludePtInOut:=" , True,
 "RefCSName:=" , "offset",
 "PtInSI:=" , False,
 "FieldInRefCS:=" , True
]

```

```

],
"Cartesian", Array("0mm", "0mm", "0mm"), False
)

```

## ExportToFile [Fields Calculator]

Evaluates the top stack element at a set of points specified in an external file and exports the data to a file.

**Note:** Regarding to legacy **ExportToFile** script which only has “IncludePtInOutput” argument (the last Boolean one), AEDT can still read it and assign other new arguments as default values. Those default values are RefCSName = “global”, PtInSI = “True”, FieldInRefCS = “False”

| UI Access         | Click <b>Export</b> , and then click <b>To File</b> . |        |                                                                                                                                                                                                   |
|-------------------|-------------------------------------------------------|--------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | Name                                                  | Type   | Description                                                                                                                                                                                       |
|                   | <ReportName>                                          | String | Name of report to be exported.                                                                                                                                                                    |
|                   | <FileName>                                            | String | Full path of the exported image file name; with extension of<br>.txt - Post processor format file<br>.csv - Comma-delimited data file<br>.tab - Tab-separated file<br>.dat - Ansys plot data file |
|                   | <solution>                                            |        | Solution Name                                                                                                                                                                                     |
|                   | <freq>                                                |        | frequency                                                                                                                                                                                         |
|                   | <phase>                                               |        | phase                                                                                                                                                                                             |
|                   | <Export Options>                                      |        | Whether to include points in output, RefCSName, Pts in SI units <boolean>, Field in RefCS, <boolean>.                                                                                             |
|                   |                                                       |        |                                                                                                                                                                                                   |

|              |       |
|--------------|-------|
| Return Value | None. |
|--------------|-------|

|                |                                                                                                                                                                                                                                                                                                                                                                                                                |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ExportToFile(<ReportName>, <FileName>, <solution>, ["Freq:=", <freq>, "Phase:=", "<phase>"], <ExportOptions>)                                                                                                                                                                                                                                                                                                  |
| Python Example | <pre>oModule.ExportToFile("C:\\offset_file_model_unit_ref.fld", "C:\\offset_SI.pts", "4500MHz : LastAdaptive",     [         "Freq:="          , "4.5GHz",         "Phase:="         , "0deg"     ],     [         "NAME:ExportOption",         "IncludePtInOutput:=" , True,         "RefCSName:="        , "offset",         "PtInSI:="           , False,         "FieldInRefCS:="     , True     ]) </pre> |

|           |                                       |
|-----------|---------------------------------------|
| VB Syntax | ExportToFile <ReportName>, <FileName> |
|-----------|---------------------------------------|



|                            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB<br/>Examp-<br/>e</b> | <pre>oModule.ExportToFile("C:\\offset_file_model_unit_ref.fld", "C:\\offset_SI.pts",<br/>"4500MHz : LastAdaptive",<br/><br/>    [<br/><br/>        "Freq:="          , "4.5GHz",<br/>        "Phase:="        , "0deg"<br/><br/>    ],<br/><br/>    [<br/><br/>        "NAME:ExportOption",<br/>        "IncludePtInOutput:=" , True,<br/>        "RefCSName:="        , "offset",<br/>        "PtInSI:="           , False,<br/>        "FieldInRefCS:="     , True<br/><br/>    ]<br/>)</pre> |
|----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## GetTopEntryValue

Gets the value of the top entry of the calculator stack.

| UI Access    | N/A                                           |        |                                      |
|--------------|-----------------------------------------------|--------|--------------------------------------|
| Parameters   | Name                                          | Type   | Description                          |
|              | <SolnName>                                    | String | Name of specified solution.          |
|              | <VarVals>                                     | Array  | Array of variable name, value pairs. |
| Return Value | Array of strings containing top entry values. |        |                                      |

|                |                                                                                                                                            |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | GetTopEntryValue(<SolnName>, <VarVals>)                                                                                                    |
| Python Example | <pre>oModule.GetTopEntryValue(<br/>    "Setup1:LastAdaptive",<br/>    ["Freq:=", "1GHz", "Phase:=", "0deg", "x_size:=", "2mm"]<br/>)</pre> |

|            |                                                                                                                                              |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | GetTopEntryValue <SolnName>, <VarVals>                                                                                                       |
| VB Example | <pre>oModule.GetTopEntryValue "Setup1:LastAdaptive", _<br/>    Array("Freq:=", "1GHz", "Phase:=", "0deg", _<br/>    "x_size:=", "2mm")</pre> |

## LoadNamedExpressions

Loads a named expression definition from a saved file.

|            |                                                                          |        |                                                                                       |
|------------|--------------------------------------------------------------------------|--------|---------------------------------------------------------------------------------------|
| UI Access  | In the Fields Calculator, click <b>Load From...</b> in the Library area. |        |                                                                                       |
| Parameters | Name                                                                     | Type   | Description                                                                           |
|            | <FileName>                                                               | String | Filename and full path to the file to hold the named expression definition.           |
|            | <FieldType>                                                              | String | For products with just one filed type, it is set to "Fields".                         |
|            | <NamedExpr>                                                              | Array  | rray of strings containing the names of expression definitions to load from the file. |

|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>LoadNamedExpressions(&lt;FileName&gt;, &lt;FieldType&gt;, &lt;NamedExpr&gt;)</code>                      |
| <b>Python Example</b> | <pre>oModule.LoadNamedExpressions(<br/>"C:\Ansoft\PersonalLib\smth.clc",<br/>"Fields", ["smoothedtemp"])</pre> |

|                   |                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>LoadNamedExpressions &lt;FileName&gt;, &lt;FieldType&gt;, &lt;NamedExpr&gt;</code>                              |
| <b>VB Example</b> | <pre>oModule.LoadNamedExpressions _<br/>"C:\Ansoft\PersonalLib\smth.clc", _<br/>"Fields", Array("smoothedtemp")</pre> |

## SaveNamedExpressions

Saves a named expression definition to a file.

|                     |                                                                        |         |                                                                                        |
|---------------------|------------------------------------------------------------------------|---------|----------------------------------------------------------------------------------------|
| <b>UI Access</b>    | In the Fields Calculator, click <b>Save To...</b> in the Library area. |         |                                                                                        |
| <b>Parameters</b>   | Name                                                                   | Type    | Description                                                                            |
|                     | <FileName>                                                             | String  | Filename and full path to the file to hold the named expression definition.            |
|                     | <NamedExprs>                                                           | Array   | Array of strings containing the names of expression definitions to load from the file. |
|                     | <Overwrite>                                                            | Boolean | Specifies whether to overwrite the file.                                               |
| <b>Return Value</b> | None.                                                                  |         |                                                                                        |

|                       |                                                                                                    |
|-----------------------|----------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SaveNamedExpressions(<FileName>, <NamedExprs>, <Overwrite>)                                        |
| <b>Python Example</b> | <pre>oModule.SaveNamedExpressions( "C:\Ansoft\PersonalLib\smth.clc", ["smoothedtemp"], True)</pre> |

|                   |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SaveNamedExpressions <FileName>, <NamedExprs>, <Overwrite>                                                |
| <b>VB Example</b> | <pre>oModule.SaveNamedExpressions _ "C:\Ansoft\PersonalLib\smth.clc", _ Array("smoothedtemp"), true</pre> |

## 20 - Motion Setup Script Commands

Motion setup commands should be executed by the **ModelSetup** module.

```
Set oModule = oDesign.GetModule("ModelSetup")
oModule.CommandName <args>
```

### Conventions Used in the Motion Setup Chapter

<AssignmentObjects>

Type: Array of strings

An array of object names.

### General Motion Setup Script Commands

Following are general script commands recognized by the **ModelSetup** module:

- [DeleteMotionSetup](#)
- [ReassignMoving](#)

## DeleteMotionSetup

**Use:** Deletes the motion setup.

**Command:** Maxwell3D or Maxwell2D>Model>Motion Setup>Unassign Band

**Syntax:** DeleteMotionSetup

**Return Value:** None

**Parameters:** List of strings. A list of names of motion setups to delete.

**Example:** oModule.DeleteMotionSetup(["MotionSetup2"])

The [] can contain more than one name as it is a list.

## ReassignMoving

**Use:** Specifies a new geometry assignment for moving objects.

**Command:** Maxwell3D or Maxwell2D>Model>Motion Setup>Add Selected Objects or  
Maxwell3D or Maxwell2D>Model>Motion Setup>Remove Selected Objects

**Syntax:** ReassignMoving Array("Name:Moving", "Objects:=", <AssignmentObjects>)

**Return Value:** None

**Example:** oModule.ReassignMoving Array("NAME:Moving", \_  
"Objects:=", Array("Box4", "Box1"))

## Commands to Create and Edit the Band

Following are script commands for creating and editing bands that are recognized by the **ModelSetup** module:

### Note:

In the following commands, all named data can be included/excluded as desired and may appear in any order.

- [AssignBand](#)
- [EditMotionSetup](#)
- [GetMotionSetupNames](#)

## AssignBand

**Use:** Assigns the selected object as a band.

**Command:** Maxwell3D or Maxwell2D>Model>Motion Setup>Assign Band

**Syntax:** AssignBand <BandDataArray>

**Return Value:** None*Parameters:*

```
<BandDataArray>

 Array("NAME:Band",
 "Move Type:=", <Translate/Rotate>,
 "Coordinate System:=", <CoordinateSystemName>,
 "Axis:=", <X/Y/Z>,
 "Is Positive:=", <bool>,
 "InitPos:=", <value>,
 "NegativePos:=", <value>,
 "PositivePos:=", <value>,
 "TranslatePeriodic:=", <bool>,
 "Consider Mechanical Transient:=", <bool>,
 "velocity:=", <value>,
 "objects:=", <AssignmentObjects>)
```

**Example:** Assign band as translate, do not consider mechanical transient.

```
oModule.AssignBand _
Array("NAME:Band", "Move Type:=", "Translate",
"Coordinate System:=", "Global", "Axis:=", "Z",
"Is Positive:=", true, "InitPos:=", "0mm", _
"NegativePos:=", "0mm", "PositivePos:=", "1mm",
"TranslatePeriodic:=", true,
```

```
"Consider Mechanical Transient:=", false,
"Velocity:=", "0m_per_sec",
"Objects:=", Array("band"))
```

**Example:** Assign band as translate, consider mechanical transient.

```
oModule.AssignBand _
Array("NAME:Band", _
"Move Type:=", "Translate",
"Coordinate System:=", "Global", "Axis:=", "Z",
"Is Positive:=", true, "InitPos:=", "0mm",
"NegativePos:=", "0mm", "PositivePos:=", "1mm",
"TranslatePeriodic:=", true,
"Consider Mechanical Transient:=", true,
"Velocity:=", "0m_per_sec", "Mass:=", "1kg",
"Damping:=", "1", "Load Force:=", "1nNewton"
"Objects:=", Array("band"))
```

**Example:** Assign band as rotate, do not consider mechanical transient.

```
oModule.AssignBand _
Array("NAME:Band", "Move Type:=", "Rotate",
"Coordinate System:=", "Global", "Axis:=", "Z",
"Is Positive:=", true, "InitPos:=", "0deg",
```



```
"HasRotateLimit:=", false, "NonCylindrical:=",
false, "Consider Mechanical Transient:=", false,
"Angular Velocity:=", "0deg_per_sec", "Objects:=",
Array("band"))
```

**Example:** Assign band as rotate, consider mechanical transient.

```
oModule.AssignBand _
Array("NAME:Band", "Move Type:=", "Rotate",
"Coordinate System:=", "Global", "Axis:=", "Z",
"Is Positive:=", true, "InitPos:=", "0deg",
"HasRotateLimit:=", false, "NonCylindrical:=",
false, "Consider Mechanical Transient:=", true,
"Angular Velocity:=", "0deg_per_sec",
"Moment of Inertia:=", "1", "Damping:=", "0",
"Load Torque:=", "0NewtonMeter", "Objects:=",
Array("band"))
```

## EditMotionSetup

**Use:** Edits the motion setup.

**Command:** Double-click the moving item in the project tree to edit it.

**Syntax:** EditMotionSetup <SetupName>, <Array>

**Return Value:** None

**Parameters:** <SetupName>

The name of the motion setup to be edited.

```
<BanddataArray>
 Array("NAME:Data",
 "Move Type:=", <Translate/Rotate>,
 "Coordinate System:=", <CoordinateSystemName>,
 "PostProcessing Coordinate System:=", <CoordinateSystemName>
 "Axis:=", <X/Y/Z>,
 "Is Positive:=", <bool>,
 "InitPos:=", <value>,
 "NegativePos:=", <value>,
 "PositivePos:=", <value>,
 "TranslatePeriodic:=", <bool>,
 "Consider Mechanical Transient:=", <bool>,
 "velocity:=", <value>,
 "Mass:=", <value>,
 "Damping:=", <value>,
 "Load Force:=", <value>)
```

**Example:** Translational Periodic motion, no mechanical transient.

```
oModule.EditMotionSetup "MotionSetup1", Array("NAME:Data", "Move Type:=", "Translate",
 "Coordinate System:=", "Global",
```

```
"PostProcessing Coordinate System:=",
"Global", "Axis:=", "X", "Is Positive:=", true,
"InitPos:=", "0meter", "NegativePos:=", "0meter",
"PositivePos:=", "0.1meter",
"TranslatePeriodic:=", true,
"Consider Mechanical Transient:=", false,
"Velocity:=", "0m_per_sec")
```

## GetMotionSetupNames

**Use:** Gets the motion setup parameters being used for the object. Applicable to 2D and 3D Transient solution types.

**Command:** None

**Syntax:** GetMotionSetupNames ()

**Return Value:** List of motion setup names

**Parameters:** None

**Example:** oModule.GetMotionSetupNames ()

## Other Commands Recognized By the ModelSetup Module

The following command is also recognized by the **ModelSetup** module:

- [GetSymmetryMultiplier](#)
- [SetSymmetryMultiplier](#)

## GetSymmetryMultiplier

**Use:** Returns the symmetry multiplier of the model. Applicable to 2D and 3D Transient solution types.

**Command:** None

**Syntax:** oModule.GetSymmetryMultiplier ()

**Return Value:** The symmetry multiplier integer value.

**Parameters:** None

**Example:** oModule.GetSymmetryMultiplier ()

## SetSymmetryMultiplier

**Use:** Sets the symmetry multiplier. This symmetry multiplier will be automatically applied to all input quantities including: input voltage, inductance, resistance, load torque, mass, damping, external circuit; and all output quantities including: induced voltages, flux linkages in every winding, stranded loss, solid loss, core loss, torque and force.

**Command:** Maxwell3D or Maxwell2D>Model>Set Symmetry Multiplier

**Syntax:** SetSymmetryMultiplier <int>

**Return Value:** None

**Parameters:** <int>

**Example:** oModule.SetSymmetryMultiplier 2

## 21 - Parameter Setup Script Commands

Parameter setup commands should be executed by the **MaxwellParameterSetup** module.

```
Set oModule = oDesign.GetModule("MaxwellParameterSetup")
```

```
oModule.CommandName <args>
```

### Conventions Used in the Parameter Chapter

<ParameterName>

Type: <string>

Name of a parameter.

<ParameterNameArray>

Type: Array of strings

An array of the names in a group of parameters.

<AssignmentObjects>

Type: Array of strings

An array of object names.

### General Parameter Setup Script Commands

Following are general script commands recognized by the **MaxwellParameterSetup** module:

- [DeleteParameters](#)
- [DeleteAllParameters](#)

- [RenameParameter](#)
- [ReassignParameter](#)

**Commands to Create and Edit Parameters**

Following are script commands for creating and editing parameters that are recognized by the **MaxwellParameterSetup** module:

- [AssignForce](#)
- [EditForce](#)
- [AssignTorque](#)
- [EditTorque](#)
- [AssignMatrix](#)
- [EditMatrix](#)
- [DeleteReduceMatrix](#)
- [RenameReduceMatrix](#)
- [AddReduceOp](#)
- [DeleteReduceOp](#)
- [EditReduceOp](#)
- [RenameReduceOp](#)

**AddReduceOp**

Adds a reduce matrix group.

| UI Access  | Right-click on the matrix and select either <b>Join in Series...</b> or <b>Join in Parallel...</b> |        |                       |
|------------|----------------------------------------------------------------------------------------------------|--------|-----------------------|
| Parameters | Name                                                                                               | Type   | Description           |
|            | <MatrixName>                                                                                       | String | Name of parent matrix |
|            | <ReduceMatrixName>                                                                                 | String | Name of reduce matrix |

|                     |                              |              |                                                                                                                                   |
|---------------------|------------------------------|--------------|-----------------------------------------------------------------------------------------------------------------------------------|
|                     | <b>&lt;ReduceOpArray&gt;</b> | <b>Array</b> | Array("NAME:<GroupName>",<br>"Type:=", one of "Join in Series" or "Join in Parallel",<br>"Sources:=", "<Source1>, <Source2, ...") |
| <b>Return Value</b> | None                         |              |                                                                                                                                   |

|                       |                                                                                                                                                                                                       |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddReduceOp (<MatrixName>, <ReduceMatrixName>, <ReduceOpArray>)                                                                                                                                       |
| <b>Python Example</b> | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.AddReduceOp ("Matrix1", "ReduceMatrix1", ["NAME:JoinSeries1", "Type:=", "Join in Series", "Sources:=", "Current1,Current3" ])</pre> |

|                   |                                                                                                                                                                                                                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AddReduceOp <MatrixName> <ReduceMatrixName> <ReduceOpArray>                                                                                                                                                                                                                                                           |
| <b>VB Example</b> | <pre>Set oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.AddReduceOp "Matrix1", "ReduceMatrix1", Array("NAME:JoinSeries1", "Type:=", "Join in Series", "Sources:=", "Current1,Current3") oModule.AddReduceOp "Matrix1", "ReduceMatrix2", Array("NAME:JoinParallel1", "Type:=", "Join in Parallel",</pre> |

|  |                                   |
|--|-----------------------------------|
|  | "Sources:=", "Current2,Current4") |
|--|-----------------------------------|

## AssignForce

Creates a force.

| UI Access    | Maxwell3D or Maxwell2D > Parameters > Assign > Force |       |                                                                                                                                                                          |
|--------------|------------------------------------------------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                 | Type  | Description                                                                                                                                                              |
|              | <ForceArray>                                         | Array | Array containing force data:<br><br>Array("NAME:<ForceName>",<br><br>"IsVirtual:=", <boolean>,<br><br>"ReferenceCS:=", <string>,<br><br>"Objects:=", <array of objects>) |
| Return Value | None                                                 |       |                                                                                                                                                                          |

**Note:** For 3D Eddy Current A-Φ simulations, only the Lorentz force type is available, so there is no type argument for force.

| Python Syntax  | AssignForce (<ForceArray>)                                                                                                                               |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup")<br/>oModule.AssignForce(<br/>    [ "NAME:Force1",<br/>      "Reference CS:=" , "Global",</pre> |



|  |                                                                     |
|--|---------------------------------------------------------------------|
|  | <pre>"Is Virtual:=" , True,<br/>"Objects:=" , ["Core"]<br/>])</pre> |
|--|---------------------------------------------------------------------|

|                   |                                                                                                                                                                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <b>AssignForce &lt;ForceArray&gt;</b>                                                                                                                                         |
| <b>VB Example</b> | <pre>oModule.AssignForce<br/>    Array("NAME:Forcel",<br/>        "Is Virtual:=", true,<br/>        "Reference CS:=", "Global",<br/>        "Objects:=", Array("Box1"))</pre> |

## [Beta] AssignLayoutForce

Creates a layout force.

|                     |                                                               |             |                                                                                                                                    |
|---------------------|---------------------------------------------------------------|-------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | <b>Maxwell3D&gt;Parameters&gt;Assign&gt;Force from Layout</b> |             |                                                                                                                                    |
| <b>Parameters</b>   | <b>Name</b>                                                   | <b>Type</b> | <b>Description</b>                                                                                                                 |
|                     | <ForceArray>                                                  | Array       | Array containing force data:<br><br>Array("NAME:<ForceName>",<br>"ReferenceCS:=", <string>,<br><array of nets and layers choices>) |
| <b>Return Value</b> | None                                                          |             |                                                                                                                                    |

| Python Syntax  | AssignLayoutForce (<ForceArray>)                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.AssignLayoutForce(     [ "NAME:LayoutForce1",       "Reference CS:=" , "Global",       ["NAME:NetsAndLayersChoices",         [           "NAME:LC1_1",           [             "NAME:NetLayerSetMap",             [               "NAME:net1",               "LayerSet:=" , ["Ground","Top"]             ],             [               "NAME:net2",               "LayerSet:=" , ["Top"]             ]           ]         ]       ]     ]   )</pre> |

|  |                           |
|--|---------------------------|
|  | <pre>    ]<br/>  ])</pre> |
|--|---------------------------|

| VB Syntax  | AssignLayoutForce <ForceArray>                                                                                                                                                                                                                                                                                                               |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AssignLayoutForce<br/><br/>Array("NAME:LayoutForce4",<br/>  "Reference CS:=", "Global",<br/>  Array("NAME:NetsAndLayersChoices", Array("NAME:LC1_1",<br/>    Array("NAME:NetLayerSetMap",<br/>      Array("NAME:net1", "LayerSet:=", Array("Ground", "Top")),<br/>      Array("NAME:net2", "LayerSet:=", Array("Top")))))</pre> |

# AssignMatrix

**Use:** Creates a matrix.

**Command:** Maxwell3D or Maxwell2D > Parameters > Assign > Matrix

**Syntax:**AssignMatrix <MatrixArray>

**Return Value:** None

**Parameters:**

```
<MatrixArray>

 Array("NAME:<MatrixName>",
 Array("NAME:MatrixEntry",
 Array("NAME:MatrixEntry",
 "Source:=", <string>,
```

```
"NumberOfTurns:=", <int>),
...
)
Array("NAME:MatrixGroup",
Array("NAME:MatrixGroup",
"GroupName:=", <string>,
"NumberOfBranches:=", <int>,
"Sources:=", <nameArray>),
...
))
```

### **Example:**

```
oModule.AssignMatrix _
Array("NAME:Matrix1", _
Array("NAME:MatrixEntry", _
Array("NAME:MatrixEntry", _
"Source:=", "Current1", _
"NumberOfTurns:=", "1"),
Array("NAME:MatrixEntry", _
"Source:=", "Current3", _
"NumberOfTurns:=", "1")), _
```

```
Array("NAME:MatrixGroup", _
Array("NAME:MatrixGroup", _
"GroupName:=", "Group1", _
"NumberOfBranches:=", "1", _
"Sources:=", "Current1,Current3")))
```

**Example: GC and RL matrices specified for Eddy Current A- $\Phi$  simulation**

```
oModule.AssignMatrix(
 [
 "NAME:Matrix1",
 [
 "NAME:RLMatrix",
 [
 "NAME:MatrixEntry",
 [
 "NAME:MatrixEntry",
 "Source:=" , 0,
 "NumberOfTurns:=" , "1"
],
 [
 "NAME:MatrixEntry",
 "Source:=" , 8,
```

```
"NumberOfTurns:=" , "1"
]
],
[
"NAME:MatrixGroup"
],
[
"NAME:GroundSources",
6,
7
]
],
[
"NAME:GCMatrix",
[
"NAME:MatrixEntry",
[
"NAME:MatrixEntry",
"Source:=" , 0,
"NumberOfTurns:=" , "1"
```

```
],
[
 "NAME:MatrixEntry",
 "Source:=" , 8,
 "NumberOfTurns:=" , "1"
]
],
[
 "NAME:MatrixGroup"
],
[
 "NAME:GroundSources",
 6,
 7
]
]
])
```

# AssignTorque

Creates a torque.

|           |                                                       |
|-----------|-------------------------------------------------------|
| UI Access | Maxwell3D or Maxwell2D > Parameters > Assign > Torque |
|-----------|-------------------------------------------------------|

| Parameters   | Name          | Type  | Description                                                                                                                                                                                                          |
|--------------|---------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <TorqueArray> | Array | Array containing torque data:<br><br>Array("NAME:<TorqueName>",<br>"Is Virtual:=", <bool>,<br>"Coordinate System:=", <string><br>"Axis:=", <string>,<br>"Is Positive:=", <bool><br>"Objects:=", <AssignmentObjects>) |
| Return Value | None          |       |                                                                                                                                                                                                                      |

**Note:** For 3D Eddy Current A- $\Phi$  simulations, only the Lorentz torque type is available, so there is no type argument for torque.

|                |                                                                                                                                                                                       |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AssignTorque (<TorqueArray>)                                                                                                                                                          |
| Python Example | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.AssignTorque(     [         "NAME:Torque1",         "Coordinate System:=" , "Global",         "Axis:=" , "Z",</pre> |



|  |                                                                   |
|--|-------------------------------------------------------------------|
|  | <pre>"Is Positive:=" , True, "Objects:=" , ["Cylinder1"] ])</pre> |
|--|-------------------------------------------------------------------|

| VB Syntax  | AssignTorque <TorqueArray>                                                                                                                                                         |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oModule.AssignTorque     Array("NAME:Torque1", "Is Virtual:=", true, "Coordinate System:=", "Global", "Axis:=", "Z", "Is Positive:=", true, "Objects:=", Array("Box3"))</pre> |

## DeleteAllParameters

**Use:** Deletes all parameters.

**Command:** Maxwell3D or Maxwell2D>Parameters>Delete All

**Syntax:** DeleteAllParameters

**Return Value:** None

**Example:** oModule.DeleteAllParameters

## DeleteParameters

Deletes one or more specified parameters.

| UI Access  | Delete button in Maxwell List dialog box (Maxwell3D or Maxwell2D>List) |        |                             |
|------------|------------------------------------------------------------------------|--------|-----------------------------|
| Parameters | Name                                                                   | Type   | Description                 |
|            | <NameArray>                                                            | string | An array of parameter names |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| <b>Python Syntax</b>  | DeleteParameters ([<NameArray>])                |
| <b>Python Example</b> | oModule.DeleteParameters(["Force1", "Torque1"]) |

|                   |                                                     |
|-------------------|-----------------------------------------------------|
| <b>VB Syntax</b>  | DeleteParameters <NameArray>                        |
| <b>VB Example</b> | oModule.DeleteParameters Array("Force1", "Torque1") |

## DeleteReduceMatrix

Deletes a specified Reduce Matrix in the project.

|                     |                                                                                      |        |                               |
|---------------------|--------------------------------------------------------------------------------------|--------|-------------------------------|
| <b>UI Access</b>    | Right-click on the desired reduce matrix item in Program Manager, then select Delete |        |                               |
| <b>Parameters</b>   | Name                                                                                 | Type   | Description                   |
|                     | <MatrixName>                                                                         | String | The name of the parent matrix |
|                     | <ReduceMatrixName>                                                                   | String | The name of the reduce matrix |
| <b>Return Value</b> | None                                                                                 |        |                               |

|                       |                                                         |
|-----------------------|---------------------------------------------------------|
| <b>Python Syntax</b>  | DeleteReduceMatrix (<MatrixName>, <ReduceMatrixName>)   |
| <b>Python Example</b> | oModule.DeleteReduceMatrix ("Matrix1", "ReduceMatrix1") |

|                   |                                                       |
|-------------------|-------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteReduceMatrix <MatrixName>, <ReduceMatrixName>   |
| <b>VB Example</b> | oModule.DeleteReduceMatrix "Matrix1", "ReduceMatrix1" |

## DeleteReduceOp

Deletes a specified Reduce Matrix group in the project.

|                     |                                                                                            |        |                                     |
|---------------------|--------------------------------------------------------------------------------------------|--------|-------------------------------------|
| <b>UI Access</b>    | Right-click on the desired reduce matrix group item in Program Manager, then select Delete |        |                                     |
| <b>Parameters</b>   | Name                                                                                       | Type   | Description                         |
|                     | <MatrixName>                                                                               | String | The name of the parent matrix       |
|                     | <ReduceMatrixName>                                                                         | String | The name of the reduce matrix       |
|                     | <GroupName>                                                                                | String | The name of the reduce matrix group |
| <b>Return Value</b> | None                                                                                       |        |                                     |

|                       |                                                                    |
|-----------------------|--------------------------------------------------------------------|
| <b>Python Syntax</b>  | DeleteReduceOp (<MatrixName>, <ReduceMatrixName>, <GroupName>)     |
| <b>Python Example</b> | oModule.DeleteReduceOp ("Matrix1", "ReduceMatrix1", "JoinSeries1") |

|                   |                                                                  |
|-------------------|------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteReduceOp <MatrixName>, <ReduceMatrixName>, <GroupName>     |
| <b>VB Example</b> | oModule.DeleteReduceOp "Matrix1", "ReduceMatrix1", "JoinSeries1" |

# EditForce

**Use:** Edits a force parameter.

**Command:** Double-click the parameter in the project tree to edit it.

**Syntax:**EditForce <ParameterName>, <ForceArray>

**Return Value:** None

# [Beta] EditLayoutForce

Creates a layout force.

|              |                                                                                                                                                                 |       |                                                                                                                                            |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------|--------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Right-click the desired Layout Force Parameter in the project manager and select <b>Properties...</b> to open the <b>Layout Force Setup</b> dialog for editing. |       |                                                                                                                                            |
| Parameters   | Name                                                                                                                                                            | Type  | Description                                                                                                                                |
|              | <ForceArray>                                                                                                                                                    | Array | Array containing force data:<br><br>Array("NAME:<ForceName>",<br><br>"ReferenceCS:=", <string>,<br><br><array of nets and layers choices>) |
| Return Value | None                                                                                                                                                            |       |                                                                                                                                            |

|                |                                                                                              |
|----------------|----------------------------------------------------------------------------------------------|
| Python Syntax  | EditLayoutForce (<ForceArray>)                                                               |
| Python Example | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup")<br/>oModule.EditLayoutForce(</pre> |

```
["NAME:LayoutForce4",
 "Reference CS:=" , "Global",
 ["NAME:NetsAndLayersChoices",
 [
 "NAME:LC1_1",
 [
 "NAME:NetLayerSetMap",
 [
 "NAME:net1",
 "LayerSet:=" , ["Ground"]
],
 [
 "NAME:net2",
 "LayerSet:=" , ["Top"]
]
]
]
]
])
```

**VB Syntax****EditLayoutForce <ForceArray>**

|                   |                                                                                                                                                                                                                                                                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>Set oModule = oDesign.GetModule("MaxwellParameterSetup")  oModule.EditLayoutForce  Array("NAME:LayoutForce4", "Reference CS:=", "Global", Array("NAME:NetsAndLayersChoices", Array("NAME:LC1_1", Array("NAME:NetLayerSetMap", Array("NAME:net1", "LayerSet:=", Array("Ground")), Array("NAME:net2", "LayerSet:=", Array("Top")))))</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

EditMatrix

**Use:** Edits a matrix parameter.

**Command:** Double-click the parameter in the project tree to edit it.

**Syntax:**EditMatrix <ParameterName>, <MatrixArray>

**Return Value:** None

EditReduceOp

Edits a reduce matrix group.

|                   |                                                                         |        |                                                                                          |
|-------------------|-------------------------------------------------------------------------|--------|------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Right-click on the reduce matrix group and select <b>Properties....</b> |        |                                                                                          |
| <b>Parameters</b> | Name                                                                    | Type   | Description                                                                              |
|                   | <MatrixName>                                                            | String | Name of parent matrix                                                                    |
|                   | <ReduceMatrixName>                                                      | String | Name of reduce matrix                                                                    |
|                   | <GroupName>                                                             | String | Current name of the reduce matrix group                                                  |
|                   | <ReduceOpArray>                                                         | Array  | Array("NAME:<NewGroupName>",<br>"Type:=", one of "Join in Series" or "Join in Parallel", |

|              |                                                                                                      |                                          |  |  |  |                                          |
|--------------|------------------------------------------------------------------------------------------------------|------------------------------------------|--|--|--|------------------------------------------|
|              | <table><tr><td></td><td></td><td>"Sources:=", "&lt;Source1&gt;, &lt;Source2, ...")</td></tr></table> |                                          |  |  |  | "Sources:=", "<Source1>, <Source2, ...") |
|              |                                                                                                      | "Sources:=", "<Source1>, <Source2, ...") |  |  |  |                                          |
| Return Value | None                                                                                                 |                                          |  |  |  |                                          |

|                       |                                                                                                                                                                                                                    |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditReduceOp (<MatrixName>, <ReduceMatrixName>, <GroupName>, <ReduceOpArray>)                                                                                                                                      |
| <b>Python Example</b> | <pre>oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.EditReduceOp ("Matrix1", "ReduceMatrix1", "Group1" ["NAME:NewSeriesGroup", "Type:=", "Join in Series", "Sources:=", "Current1,Current3" ])</pre> |

|                   |                                                                                                                                                                                                                          |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | EditReduceOp <MatrixName>, <ReduceMatrixName>, <GroupName>, <ReduceOpArray>                                                                                                                                              |
| <b>VB Example</b> | <pre>Set oModule = oDesign.GetModule("MaxwellParameterSetup") oModule.EditReduceOp "Matrix1", "ReduceMatrix1", "Group1" Array("NAME:NewSeriesGroup", "Type:=", "Join in Series", "Sources:=", "Current1,Current3")</pre> |

## EditTorque

**Use:** Edits a torque parameter.

**Command:** Double-click the parameter in the project tree to edit it.

**Syntax:** EditTorque <ParameterName>, <TorqueArray>

**Return Value:** None

## ReassignParameter

**Use:** Specifies a new geometry assignment for a parameter.

**Command:** Maxwell3D or Maxwell2D>Parameters>Reassign

**Syntax:** ReassignParameter Array("Name:<ParameterName>", "Objects:=", <AssignmentObjects>)

**Return Value:** None

**Example:** oModule.ReassignParameter Array("NAME:Forcel", \_  
"Objects:=", Array("Box2"))

## RenameParameter

**Use:** Renames a parameter.

**Command:** Right-click the parameter item in the project tree, and click **Rename**.

**Syntax:** RenameParameter <OldName>, <NewName>

**Return Value:** None

*Parameters:*  
<OldName>

Type: <string>

<NewName>

Type: <string>

**Example:** oModule.RenameParameter "Forcel", "test"



## RenameReduceMatrix

Renames a specified Reduce Matrix in the project.

|                     |                                                                                      |        |                                   |
|---------------------|--------------------------------------------------------------------------------------|--------|-----------------------------------|
| <b>UI Access</b>    | Right-click on the desired reduce matrix item in Program Manager, then select Rename |        |                                   |
| <b>Parameters</b>   | Name                                                                                 | Type   | Description                       |
|                     | <MatrixName>                                                                         | String | The name of the parent matrix     |
|                     | <OldReduceMatrixName>                                                                | String | The old name of the reduce matrix |
|                     | <NewReduceMatrixName>                                                                | String | The new name of the reduce matrix |
| <b>Return Value</b> | None                                                                                 |        |                                   |

|                       |                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameReduceMatrix (&lt;MatrixName&gt;, &lt;OldReduceMatrixName&gt;, &lt;NewReduceMatrixName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.RenameReduceMatrix ("Matrix1", "ReduceMatrix1", "ReduceMatrix2")</code>                          |

|                   |                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameReduceMatrix &lt;MatrixName&gt;, &lt;OldReduceMatrixName&gt;, &lt;NewReduceMatrixName&gt;</code> |
| <b>VB Example</b> | <code>oModule.RenameReduceMatrix "Matrix1", "ReduceMatrix1", "ReduceMatrix2"</code>                          |

## RenameReduceOp

Renames a specified Reduce Matrix group in the project.

|                   |                                                                                            |        |                               |
|-------------------|--------------------------------------------------------------------------------------------|--------|-------------------------------|
| <b>UI Access</b>  | Right-click on the desired reduce matrix group item in Program Manager, then select Rename |        |                               |
| <b>Parameters</b> | Name                                                                                       | Type   | Description                   |
|                   | <MatrixName>                                                                               | String | The name of the parent matrix |

|                     |                    |        |                                         |
|---------------------|--------------------|--------|-----------------------------------------|
|                     | <ReduceMatrixName> | String | The name of the reduce matrix           |
|                     | <OldGroupName>     | String | The old name of the reduce matrix group |
|                     | <NewGroupName>     | String | The new name of the reduce matrix group |
| <b>Return Value</b> | None               |        |                                         |

|                       |                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameReduceOp (&lt;MatrixName&gt;, &lt;ReduceMatrixName&gt;, &lt;OldGroupName&gt;, &lt;NewGroupName&gt;)</code> |
| <b>Python Example</b> | <code>oModule.RenameReduceOp ("Matrix1", "ReduceMatrix1", "Series1", "Series2")</code>                                 |

|                   |                                                                                                                      |
|-------------------|----------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameReduceOp &lt;MatrixName&gt;, &lt;ReduceMatrixName&gt;, &lt;OldGroupName&gt;, &lt;NewGroupName&gt;</code> |
| <b>VB Example</b> | <code>oModule.RenameReduceOp "Matrix1", "ReduceMatrix1", "Series1", "Series2"</code>                                 |

## 22 - User Defined Document Script Commands

The product has to implement the [GetModule](#) call to create the UserDefinedDocument scripting object (e.g., Check AltraSimDesign.cpp (function GetMgrDispatch())). To access the UserDefinedDocuments scripting object, use:

```
Set oModule = oDesign.GetModule("UserDefinedDocuments")
```

Once you have the scripting object, you can use the following methods:

- [AddDocument](#)
- [EditDocument](#)
- [RenameDocument](#)
- [DeleteDocument](#)
- [UpdateDocument](#)
- [ViewHtmlDocument](#)
- [ViewPdfDocument](#)
- [SaveHtmlDocumentAs](#)
- [SavePdfDocumentAs](#)
- [GetDocumentDefinitionNames](#)
- [DeleteAllDocuments](#)
- [UpdateAllDocuments](#)

You can find examples of how to use these methods on each of the methods' pages. A complete [example of a Python script](#) is also available, as is an [example with a line by line explanation](#).

### AddDocument

Creates a new document based on provided data and traces. The document names come from UserDefinedDocument folder under syslib, userlib, and personallib. This creates a document and places it in the Project Manager under **Results > Documents**.

|              |                                                                              |       |                                            |
|--------------|------------------------------------------------------------------------------|-------|--------------------------------------------|
| UI Access    | Right click on <b>Results &gt; Create Document</b> . Choose a document name. |       |                                            |
| Parameters   | Name                                                                         | Type  | Description                                |
|              | <data>                                                                       | Array | Data the defines the document.             |
|              | <traces>                                                                     | Array | trace data for the inputs in the document. |
| Return Value | None                                                                         |       |                                            |

|                |                                                                                                                                                      |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AddDocument (<data>, <traces>)                                                                                                                       |
| Python Example | <pre>oModule.AddDocument ( [     "NAME:Design Summary",     "",     "SysLib",     "DesignSummary",     ["NAME:Inputs"] ], ["NAME:DocTraces"] )</pre> |

|           |                              |
|-----------|------------------------------|
| VB Syntax | AddDocument <data>, <traces> |
|-----------|------------------------------|

**VB  
Example**

In this example, the document names come from the UserDefinedDocument folder in the syslib, userlib, and personallib folders. This creates a document and places it in the Documents folder under results.

```
oModule.AddDocument _
Array("NAME:Design Summary", _
"", "SysLib", "DesignSummary", _
Array("NAME:Inputs")), _
Array("NAME:DocTraces")
```

The following example is explained in [Explication of a Sample UDD Script](#).

```
oModule.AddDocument Array("NAME:Test Report1", "Test Report", "SysLib", _
"Examples/TestUDDInputs", Array("NAME:Inputs", Array("NAME:DLMetrics", "Solution", _
"Data Line Metrics", -1, -1), Array("NAME:DQ0", "Trace", "DQ0", -1, -1), _
Array("NAME:DQS", "Trace", "DQS", -1, -1), _
Array("NAME:Name", "Text", "User Name", Array("Sita Ramesh")), _
Array("NAME:Summary", "Bool", "Display Summary", Array(true)), _
Array("NAME:Version", "Number", "Script Version")), _
Array("NAME:DocTraces", Array("NAME:DLMetrics", _
Array("User Defined", "", "DDR3 AC-Timing 4-DQ1", Array("Context:=", ""), _
Array("Index:=", Array("All"), "Trise:=", Array("Nominal"), "Tfall:=", _
Array("Nominal"), "Pulse_Width:=", Array("Nominal"), _
"Data_Rate:=", Array("Nominal"), "Length:=", Array("Nominal")), _
Array("Probe Component:=", Array("")), Array()), _
Array("NAME:DQ0", Array(_
```

|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Standard", "DQ0", "NexximTransient", Array("NAME:Context", "SimValueContext:=", Array( _ 1, 0, 2, 0, false, false, -1, 1, 0, 1, 1, "", 0, 0, _ "DE", false, "0", "DP", false, "20000000", "DT", false, "0.001", "WE", _ false, "100ns", "WM", false, "100ns", "WN", false, "0ps", "WS", false, "0ps")), _ Array("Time:=", Array("All"), "Trise:=", Array("Nominal"), "Tfall:=", Array("Nom- inal"), _ "Pulse_Width:=", Array("Nominal"), "Data_Rate:=", Array("Nominal"), _ "Length:=", Array("Nominal")), Array("Probe Component:=", Array("DQ0")), Array()))</pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

DeleteAllDocuments

Deletes all documents in the object.

|              |                                                                                                |
|--------------|------------------------------------------------------------------------------------------------|
| UI Access    | Right click on <b>Documents</b> in the Project Manager and click <b>Delete All Documents</b> . |
| Parameters   | None.                                                                                          |
| Return Value | None.                                                                                          |

|                |                              |
|----------------|------------------------------|
| Python Syntax  | DeleteAllDocuments()         |
| Python Example | oModule.DeleteAllDocuments() |

|           |                    |
|-----------|--------------------|
| VB Syntax | DeleteAllDocuments |
|-----------|--------------------|

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>VB Example</b> | <code>oModule.DeleteAllDocuments</code> |
|-------------------|-----------------------------------------|

## DeleteDocument

Deletes a specified document.

|                     |                                                                                                                          |        |                                     |
|---------------------|--------------------------------------------------------------------------------------------------------------------------|--------|-------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>Delete</b> . |        |                                     |
| <b>Parameters</b>   | Name                                                                                                                     | Type   | Description                         |
|                     | <name>                                                                                                                   | String | Name of the document to be deleted. |
| <b>Return Value</b> | None.                                                                                                                    |        |                                     |

|                       |                                                               |
|-----------------------|---------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>DeleteDocument(&lt;names&gt;)</code>                    |
| <b>Python Example</b> | <code>oModule.DeleteDocument("Project Design Summary")</code> |

|                   |                                                              |
|-------------------|--------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>DeleteDocument &lt;names&gt;</code>                    |
| <b>VB Example</b> | <code>oModule.DeleteDocument "Project Design Summary"</code> |

## EditDocument

Edits specified documents. If the document pops up a dialog box, the user can make a change the inputs for the document. The document is regenerated and updated. A new one is *not* created.

|                  |                                                                                                                                   |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b> | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>Modify document</b> . |
|------------------|-----------------------------------------------------------------------------------------------------------------------------------|

|                     |                  |             |                                           |
|---------------------|------------------|-------------|-------------------------------------------|
| <b>Parameters</b>   | <b>Name</b>      | <b>Type</b> | <b>Description</b>                        |
|                     | <originalName>   | String      | Name of the original document             |
|                     | <modifiedData>   | Array       | New data to add to or modify the document |
|                     | <modifiedtraces> | Array       | Trace data for the inputs of the document |
| <b>Return Value</b> | None.            |             |                                           |

|                       |                                                                                                                                                                                                           |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | EditDocument( <name>,<data>,<traces>)                                                                                                                                                                     |
| <b>Python Example</b> | <pre>oModule.EditDocument("Design Summary",<br/>[<br/>    "NAME:Design Summary",<br/>    "",<br/>    "SysLib",<br/>    "DesignSummary",<br/>    ["NAME:Inputs"]<br/>],<br/>["NAME:DocTraces"]<br/>)</pre> |

|                  |                                       |
|------------------|---------------------------------------|
| <b>VB Syntax</b> | EditDocument <name>, <data>, <traces> |
|------------------|---------------------------------------|



|                   |                                                                                                                                                                  |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oModule.EditDocument "Design Summary", _ Array("NAME:Design Summary", "", "SysLib", _ "DesignSummary", Array("NAME:Inputs")), Array("NAME:DocTraces")</pre> |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## GetDocumentDefinitionNames

Document definition names are the list of names that can be used to create a document. They appear when you click on **Create document**. This method returns the filenames of document definitions according to the files in the installation directories:

- syslib/UserDefinedDocuments
- userlib/UserDefinedDocuments
- personallib/UserDefinedDocuments

|                     |             |        |                                                |
|---------------------|-------------|--------|------------------------------------------------|
| <b>UI Access</b>    | NA          |        |                                                |
| <b>Parameters</b>   | Name        | Type   | Description                                    |
|                     | <separator> | String | Separator used to convey the directory "level" |
| <b>Return Value</b> | None.       |        |                                                |

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | GetDocumentDefinitionNames(<separator>) |
| <b>Python Example</b> | oModule.GetDocumentDefinitionNames("/") |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | GetDocumentDefinitionNames <separator> |
| <b>VB Example</b> | oModule.GetDocumentDefinitionNames "/" |

## GetDocumentNames

Retrieves the names for all documents.

|                     |                                             |
|---------------------|---------------------------------------------|
| <b>UI Access</b>    | N/A                                         |
| <b>Parameters</b>   | None.                                       |
| <b>Return Value</b> | Array of strings containing document names. |

|                       |                                         |
|-----------------------|-----------------------------------------|
| <b>Python Syntax</b>  | GetDocumentNames()                      |
| <b>Python Example</b> | <code>oModule.GetDocumentNames()</code> |

|                   |                                       |
|-------------------|---------------------------------------|
| <b>VB Syntax</b>  | GetDocumentNames                      |
| <b>VB Example</b> | <code>oModule.GetDocumentNames</code> |

## RenameDocument

Changes the name of a document.

|                   |                                                                                                                         |             |                              |
|-------------------|-------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------|
| <b>UI Access</b>  | Right click on the created document in the Project Manager under <b>Results&gt; Documents</b> and click <b>Rename</b> . |             |                              |
| <b>Parameters</b> | <b>Name</b>                                                                                                             | <b>Type</b> | <b>Description</b>           |
|                   | <oldName>                                                                                                               | String      | Current name of the document |
|                   | <newName>                                                                                                               | String      | New name of the document     |

|                     |       |
|---------------------|-------|
| <b>Return Value</b> | None. |
|---------------------|-------|

|                       |                                                                                 |
|-----------------------|---------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>RenameDocument(&lt;oldName&gt;, &lt;newName&gt;)</code>                   |
| <b>Python Example</b> | <code>oModule.RenameDocument("Design Summary", "Project Design Summary")</code> |

|                   |                                                                                |
|-------------------|--------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>RenameDocument &lt;oldName&gt;, &lt;newName&gt;</code>                   |
| <b>VB Example</b> | <code>oModule.RenameDocument "Design Summary", "Project Design Summary"</code> |

## SaveHtmlDocumentAs

Saves a pre-existing HTML file to a different name and/or location.

|                     |                                                                                                                                     |             |                                    |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>Save As &gt; HTML</b> . |             |                                    |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                         | <b>Type</b> | <b>Description</b>                 |
|                     | <code>&lt;name&gt;</code>                                                                                                           | String      | Name of the document to be saved.  |
|                     | <code>&lt;saveTo&gt;</code>                                                                                                         | String      | File path to save the document to. |
| <b>Return Value</b> | none                                                                                                                                |             |                                    |

|                       |                                                                         |
|-----------------------|-------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>SaveHtmlDocumentAs(&lt;name&gt;, &lt;saveTo&gt;)</code>           |
| <b>Python Example</b> | <code>oModule.SaveHtmlDocumentAs("Design Summary 1", "DS1.html")</code> |

|                   |                                                                       |
|-------------------|-----------------------------------------------------------------------|
| <b>VB Syntax</b>  | SaveHtmlDocumentAs <name> <saveTo>                                    |
| <b>VB Example</b> | <code>oModule.SaveHtmlDocumentAs "Design Summary 1" "DS1.html"</code> |

## SavePdfDocumentAs

Saves a pre-existing PDF file to a different name and/or location.

|                     |                                                                                                                                    |        |                                    |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------|--------|------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>Save As &gt; PDF</b> . |        |                                    |
| <b>Parameters</b>   | Name                                                                                                                               | Type   | Description                        |
|                     | <name>                                                                                                                             | String | Name of the document to be saved.  |
|                     | <saveTo>                                                                                                                           | String | File path to save the document at. |
| <b>Return Value</b> | None.                                                                                                                              |        |                                    |

|                       |                                                                       |
|-----------------------|-----------------------------------------------------------------------|
| <b>Python Syntax</b>  | SavePdfDocumentAs(<name>, <saveTo>)                                   |
| <b>Python Example</b> | <code>oModule.SavePdfDocumentAs("Design Summary 1", "DS1.pdf")</code> |

|                   |                                                                      |
|-------------------|----------------------------------------------------------------------|
| <b>VB Syntax</b>  | SavePdfDocumentAs <name>, <saveTo>                                   |
| <b>VB Example</b> | <code>oModule.SavePdfDocumentAs "Design Summary 1", "DS1.pdf"</code> |

## UpdateAllDocuments

Refreshes the contents of all created documents. This action is made on the folder rather than the individual document.

|                     |                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Right click on <b>Results &gt; Documents</b> in the Project Manager and click <b>Update All Documents</b> . |
| <b>Parameters</b>   | None.                                                                                                       |
| <b>Return Value</b> | None.                                                                                                       |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | UpdateAllDocuments()                      |
| <b>Python Example</b> | <code>oModule.UpdateAllDocuments()</code> |

|                   |                                         |
|-------------------|-----------------------------------------|
| <b>VB Syntax</b>  | UpdateAllDocuments                      |
| <b>VB Example</b> | <code>oModule.UpdateAllDocuments</code> |

## UpdateDocument

Refreshes the contents of the selected document.

|                     |                                                                                                                                   |        |                                    |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------|--------|------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>Update Document</b> . |        |                                    |
| <b>Parameters</b>   | Name                                                                                                                              | Type   | Description                        |
|                     | <name>                                                                                                                            | String | Name of the document to be updated |
| <b>Return Value</b> | None.                                                                                                                             |        |                                    |

|                       |                                           |
|-----------------------|-------------------------------------------|
| <b>Python Syntax</b>  | UpdateDocument(<name>)                    |
| <b>Python Example</b> | oModule.UpdateDocument("Test UDD Report") |

|                   |                                          |
|-------------------|------------------------------------------|
| <b>VB Syntax</b>  | UpdateDocument <name>                    |
| <b>VB Example</b> | oModule.UpdateDocument "Test UDD Report" |

## ViewHtmlDocument

Displays a pre-existing document as HTML.

|                     |                                                                                                                                     |        |                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------|--------|---------------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>View Xml Document</b> . |        |                                             |
| <b>Parameters</b>   | Name                                                                                                                                | Type   | Description                                 |
|                     | <name>                                                                                                                              | String | Name of the document to be viewed as a HTML |
| <b>Return Value</b> | None                                                                                                                                |        |                                             |

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | ViewHtmlDocument(<name>)                     |
| <b>Python Example</b> | oModule.ViewHtmlDocument("Design Summary 1") |

|                   |                                             |
|-------------------|---------------------------------------------|
| <b>VB Syntax</b>  | ViewHtmlDocument <name>                     |
| <b>VB Example</b> | oModule.ViewHtmlDocument "Design Summary 1" |

## ViewPdfDocument

Displays a pre-existing document as a PDF file.

|                     |                                                                                                                                     |        |                                            |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------|--------|--------------------------------------------|
| <b>UI Access</b>    | Right click on the created document in the Project Manager under <b>Results &gt; Documents</b> and click <b>View PDF Document</b> . |        |                                            |
| <b>Parameters</b>   | Name                                                                                                                                | Type   | Description                                |
|                     | <name>                                                                                                                              | String | Name of the document to be viewed as a PDF |
| <b>Return Value</b> | None.                                                                                                                               |        |                                            |

|                       |                                             |
|-----------------------|---------------------------------------------|
| <b>Python Syntax</b>  | ViewPdfDocument(<name>)                     |
| <b>Python Example</b> | oModule.ViewPdfDocument("Design Summary 1") |

|                   |                                            |
|-------------------|--------------------------------------------|
| <b>VB Syntax</b>  | ViewPdfDocument <name>                     |
| <b>VB Example</b> | oModule.ViewPdfDocument "Design Summary 1" |

## Explication of a Sample UDD Script

This VB script defines a document. It is a portion of one of the UDD example VB scripts.

```

Array("NAME:Test Report", ' Name of the document
"Test Report", ' Description of the document
"SysLib", ' Location of the python script: Syslib, Userlib, PeronalLib, etc.
"TestUDDReport", ' Relative path of the script in the UserDefinedDocuments folder

' This array is the start of the input definition.
```

Array("NAME:Inputs", ' Document Inputs keyword

' This array contains the Solution input.

Array("NAME:DLMetrics", ' Input name  
"Solution", ' Solution Input Type  
"Data Line Metrics", ' Input Description  
-1, ' Solution ID  
-1), ' Report ID

' This array contains the trace input.

Array("NAME:DQ0", ' Input name  
"Trace", ' Trace Input Type  
"DQ0", ' Input Description  
-1, ' Solution ID  
-1), ' Report ID

' This array contains the text input.

Array("NAME:Name", ' Input name  
"Text", ' Text Input Type  
"User Name", ' Input Description  
Array("Sita Ramesh")), ' Default Value

' This array contains the Bool input.

Array("NAME:Summary", ' Input name  
"Bool", ' Boolean Input Type  
"Display Summary", ' Input Description  
Array(true)), ' Default Value

' This array contains the number input.



```

Array("NAME:Version", ' Input name
"Number", ' Number Input Type
"Script Version", ' Input Description
Array(1021))), ' Default Value

' This array contains trace selection for the solution and trace inputs.
Array("NAME:DocTraces", ' Document traces keyword

' This array has input for "DLMetrics".
Array("NAME:DLMetrics", ' Input name

' This array defines a trace similar to the UDO. This trace definition is a User defined solution
Array("User Defined", "", "DDR3 AC-Timing 4-DQ1", Array("Context:=", ""), Array("Index:=", Array
("All"), "Trise:=", Array("Nominal"), "Tfall:=", Array("Nominal"), "Pulse_Width:=", Array("Nom-
inal"), "Data_Rate:=", Array("Nominal"), "Length:=", Array("Nominal")), Array("Probe Com-
ponent:=", Array(""))), Array()),

' This array is for input "DQ0".
Array("NAME:DQ0",

' This array defines a trace similar to the UDO. This trace definition is a Standard solution.
Array("Standard", "DQ0", "NexximTransient", Array("NAME:Context", "SimValueContext:=", Array(1,
0, 2, 0, false, false, -1, 1, 0, 1, 1, "", 0, 0, "DE", false, "0", "DP", _
false, "20000000", "DT", false, "0.001", "WE", false, "100ns", "WM", false, _
"100ns", "WN", false, "0ps", "WS", false, "0ps")), Array("Time:=", Array("All"), "Trise:=",
Array(_
"Nominal"), "Tfall:=", Array("Nominal"), "Pulse_Width:=", Array("Nominal"), "Data_Rate:=", Array
("Nominal"), "Length:=", Array("Nominal")), Array("Probe Component:=", Array(_
"DQ0")), Array()))

```

## Example Python Script: Defining a Document

This script creates a user-defied solution and a document.

```
import ScriptEnv
ScriptEnv.Initialize("Ansoft.ElectronicsDesktop")
oDesktop.RestoreWindow()
oProject = oDesktop.SetActiveProject("BJT_Inverter")
oDesign = oProject.SetActiveDesign("Nexxim1")
oModule = oDesign.GetModule("UserDefinedSolutionModule")

oModule.CreateUserDefinedSolution("UDS Distance Trace Arithmetic Result1", "SysLib", "TraceArithmetic/Distance Sweep Trace Arithmetic",
[
 "Offset 1:=", "0",
 "Scale 1:=", "1",
 "Offset 2:=", "0",
 "Scale 2:=", "1",
 "Operation:=", "Add"
],
[
 [
 "Standard",
 "probel",
 "Transient",
 [
 style="font-family: monospace;">"NAME:Context",
 style="font-family: monospace;">"SimValueContext:=", [1,0,2,0,False,False,-
1,1,0,1,1,"",0,0,"DE",False,"0","DP",False,"500000000","DT",False,"0.001","NUMLEVELS",False,"0","-
WE",False,"10us","WM",False,"10us","WN",False,"0ns","WS",False,"0ns"]
],
 [
 "Time:=", ["All"]
],
 [
 "Probe Component:=", ["V(Port1)"]
]
]
]
```

```

],
 []
],
 [
 "Standard",
 "probe2",
 "Transient",
 [
 "NAME:Context",
 "SimValueContext:=", [1,0,2,0,False,False,-
1,1,0,1,1,"",0,0,"DE",False,"0","DP",False,"500000000","DT",False,"0.001","NUMLEVELS",False,"0",-
"WE",False,"10us","WM",False,"10us","WN",False,"0ns","WS",False,"0ns"]
],
 [
 "Time:=", ["All"]
],
 [
 "Probe Component:=", ["V(Port1)"]
],
 []
]
],
[])
oModule = oDesign.GetModule("UserDefinedDocuments")
oModule.AddDocument(
[
 "NAME:Test Report",
 "Test Report",
 "SysLib",
 "TestUDDInputs",
 [
 "NAME:Inputs",
 [
 "NAME:UDS1",

```

```

 "Solution",
 "UDS Distance Trace Arithmetic Result1",
 1000000,
 0
],
 [
 "NAME:UDS2",
 "Solution",
 "UDS Distance Trace Arithmetic Result2",
 1000000,
 2
]
],
[
 "NAME:DocTraces",
 [
 "NAME:UDS1",
 [
 "User Defined",
 "",
 "UDS Distance Trace Arithmetic Result1",
 [
 "Context:=", ""
],
 [
 "Distance:=", ["All"]
],
 [
 "Probe Component:=", [""]
],
 []
]
]
]

```

```
],
 [
 "NAME:UDS2",
 [
 "User Defined",
 "",
 "UDS Distance Trace Arithmetic Result1",
 [
 "Context:=", ""
],
 [
 "Distance:=", ["All"]
],
 [
 "Probe Component:=", [""]
],
 []
]
]
])
```

This page intentionally  
left blank.

## 23 - User Defined Solutions Commands

User Defined Solution commands should be executed by the "UserDefinedSolutionModule" module.

```
Set oDesign = oProject.SetActiveDesign("TestDesign1")

Set oModule =

oDesign.GetModule("UserDefinedSolutionModule")
```

The list of commands is as follows:

- [CreateUserDefinedSolution](#)
- [DeleteUserDefinedSolutions](#)
- [EditUserDefinedSolution](#)

### CreateUserDefinedSolution

Creates a new user defined solution.

| UI Access  | Right-click on <b>Results</b> > <b>Create User Defined Solution</b> . |        |                                                                                                                                                     |
|------------|-----------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                                                                  | Type   | Description                                                                                                                                         |
|            | <SoluName>                                                            | String | Name of user defined solution.                                                                                                                      |
|            | <LibType>                                                             | String | Indicates the library where the UDS plugin file is located. This parameter must be one of the following values: "SysLib", "UserLib", "PersonalLib". |
|            | <RelativePath>                                                        | String | The path of the UDS plugin file relative to the "UserDefinedOutputs" sub-directory of the library specified by <LibType>.                           |
|            | <PropList>                                                            | Array  | Strings specify name-value pairs corresponding to the UDS properties specified in the plugin file.                                                  |

|                     |                                               |       |                                                                                                         |
|---------------------|-----------------------------------------------|-------|---------------------------------------------------------------------------------------------------------|
|                     |                                               |       | For example:<br><br>Array("multiply_factor:=", "2.0", "component_name:=", "resistor1")                  |
|                     | <ProbeSelections>                             | Array | Name of the probe being specified. Note: this must match a probe name specified in the UDS plugin file. |
|                     | <DynamicProbes>                               | Array | Array of <ProbeSelection>'s, representing the probes that are used by dynamic-probes.                   |
| <b>Return Value</b> | String name of created user defined solution. |       |                                                                                                         |

|                       |                                                                                                                  |  |  |
|-----------------------|------------------------------------------------------------------------------------------------------------------|--|--|
| <b>Python Syntax</b>  | CreateUserDefinedSolution(<SoluName>, <LibType>, <RelativePath>, <PropList>, <ProbeSelections>, <DynamicProbes>) |  |  |
| <b>Python Example</b> | <pre>oModule.CreateUserDefinedSolution( "ConstantTimestep1", "SysLib", "ConstantTimestep", [], [], [])</pre>     |  |  |

|                   |                                                                                                                                                                                                                      |  |  |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | CreateUserDefinedSolution <SoluName>, <LibType>, <RelativePath>, <PropList>, <ProbeSelections>, <DynamicProbes>                                                                                                      |  |  |
| <b>VB Example</b> | <pre>oModule.CreateUserDefinedSolution "User Defined Solution 1", _ "SysLib", "Example", _ Array("multiplication_factor:=", "1.2"), _ Array(Array("Modal Solution Data", "Probe 1", "Setup1 : LastAdaptive", _</pre> |  |  |



|  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>Array(), Array("Freq=", Array( "All")), _<br/>Array("Probe Component=", Array("dB(S(1,1))")), Array()), _<br/>Array( "Modal Solution Data", "Probe 2",<br/>"Setup1 : LastAdaptive", Array(), _<br/>Array("Freq=", Array( "All")), _<br/>Array("Probe Component=", Array("mag(S(1,1))")), Array()), _<br/>Array(Array("Modal Solution Data", _<br/>"Dynamic Probe 1", "Setup1 : LastAdaptive", Array(), _<br/>Array("Freq=", Array( "All")), _<br/>Array("Probe Component=", Array("Freq")), Array()))</pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## DeleteUserDefinedSolutions

Deletes one or more user defined solutions.

|              |                                                              |       |                                                                            |
|--------------|--------------------------------------------------------------|-------|----------------------------------------------------------------------------|
| UI Access    | Delete button from the <b>User Defined Solutions</b> dialog. |       |                                                                            |
| Parameters   | Name                                                         | Type  | Description                                                                |
|              | <SoluNames>                                                  | Array | Array of strings containing names of User Defined Solutions to be deleted. |
| Return Value | None.                                                        |       |                                                                            |

|                |                                                                |
|----------------|----------------------------------------------------------------|
| Python Syntax  | DeleteUserDefinedSolutions(<SoluNames>)                        |
| Python Example | oModule.DeleteUserDefinedSolutions(["Solution1", "Solution2"]) |

|                   |                                                                                     |
|-------------------|-------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeleteUserDefinedSolutions < <i>SoluNames</i> >                                     |
| <b>VB Example</b> | <pre>oModule.DeleteUserDefinedSolutions _<br/>Array("Solution1", "Solution2")</pre> |

## EditUserDefinedSolution

Modifies an existing user defined solution.

|                     |                                                                       |        |                                                                                                                                                                                                                 |
|---------------------|-----------------------------------------------------------------------|--------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | <b>Edit</b> button from the <b>User Defined Solutions</b> dialog box. |        |                                                                                                                                                                                                                 |
| <b>Parameters</b>   | Name                                                                  | Type   | Description                                                                                                                                                                                                     |
|                     | < <i>SoluName</i> >                                                   | String | Name of user defined solution to be edited.                                                                                                                                                                     |
|                     | < <i>NewSoluName</i> >                                                | String | New name for the specified user defined solution.                                                                                                                                                               |
|                     | < <i>LibType</i> >                                                    | String | Indicates the library where the UDS plugin file is located. This parameter must be one of the following values: "SysLib", "UserLib", "PersonalLib".                                                             |
|                     | < <i>RelativePath</i> >                                               | String | The path of the UDS plugin file relative to the "UserDefinedOutputs" sub-directory of the library specified by < <i>LibType</i> >.                                                                              |
|                     | < <i>PropList</i> >                                                   | Array  | Strings specify name-value pairs corresponding to the UDS properties specified in the plugin file.<br><br>For example:<br><br><pre>Array("multiply_factor:=", "2.0", "component_name:=",<br/>"resistor1")</pre> |
|                     | < <i>ProbeSelections</i> >                                            | Array  | Name of the probe being specified. Note: this must match a probe name specified in the UDS plugin file.                                                                                                         |
|                     | < <i>DynamicProbes</i> >                                              | Array  | Array of <ProbeSelection>'s, representing the probes that are used by dynamic-probes.                                                                                                                           |
| <b>Return Value</b> | String name of update user defined solution.                          |        |                                                                                                                                                                                                                 |

|                       |                                                                                                                                                                                      |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>EditUserDefinedSolution(&lt;SoluName&gt;, &lt;NewSoluName&gt;, &lt;LibType&gt;, &lt;RelativePath&gt;, &lt;PropList&gt;, &lt;ProbeSelections&gt;, &lt;DynamicProbes&gt;)</code> |
| <b>Python Example</b> | <pre>oModule.EditUserDefinedSolution("ConstantTimestep1" "ConstantTimestep1After", "SysLib", "ConstantTimestep", [], [], [])</pre>                                                   |

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EditUserDefinedSolution &lt;SoluName&gt;, &lt;NewSoluName&gt;, &lt;LibType&gt;, &lt;RelativePath&gt;, &lt;PropList&gt;, &lt;ProbeSelections&gt;, &lt;DynamicProbes&gt;</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>VB Example</b> | <pre>oModule.CreateUserDefinedSolution "User Defined Solution 1", _ "User Defined Solution 1", "SysLib", "Example", _ Array("multiplication_factor:=", "1.2"), _ Array(Array("Modal Solution Data", "Probe 1", "Setup1 : LastAdaptive", _ Array(), Array("Freq:=", Array( "All"))), _ Array("Probe Component:=", Array("dB(S(1,1))")), Array()), _ Array( "Modal Solution Data", "Probe 2", "Setup1 : LastAdaptive", Array(), _ Array("Freq:=", Array( "All"))), _ Array("Probe Component:=", Array("mag(S(1,1))")), Array()), _ Array(Array("Modal Solution Data", _ "Dynamic Probe 1", "Setup1 : LastAdaptive", Array(), _</pre> |

|  |                                                                                                      |
|--|------------------------------------------------------------------------------------------------------|
|  | <pre>Array("Freq=", Array( "All")), _<br/>Array("Probe Component:=", Array("Freq")), Array()))</pre> |
|--|------------------------------------------------------------------------------------------------------|

## 24 - Network Data Explorer Script Commands

Network Data Explorer (NDE) scripting uses three objects, each with unique script commands:

- **Network Data Explorer** – top-level object obtained by calling `oNDE=oDesktop.GetTool("ndExplorer")`. Network Data Explorer commands are called using `oNDE`.
- **Network Data** – single set of S-parameters, corresponding to a single entry in the UI tree. Network Data commands are called using `oData`.
- **Post Process Settings** – settings that can be applied and removed from network data without making permanent changes to the underlying data. Post Process Settings commands are called using `oPostProc`.

Examples using the above objects:

```
oNDE = oDesktop.GetTool("ndExplorer")

oData = oNDE.Open("D:\folder\test.s2p")

success = oPostProc.AddDiffPair(2, 1, "Diff1", "Comm1", 100, 25)
```

The following commands are described in this section:

**Warning: The following commands are preliminary. Their syntax may change. Be prepared to make changes to legacy scripts.**

[AddDiffPair](#)

[Cascade \(SPISim\)](#)

[ClearDiffPairs](#)

[Clone](#)

[Close](#)

[Combine \(SPISim\)](#)

[Deembed \(SPISim\)](#)

[DeembedBack \(SPISim\)](#)

[DeembedFront \(SPISim\)](#)

[DisableDiffPairs](#)

[EnableDiffPairs](#)

[ExportCitiFile](#)

[ExportFullWaveSpice](#)

[ExportMatlab](#)

[ExportNMFData](#)

[ExportNetworkData](#)

[ExportSpreadsheet](#)

[ExportTouchstone](#)

[ExportTouchstone2](#)

[Extract \(SPISim\)](#)

[GetFrequencies](#)

[GetFrequencyCount](#)

[GetName](#)

[GetPortCount](#)

[GetPortNumber](#)

[GetPostProcSettings](#)

[GetSolutionVariation](#)

[GetVariation](#)

[HasSameData](#)

[LoadSolution](#)

[Open](#)

[Rename \(SPISim\)](#)

[Renormalize \(SPISim\)](#)

[Reorder](#)

[Reorder \(SPISim\)](#)

[Reset](#)

[SetAllPortImpedances](#)

[SetAllPortImpedances](#)

[SetPortDeembedDistance](#)

[SetPortImpedance](#)

[SetPostProcSettings](#)

[Smooth](#)

[Stretch \(SPISim\)](#)

[Terminate](#)

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

# AddDiffPair

Specifies a differential pair from two terminal ports.

|                                                  |                                                                                                                                                                                                                                                         |             |                                                     |
|--------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|-----------------------------------------------------|
| UI Access                                        | From the <b>Network Data Explorer</b> tab, select <b>Differential Pairs</b> in the <b>NDE</b> ribbon to open the <b>Differential Pairs</b> window. Then select differential pairs from the <b>Pairs</b> group box and click <b>Add Pairs &gt;&gt;</b> . |             |                                                     |
| Parameters                                       | <b>Name</b>                                                                                                                                                                                                                                             | <b>Type</b> | <b>Description</b>                                  |
|                                                  | <positiveTerminal>                                                                                                                                                                                                                                      | Integer     | The portNumber of the positive terminal.            |
|                                                  | <negativeTerminal>                                                                                                                                                                                                                                      | Integer     | The portNumber of the negative terminal.            |
|                                                  | <diffName>                                                                                                                                                                                                                                              | String      | The new name of the differential pin (e.g., Diff1). |
|                                                  | <commName>                                                                                                                                                                                                                                              | String      | The new name of the common pin (e.g., Comm1).       |
|                                                  | <diffImpedance>                                                                                                                                                                                                                                         | Double      | The differential pin characteristic impedance.      |
|                                                  | <commImpedance>                                                                                                                                                                                                                                         | Double      | The common pin characteristic impedance.            |
|                                                  | <matched>                                                                                                                                                                                                                                               | Boolean     | Specify whether to use matched pair.                |
| <b>Note:</b> The default value is <b>False</b> . |                                                                                                                                                                                                                                                         |             |                                                     |
| Return Value                                     | Boolean.                                                                                                                                                                                                                                                |             |                                                     |

|                |                                                                             |
|----------------|-----------------------------------------------------------------------------|
| Python Syntax  | AddDiffPair()                                                               |
| Python Example | <pre>success = oPostProc.AddDiffPair(2, 1, "Diff1", "Comm1", 100, 25)</pre> |

|           |             |
|-----------|-------------|
| VB Syntax | AddDiffPair |
|-----------|-------------|



|                   |                                                                                  |
|-------------------|----------------------------------------------------------------------------------|
| <b>VB Example</b> | <code>Set success = oPostProc.AddDiffPair 2, 1, "Diff1", "Comm1", 100, 25</code> |
|-------------------|----------------------------------------------------------------------------------|

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Cascade (SPISim)

Creates new network data from a group of network data objects cascaded together. The network data must have an even number of terminal ports and must have the same number of ports.

**Note:** Cascade opens and interacts with **SPISim** to complete.

|                     |                                                                                                                                                   |             |                                                                                  |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Cascade</b> from the drop-down menu. |             |                                                                                  |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                       | <b>Type</b> | <b>Description</b>                                                               |
|                     | <dataArray>                                                                                                                                       | Array       | These network data objects are cascaded together to create the new network data. |
| <b>Return Value</b> | Network IDispatch.<br><br><b>Note:</b> Cascade returns <b>None</b> if the specified network data does not have compatible terminal ports defined. |             |                                                                                  |

|                       |                                                                                                  |
|-----------------------|--------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>Cascade()</code>                                                                           |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Cascade([oData1, oData2, oData3])</pre> |

|                   |                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Cascade                                                                                                               |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/><br/>Set oData2 = oNDE.Cascade Array(oData1, oData2, oData3)</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## ClearDiffPairs

Clears all differential pair definitions. All other post-processing settings remain the same.

|                     |                                                                                                                                                                                                                                             |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Differential Pairs</b> in the <b>NDE</b> ribbon to open the <b>Differential Pairs</b> window. Then select differential pairs from the rightmost box and click << <b>Remove Pairs</b> . |
| <b>Parameters</b>   | None.                                                                                                                                                                                                                                       |
| <b>Return Value</b> | None.                                                                                                                                                                                                                                       |

|                       |                                                                                      |
|-----------------------|--------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ClearDiffPairs()                                                                     |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/><br/>oPostProc.ClearDiffPairs()</pre> |

|                   |                                                                                   |
|-------------------|-----------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ClearDiffPairs                                                                    |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>oPostProc.ClearDiffPairs</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Clone

Creates a copy of the network data object.

|                     |                                                                                                                                          |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select the network data object to clone. Then select <b>Clone</b> from the <b>NDE</b> ribbon. |
| <b>Parameters</b>   | None.                                                                                                                                    |
| <b>Return Value</b> | Network IDispatch.                                                                                                                       |

|                       |                                                                             |
|-----------------------|-----------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Clone()                                                                     |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData1 = oData.Clone()</pre> |

|                   |                                                                                   |
|-------------------|-----------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Clone                                                                             |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set oData1 = oData.Clone</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Close

Closes the network data object. The object will no longer be accessible.

|              |                                                                                                                            |
|--------------|----------------------------------------------------------------------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, click <b>Close</b> in the <b>NDE</b> ribbon, or select <b>File &gt; Close</b> . |
| Parameters   | None.                                                                                                                      |
| Return Value | None.                                                                                                                      |

|                |                                                                    |
|----------------|--------------------------------------------------------------------|
| Python Syntax  | Close()                                                            |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData.Close()</pre> |

|            |                                                                      |
|------------|----------------------------------------------------------------------|
| VB Syntax  | Close                                                                |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>oData.Close</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Combine (SPISim)

Combines frequencies from multiple network data objects to create a new network data with all of the frequencies.

**Note:** Combine opens and interacts with **SPISim** to complete.

|              |                                                                                                                                                   |                      |                                                                                                                                                |
|--------------|---------------------------------------------------------------------------------------------------------------------------------------------------|----------------------|------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Combine</b> from the drop-down menu. |                      |                                                                                                                                                |
| Parameters   | <b>Name</b><br><dataArray>                                                                                                                        | <b>Type</b><br>Array | <b>Description</b><br>These network data objects are combined to create the new network data.                                                  |
|              | <freqTol>                                                                                                                                         | Double               | If $\text{abs}(f1 - f2) < \text{freqTol}$ , f1 and f2 are considered the same frequency.<br><br><b>Note:</b> The default value is <b>100</b> . |
| Return Value | Network IDispatch.<br><br><b>Note:</b> Combine returns <b>None</b> if the network data does not have the same number of ports.                    |                      |                                                                                                                                                |

|                |                                                    |
|----------------|----------------------------------------------------|
| Python Syntax  | Combine()                                          |
| Python Example | <code>oNDE = oDesktop.GetTool("ndExplorer")</code> |

|  |                                                                    |
|--|--------------------------------------------------------------------|
|  | <code>oData4 = oNDE.Combine([oData1, oData2, oData3], 1000)</code> |
|--|--------------------------------------------------------------------|

|                   |                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Combine                                                                                                                              |
| <b>VB Example</b> | <code>Set oNDE = oDesktop.GetTool("ndExplorer")</code><br><code>Set oData4 = oNDE.Combine Array(oData1, oData2, oData3), 1000</code> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Deembed (SPISim)

Creates a new network data, dembedding the frontData and the backData from the front and back of the originalData.

**Note:** Deembed opens and interacts with **SPISim** to complete.

|                     |                                                                                                                                                                                                                                                                                               |             |                                                                |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Deembed</b> from the drop-down menu to open the <b>Deembed...</b> window, and choose <b>Given 3 S-params in order of: Total, Front, Back</b> from the <b>Settings</b> group box. |             |                                                                |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                                                                   | <b>Type</b> | <b>Description</b>                                             |
|                     | <code>&lt;originalData&gt;</code>                                                                                                                                                                                                                                                             | Object      | Original network data that is deembeded.                       |
|                     | <code>&lt;frontData&gt;</code>                                                                                                                                                                                                                                                                | Object      | Network data that is deembeded from the front of the original. |
|                     | <code>&lt;backData&gt;</code>                                                                                                                                                                                                                                                                 | Object      | Network data that is deembeded from the back of the original.  |
| <b>Return Value</b> | Network IDispatch.                                                                                                                                                                                                                                                                            |             |                                                                |

**Note:** Deembed returns **None** if the specified network data does not have compatible terminal ports defined.

|                       |                                                                                                       |
|-----------------------|-------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Deembed()                                                                                             |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Deembed(oData1, oDataFront, oDataBack)</pre> |

|                   |                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Deembed                                                                                                      |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set oData2 = oNDE.Deembed oData1, oDataFront, oDataBack</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## DeembedBack (SPISim)

Creates a new network data, dembedding the backData from the back of the originalData.

**Note:** DeembedBack opens and interacts with **SPISim** to complete.

|                  |                                                                                                                                                                                                                                                                                        |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b> | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Deembed</b> from the drop-down menu to open the <b>Deembed...</b> window, and choose <b>Given 3 S-params in order of: Total, Back</b> from the <b>Settings</b> group box. |
|------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| Parameters   | Name                                                                                                                        | Type   | Description                                                   |
|--------------|-----------------------------------------------------------------------------------------------------------------------------|--------|---------------------------------------------------------------|
|              | <originalData>                                                                                                              | Object | Original network data that is deembeded.                      |
|              | <backData>                                                                                                                  | Object | Network data that is deembeded from the back of the original. |
| Return Value | Network IDispatch.                                                                                                          |        |                                                               |
|              | <b>Note:</b> DeembedBack returns <b>None</b> if the specified network data does not have compatible terminal ports defined. |        |                                                               |

|                |                                                                                               |
|----------------|-----------------------------------------------------------------------------------------------|
| Python Syntax  | DeembedBack()                                                                                 |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.DeembedBack(oData1, oDataBack)</pre> |

|            |                                                                                                      |
|------------|------------------------------------------------------------------------------------------------------|
| VB Syntax  | DeembedBack                                                                                          |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set oData2 = oNDE.DeembedBack oData1, oDataBack</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.



## DeembedFront (SPISim)

Creates a new network data, dembedding the frontData from the front of the originalData.

**Note:** DeembedFront opens and interacts with **SPISim** to complete.

|              |                                                                                                                                                                                                                                                                                         |             |                                                                |
|--------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|----------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Deembed</b> from the drop-down menu to open the <b>Deembed...</b> window, and choose <b>Given 3 S-params in order of: Total, Front</b> from the <b>Settings</b> group box. |             |                                                                |
| Parameters   | <b>Name</b>                                                                                                                                                                                                                                                                             | <b>Type</b> | <b>Description</b>                                             |
|              | <originalData>                                                                                                                                                                                                                                                                          | Object      | Original network data that is deembeded.                       |
|              | <frontData>                                                                                                                                                                                                                                                                             | Object      | Network Data that is deembeded from the front of the original. |
| Return Value | Network IDispatch.<br><br><b>Note:</b> DeembedFront returns <b>None</b> if the specified network data does not have compatible terminal ports defined.                                                                                                                                  |             |                                                                |

|                |                                                                                                     |
|----------------|-----------------------------------------------------------------------------------------------------|
| Python Syntax  | DeembedFront()                                                                                      |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData2 = oNDE.DeembedFront(oData1, oDataFront)</pre> |

|            |                                                      |
|------------|------------------------------------------------------|
| VB Syntax  | DeembedFront                                         |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")</pre> |

|  |                                                   |
|--|---------------------------------------------------|
|  | Set oData2 = oNDE.DeembedFront oData1, oDataFront |
|--|---------------------------------------------------|

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## DisableDiffPairs

Deactivates all differential pair definitions. This script does not remove them, so they [can be reactivated](#).

|              |                                                                                                                                                                                                                                                                                                                                                                                                                        |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, select <b>Differential Pairs</b> in the <b>NDE</b> ribbon to open the <b>Differential Pairs</b> window. Once differential pairs have been added to the rightmost box (i.e., select differential pairs from the Pairs group box and click <b>Add Pairs &gt;&gt;</b> ), remove check marks from the <b>Enabled</b> column to deactivate the corresponding differential pairs. |
| Parameters   | None.                                                                                                                                                                                                                                                                                                                                                                                                                  |
| Return Value | None.                                                                                                                                                                                                                                                                                                                                                                                                                  |

|                |                                                                               |
|----------------|-------------------------------------------------------------------------------|
| Python Syntax  | DisableDiffPairs()                                                            |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer") oPostProc.DisableDiffPairs()</pre> |

|            |                                                      |
|------------|------------------------------------------------------|
| VB Syntax  | DisableDiffPairs                                     |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")</pre> |

|  |                                         |
|--|-----------------------------------------|
|  | <code>oPostProc.DisableDiffPairs</code> |
|--|-----------------------------------------|

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## EnableDiffPairs

Enables all differential pair definitions.

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                 |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Differential Pairs</b> in the <b>NDE</b> ribbon to open the <b>Differential Pairs</b> window. Once differential pairs have been added to the rightmost box (i.e., select differential pairs from the Pairs group box and click <b>Add Pairs &gt;&gt;</b> ), add check marks to the <b>Enabled</b> column to activate the corresponding differential pairs. |
| <b>Parameters</b>   | None.                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>Return Value</b> | None.                                                                                                                                                                                                                                                                                                                                                                                                           |

|                       |                                                                                        |
|-----------------------|----------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>EnableDiffPairs()</code>                                                         |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") success = oPostProc.EnableDiffPairs()</pre> |

|                   |                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>EnableDiffPairs</code>                                                                 |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set success = oPostProc.EnableDiffPairs</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## ExportCitiFile

Writes the S-parameters to disk in Citifile format (\*.cit text file).

|            |                                                                                                                                                                                                        |             |                                                                                                                                                                                                                                            |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | From the <b>Network Data Explorer</b> tab, select <b>File &gt; Save As</b> to open a <b>Save As</b> explorer window. Then select <b>Citifile (*.cit)</b> from the <b>Save as type:</b> drop-down menu. |             |                                                                                                                                                                                                                                            |
| Parameters | <b>Name</b>                                                                                                                                                                                            | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                         |
|            | <filePath>                                                                                                                                                                                             | String      | The full path to the file.                                                                                                                                                                                                                 |
|            | <matrixType>                                                                                                                                                                                           | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – S-Parameters</li><li>• 1 – Y-Parameters</li><li>• 2 – Z-Parameters</li><li>• 3 – Gamma</li><li>• 4 – Impedance</li></ul> <div><b>Note:</b> The default value is 0.</div> |
|            | <formalType>                                                                                                                                                                                           | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – Magnitude/Angle (degrees)</li><li>• 1 – Real/Imaginary</li><li>• 2 – dB/Angle (degrees)</li></ul>                                                                        |

|                     |                                                            |                  |                                                                                                                                                                                                                         |
|---------------------|------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                                                            |                  | <b>Note:</b> The default value is <b>0</b> .                                                                                                                                                                            |
|                     | <precision>                                                | Integer          | The number of significant figures.<br><b>Note:</b> The default value is <b>10</b> .                                                                                                                                     |
|                     | <frequencies>                                              | Array of Doubles | A subset of the calculated frequencies. Any array values that don't have calculated data will be ignored. An empty array will cause all frequencies to be written.<br><b>Note:</b> The default value is an empty array. |
|                     |                                                            |                  |                                                                                                                                                                                                                         |
| <b>Return Value</b> | Boolean True if file was successfully written; else False. |                  |                                                                                                                                                                                                                         |

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportCitiFile()                                                                                                      |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") success = oData.ExportCitiFile("D:\folder\nel133.cit", 0, 1, 12, [])</pre> |

|                   |                                                                                                                                   |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportCitiFile                                                                                                                    |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set success = oData.ExportCitiFile "D:\folder\nel133.cit", 0, 1, 12, Array()</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## ExportMatlab

Writes the S-parameters to disk in Matlab format (\*.m text file).

|            |                                                                                                                                                                                                    |             |                                                                                                                                                                                                                                            |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | From the <b>Network Data Explorer</b> tab, select <b>File &gt; Save As</b> to open a <b>Save As</b> explorer window. Then select <b>MATLAB (*.m)</b> from the <b>Save as type:</b> drop-down menu. |             |                                                                                                                                                                                                                                            |
| Parameters | <b>Name</b>                                                                                                                                                                                        | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                         |
|            | <filePath>                                                                                                                                                                                         | String      | The full path to the file.                                                                                                                                                                                                                 |
|            | <matrixType>                                                                                                                                                                                       | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – S-Parameters</li><li>• 1 – Y-Parameters</li><li>• 2 – Z-Parameters</li><li>• 3 – Gamma</li><li>• 4 – Impedance</li></ul> <div><b>Note:</b> The default value is 0.</div> |
|            | <formalType>                                                                                                                                                                                       | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – Magnitude/Angle (degrees)</li><li>• 1 – Real/Imaginary</li><li>• 2 – dB/Angle (degrees)</li></ul>                                                                        |

|                     |                                                            |                  |                                                                                                                                                                                                                         |
|---------------------|------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                                                            |                  | <b>Note:</b> The default value is <b>0</b> .                                                                                                                                                                            |
|                     | <i>&lt;precision&gt;</i>                                   | Integer          | The number of significant figures.<br><b>Note:</b> The default value is <b>10</b> .                                                                                                                                     |
|                     | <i>&lt;frequencies&gt;</i>                                 | Array of Doubles | A subset of the calculated frequencies. Any array values that don't have calculated data will be ignored. An empty array will cause all frequencies to be written.<br><b>Note:</b> The default value is an empty array. |
| <b>Return Value</b> | Boolean True if file was successfully written; else False. |                  |                                                                                                                                                                                                                         |

|                       |                                                                                                                  |
|-----------------------|------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportMatlab()                                                                                                   |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") success = oData.ExportMatlab("D:\folder\ne133.m", 0, 1, 12, [])</pre> |

|                   |                                                                                                                             |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportMatlab                                                                                                                |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool "ndExplorer" Set success = oData.ExportMatlab "D:\folder\ne133.m", 0, 1, 12, Array()</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## ExportSpreadsheet

Writes the S-parameters to disk in spreadsheet format (\*.tab text file).

|            |                                                                                                                                                                                                                        |             |                                                                                                                                                                                                                                            |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | From the <b>Network Data Explorer</b> tab, select <b>File &gt; Save As</b> to open a <b>Save As</b> explorer window. Then select <b>Data Table (spreadsheet) (*.tab)</b> from the <b>Save as type:</b> drop-down menu. |             |                                                                                                                                                                                                                                            |
| Parameters | <b>Name</b>                                                                                                                                                                                                            | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                         |
|            | <filePath>                                                                                                                                                                                                             | String      | The full path to the file.                                                                                                                                                                                                                 |
|            | <matrixType>                                                                                                                                                                                                           | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – S-Parameters</li><li>• 1 – Y-Parameters</li><li>• 2 – Z-Parameters</li><li>• 3 – Gamma</li><li>• 4 – Impedance</li></ul> <div><b>Note:</b> The default value is 0.</div> |
|            | <formalType>                                                                                                                                                                                                           | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – Magnitude/Angle (degrees)</li><li>• 1 – Real/Imaginary</li><li>• 2 – dB/Angle (degrees)</li></ul>                                                                        |



|                     |                                                            |                  |                                                                                                                                                                                                                         |
|---------------------|------------------------------------------------------------|------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                                                            |                  | <b>Note:</b> The default value is <b>0</b> .                                                                                                                                                                            |
|                     | <precision>                                                | Integer          | The number of significant figures.<br><b>Note:</b> The default value is <b>10</b> .                                                                                                                                     |
|                     | <frequencies>                                              | Array of Doubles | A subset of the calculated frequencies. Any array values that don't have calculated data will be ignored. An empty array will cause all frequencies to be written.<br><b>Note:</b> The default value is an empty array. |
|                     | <renormalize>                                              | Boolean          | If <b>True</b> , will renormalize the data to the value of renormImpedance before writing.<br><b>Note:</b> The default value is <b>False</b> .                                                                          |
|                     | <renormImpedance>                                          | Double           | Data will be renormalized to this impedance before writing only if renormalize is <b>True</b> .<br><b>Note:</b> The default value is <b>50 ohms</b> .                                                                   |
| <b>Return Value</b> | Boolean True if file was successfully written; else False. |                  |                                                                                                                                                                                                                         |

|                       |                                                                                        |
|-----------------------|----------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportSpreadsheet()                                                                    |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData = oNDE.Open(&lt;model path&gt;)</pre> |

|  |                                                                                               |
|--|-----------------------------------------------------------------------------------------------|
|  | <code>success = oData.ExportSpreadsheet("D:\folder\ne133.tab", 0, 1, 12, [], True, 23)</code> |
|--|-----------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                            |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportSpreadsheet                                                                                                                                          |
| <b>VB Example</b> | <code>Set oData=oDesktop.GetTool("ndExplorer")</code><br><code>success = oData.ExportSpreadsheet "D:\folder\ne133.tab", 0, 1, 12, Array(), True, 23</code> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## ExportTouchstone

Writes the S-parameters to disk in Touchstone 1.0 format (\*.snp text file).

|                   |                                                                                                                                                                                                                     |             |                                                                                                                                                                                            |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | From the <b>Network Data Explorer</b> tab, select <b>File &gt; Save As</b> to open a <b>Save As</b> explorer window. Then select <b>Touchstone Format 1.0 (*.snp)</b> from the <b>Save as type:</b> drop-down menu. |             |                                                                                                                                                                                            |
| <b>Parameters</b> | <b>Name</b>                                                                                                                                                                                                         | <b>Type</b> | <b>Description</b>                                                                                                                                                                         |
|                   | <filePath>                                                                                                                                                                                                          | String      | The full path to the file.                                                                                                                                                                 |
|                   | <matrixType>                                                                                                                                                                                                        | Integer     | One of the following: <ul style="list-style-type: none"><li>• 0 – S-Parameters</li><li>• 1 – Y-Parameters</li><li>• 2 – Z-Parameters</li><li>• 3 – Gamma</li><li>• 4 – Impedance</li></ul> |

|                                      |                  |                                                                                                                                                                                                                                    |
|--------------------------------------|------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                      |                  | <b>Note:</b> The default value is <b>0</b> .                                                                                                                                                                                       |
| <code>&lt;formalType&gt;</code>      | Integer          | <p>One of the following:</p> <ul style="list-style-type: none"> <li>• 0 – Magnitude/Angle (degrees)</li> <li>• 1 – Real/Imaginary</li> <li>• 2 – dB/Angle (degrees)</li> </ul> <p><b>Note:</b> The default value is <b>0</b>.</p>  |
| <code>&lt;precision&gt;</code>       | Integer          | <p>The number of significant figures.</p> <p><b>Note:</b> The default value is <b>10</b>.</p>                                                                                                                                      |
| <code>&lt;frequencies&gt;</code>     | Array of Doubles | <p>A subset of the calculated frequencies. Any array values that don't have calculated data will be ignored. An empty array will cause all frequencies to be written.</p> <p><b>Note:</b> The default value is an empty array.</p> |
| <code>&lt;renormalize&gt;</code>     | Boolean          | <p>If <b>True</b>, will renormalize the data to the value of renormImpedance before writing.</p> <p><b>Note:</b> The default value is <b>False</b>.</p>                                                                            |
| <code>&lt;renormImpedance&gt;</code> | Double           | <p>Data will be renormalized to this impedance before writing only if renormalize is <b>True</b>.</p> <p><b>Note:</b> The default value is <b>50 ohms</b>.</p>                                                                     |
| <code>&lt;freqUnits&gt;</code>       | String           | One of the following: "Hz", "KHz", "MHz", "GHz".                                                                                                                                                                                   |

|                     |                                                                            |
|---------------------|----------------------------------------------------------------------------|
|                     | <div></div> <div></div> <div><b>Note:</b> The default value is "Hz".</div> |
| <b>Return Value</b> | Boolean True if file was successfully written; else False.                 |

|                       |                                                                                                                                             |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportTouchstone()                                                                                                                          |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>success = oData.ExportTouchstone("D:\folder\ne133.s2p", 0, 1, 12, [], True, 23, "GHz")</pre> |

|                   |                                                                                                                                                         |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportTouchstone                                                                                                                                        |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>success = oData.ExportTouchstone "D:\folder\ne133.s2p", 0, 1, 12, Array(), True, 23,<br/>"GHz"</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## ExportTouchstone2

Writes the S-parameters to disk in Touchstone 2 format (\*.ts text file).

|            |                                                                                                                                                                                                                  |                  |                                                                                                                                                                                                                                                   |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access  | From the <b>Network Data Explorer</b> tab, select <b>File &gt; Save As</b> to open a <b>Save As</b> explorer window. Then select <b>Touchstone 2 Format (*.ts)</b> from the <b>Save as type:</b> drop-down menu. |                  |                                                                                                                                                                                                                                                   |
| Parameters | <b>Name</b>                                                                                                                                                                                                      | <b>Type</b>      | <b>Description</b>                                                                                                                                                                                                                                |
|            | <filePath>                                                                                                                                                                                                       | String           | The full path to the file.                                                                                                                                                                                                                        |
|            | <matrixType>                                                                                                                                                                                                     | Integer          | One of the following: <ul style="list-style-type: none"><li>• 0 – S-Parameters</li><li>• 1 – Y-Parameters</li><li>• 2 – Z-Parameters</li><li>• 3 – Gamma</li><li>• 4 – Impedance</li></ul> <div><b>Note:</b> The default value is <b>0</b>.</div> |
|            | <formalType>                                                                                                                                                                                                     | Integer          | One of the following: <ul style="list-style-type: none"><li>• 0 – Magnitude/Angle (degrees)</li><li>• 1 – Real/Imaginary</li><li>• 2 – dB/Angle (degrees)</li></ul> <div><b>Note:</b> The default value is <b>0</b>.</div>                        |
|            | <precision>                                                                                                                                                                                                      | Integer          | The number of significant figures. <div><b>Note:</b> The default value is <b>10</b>.</div>                                                                                                                                                        |
|            | <frequencies>                                                                                                                                                                                                    | Array of Doubles | A subset of the calculated frequencies. Any array values that don't have calculated data will be ignored. An empty array will cause all frequencies to be written.                                                                                |

|                     |                                                            |         |                                                                                                                                                                        |
|---------------------|------------------------------------------------------------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                                                            |         | <b>Note:</b> The default value is an empty array.                                                                                                                      |
|                     | <code>&lt;renormalize&gt;</code>                           | Boolean | If <b>True</b> , will renormalize the data to the value of <code>renormImpedance</code> before writing.<br><br><b>Note:</b> The default value is <b>False</b> .        |
|                     | <code>&lt;renormImpedance&gt;</code>                       | Double  | Data will be renormalized to this impedance before writing only if <code>renormalize</code> is <b>True</b> .<br><br><b>Note:</b> The default value is <b>50 ohms</b> . |
|                     | <code>&lt;freqUnits&gt;</code>                             | String  | One of the following: "Hz", "KHz", "MHz", "GHz".<br><br><b>Note:</b> The default value is <b>"Hz"</b> .                                                                |
| <b>Return Value</b> | Boolean True if file was successfully written; else False. |         |                                                                                                                                                                        |

|                       |                                                                                                                                         |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>ExportTouchstone2()</code>                                                                                                        |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") success = oData.ExportTouchstone2("D:\folder\nel33.ts", 0, 1, 12, [], True, 23, "GHz")</pre> |

|                |                                |
|----------------|--------------------------------|
| <b>VB Syn-</b> | <code>ExportTouchstone2</code> |
|----------------|--------------------------------|

|                   |                                                                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>tax</b>        |                                                                                                                                                     |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set success = oData.ExportTouchstone2 "D:\folder\nel33.ts", 0, 1, 12, Array(), True, 23, "GHz"</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Extract (SPISim)

Creates a new smaller network data after removing data for the specified terminal port numbers.

**Note:** Extract opens and interacts with **SPISim** to complete.

|                     |                                                                                                                                                   |                   |                                                                                    |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Extract</b> from the drop-down menu. |                   |                                                                                    |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                       | <b>Type</b>       | <b>Description</b>                                                                 |
|                     | <oData>                                                                                                                                           | Object            | Data from this object is copied to a new network data and the copy is transformed. |
|                     | <portNumbers>                                                                                                                                     | Array of Integers | The port numbers that will be retained (integers between 1 and <i>n</i> ).         |
| <b>Return Value</b> | Network IDispatch.                                                                                                                                |                   |                                                                                    |

|                       |                                                  |
|-----------------------|--------------------------------------------------|
| <b>Python Syntax</b>  | Extract()                                        |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")</pre> |

|  |                                                    |
|--|----------------------------------------------------|
|  | <code>oData2 = oNDE.Extract(oData1, [3, 2])</code> |
|--|----------------------------------------------------|

|                   |                                                                                                                     |
|-------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Extract                                                                                                             |
| <b>VB Example</b> | <code>Set oNDE = oDesktop.GetTool "ndExplorer"</code><br><code>Set oData2 = oNDE.Extract oData1, Array(3, 2)</code> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## GetFrequencies

Returns the frequency values with calculated data in the network data.

|                     |                   |
|---------------------|-------------------|
| <b>UI Access</b>    | None.             |
| <b>Parameters</b>   | None.             |
| <b>Return Value</b> | Array of doubles. |

|                       |                                                                                                   |
|-----------------------|---------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetFrequencies()                                                                                  |
| <b>Python Example</b> | <code>oNDE = oDesktop.GetTool("ndExplorer")</code><br><code>Freqs = oData.GetFrequencies()</code> |



|                   |                                                                                           |
|-------------------|-------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetFrequencies                                                                            |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set Freqs = oData.GetFrequencies</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## GetFrequencyCount

Returns the number of frequencies with calculated data in the network data.

|                     |                                |
|---------------------|--------------------------------|
| <b>UI Access</b>    | None.                          |
| <b>Parameters</b>   | None.                          |
| <b>Return Value</b> | Integer number of frequencies. |

|                       |                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetFrequencyCount()                                                                        |
| <b>Python Example</b> | <pre>oData = oDesktop.GetTool("ndExplorer")<br/>numFreqs = oData.GetFrequencyCount()</pre> |

|                   |                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetFrequencyCount                                                                                |
| <b>VB Example</b> | <pre>Set oData = oDesktop.GetTool("ndExplorer")<br/>Set numFreqs = oData.GetFrequencyCount</pre> |

## GetName

Returns the design name of the active design, in that order separated by a semicolon.

|              |                                                  |
|--------------|--------------------------------------------------|
| UI Access    | N/A                                              |
| Parameters   | None.                                            |
| Return Value | String indicating the name of the active design. |

|                |                                              |
|----------------|----------------------------------------------|
| Python Syntax  | GetName()                                    |
| Python Example | <code>design_name = oDesign.GetName()</code> |

|            |                                            |
|------------|--------------------------------------------|
| VB Syntax  | GetName                                    |
| VB Example | <code>design_name = oDesign.GetName</code> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## GetPortCount

Returns the total number of ports.

|                     |                                                                                                                                         |
|---------------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Edit Ports</b> in the <b>NDE</b> ribbon to open the <b>Port properties</b> window. |
| <b>Parameters</b>   | None.                                                                                                                                   |
| <b>Return Value</b> | Integer number of ports.                                                                                                                |

|                       |                                                                             |
|-----------------------|-----------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetPortCount()                                                              |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") num = oData.GetPortCount()</pre> |

|                   |                                                                               |
|-------------------|-------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetPortCount                                                                  |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") num = oData.GetPortCount</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## GetPortNumber

Returns the terminal port number for a given terminal port name.

**Note:** GetPortNumber can be used in other commands that require a port number, if a port name is preferable.

|                  |                                                                                                                           |
|------------------|---------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b> | From the <b>Network Data Explorer</b> tab, ensure the <b>Full Port Names</b> check box is activated. Port numbers will be |
|------------------|---------------------------------------------------------------------------------------------------------------------------|

|              |                                      |             |                                     |
|--------------|--------------------------------------|-------------|-------------------------------------|
|              | represented in the object portNames. |             |                                     |
| Parameters   | <b>Name</b>                          | <b>Type</b> | <b>Description</b>                  |
|              | <portName>                           | String      | The designation of the port number. |
| Return Value | Integer.                             |             |                                     |

|                |                                                                                                 |  |  |
|----------------|-------------------------------------------------------------------------------------------------|--|--|
| Python Syntax  | GetPortNumber()                                                                                 |  |  |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>index = oPostProc.GetPortNumber("Input35")</pre> |  |  |

|            |                                                                                                    |  |  |
|------------|----------------------------------------------------------------------------------------------------|--|--|
| VB Syntax  | GetPortNumber                                                                                      |  |  |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>index = oPostProc.GetPortNumber "Input35"</pre> |  |  |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## GetPostProcSettings

Returns a copy of the IDispatch to the postprocessing settings for the network data (oPostProc).

**Note:** Making changes to the PostProcSettings will not change the network data. To make changes, call `oData.SetPostProcSettings` to change the network data.

|                     |                            |
|---------------------|----------------------------|
| <b>UI Access</b>    | None.                      |
| <b>Parameters</b>   | None.                      |
| <b>Return Value</b> | PostProcSettings IDispatch |

|                       |                                                                                              |
|-----------------------|----------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetPostProcSettings()</code>                                                           |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oPostProc = oData.GetPostProcSettings()</pre> |

|                   |                                                                                                    |
|-------------------|----------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetPostProcSettings</code>                                                                   |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set oPostProc = oData.GetPostProcSettings</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## GetVariation

Returns the variation key for the network data.

|                     |                       |
|---------------------|-----------------------|
| <b>UI Access</b>    | None.                 |
| <b>Parameters</b>   | None.                 |
| <b>Return Value</b> | String variation key. |

|                       |                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetVariation()                                                                        |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>variation = oData.GetVariation()</pre> |

|                   |                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetVariation                                                                                |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set variation = oData.GetVariation</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## HasSameData

Compares one network data object with another network data. Actual calculated values will be compared and no interpolation will be done.

|                  |       |
|------------------|-------|
| <b>UI Access</b> | None. |
|------------------|-------|

| Parameters | Name                | Type    | Description                                                                                                                                                                                                                                              |
|------------|---------------------|---------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|            | <NetworkData>       | Object  | The second network data to compare against.                                                                                                                                                                                                              |
|            | <matrixType>        | Integer | One of the following: <ul style="list-style-type: none"> <li>• 0 – S-Parameters</li> <li>• 1 – Y-Parameters</li> <li>• 2 – Z-Parameters</li> <li>• 3 – Gamma</li> <li>• 4 – Impedance</li> </ul> <p><b>Note:</b> The default value is <b>0</b>.</p>      |
|            | <comparePortNames>  | Boolean | If <b>True</b> , port names must be identical. <p><b>Note:</b> The default value is <b>True</b>.</p>                                                                                                                                                     |
|            | <compareNoise>      | Boolean | If <b>True</b> , and both network data have noise data, the comparison will fail unless the noise values also match. <p><b>Note:</b> The default value is <b>10</b>.</p>                                                                                 |
|            | <relativeTolerance> | Double  | If the absoluteTolerance test fails, then $\text{abs}(\text{value1} - \text{value2}) / \max(\text{abs}(\text{value1}), \text{abs}(\text{value2}))$ must be less than this to be considered equal. <p><b>Note:</b> The default value is <b>1e-14</b>.</p> |
|            | <absoluteTolerance> | Double  | If <b>True</b> , then $\text{abs}(\text{value1} - \text{value2}) < \text{absoluteTolerance}$ . <p><b>Note:</b> The default value is <b>0</b>.</p>                                                                                                        |

|              |                                                                                                        |
|--------------|--------------------------------------------------------------------------------------------------------|
| Return Value | Boolean True if the compared network data have the same frequency and matrix values; otherwise, False. |
|--------------|--------------------------------------------------------------------------------------------------------|

|                |                                                                                                                    |
|----------------|--------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | HasSameData()                                                                                                      |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>success = oData.HasSameData(oData2, 0, True, False, 1e-10, 0)</pre> |

|            |                                                                                                                           |
|------------|---------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | HasSameData                                                                                                               |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set success = oData.HasSameData oData2, 0, True, False, 1e-10, 0</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## LoadSolution

Loads the specified design (all variations) and returns an integer ID.

|            |                                                                                                                                                                                                                                                                              |             |                                            |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------|
| UI Access  | After analyzing a design, from the <b>Project Manager</b> window, expand the <b>Project Tree</b> and Analysis folders. Then right-click on the completed analysis icon and select <b>Network Data Explorer</b> to open the solution in the <b>Network Data Explorer</b> tab. |             |                                            |
| Parameters | <b>Name</b>                                                                                                                                                                                                                                                                  | <b>Type</b> | <b>Description</b>                         |
|            | <projectName>                                                                                                                                                                                                                                                                | String      | Full path to the Electronics Desktop file. |



|                     |                                                                           |        |                       |
|---------------------|---------------------------------------------------------------------------|--------|-----------------------|
|                     | <designName>                                                              | String | Name of the design.   |
|                     | <solutionName>                                                            | String | Name of the solution. |
| <b>Return Value</b> | Integer ID (identifying the solution); < 0 if the solution is not loaded. |        |                       |

|                       |                                                                                                                             |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | LoadSolution()                                                                                                              |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") id = oNDE.LoadSolution("D:\folder\Tee.aedt", "HFSS_Test", "Setup1:Sweep1")</pre> |

|                   |                                                                                                                                |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | LoadSolution                                                                                                                   |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") id = oNDE.LoadSolution "D:\folder\Tee.aedt", "HFSS_Test", "Setup1:Sweep1"</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Open

Reads contents of file and returns the IDispatch for the new network data.

|                   |                                                                                                                                                                                                     |             |                    |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------|
| <b>UI Access</b>  | From the <b>Network Data Explorer</b> tab, select <b>Open</b> in the <b>NDE</b> ribbon, or select <b>File &gt; Open</b> to open an explorer window. Then navigate to the required S-parameter file. |             |                    |
| <b>Parameters</b> | <b>Name</b>                                                                                                                                                                                         | <b>Type</b> | <b>Description</b> |

|              |                                                                                                                         |                                                |  |            |        |                                                |
|--------------|-------------------------------------------------------------------------------------------------------------------------|------------------------------------------------|--|------------|--------|------------------------------------------------|
|              | <table><tr><td>&lt;fileName&gt;</td><td>String</td><td>Full path to the file containing S-parameters.</td></tr></table> |                                                |  | <fileName> | String | Full path to the file containing S-parameters. |
| <fileName>   | String                                                                                                                  | Full path to the file containing S-parameters. |  |            |        |                                                |
| Return Value | Network IDispatch.                                                                                                      |                                                |  |            |        |                                                |

|                |                                                                                              |
|----------------|----------------------------------------------------------------------------------------------|
| Python Syntax  | Open()                                                                                       |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData = oNDE.Open("D:\folder\test.s2p")</pre> |

|            |                                                                                                     |
|------------|-----------------------------------------------------------------------------------------------------|
| VB Syntax  | Open                                                                                                |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set oData = oNDE.Open "D:\folder\test.s2p"</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Rename (SPISim)

Creates a new network data of the same size with the terminal ports renamed.

**Note:** Rename opens and interacts with **SPISim** to complete.

|           |                                                                                                                                  |
|-----------|----------------------------------------------------------------------------------------------------------------------------------|
| UI Access | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Rename</b> from the |
|-----------|----------------------------------------------------------------------------------------------------------------------------------|

|              |                    |                  |                                                                                    |
|--------------|--------------------|------------------|------------------------------------------------------------------------------------|
|              | drop-down menu.    |                  |                                                                                    |
| Parameters   | <b>Name</b>        | <b>Type</b>      | <b>Description</b>                                                                 |
|              | <oData>            | Object           | Data from this object is copied to a new network data and the copy is transformed. |
|              | <portNames>        | Array of Strings | The new names (one for each port).                                                 |
| Return Value | Network IDispatch. |                  |                                                                                    |

|                |                                                                                                               |
|----------------|---------------------------------------------------------------------------------------------------------------|
| Python Syntax  | Rename()                                                                                                      |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Rename(oData1, ["input", "control", "output"])</pre> |

|            |                                                                                                                          |
|------------|--------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | Rename                                                                                                                   |
| VB Example | <pre>Set oNDE = oDesktop.GetTool "ndExplorer" Set oData2 = oNDE.Rename oData1, Array("input", "control", "output")</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Renormalize (SPISim)

Creates a new network data of the same size after renormalizing the terminal ports using the specified impedance values.

**Note:** Rename opens and interacts with **SPISim** to complete.

|              |                                                                                                                                                       |                  |                                                                                    |
|--------------|-------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|------------------------------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Renormalize</b> from the drop-down menu. |                  |                                                                                    |
| Parameters   | <b>Name</b>                                                                                                                                           | <b>Type</b>      | <b>Description</b>                                                                 |
|              | <oData>                                                                                                                                               | Object           | Data from this object is copied to a new network data and the copy is transformed. |
|              | <portImpedances>                                                                                                                                      | Array of Doubles | The new impedance values to be applied to each port.                               |
| Return Value | Network IDispatch.<br><br><b>Note:</b> Renormalize returns <b>None</b> if there is not an impedance value for each port.                              |                  |                                                                                    |

|                |                                                                                                |
|----------------|------------------------------------------------------------------------------------------------|
| Python Syntax  | Renormalize()                                                                                  |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData2 = oNDE.Renormalize(oData1, [3, 2])</pre> |

|            |                                                                                                        |
|------------|--------------------------------------------------------------------------------------------------------|
| VB Syntax  | Renormalize                                                                                            |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>oData2 = oNDE.Renormalize oData1, Array(3, 2)</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Reorder

Rearranges the terminal port entries but does not change the port names (i.e., switching first and third terminal port positions will give switched values for S(1,1) and S(3,3) but S(Port1,Port1) and S(Port3,Port3) do not change).

|                     |                                                                                                                                                                                                                                                      |                   |                                                                          |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|--------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Edit Ports</b> in the <b>NDE</b> ribbon to open the <b>Port properties</b> window. <b>Click+drag</b> the terminal port rows to reorder them. Make changes, as necessary, then click <b>OK</b> . |                   |                                                                          |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                          | <b>Type</b>       | <b>Description</b>                                                       |
|                     | <newOrder>                                                                                                                                                                                                                                           | Array of Integers | Must have one entry for each port; all numbers 1 to $n$ must be present. |
| <b>Return Value</b> | None.                                                                                                                                                                                                                                                |                   |                                                                          |

|                       |                                                                               |
|-----------------------|-------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Reorder()                                                                     |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oPostProc.Reorder([3, 2, 1])</pre> |

|                   |                                                                                       |
|-------------------|---------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Reorder                                                                               |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") oPostProc.Reorder Array(3, 2, 1)</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

Reorder (SPISim)

Creates a new network data with the same number of terminal ports and with the port names in the same order. The data is also reordered so it corresponds to the new order (e.g., the data for S(Port1, Port1) may correspond to S(Port3, Port3).

**Note:** Reorder opens and interacts with **SPISim** to complete.

|               |                                                                                                                                                        |                   |                                                                                    |
|---------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|------------------------------------------------------------------------------------|
| UI Access     | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Reorder</b> from the drop-down menu.      |                   |                                                                                    |
| Parameters    | <b>Name</b>                                                                                                                                            | <b>Type</b>       | <b>Description</b>                                                                 |
|               | <oData>                                                                                                                                                | Object            | Data from this object is copied to a new network data and the copy is transformed. |
|               | <portNumbers>                                                                                                                                          | Array of Integers | The port numbers in new order (integers between 1 and <i>n</i> ).                  |
| Return Value  | Network IDispatch.<br><br><b>Note:</b> Reorder returns <b>None</b> if the array argument does not contain all the terminal port numbers, in any order. |                   |                                                                                    |
| Python Syntax | Reorder()                                                                                                                                              |                   |                                                                                    |

|                       |                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData2 = oNDE.Reorder([3, 2, 1])</pre> |
|-----------------------|---------------------------------------------------------------------------------------|

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Reorder                                                                                           |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set oData2 = oNDE.Reorder Array(3, 2, 1)</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Reset

Resets the post-processing to what it was when the network data was created.

**Note:** This is not equivalent to setting post-processing to an empty state. The network data may have been read from a file or created from solution data that already had post-processing settings.

|                     |                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Reset postprocessing</b> in the <b>NDE</b> ribbon. |
| <b>Parameters</b>   | None.                                                                                                   |
| <b>Return Value</b> | None.                                                                                                   |

|                       |                                                  |
|-----------------------|--------------------------------------------------|
| <b>Python Syntax</b>  | Reset()                                          |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer")</pre> |

|  |                                |
|--|--------------------------------|
|  | <code>oPostProc.Reset()</code> |
|--|--------------------------------|

|                   |                                                                          |
|-------------------|--------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Reset                                                                    |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>oPostProc.Reset</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## SetAllPortImpedances

Sets all terminal port impedances in a single call. The impedances are in an array of real or complex values.

|                     |                                                                                                                                                                                                                                                         |                            |                                                                                                           |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------|-----------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Through the UI, the terminal ports can <b>only</b> be edited individually. From the <b>Network Data Explorer</b> tab, select <b>Edit Ports</b> in the <b>NDE</b> ribbon to open the <b>Port properties</b> window. Make changes, then click <b>OK</b> . |                            |                                                                                                           |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                             | <b>Type</b>                | <b>Description</b>                                                                                        |
|                     | <impedances>                                                                                                                                                                                                                                            | Array of Doubles or String | Must have one entry for each port or a single complex (or real) value which will be applied to all ports. |
| <b>Return Value</b> | None.                                                                                                                                                                                                                                                   |                            |                                                                                                           |

|                      |                        |
|----------------------|------------------------|
| <b>Python Syntax</b> | SetAllPortImpedances() |
|----------------------|------------------------|



|                       |                                                                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oPostProc.SetAllPortImpedances([23, "2+3i", 50]) -or- oData.SetAllPortImpedances(50)</pre> |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetAllPortImpedances                                                                                                                        |
| <b>VB Example</b> | <pre>Set oData=oDesktop.GetTool("ndExplorer") oPostProc.SetAllPortImpedances Array(23, "2+3i", 50) -or- oData.SetAllPortImpedances 50</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## SetPortDeembedDistance

Sets terminal port impedance for a single port.

|                     |                                                                                                                                                                              |             |                                                                           |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|---------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Edit Ports</b> in the <b>NDE</b> ribbon to open the <b>Port properties</b> window. Make changes, then click <b>OK</b> . |             |                                                                           |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                  | <b>Type</b> | <b>Description</b>                                                        |
|                     | <portNumber>                                                                                                                                                                 | Integer     | The port number of the changed terminal port (integers between 1 and n) . |
|                     | <distance>                                                                                                                                                                   | String      | Impedance with units (e.g., "20mm"); if no units, meters are assumed      |
| <b>Return Value</b> | None.                                                                                                                                                                        |             |                                                                           |

|                       |                                                                                              |
|-----------------------|----------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SetPortDeembedDistance()                                                                     |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oPostProc.SetPortDeembedDistance(2, "20mm")</pre> |

|                   |                                                                                                 |
|-------------------|-------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetPortDeembedDistance                                                                          |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") oPostProc.SetPortDeembedDistance 2, "20mm"</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## SetPortImpedance

Sets port impedance for a single port.

|                     |                                                                                                                                                                              |                  |                                                                                        |
|---------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|----------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Edit Ports</b> in the <b>NDE</b> ribbon to open the <b>Port properties</b> window. Make changes, then click <b>OK</b> . |                  |                                                                                        |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                  | <b>Type</b>      | <b>Description</b>                                                                     |
|                     | <portNumber>                                                                                                                                                                 | Integer          | The port number of the changed terminal port (integers between 1 and <i>n</i> ).       |
|                     | <impedance>                                                                                                                                                                  | Double or String | Double if impedance value is real; string format (e.g., 24+10i) for complex impedance. |
| <b>Return Value</b> | None.                                                                                                                                                                        |                  |                                                                                        |

|                       |                                                                                         |
|-----------------------|-----------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SetPortImpedance()                                                                      |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oPostProc.SetPortImpedance(2, "25+3i")</pre> |

|                   |                                                                                            |
|-------------------|--------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetPortImpedance                                                                           |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") oPostProc.SetPortImpedance 2, "25+3i"</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## SetPostProcSettings

Applies postprocessing settings (oPostProc) to the specified network data.

**Note:** Making changes to the PostProcSettings will not change the network data. To make changes, call oData.SetPostProcSettings to change the network data.

|                     |                    |             |                                                |
|---------------------|--------------------|-------------|------------------------------------------------|
| <b>UI Access</b>    | None.              |             |                                                |
| <b>Parameters</b>   | <b>Name</b>        | <b>Type</b> | <b>Description</b>                             |
|                     | <PostProcSettings> | Object      | The settings that will be applied to the data. |
| <b>Return Value</b> | None.              |             |                                                |

|                |                                                                                           |
|----------------|-------------------------------------------------------------------------------------------|
| Python Syntax  | SetPostProcSettings()                                                                     |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer")<br/>oData.SetPostProcSettings(oPostProc)</pre> |

|            |                                                                                              |
|------------|----------------------------------------------------------------------------------------------|
| VB Syntax  | SetPostProcSettings                                                                          |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>oData.SetPostProcSettings oPostProc</pre> |

**Warning: The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.**

## Smooth

Creates a new network data with values smoothed. The number of adjacent points smoothed is user-specified.

|            |                                                                                                                                                  |             |                                                                                    |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------------------------------------------------------------|
| UI Access  | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Smooth</b> from the drop-down menu. |             |                                                                                    |
| Parameters | <b>Name</b>                                                                                                                                      | <b>Type</b> | <b>Description</b>                                                                 |
|            | <oData>                                                                                                                                          | Object      | Data from this object is copied to a new network data and the copy is transformed. |
|            | <order>                                                                                                                                          | Integer     | The number of adjacent points that are smoothed.                                   |

|                     |                                       |         |                                                  |
|---------------------|---------------------------------------|---------|--------------------------------------------------|
|                     |                                       |         | <b>Note:</b> The default value is <b>10</b> .    |
|                     | <code>&lt;enforceCausality&gt;</code> | Boolean | Casual data is calculated before smoothing.      |
|                     |                                       |         | <b>Note:</b> The default value is <b>False</b> . |
| <b>Return Value</b> | Network IDispatch.                    |         |                                                  |

|                       |                                                                                        |
|-----------------------|----------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Smooth()                                                                               |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Smooth(oData1, 3, True)</pre> |

|                   |                                                                                               |
|-------------------|-----------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Smooth                                                                                        |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set oData2 = oNDE.Smooth oData1, 3, True</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Stretch (SPISim)

Creates a new network data representing a matrix of transmission lines with the length of those transmission lines multiplied by a factor of  $n$ .

**Note:** Stretch opens and interacts with **SPISim** to complete.

|              |                                                                                                                                                        |             |                                                                                    |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|------------------------------------------------------------------------------------|
| UI Access    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Stretch</b> from the drop-down menu.      |             |                                                                                    |
| Parameters   | <b>Name</b>                                                                                                                                            | <b>Type</b> | <b>Description</b>                                                                 |
|              | <oData>                                                                                                                                                | Object      | Data from this object is copied to a new network data and the copy is transformed. |
|              | <factor>                                                                                                                                               | Double      | The transmission line lengths are multiplied by this number.                       |
| Return Value | Network IDispatch. <div><b>Note:</b> Stretch returns <b>None</b> if the network data argument does not represent a matrix of transmission lines.</div> |             |                                                                                    |

|                |                                                                                     |
|----------------|-------------------------------------------------------------------------------------|
| Python Syntax  | Stretch()                                                                           |
| Python Example | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Stretch(oData1, 3.1)</pre> |

|            |                                                                                            |
|------------|--------------------------------------------------------------------------------------------|
| VB Syntax  | Stretch                                                                                    |
| VB Example | <pre>Set oNDE = oDesktop.GetTool("ndExplorer") Set oData2 = oNDE.Stretch oData1, 3.1</pre> |

**Warning:** The following command is preliminary. Its syntax may change. Be prepared to make changes to legacy scripts.

## Terminate

Creates a new network data with specified ports terminated.

|                     |                                                                                                                                                              |                          |                                                                                    |
|---------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|------------------------------------------------------------------------------------|
| <b>UI Access</b>    | From the <b>Network Data Explorer</b> tab, select <b>Transforms</b> in the <b>NDE</b> ribbon. Then select <b>Terminate</b> from the drop-down menu.          |                          |                                                                                    |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                  | <b>Type</b>              | <b>Description</b>                                                                 |
|                     | <oData>                                                                                                                                                      | Object                   | Data from this object is copied to a new network data and the copy is transformed. |
|                     | <portNumbers>                                                                                                                                                | Array of Integers        | The port numbers that are terminated (integers between 1 and $n$ ).                |
|                     | <termImpedances>                                                                                                                                             | Array of Complex Numbers | The impedance values used to terminate the specified ports.                        |
| <b>Return Value</b> | Network IDispatch.<br><br><b>Note:</b> Terminate returns <b>None</b> if the port numbers are not valid or there are no impedance value for the port numbers. |                          |                                                                                    |

|                       |                                                                                                         |
|-----------------------|---------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Terminate()                                                                                             |
| <b>Python Example</b> | <pre>oNDE = oDesktop.GetTool("ndExplorer") oData2 = oNDE.Terminate(oData1, [2, 3], [23, "10+2i"])</pre> |

|                   |                                                                                                                              |
|-------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Terminate                                                                                                                    |
| <b>VB Example</b> | <pre>Set oNDE = oDesktop.GetTool("ndExplorer")<br/>Set oData2 = oNDE.Terminate oData1, Array(2, 3), Array(23, "10+2i")</pre> |



## 25 - CompInstance Script Commands

CompInstance commands should be executed by the **oDesign2** or **oPropHost2** object.

```
Set oDesign2 = CompInstance.GetParentDesign
```

```
Set oPropHost2 = CompInstance.GetPropHost
```

### Callback Scripting Using CompInstance Object

Callback scripts are scripts that can be set in the Property Dialog for individual properties by clicking the button in the Callback column and choosing a script that is saved with the project. Callback scripts can contain any legal script commands including general Ansys script function calls ( e.g., `GetApplicationName()` ). In addition, Callback scripts can also call functions on a special object named `CompInstance`.

You can obtain an interface to a `CompInstance` in a schematic or layout by calling `oEditor.GetCompInstanceFromRefDes(refDes)`. For more information see Layout Scripting and [Schematic Scripting](#). This interface is also available as a `CompInstance` object in [CompInstance event callbacks](#), such as placing a component in a layout or schematic.

#### Definitions

<propName> = text string

<value> = double

<valueText> = text string

<fileName> = full path file name

<choices> = string containing menu choices separated by commas

<initialChoice> = string containing initial choice for menu; must be one of the <choices>

<scriptName> = string containing name of script stored in project

<bool> is 1 for true or 0 for false

<editorName> is either "Layout" or "SchematicEditor"

The topics for this section include:

[ComplInstance Functions](#)

## ComplInstance Functions

Following are commands that can be used to manipulate properties from a ComplInstance script.

The topics for this section include:

[GetComponentName](#)

GetEditor

[GetInstanceID](#)

[GetInstanceName](#)

[GetParentDesign](#)

[GetPropHost](#)

[GetPropServerName](#)

### GetComponentName

Returns the name of the component corresponding to this ComplInstance.

|            |      |      |             |
|------------|------|------|-------------|
| UI Access  | NA   |      |             |
| Parameters | Name | Type | Description |

|                     |                                                                   |
|---------------------|-------------------------------------------------------------------|
|                     | None                                                              |
| <b>Return Value</b> | String<br>name of component (e.g. MS_TRL) and stores it in "name" |

|                       |                                                     |
|-----------------------|-----------------------------------------------------|
| <b>Python Syntax</b>  | GetComponentName()                                  |
| <b>Python Example</b> | <code>name = CompInstance.GetComponentName()</code> |

|                   |                                                     |
|-------------------|-----------------------------------------------------|
| <b>VB Syntax</b>  | GetComponentName()                                  |
| <b>VB Example</b> | <code>name = CompInstance.GetComponentName()</code> |

### GetInstanceID [Component Instance]

*Use:* Returns the instanceID of the CompInstance.

*Command:* None

*Syntax:* GetInstanceID()

*Return Value:* String

*VB Example:* `id = CompInstance.GetInstanceID();`

Returns id of compInstance (e.g. 7) and stores it in "id".

Note that this is not the same number as the one used in the RefDes.

|                      |                 |
|----------------------|-----------------|
| <b>Python Syntax</b> | GetInstanceID() |
|----------------------|-----------------|

|                       |                                                |
|-----------------------|------------------------------------------------|
| <b>Python Example</b> | <code>id = CompInstance.GetInstanceID()</code> |
|-----------------------|------------------------------------------------|

## **GetInstanceName [Component Instance]**

*Use:* Returns the instance name of the component corresponding to this CompInstance.

*Command:* None

*Syntax:* GetInstanceName()

*Return Value:* String

*VB Example:* `name = CompInstance.GetInstanceName()` ;

Returns instanceName (e.g. A7) of compInstance and stores it in "name".

Note that the Instance Name is not the same as the RefDes.

## **GetParentDesign**

*Use:* Returns an interface to the compInstance's parent design.

*Command:* None

*Syntax:* GetParentDesign()

*Return Value:* Returns interface to design.

*Example:* `Set oDesign2 = CompInstance.GetParentDesign()` ;

Returns the interface to the design containing the compInstance.

This interface can be used to call Design functions. See: [Design Object Script Commands](#).

|                       |                                                        |
|-----------------------|--------------------------------------------------------|
| <b>Python Syntax</b>  | GetParentDesign()                                      |
| <b>Python Example</b> | <code>oDesign2 = CompInstance.GetParentDesign()</code> |

## GetPropHost

*Use:* Returns an interface to the PropHost of the CompInstance, which gives access to its properties.

*Command:* None

*Syntax:* GetPropHost()

*VB Example:* `Set oPropHost2 = CompInstance.GetPropHost()`;

Returns the interface to the properties of the compInstance.

This interface can be used to call PropHost functions; for more information see Callback Scripting Using PropHost Object.

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | GetPropHost()                                        |
| <b>Python Example</b> | <code>oPropHost2 = CompInstance.GetPropHost()</code> |

## GetPropServerName

*Use:* Returns the PropServerName of the Component corresponding to this CompInstance.

*Command:* None

*Syntax:* GetPropServerName()

*Return Value:* String

*VB Example:*

`name = CompInstance.GetPropServerName()`;

Returns propserver name of compInstance (e.g., CompInst@MS\_TRL;7) and stores it in "name".

|                       |                                                      |
|-----------------------|------------------------------------------------------|
| <b>Python Syntax</b>  | GetPropServerName()                                  |
| <b>Python Example</b> | <code>name = CompInstance.GetPropServerName()</code> |

## 26 - Schematic Scripting

The Schematic scripting interface is a set of commands that match the data changing methods available in the UI of the Schematic, plus some selection and query methods. Examples of the commands available through the scripting interface are adding items, removing items, and modifying items on the Schematic. Identifying objects on the Schematic is done by a unique ID that is generated and returned by all methods that add objects, and by the FindElements and GetSelections methods. This ID can then be passed into commands to modify or remove Schematic objects.

- Since most Schematic objects can have sub-components (such as property displays or segments on a wire, etc), IDs can also be in the format of "TopLevelID:SubComponentIndex".
- The SubComponentIndex is a one-based index into the sub components of the item thus making the second segment of component CAP1 identified by "CAP1:2".
- A SubComponentIndex of zero will refer to the TopLevel item only.
- When an ID is mentioned later in this document, it will refer to either a simple string ID representing a top-level Schematic item or a "TopLevelID:SubComponentIndex" pair unless otherwise stated.
- If a specified ID or SubComponentIndex does not exist, then the function will ignore that entry and proceed with other specified IDs. Even if no valid IDs are specified, the functions will still return success.

**The topics for this section include:**

- [Method Format](#)
- [Editor Scripting IDs](#)
- [Create Method List](#)
- [General Method List](#)
- [Property Method List](#)
- [Information Method List](#)

### Method Format

In the following formats, [Opt=x] appearing after a parameter indicates that the parameter is optional and the default value is x.

When calling object-creation methods as functions, parenthesis are required in order to retrieve the name of the created object. Parenthesis are also required in JavaScript. When using VB/VBScript to call a method as a subroutine, however, parenthesis are not required.

All methods to create Schematic objects have the following form:

```
Create[type] (VARIANT parameters,
// Array of type specific parameters
VARIANT attributes,
// attributes is in the format of:
Array("NAME:Attributes", _
"Page:=", int, _ // [Opt=1] Page number (one-based)
// Note: Page 1 always exists
"X:=", double, _ // X position of the object
"Y:=", double, _ // Y position of the object
"Angle:=", double, _ // Rotation angle (radians)
"Flip:=", bool) // True if mirrored
[out,retval] string id)
```

The algorithm used in the create methods is:

- 1) Create SchAdd[type]Command object with parameters passed in
- 2) Execute the command

All methods to modify Schematic objects have the following form:



```
[modification] (Array("NAME:Selections",
"Page:=", Page number [optional: Default=1]
"Selections:=", IDs to modify see format below),
VARIANT params
// Array of modification specific parameters)
```

The algorithms in these types of methods will be:

- 1) Go through ids and get the SelectableObjs they correspond to
- 2) Select the SelectableObjs found in (1)
- 3) Select any sub components specified
- 4) Create the Sch[*modification*]Command
- 5) Execute the command

### ID Format

When IDs are passed to a function it can be in one of the following formats:

- Array of integers of top-level items only
- String of IDs separated by spaces or commas
- Array of strings with each string element being an ID

### Point Format

When a method takes an array of points, each element in the array is a string in the format of: "x y", "x,y", or "(x,y)".

## Editor Scripting IDs

Objects in the schematic are identified by text IDs. These IDs are used in scripts to perform actions on objects. These IDs are returned by all methods that add objects, and by the FindElements and GetSelections methods. The IDs are then passed into commands to modify or remove Schematic objects.

### Format of IDs for different schematic objects:

**Components:** CompInst@<CompName>;<compInstID>;<optional schematicID>

**Geometric primitive:** SchObj@<schematicID>

**Global Port/ground ::** GPort@<portName>;<schematicID>

**Interface Port ::** IPort@<portName>;<schematicID>

**Page Port:** PagePort@<portName>;<schematicID>

**Wire :** Wire@<netName>;<schematicID>;<segment index list>

### Find this information in the Properties Window for the selected object as follows:

<CompName> General tab/CompName

<compInstID> General tab/ID

<schematicID> Symbol tab/SchematicID

<portName> Param Values tab/PortName

<netName> General tab/NetName

<segment index list> numbers separated by commas; Symbol tab/Segmentn - e.g. Segment0 refers to index "0"

In the Layout, the ID for a selected object is always the Name or Net property in the Footprint tab of the Properties Window

### Format for Components

```
Array("CompInst@CAP_ ; 2 ; 4")
```

The format for Components is `CompInst@<CompName>;<compInstID>;<optional schematicID>`. In the documented example, `<CompName>` stands for the component name `CAP_`, `<compInstID>` stands for its ID 2, and `<optional schematicID>` is the schematic ID 4. If you select the component, the **Properties** window gets updated and displays its details. For example, the selected object's component name and ID appear in the **General** tab of the **Properties** window as shown below.

| Properties                  |                                                  |
|-----------------------------|--------------------------------------------------|
| Name                        | Value                                            |
| ID                          | 2                                                |
| CompName                    | CAP_                                             |
| Description                 | Capacitor                                        |
| Manufacturer                |                                                  |
| Datasource                  | Ansoft built-in component                        |
| Date                        | 15:30:47 11/14/2005                              |
| PinCount                    | 2                                                |
| Refdes                      | C1                                               |
| Symbol                      | nexx_cap                                         |
| Footprint                   | __symbolFootprint_nexx_cap__positive__negative_0 |
| Param Values General Symbol |                                                  |

The SchematicID is shown in the **Symbol** tab.

## Create Method List

This section lists the scripts that are available in the Schematic scripting interface to create and change data.

### Script Position Parameters

In Create Method scripts, X and Y position parameters are expressed in meters.

- If your layout grid units are metric, you may enter the X and Y positions directly into a script as parameters.
- If your layout grid is expressed in mils, you must first convert the component's positional coordinates to meters.

To convert an X or Y position to meters based on the minor grid size:

$$\text{X or Y position in meters} = \text{desired\_position} \times (0.00254 / \text{minor\_grid\_size})$$

where desired\_position and minor\_grid\_size have the same units. For example, to position an object at 500mm with a minor grid size of 20mm:

$$\text{X or Y position in meters} = 500 \times (0.00254 / 20)$$

This section lists the following commands:

[CreateArc \(Schematic Editor\)](#)

[CreateCircle \(Schematic Editor\)](#)

[CreateComponent \(Schematic Editor\)](#)

[CreateCurve \(Schematic Editor\)](#)

[CreateGlobalPort \(Schematic Editor\)](#)

[CreateGround \(Schematic Editor\)](#)

[CreateLine \(Schematic Editor\)](#)

[CreatePagePort \(Schematic Editor\)](#)

[CreateIPort \(Schematic Editor\)](#)

[CreatePolygon \(Schematic Editor\)](#)

[CreateRectangle \(Schematic Editor\)](#)

[CreateText \(Schematic Editor\)](#)

[CreateWire \(Schematic Editor\)](#)

## CreateArc (Schematic Editor)

Create an arc

| UI Access    | Draw>Primitive>Arc  |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------|---------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|              | <ArcData>           | Array | <pre>Array("NAME:ArcData" _       "x:=", double, _ // X position of the object       "y:=", double, _ // Y position of the object       "Radius:=", double, _ // Radius of the circle       "StartAng:=", double, _ // Start angle of the arc (radians)       "EndAng:=", double, _ // End angle of the arc (radians)       "LineWidth:=", double, // the width of the line, in meters       "Color:=", int, // the RBG value of the arc color       "Id:=", int), // [Opt=New id] Id for this item</pre> |
|              | <Attributes>        | Array | <pre>Array("NAME:Attributes", _       "Page:=", int) _ // [Opt=1] Page number (one-based) [out,retval] string id)</pre>                                                                                                                                                                                                                                                                                                                                                                                   |
| Return Value | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |

Arc is created with the format SchObj@<schematicID>

The Schematic ID can be found on the **Symbols** tab of the **Properties** window when you select the arc.

| Properties  |                 |      |
|-------------|-----------------|------|
| Name        | Value           | Unit |
| SchematicID | 48              |      |
| Color       |                 |      |
| Linewidth   | 0               | mil  |
| Fill Style  | Hollow          |      |
| Center      | 4100 , 3300     | mil  |
| Radius      | 360.5551275464  | mil  |
| StartAngle  | 123.69006752598 | deg  |
| EndAngle    | 315             | deg  |
| ◀           |                 |      |
| Symbol      |                 |      |

| Python Syntax  | CreateArc()                                                                                                              |
|----------------|--------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateArc (["NAME:ArcData", _ "X:=", -0.004318, "Y:=", -0.00127, _ "Radius:=", 0.00297299377732279, _</pre> |

|  |                                                                                                                                          |
|--|------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"StartAng:=", 1.9195673303788, _<br/>"EndAng:=", 3.32144615338227, _<br/>"Id:=", 10], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |
|--|------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax         | CreateArc()                                                                                                                                                                                                                                                                    |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateArc<br/>Array("NAME:ArcData", _<br/>"X:=", -0.004318, "Y:=", -0.00127, _<br/>"Radius:=", 0.00297299377732279, _<br/>"StartAng:=", 1.9195673303788, _<br/>"EndAng:=", 3.32144615338227, _<br/>"Id:=", 10), _<br/>Array("NAME:Attributes", "Page:=", 1)</pre> |

**VB Example:**

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
```

```
Dim oEditor
Dim oModule
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
Dim ArcID
ArcID = oEditor.CreateArc(Array("NAME:ArcData", "X=", 0.10414, "Y=", 0.08382, "Radius=", _
0.00915810023967853, "StartAng=", 2.15879893034246, "EndAng=", _
5.49778714378214, "LineWidth=", 0, "Color=", 0, "Id=", 48), Array("NAME:Attributes", "Page=", _
1))
MsgBox "Arc ID = " & ArcID
```

**CreateCircle (Schematic Editor)**

Create a circle

| UI Access  | Draw>Primitive>Circle |       |                           |
|------------|-----------------------|-------|---------------------------|
| Parameters | Name                  | Type  | Description               |
|            | <CircleData>          | Array | Array("NAME:CircleData" _ |



|                     |                     |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|---------------------|---------------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                     |       | "x:=", double, _ // X position of the object<br>"y:=", double, _ // Y position of the object<br>"Radius:=", double, _ // Radius of the circle<br>"LineWidth:=", double, // the width of the circle border, in meters<br>"Bordercolor:=", int, // the RGB value of the border color<br>"Fill:=", int, // the fill pattern id, 0 = hollow, 1 = solid, 2 = NEDiagonal,<br>3 = OrthoCross, 4 = DiagCross, 5 = NWDiagonal, 6 = Horizontal, 7 = Vertical<br>"Color:=", int, // the RGB value of the circle fill color<br>"Id:=", int), // [Opt=New id] Id for this item |
|                     | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int) _ // [Opt=1] Page number (one-based)<br>[out,retval] string id)                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
| <b>Return Value</b> | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |

Circle is created and the returned value has the format SchObj@<schematicID>

The schematic ID of the circle can be located in the **Properties** window on the **Symbols** tab.

| Properties  |             |      |
|-------------|-------------|------|
| Name        | Value       | Unit |
| SchematicID | 7           |      |
| Color       |             |      |
| BorderWidth | 0           | mil  |
| BorderColor |             |      |
| Fill Style  | Hollow      |      |
| Center      | 3800 , 2400 | mil  |
|             |             |      |
| Symbol      |             |      |

| Python Syntax  | CreateCircle()                                                                                                                                                                 |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateCircle(["NAME:CircleData", _     "X:=", -0.004572, "Y:=", -0.000508, _     "Radius:=", 0.001778, "Id:=", 12], _     ["NAME:Attributes", "Page:=", 1])</pre> |

| VB Syntax | CreateCircle() |
|-----------|----------------|
|-----------|----------------|

|                   |                                                                                                                                                                             |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateCircle Array("NAME:CircleData", _ "X:=", -0.004572, "Y:=", -0.000508, _ "Radius:=", 0.001778, "Id:=", 12), _ Array("NAME:Attributes", "Page:=", 1)</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## CreateComponent (Schematic Editor)

Creates a component of a given type.

| UI Access         | N/A.         |       |                                                                                                                                                                                                                                                                                                                                                                               |
|-------------------|--------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | Name         | Type  | Description                                                                                                                                                                                                                                                                                                                                                                   |
|                   | <params>     | Array | Structured array.<br><pre>Array("NAME:ComponentProps",       "Name:=", &lt;String path to component in library&gt;,       "Id:=", &lt;String Component ID&gt;)</pre>                                                                                                                                                                                                          |
|                   | <attributes> | Array | Structured array.<br><pre>Array("NAME:Attributes",       "Page:=", &lt;Integer: page number&gt;,</pre> <div style="background-color: #f0f0f0; padding: 10px; margin: 10px 0;"> <p><b>Note:</b><br/>For the page number, page 1 always exists.</p> </div> <pre>      "X:=", &lt;Double: X position of object&gt;,       "Y:=", &lt;Double: Y position of the object&gt;,</pre> |

|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                                                                                                                                                                                                                                                                                                                                                                                                                                                   |  | <pre>"Angle:=", &lt;Double: rotation angle in degrees&gt;, "Flip:=", &lt;boolean: True if mirrored; else False&gt;, "PinNumber:=", &lt;Integer: index of pin in symbol 0 to n-1&gt;</pre> <div><b>Note:</b><br/><br/>PinNumber is optional and is set to -1 by default. PinNumber affects where the symbol is placed relative to the X and Y parameters. If PinNumber is set to -1, the symbol is placed with its center at (X,Y). If it is a pin index, the symbol is placed so that the pin is located at (X,Y). The PinNumber is not recorded.</div> |
| <b>Return Value</b> | <p>String containing unique ID in the format:</p> <pre>CompInst@&lt;CompName&gt;;&lt;compInstID&gt;;&lt;optional schematicID&gt;</pre> <p>For example, the return value <code>CompInst@CAP;4;35</code> indicates that the component name is <code>CAP</code>, its ID is <code>4</code>, and its Schematic ID is <code>35</code>.</p> <p>You can also see these details in the <b>Properties</b> window when you select the created component.</p> |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

|                       |                                                                                                                                 |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>CreateComponent(&lt;params&gt;, &lt;attributes&gt;)</code>                                                                |
| <b>Python Example</b> | <pre>oEditor.CreateComponent(     ["NAME:ComponentProps",      "Name:=", "Maxwell Circuit Elements\Passive Elements:Res",</pre> |

|  |                                                                                                                                                                                             |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>         "Id:=", "1"     ],     ["NAME:Attributes",         "Page:=", 1,         "X:=", -0.01524,         "Y:=", 0.00508,         "Angle:=", 0,         "Flip:=", false     ] ) </pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | CreateComponent <params>, <attributes>                                                                                                                                                                                                                                                                                                                                                                                                  |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> Set oProject = oDesktop.SetActiveProject("Project1") Set oDesign = oProject.SetActiveDesign("Circuit1") Set oEditor = oDesign.SetActiveEditor("SchematicEditor")  Dim CompID  CompID = oEditor.CreateComponent Array("NAME:ComponentProps", "Name:=", "Maxwell Circuit Elements\Passive Elements:Res", "Id:=", "1"), Array("NAME:Attributes", "Page:=", 1, "X:=", -0.01524, "Y:=", 0.00508, "Angle:=", 0, "Flip:=", false) </pre> |

## CreateGlobalPort (Schematic Editor)

Create a global port and enable placement on the schematic

| UI Access    | Draw>Global Port                                                                                              |         |                                                                                                                                               |
|--------------|---------------------------------------------------------------------------------------------------------------|---------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                                                                          | Type    | Description                                                                                                                                   |
|              | <portname>                                                                                                    | String  | Name for this port - leave empty to use default unique name                                                                                   |
|              | <id>                                                                                                          | Integer | ID number for the port - leave empty to use default (recommended)                                                                             |
|              | <pagenumber>                                                                                                  | Integer | The (1-based) page number of the schematic page this port should be added to - leave empty to use default (1)                                 |
|              | <xval>                                                                                                        | Double  | X position of the object                                                                                                                      |
|              | <yval>                                                                                                        | Double  | Y position of the object                                                                                                                      |
|              | <angle>                                                                                                       | Double  | Rotation of the object, in radians. Rotations of multiples of 90 degrees are valid. Other values will be adjusted to the nearest valid angle. |
|              | <degrees>                                                                                                     | Double  | Rotation of the object, in degrees. Rotations of multiples of 90 degrees are valid. Other values will be adjusted to the nearest valid angle  |
|              | <flip>                                                                                                        | Boolean | True if mirrored                                                                                                                              |
| Return Value | String<br>PortID, the identifier that can be used for this port in script operations involving this schematic |         |                                                                                                                                               |

Global port is created and the return value has the format GPort@<portName>;<schematicID>

The Schematic ID can be found in the **Properties** window on the **Symbols** tab when you select the global port.

| Properties         |                                                                                   |      |
|--------------------|-----------------------------------------------------------------------------------|------|
| Name               | Value                                                                             | Unit |
| GlobalPortSymbol   | defaultglobalport                                                                 |      |
| SchematicID        | 68                                                                                |      |
| Levels             | 0.1                                                                               |      |
| Component Location | 4600 , 2500                                                                       | mil  |
| Component Angle    | 0°                                                                                |      |
| Component Mirror   |  |      |
| Component Color    |  |      |

Param Values Symbol

|                       |                                                                                                                                                                                                                               |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | CreateGlobalPort(["NAME:GlobalPortProps", "Name:=", <portname>, "Id:=", <id>], ["NAME:Attributes", "Page:=", <pagenumber>, "X:=", <xval>, "Y:=", <yval>, "Angle:=", <angle>, [or "Degrees:=", <degrees> ,] "Flip:=", <flip>]) |
| <b>Python Example</b> | <pre>oEditor.CreateGlobalPort (["NAME:GlobalPortProps", "Name:=", "G_0", "Id:=", 1],_ ["NAME:Attributes", "Page:=", 1, "X:=", -0.02286, "Y:=", 0.00254, "Angle:=", 0, "Flip:=", false])</pre>                                 |

|                   |                                                                                                                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | CreateGlobalPort(Array("NAME:GlobalPortProps", "Name:=", <portname>, "Id:=", <id>), Array("NAME:Attributes", "Page:=", <pagenumber>, "X:=", <xval>, "Y:=", <yval>, "Angle:=", <angle>, [or "Degrees:=", <degrees>], "Flip:=", <flip>) |
| <b>VB Example</b> | <pre>oEditor.CreateGlobalPort Array("NAME:GlobalPortProps", "Name:=", "G_0", "Id:=", 1), _   Array("NAME:Attributes", "Page:=", 1, "X:=", -0.02286,     "Y:=", 0.00254, "Angle:=", 0, "Flip:=", false)</pre>                          |

**VB Example:**

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
```



```
Dim GPort
```

```
GPort = oEditor.CreateGlobalPort(Array("NAME:GlobalPortProps", "Name:=", "G_1", "Id:=", 68),
Array("NAME:Attributes", "Page:=", _
```

```
1, "X:=", 0.11684, "Y:=", 0.0635, "Angle:=", 0, "Flip:=", false))
```

```
MsgBox "Port = " & GPort
```

## CreateGround (Schematic Editor)

Create a ground

| UI Access    | Draw>Ground         |       |                                                                                                                                                                                                                                                                                                                                        |
|--------------|---------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                | Type  | Description                                                                                                                                                                                                                                                                                                                            |
|              | <GroundProps>       | Array | Array("NAME:GroundProps", _<br>"Id:=", int), // [Opt=New id] Id for this item                                                                                                                                                                                                                                                          |
|              | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int, _ // [Opt=1] Page number (one-based)<br>Note: Page 1 always exists<br>"X:=", double, _ // X position of the object<br>"Y:=", double, _ // Y position of the object<br>"Angle:=", double, _ // Rotation angle (radians)<br>"Flip:=", bool), // True if mirrored<br>[out,retval] string id) |
| Return Value | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                        |

Ground is created and the returned value has the format GPort@<portname>;<schematicID>

The Schematic ID for ground can be found on the **Symbols** tab of the **Properties** window when you select the object.

| Properties             |              |      |
|------------------------|--------------|------|
| Name                   | Value        | Unit |
| GlobalPortSymbol       | ground_earth |      |
| SchematicID            | 144          |      |
| Levels                 | 0.1          |      |
| Component Location     | 4600 , 2600  | mil  |
| Component Angle        | 0°           |      |
| <div><div></div></div> |              |      |
| Param Values Symbol    |              |      |

| Python Syntax  | CreateGround()                                                                                                                                                       |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateGround (["NAME:GroundProps", "Id:=", 8), ["NAME:Attributes", "Page:=", 1, "X:=", -0.04064, "Y:=", -0.00254, "Angle:=", 0, "Flip:=", false])</pre> |

| VB Syntax         | CreateGround()                                                                                                                                                         |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateGround Array("NAME:GroundProps", "Id=", 8), Array("NAME:Attributes", "Page=", 1, "X=", -0.04064, "Y=", -0.00254, "Angle=", 0, "Flip=", false)</pre> |

***VB Example:***

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
```

```
Gnd = oEditor.CreateGround(Array("NAME:GroundProps", "Id:=", 144), Array("NAME:Attributes",
"Page:=", _
1, "X:=", 0.11684, "Y:=", 0.06604, "Angle:=", 0, "Flip:=", false))
```

```
MsgBox "GndIndex =" & Gnd
```

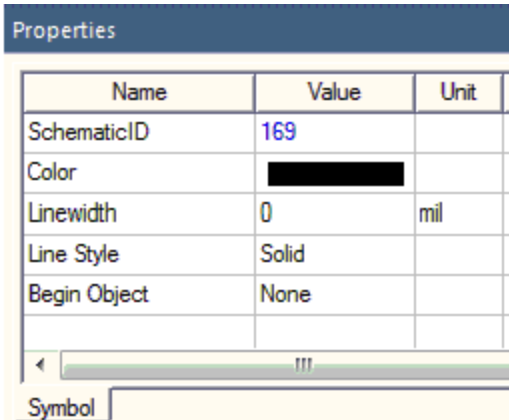
CreateLine (Schematic Editor)

Creates a line.

|              |                     |       |                                                                                                                                                                                                                                 |
|--------------|---------------------|-------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Draw>Primitive>Line |       |                                                                                                                                                                                                                                 |
| Parameters   | Name                | Type  | Description                                                                                                                                                                                                                     |
|              | <LineData>          | Array | Array("NAME:LineData" _<br>"Points:=", Array of points, _<br>"LineWidth:=", double, // the width of the line, in meters<br>"Color:=", int, // the RGB value of the line color<br>"Id:=", int), // [Opt=New id] Id for this item |
|              | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int) _ // [Opt=1] Page number (one-based)<br>[out,retval] string id)                                                                                                                    |
| Return Value | String<br>Unique id |       |                                                                                                                                                                                                                                 |

Line is created and the returned value has the format SchObj@<schematicID>

The Schematic ID can be found on the **Symbol** tab of the **Properties** window when you select the line.



| Python Syntax  | CreateLine()                                                                                                                                                                                                                     |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateLine (["NAME:LineData",_ "Points:=", [ "(-0.055652, 0.020669)", _ "(-0.036923, 0.011257)", "(-0.023144, 0.018049)"],_ "LineWidth:=", 0, "Color:=", 0, "Id:=", 22)], _ ["NAME:Attributes", "Page:=", 1])</pre> |

| VB Syntax         | CreateLine()                                                                                                                                                                                                                                                  |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateLine Array("NAME:LineData", _<br/>"Points:=", Array("(-0.055652, 0.020669)", _<br/>"(-0.036923, 0.011257)", "(-0.023144, 0.018049)"), _<br/>"LineWidth:=", 0, "Color:=", 0, "Id:=", 22), _<br/>Array("NAME:Attributes", "Page:=", 1)</pre> |

***VB Example:***

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")

LineID = oEditor.CreateLine(Array("NAME:LineData", "Points:=", Array("(0.050800, 0.106680)", _
```

```
"(0.124460, 0.106680)", "(0.124460, 0.106680)"), "LineWidth:=", 0, "Color:=", _
0, "Id:=", 169), Array("NAME:Attributes", "Page:=", 1))
MsgBox "LineID = " & LineID
```

CreatePagePort (Schematic Editor)

Create a PagePort

| UI Access    | Draw>Page Connector |       |                                                                                                                                                                                                                                                                                                                                                                        |
|--------------|---------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                | Type  | Description                                                                                                                                                                                                                                                                                                                                                            |
|              | <PagePortProps>     | Array | Array("NAME:PagePortProps"),<br>Array("NAME:Attributes", _<br>"Page:=", int, _ // [Opt=1] Page number (one-based)<br>Note: Page 1 always exists<br>"X:=", double, _ // X position of the object<br>"Y:=", double, _ // Y position of the object<br>"Angle:=", double, _ // Rotation angle (radians)<br>"Flip:=", bool), // True if mirrored<br>[out,retval] string id) |
| Return Value | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                                                        |

The pageport is created and it has the following format:

```
PagePort@<portName>;<schematicID>
```

The schematic ID can be found in the **Properties** window on the **Symbols** tab when you select the pageport.

| Name               | Value                                                                             | Unit |
|--------------------|-----------------------------------------------------------------------------------|------|
| SchematicID        | 177                                                                               |      |
| Levels             | 0.1                                                                               |      |
| Component Location | 3500 , 6500                                                                       | mil  |
| Component Angle    | 0°                                                                                |      |
| Component Mirror   |  |      |

Param Values   Symbol

| Python Syntax  | CreatePagePort()                                                                                                                                                                                      |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreatePagePort (["NAME:PagePortProps", "Name:=", "pageport_0", "Id:=", _ 12], ["NAME:Attributes", "Page:=", _ 1, "X:=", -0.0381, "Y:=", -0.0127, "Angle:=", _ 0, "Flip:=", false])</pre> |



| VB Syntax         | CreatePagePort()                                                                                                                                                                                              |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreatePagePort Array("NAME:PagePortProps", "Name:=", "pageport_0", "Id:=", _ 12), Array("NAME:Attributes", "Page:=", _ 1, "X:=", -0.0381, "Y:=", -0.0127, "Angle:=", _ 0, "Flip:=", false)</pre> |

**VB Example:**

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
```

```
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
PagePortID = oEditor.CreatePagePort(Array("NAME:PagePortProps", "Name:=", "pageport_0", "Id:=", _
177), Array("NAME:Attributes", "Page:=", 1, "X:=", 0.0889, "Y:=", 0.1651, "Angle:=", _
0, "Flip:=", false))
MsgBox "PagePort = " & PagePortID
```

**CreteIPort (Schematic Editor)**

Create an interface port

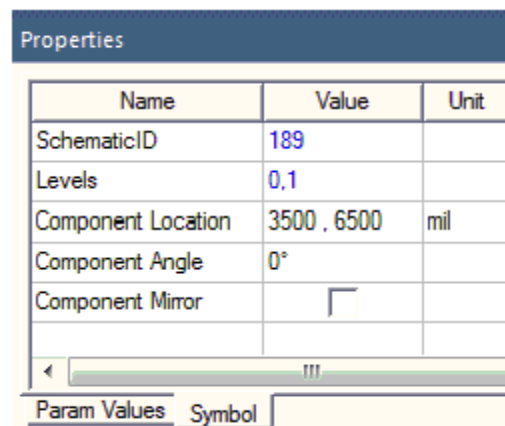
| UI Access  | Draw>Interface Port |       |                                                                                                                             |
|------------|---------------------|-------|-----------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                | Type  | Description                                                                                                                 |
|            | <IPortProps>        | Array | Array("NAME:IPortProps"),<br>"Name:=", string), // [Opt=default name] Name for this port<br>"Id:=", int), // Port id number |
|            | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int, _ // [Opt=1] Page number (one-based)                                           |

|                     |                         |  |                                                                                                                                                                                                                                                                |
|---------------------|-------------------------|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                         |  | <b>Note:</b><br><br>Page 1 always exists<br>"X:=", double, _ // X position of the object<br>"Y:=", double, _ // Y position of the object<br>"Angle:=", double, _ // Rotation angle (radians)<br>"Flip:=", bool, // True if mirrored<br>[out,retval] string id) |
| <b>Return Value</b> | String<br><br>Unique id |  |                                                                                                                                                                                                                                                                |

Interface port is created and the returned value has the following format:

IPort@<portName>;<schematicID>

The interface port schematic ID can be found in the **Properties** window on the **Symbols** tab.



| Python Syntax         | CreateIPort()                                                                                                                                                                           |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oEditor.CreateIPort (["NAME:IPortProps", "Name:=", _ "Port1", "Id:=", 4], ["NAME:Attributes","Page:=", _ 1, "X:=", -0.0381, "Y:=", 0.0127, "Angle:=", _ 0, "Flip:=", false])</pre> |

| VB Syntax         | CreateIPort()                                                                                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateIPort Array("NAME:IPortProps", "Name:=", _ "Port1", "Id:=", 4), Array("NAME:Attributes","Page:=", _ 1, "X:=", -0.0381, "Y:=", 0.0127, "Angle:=", _ 0, "Flip:=", false)</pre> |

*VB Example:*

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule
```

```
Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")

IPortID = oEditor.CreateIPort(Array("NAME:IPortProps", "Name:=", "Port1", "Id:=", 189), Array
("NAME:Attributes", "Page:=", _
1, "X:=", 0.0889, "Y:=", 0.1651, "Angle:=", 0, "Flip:=", false))

MsgBox "IPort ID = " & IPortID
```

CreatePolygon (Schematic Editor)

Create a polygon

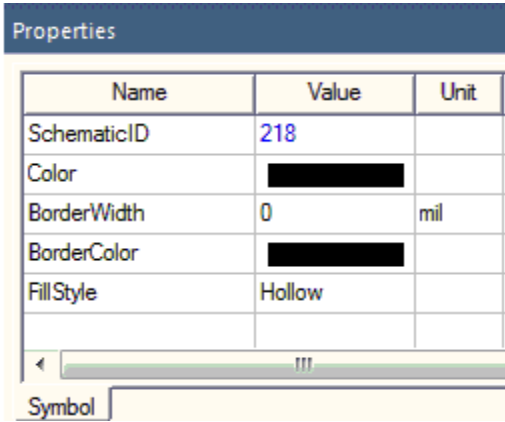
| UI Access  | Draw>Primitive>Polygon |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|------------|------------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                   | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|            | <PolygonData>          | Array | Array("NAME:PolygonData" _<br>"Points:=", Array of points, _<br>"LineWidth:=", double, // the width of the Polygon border, in meters<br>"Bordercolor:=", int, // the RBG value of the border color<br>"Fill:=", int, // the fill pattern id, 0 = hollow, 1 = solid, 2 = NEDiagonal,<br>3 = OrthoCross, 4 = DiagCross, 5 = NWDiagonal, 6 = Horizontal, 7 = Vertical<br>"Color:=", int, // the RBG value of the Polygon fill color<br>"Id:=", int), // [Opt=New id] Id for this item |

|              |                     |       |                                                                                                              |
|--------------|---------------------|-------|--------------------------------------------------------------------------------------------------------------|
|              | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int) _ // [Opt=1] Page number (one-based)<br>[out,retval] string id) |
| Return Value | String<br>Unique id |       |                                                                                                              |

Polygon is created and the return value has the format

SchObj@<schematicID>

The Schematic ID can be found in the **Properties** window on the **Symbol** tab when you select the polygon.



| Python Syntax         | CreatePolygon()                                                                                                                                                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oEditor.CreatePolygon( ["NAME:PolygonData", "Points:=", [ _ "(-0.058951, 0.006308)", "(-0.046142, 0.010480)",_ "(-0.046239, 0.001067)"), "LineWidth:=", _ 0, "BorderColor:=", 0, "Fill:=", 0, "Color:=",_ 0, "Id:=", 27], ["NAME:Attributes", "Page:=", 1])</pre> |

| VB Syntax         | CreatePolygon()                                                                                                                                                                                                                                                                     |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreatePolygon Array("NAME:PolygonData", "Points:=", Array( _ "(-0.058951, 0.006308)", "(-0.046142, 0.010480)",_ "(-0.046239, 0.001067)"), "LineWidth:=", _ 0, "BorderColor:=", 0, "Fill:=", 0, "Color:=",_ 0, "Id:=", 27), Array("NAME:Attributes", "Page:=", 1)</pre> |

**VB Example:**

```
Dim oAnsoftApp
Dim oDesktop
```

```
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow

Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")

polygonID = oEditor.CreatePolygon(Array("NAME:PolygonData", "Points:=", Array(_
"(0.101600, 0.058420)", "(0.101600, 0.060960)", "(0.101600, 0.058420)", _
"(0.099060, 0.060960)", "(0.101600, 0.060960)", "(0.101600, 0.058420)", _
"(0.101600, 0.060960)", "(0.101600, 0.060960)"), "LineWidth:=", 0, "BorderColor:=", _
0, "Fill:=", 0, "Color:=", 0, "Id:=", 218), Array("NAME:Attributes", "Page:=", 1))

MsgBox "Polygon ID = " & polygonID
```

**CreateRectangle (Schematic Editor)**

Create a rectangle

|           |                          |
|-----------|--------------------------|
| UI Access | Draw>Primitive>Rectangle |
|-----------|--------------------------|

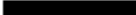


| Parameters   | Name                | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |
|--------------|---------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <RectData>          | Array | <pre>Array("NAME:RectData" _       "x1:=", double, _ // X position of the upper left       "y1:=", double, _ // Y position of the upper left       "x2:=", double, _ // X position of the lower right       "y2:=", double, _ // Y position of the lower right       "LineWidth:=", double, // the width of the Rectangle border, in meters       "Bordercolor:=", int, // the RBG value of the border color       "Fill:=", int, // the fill pattern id, 0 = hollow, 1 = solid, 2 = NEDiagonal,       3 = OrthoCross, 4 = DiagCross, 5 = NWDiagonal, 6 = Horizontal, 7 = Vertical       "Color:=", int, // the RBG value of the Rectangle fill color       "Id:=", int), // [Opt=New id] Id for this item</pre> |
|              | <Attributes>        | Array | <pre>Array("NAME:Attributes", _       "Page:=", int) _ // [Opt=1] Page number (one-based) [out,retval] string id)</pre>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| Return Value | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |

Rectangle is created and the return value has the format

SchObj@<schematicID>

The Schematic ID can be found in the **Properties** window on the **Symbol** tab when you select the rectangle.

| Name        | Value                                                                             | Unit |
|-------------|-----------------------------------------------------------------------------------|------|
| SchematicID | 9                                                                                 |      |
| Color       |  |      |
| BorderWidth | 0                                                                                 | mil  |
| BorderColor |  |      |
| FillStyle   | Hollow                                                                            |      |

| Python Syntax  | CreateRectangle()                                                                                                                                                                                                                                                                                                        |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateRectangle<br/><br/>(["NAME:RectData", "X1:=", -0.0755449856733525, "Y1:=", _<br/>0.0335755491881566, "X2:=", -0.0611831900668577, "Y2:=", _<br/>0.0254242597898758, "LineWidth:=", _<br/>0, "BorderColor:=", 0, "Fill:=", 0, "Color:=", 0, "Id:=", 31], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |

|                  |                   |
|------------------|-------------------|
| <b>VB Syntax</b> | CreateRectangle() |
|------------------|-------------------|

|                   |                                                                                                                                                                                                                                                                                                         |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateRectangle Array("NAME:RectData", "X1:=", -0.0755449856733525, "Y1:=", _ 0.0335755491881566, "X2:=", -0.0611831900668577, "Y2:=", _ 0.0254242597898758, "LineWidth:=", _ 0, "BorderColor:=", 0, "Fill:=", 0, "Color:=", 0, "Id:=", 31), _ Array("NAME:Attributes", "Page:=", 1)</pre> |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**VB Example:**

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
```

```
rectangleID = oEditor.CreateRectangle(Array("NAME:RectData", "X1:=", 0.1016, "Y1:=", 0.0635,
"X2:=", _
0.10414, "Y2:=", 0.06096, "LineWidth:=", 0, "BorderColor:=", 0, "Fill:=", 0, "Color:=", _
0, "Id:=", 9), Array("NAME:Attributes", "Page:=", 1))
MsgBox "Rectangle ID = " & rectangleID
```

CreateText (Schematic Editor)

Create text

| UI Access  | Draw>Primitive>Text |       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |
|------------|---------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters | Name                | Type  | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|            | <TextData>          | Array | Array("NAME:TextData" _<br>"x:=", double, _ // X position of the object<br>"y:=", double, _ // Y position of the object<br>"Text:=", string, _ // Text to display<br>"Color:=", int, // the RGB value of the text color<br>"Id:=", int), // Id for this item<br>"ShowRect:=", bool, // true or false, whether to show the highlight rectangle for the text<br>"X1:=", double, // the text rectangle left X value, in meters<br>"Y1:=", double, // the text rectangle upper Y value, in meters<br>"X2:=", double, // the text rectangle right X value, in meters<br>"Y2:=", double, // the text rectangle lower Y value, in meters |

|                     |                     |       |                                                                                                                                                                                                                                                                                                                                                                                                          |
|---------------------|---------------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     |                     |       | "RectLineWidth:=", double, // the width of the rectangle border, in meters<br>"RectBordercolor:=", int, // the RGB value of the rectangle border color<br>"RectFill:=", int, // the rectangle fill pattern id, 0 = hollow, 1 = solid,<br>2 = NEDiagonal, 3 = OrthoCross, 4 = DiagCross, 5 = NWDiagonal, 6 = Horizontal, 7 = Vertical<br>"RectColor:=", int, // the RGB value of the rectangle fill color |
|                     | <Attributes>        | Array | Array("NAME:Attributes", _<br>"Page:=", int) _ // [Opt=1] Page number (one-based)<br>[out,retval] string id)                                                                                                                                                                                                                                                                                             |
| <b>Return Value</b> | String<br>Unique id |       |                                                                                                                                                                                                                                                                                                                                                                                                          |

Text is created and the return value has the format: SchObj@<SchematicID>

The Schematic ID can be found in the **Properties** window on the **Symbols** tab when you select the text.

| Properties                              |             |      |
|-----------------------------------------|-------------|------|
| Name                                    | Value       | Unit |
| SchematicID                             | 286         |      |
| Color                                   |             |      |
| Location                                | 4100 , 2600 | mil  |
| Angle                                   | 0           | deg  |
| TextSize                                | 12          |      |
| <div> <div></div> <div>'''</div> </div> |             |      |
| Symbol                                  |             |      |

| Python Syntax         | CreateText()                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oEditor.CreateText(<br/>    ["NAME:TextData", "X:=", -0.0764183381088826, "Y:=", _<br/>    0.040174212034384, "Size:=", 12, "Angle:=", _<br/>    0, "Text:=", "Control Circuit", "Color:=", _<br/>    0, "Id:=", 34, "ShowRect:=", false, "X1:=", _<br/>    -0.0793287547755609, "Y1:=", _<br/>    0.0407033787010528, "X2:=", -0.0502245881087778, _<br/>    "Y2:=", 0.035411712034365, "RectLineWidth:=", _<br/>    0, "RectBorderColor:=", 0, "RectFill:=",<br/>    0, "RectColor:=", 0], ["NAME:Attributes", "Page:=", 1])</pre> |

| VB Syntax         | CreateText()                                                                                                                                                                                             |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateText<br/>Array("NAME:TextData", "X:=", -0.0764183381088826, "Y:=", _<br/>    0.040174212034384, "Size:=", 12, "Angle:=", _<br/>    0, "Text:=", "Control Circuit", "Color:=", _</pre> |

|  |                                                                                                                                                                                                                                                                                                   |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> 0, "Id:=", 34, "ShowRect:=", false, "X1:=", _ -0.0793287547755609, "Y1:=", _ 0.0407033787010528, "X2:=", -0.0502245881087778, _ "Y2:=", 0.035411712034365, "RectLineWidth:=", _ 0, "RectBorderColor:=", 0, "RectFill:=", 0, "RectColor:=", 0), Array("NAME:Attributes", "Page:=", 1) </pre> |
|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**VB Example:**

```

Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
textobj = oEditor.CreateText(Array("NAME:TextData", "X:=", 0.10414, "Y:=", 0.06604, "Size:=", _

```

```
12, "Angle:=", 0, "Text:=", "CreateDefaultTectxt" & Chr(13) & Chr(10) & "", "Color:=", _
0, "Id:=", 286, "ShowRect:=", false, "X1:=", 0.100141851851836, "Y1:=", _
0.0670983333333376, "X2:=", 0.140123333333477, "Y2:=", 0.0565149999999619, "RectLineWidth:=", _
0, "RectBorderColor:=", 0, "RectFill:=", 0, "RectColor:=", 0), Array("NAME:Attributes", "Page:=", _
1))

MsgBox "text = " & textobj
```

CreateWire (Schematic Editor)

Create a wire

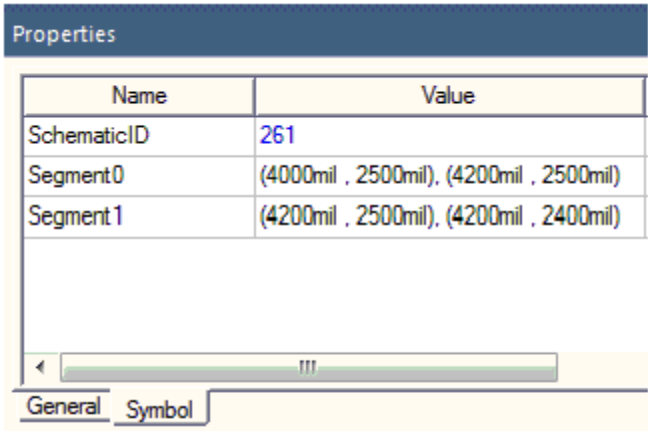
| UI Access    | Draw>Wire    |       |                                                                                                                                                                  |
|--------------|--------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name         | Type  | Description                                                                                                                                                      |
|              | <WireData>   | Array | Array("NAME:WireData" _<br><br>"Name:=", string), // [Opt=default name] Name for this wire<br><br>"Id:=", int), // Wire idnum<br><br>"Points:=", Points on wire) |
|              | <Attributes> | Array | Array("NAME:Attributes", _<br><br>"Page:=", int, _ // [Opt=1] Page number (one-based)<br><br>Note: Page 1 always exists<br><br>[out,retval] string id)           |
| Return Value | String       |       |                                                                                                                                                                  |



|  |           |
|--|-----------|
|  | Unique id |
|--|-----------|

Wire is created and its return value has the format Wire@<netName>;<schematicID>;<segment index list>

The Schematic ID can be found in the **Symbol** tab of the **Properties** window when you select the wire. The wire net name appears in the **General** tab. The parameter segment index list indicates the number of segments i.e. the segment count.



| Python Syntax  | CreateWire()                                                                                                                                                                                                                  |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CreateWire(<br/>["NAME:WireData", "Name:=", "", "Id:=", 41, "Points:=", [ _<br/>    "(-0.033020, 0.015240)", "(-0.017780, 0.015240)",<br/>    "(-0.017780, 0.012700)"]], ["NAME:Attributes", "Page:=", 1])</pre> |

| VB Syntax         | CreateWire()                                                                                                                                                                                                                                    |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.CreateWire<br/><br/>Array("NAME:WireData", "Name:=", "", "Id:=", 41, "Points:=", Array( _<br/>    "(-0.033020, 0.015240)", "(-0.017780, 0.015240)",<br/>    "(-0.017780, 0.012700)")), Array("NAME:Attributes", "Page:=", 1)</pre> |

***VB Example:***

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("Project4")
Set oDesign = oProject.SetActiveDesign("Circuit1")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
```

```
WireID = oEditor.CreateWire(Array("NAME:WireData", "Name:=", "", "Id:=", 261, "Points:=", Array(
_
"(0.101600, 0.063500)", "(0.106680, 0.063500)", "(0.106680, 0.060960)")), Array("NAME:At-
tributes", "Page:=", _
1))
MsgBox "Wire ID = " & WireID
```

## General Method List

This section presents the general script methods that are available.

The general method script commands are listed here.

[Activate \(Schematic Editor\)](#)

[AddPinGrounds \(Schematic Editor\)](#)

[AddPinIPorts \(Schematic Editor\)](#)

[AddPinPageConnectors \(Schematic Editor\)](#)

[AddPinCustomTerminations \(Schematic Editor\)](#)

[AlignHorizontal \(Schematic Editor\)](#)

[AlignVertical \(Schematic Editor\)](#)

[BringToFront \(Schematic Editor\)](#)

[Close Editor \(Schematic Editor\)](#)

[Copy \(Schematic Editor\)](#)

[CopyData \[Schematic Editor\]](#)

[CopySubdesign \[Schematic Editor\]](#)

[CreatePage \(Schematic Editor\)](#)

[Cut \(Schematic Editor\)](#)

[DeactivateOpen \(Schematic Editor\)](#)

[DeactivateShort \(Schematic Editor\)](#)

[Delete \(Schematic Editor\)](#)

[DeletePage \(Schematic Editor\)](#)

[ElectricRuleCheck \(Schematic Editor\)](#)

[ExportImage \(Schematic Editor\)](#)

[FindElements \(Schematic Editor\)](#)

[FitToBorder \(Schematic Editor\)](#)

[GridSetup \(Schematic Editor\)](#)

[FlipHorizontal \(Schematic Editor\)](#)

[FlipVertical \(Schematic Editor\)](#)

[Move \(Schematic Editor\)](#)

[NameNets \(Schematic Editor\)](#)

[PageBorders \(Schematic Editor\)](#)

[Pan \(Schematic Editor\)](#)

[Paste \(Schematic Editor\)](#)

[PasteData \(Schematic Editor\)](#)

[PasteDesign \(Schematic Editor\)](#)

[PushExcitations \(Schematic Editor\)](#)

[Rotate \(Schematic Editor\)](#)

[ShowVariableBlock \(Schematic Editor\)](#)

[SelectAll \(Schematic Editor\)](#)

[SelectPage \(Schematic Editor\)](#)

[SendToBack\(Schematic Editor\)](#)

[SetPageData \[Schematic\]](#)

[SortComponents \(Schematic Editor\)](#)

[ZoomArea \(Schematic Editor\)](#)

[ZoomIn \(Schematic Editor\)](#)

[ZoomOut \(Schematic Editor\)](#)

[ZoomPrevious \(Schematic Editor\)](#)

[ZoomToFit \(Schematic Editor\)](#)

[Wire \(Schematic Editor\)](#)

## Activate (Schematic Editor)

Enable items previously disabled

| UI Access    | Edit>Activate |      |             |
|--------------|---------------|------|-------------|
| Parameters   | Name          | Type | Description |
|              | Name          | Type | Description |
| Return Value | Value         |      |             |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs.](#)

|                |                                                                             |
|----------------|-----------------------------------------------------------------------------|
| Python Syntax  | Activate(["NAME:Selections","Selections:=", [<components>] ])               |
| Python Example | oEditor.Activate(["NAME:Selections","Selections:=", ["CompInst@STEP;1;1"]]) |

|            |                                                                                          |
|------------|------------------------------------------------------------------------------------------|
| VB Syntax  | Activate Array("NAME:Selections", "Selections:=", Array("<components>"))                 |
| VB Example | oEditor.Activate Array("NAME:Selections",<br>"Selections:=", Array("CompInst@STEP;1;1")) |

**AddPinGrounds (Schematic Editor)**

```
Adds grounds at all unconnected pins

AddPinGrounds (
Array("NAME:Selections", _ // PagePort Name
"Selections:=", _ // Net Name
Array("CompInst@CAP_;2;4"))
```

**Note:**

The format for Components is `CompInst@<CompName>;<compInstID>;<optional schematicID>`. In the documented example, `<CompName>` stands for the component name `CAP_`, `<compInstID>` stands for its ID 2, and `<optional schematicID>` is the schematic ID 4. If you select the component, the **Properties** window gets updated and displays its details. For example, the selected object's component name and ID appear in the **General** tab of the **Properties** window as shown below.

| Properties                      |                                                  |
|---------------------------------|--------------------------------------------------|
| Name                            | Value                                            |
| ID                              | 2                                                |
| CompName                        | CAP_                                             |
| Description                     | Capacitor                                        |
| Manufacturer                    |                                                  |
| Datasource                      | Ansoft built-in component                        |
| Date                            | 15:30:47 11/14/2005                              |
| PinCount                        | 2                                                |
| Refdes                          | C1                                               |
| Symbol                          | nexx_cap                                         |
| Footprint                       | __symbolFootprint_nexx_cap__positive__negative_0 |
| Param Values   General   Symbol |                                                  |

To view the SchematicID click the **Symbol** tab.

Properties

| Name               | Value                               | Unit |
|--------------------|-------------------------------------|------|
| SchematicID        | 4                                   |      |
| Levels             | 0.1                                 |      |
| Component Location | 3900 , 2300                         | mil  |
| Component Angle    | 0°                                  |      |
| Component Mirror   | <input type="checkbox"/>            |      |
| Use Symbol Color   | <input checked="" type="checkbox"/> |      |

Param ValuesGeneralSymbol

|                |                                                                                                           |
|----------------|-----------------------------------------------------------------------------------------------------------|
| Python Syntax  | AddPinPageConnectors ()                                                                                   |
| Python Example | <pre>oEditor.AddPinPageconnectors["NAME:Selections",<br/>"Selections:=", [ _<br/>"CompInst@C;1;1"]]</pre> |



## AddPinIPorts (Schematic Editor)

Adds Interface Ports at unconnected pins of all selected components. Each port has a unique name.

|                     |                                              |       |                                                                                    |
|---------------------|----------------------------------------------|-------|------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Draw>Add at unconnected pins>Interface Ports |       |                                                                                    |
| <b>Parameters</b>   | Name                                         | Type  | Description                                                                        |
|                     | <IntPortName,NetName>                        | Array | Array("NAME:Selections", _ // Interface Port Name<br>"Selections:=", _ // Net Name |
| <b>Return Value</b> | None                                         |       |                                                                                    |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                   |
|-----------------------|---------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddPinIPorts ()                                                                                   |
| <b>Python Example</b> | <pre>oEditor.AddPinIPorts<br/><br/>["NAME:Selections", "Selections:=", ["CompInst@R;2;37"]]</pre> |

|                   |                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AddPinIPorts ()                                                                                             |
| <b>VB Example</b> | <pre>oEditor.AddPinIPorts<br/><br/>Array("NAME:Selections", "Selections:=", Array("CompInst@R;2;37"))</pre> |

AddPinPageConnectors (Schematic Editor)

Adds Page Ports at unconnected pins of all selected components

|              |                                      |      |             |
|--------------|--------------------------------------|------|-------------|
| UI Access    | Draw>Add at unconnected pins>Grounds |      |             |
| Parameters   | Name                                 | Type | Description |
|              | Name                                 | Type | Description |
| Return Value | None                                 |      |             |

Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                |                                                                                                        |
|----------------|--------------------------------------------------------------------------------------------------------|
| Python Syntax  | AddPinPageConnectors ()                                                                                |
| Python Example | <pre>oEditor.AddPinPageconnectors ["NAME:Selections", "Selections:=", [ _     "CompInst@C;1;1"]]</pre> |

|            |                                                                                                                  |
|------------|------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | AddPinPageConnectors ()                                                                                          |
| VB Example | <pre>oEditor.AddPinPageconnectors Array("NAME:Selections", "Selections:=", Array( _     "CompInst@C;1;1"))</pre> |

## AlignHorizontal (Schematic Editor)

Align items horizontally

| UI Access    | Draw>Align Horizontal |       |                                                                                                                                                                                                                       |
|--------------|-----------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                  | Type  | Description                                                                                                                                                                                                           |
|              | <Selections>          | Array | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)                                                                                                       |
|              | <AlignParameters>     | Array | Array("NAME:AlignParameters", _<br>"Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool) _ // [Opt=1] Should wires staircase<br>Note: Alignment occurs relative to the first item in ids |
| Return Value | Value                 |       |                                                                                                                                                                                                                       |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

| Python Syntax  | AlignHorizontal()                                                                                                                                                                          |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.AlignHorizontal(["NAME:Selections", "Selections:=", [ _ "CompInst@R;2;4", "CompInst@C;1;1"]], ["NAME:AlignParameters", "Disconnect:=", _ false, "Rubberband:=", false])</pre> |

|                   |                                                                                                                                                                                                                  |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AlignHorizontal()                                                                                                                                                                                                |
| <b>VB Example</b> | <pre>oEditor.AlignHorizontal Array("NAME:Selections", "Selections:=", Array( _<br/>"CompInst@R;2;4", "CompInst@C;1;1")), Array("NAME:AlignParameters", "Disconnect:=", _<br/>false, "Rubberband:=", false)</pre> |

**AlignVertical (Schematic Editor)**

Align items vertically

|                     |                     |             |                                                                                                                                                                                                                       |
|---------------------|---------------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Draw>Align Vertical |             |                                                                                                                                                                                                                       |
| <b>Parameters</b>   | <b>Name</b>         | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                    |
|                     | <Selections>        | Array       | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)                                                                                                       |
|                     | <Align Parameters>  | Array       | Array("NAME:AlignParameters", _<br>"Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool) _ // [Opt=1] Should wires staircase<br>Note: Alignment occurs relative to the first item in ids |
| <b>Return Value</b> | None                |             |                                                                                                                                                                                                                       |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AlignVertical()                                                                                                                                                                          |
| <b>Python Example</b> | <pre>oEditor.AlignVertical (["NAME:Selections", "Selections:=", ["CompInst@R;2;4", _ "CompInst@C;1;1"]], ["NAME:AlignParameters", "Disconnect:=", false, "Rubberband:=", _ false])</pre> |

|                   |                                                                                                                                                                                                         |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | AlignVertical()                                                                                                                                                                                         |
| <b>VB Example</b> | <pre>oEditor.AlignVertical Array("NAME:Selections", "Selections:=", Array("CompInst@R;2;4", _ "CompInst@C;1;1")), Array("NAME:AlignParameters", "Disconnect:=", false, "Rub- berband:=", _ false)</pre> |

**BringToFront (Schematic Editor)**

Bring the selected object to the front of the view

|                  |                     |
|------------------|---------------------|
| <b>UI Access</b> | Draw>Bring to Front |
|------------------|---------------------|

| Parameters   | Name         | Type  | Description                                                                                                                                                                                     |
|--------------|--------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              | <Selections> | Array | ["NAME: Selections", "Selections:=", [<Selected Components>]]<br><br><Selected Components><br><br>CompInst@DefName; ID, schematic ID , CompInst@DefName; ID, schematic ID<br><br>CompInst@R;1,2 |
| Return Value | None         |       |                                                                                                                                                                                                 |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                |                                                                                         |
|----------------|-----------------------------------------------------------------------------------------|
| Python Syntax  | BringToFront( )                                                                         |
| Python Example | <pre>oEditor.BringToFront(<br/>["NAME:Selections", "Selections:=", ["SchObj@1"]])</pre> |

|            |                                                                                                 |
|------------|-------------------------------------------------------------------------------------------------|
| VB Syntax  | BringToFront( )                                                                                 |
| VB Example | <pre>oEditor.BringToFront<br/>Array("NAME:Selections", "Selections:=", Array("SchObj@1"))</pre> |

## CloseEditor (Schematic Editor)

Close an Editor

|                     |      |      |             |
|---------------------|------|------|-------------|
| <b>UI Access</b>    | NA   |      |             |
| <b>Parameters</b>   | Name | Type | Description |
|                     | None |      |             |
| <b>Return Value</b> | None |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                              |
|-----------------------|----------------------------------------------|
| <b>Python Syntax</b>  | CloseEditor()                                |
| <b>Python Example</b> | <code>oDefinitionEditor.CloseEditor()</code> |

|                   |                                            |
|-------------------|--------------------------------------------|
| <b>VB Syntax</b>  | CloseEditor                                |
| <b>VB Example</b> | <code>oDefinitionEditor.CloseEditor</code> |

## Copy (Schematic Editor)

GeneralCopy

```
// Copy items for pasting
```

```
Copy(
Array("NAME:Selections", _
"Page:=", page number, _ // [Opt=1] Page number
"Selections:=", IDs to modify))
```

**Note:**  
For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                |                                                                                                |
|----------------|------------------------------------------------------------------------------------------------|
| Python Syntax  | Copy( )                                                                                        |
| Python Example | oEditor.Copy (["NAME:Selections", "Selections:=",<br>["CompInst@R;2;37", _ "CompInst@C;1;1"]]) |

**CopyData [Schematic Editor]**

**Use:** Copy graphic object or component properties for pasting.

**Command:** Edit> Copy Data (or Right-click on graphic object or component and select Copy Data)

**Syntax:** CopyData Array("NAME:Selections", "Selections:=", Array(<selections>))

**Return Value:** None.

**Parameters:** <selections>

Type: Comma-separated list of strings



Identifiers for each selected item to be copied, in the form `CompInst@<id>`, where `id` is the `compname;ID;schematicID` for each element.

**Example:** `Set oEditor = oDesign.SetActiveEditor("SchematicEditor")`

```
oEditor.CopyData Array("NAME:Selections", "Selections:=", Array("CompInst@Cap;1;1"))
```

## CopySubdesign [Schematic Editor]

Copy components that represent subdesigns, as well as their represented designs, for pasting. The designs, when pasted, will be independent, with unique names based on the original design.

| UI Access    | (Right-click on hierarchical component) <b>Copy as New Design</b> |                                 |                                                                                                                                                                                     |
|--------------|-------------------------------------------------------------------|---------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                                              | Type                            | Description                                                                                                                                                                         |
|              | <selections>                                                      | Comma-separated list of strings | Identifiers for each selected item to be copied, in the form <code>CompInst@&lt;id&gt;</code> , where <code>id</code> is the <code>compname;ID;schematicID</code> for each element. |
| Return Value | None                                                              |                                 |                                                                                                                                                                                     |

| Python Syntax  | <code>CopySubdesign(["NAME:Selections", "Selections:=", [&lt;selections&gt;]])</code>                                  |
|----------------|------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.CopySubdesign ([     "NAME:Selections", "Selections:=", [         "CompInst@Simplorer2;23;17" ] ] )</pre> |

|                   |                                                                                                                        |
|-------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>CopySubdesign Array("NAME:Selections", "Selections:=", Array(&lt;selections&gt;))</code>                         |
| <b>VB Example</b> | <pre>oEditor.CopySubdesign Array(<br/>"NAME:Selections", "Selections:=",<br/>Array("CompInst@Simplorer2;23;17"))</pre> |

## CreatePage (Schematic Editor)

Create a page at the end of the existing pages

|                     |                    |         |                                        |
|---------------------|--------------------|---------|----------------------------------------|
| <b>UI Access</b>    | Schematic>New Page |         |                                        |
| <b>Parameters</b>   | Name               | Type    | Description                            |
|                     | <PageName>         | String  | Name of the page that has been created |
|                     | <PageNum>          | Integer | Page Number                            |
| <b>Return Value</b> | None               |         |                                        |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>CreatePage(string name, // Page name<br/>[out,retval] int num) // Page number</code> |
| <b>Python Example</b> | <pre>oEditor.CreatePage ("TestPage1")</pre>                                                |

|                   |                                                                               |
|-------------------|-------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | CreatePage( string name, // Page name<br>[out,retval] int num) // Page number |
| <b>VB Example</b> | oEditor.CreatePage "TestPage1"                                                |

## Cut (Schematic Editor)

### Cut Page Selection

|                     |              |             |                                                                                                                   |
|---------------------|--------------|-------------|-------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Edit>Cut     |             |                                                                                                                   |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>                                                                                                |
|                     | <Selections> | Array       | Array ("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)) |
| <b>Return Value</b> | None         |             |                                                                                                                   |

#### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                         |
|-----------------------|-------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Cut()                                                                   |
| <b>Python Example</b> | oEditor.Cut (["NAME:Selections", "Selections:=", ["CompInst@R;10;93"]]) |

|                  |       |
|------------------|-------|
| <b>VB Syntax</b> | Cut() |
|------------------|-------|

|                   |                                                                                                |
|-------------------|------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oEditor.Cut Array("NAME:Selections",<br/>"Selections:=", Array("CompInst@R;10;93"))</pre> |
|-------------------|------------------------------------------------------------------------------------------------|

**DeactivateOpen (Schematic Editor)**

Disable items and replace with an open circuit

|                     |                        |             |                                                                                                                 |
|---------------------|------------------------|-------------|-----------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Edit>Deactivate (Open) |             |                                                                                                                 |
| <b>Parameters</b>   | <b>Name</b>            | <b>Type</b> | <b>Description</b>                                                                                              |
|                     | <Selections>           | Array       | Array("NAME:Selections", _<br>"Page:=", page number, _// [Opt=1] Page number<br>"Selections:=", IDs to modify)) |
| <b>Return Value</b> | None                   |             |                                                                                                                 |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                   |
|-----------------------|---------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | DeactivateOpen()                                                                                  |
| <b>Python Example</b> | <pre>oEditor.DeactivateOpen (["NAME:Selections",<br/>"Selections:=", [ "CompInst@R;2;37"]])</pre> |

|                   |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeactivateOpen()                                                                                          |
| <b>VB Example</b> | <pre>oEditor.DeactivateOpen Array("NAME:Selections",<br/>"Selections:=", Array( "CompInst@R;2;37"))</pre> |

## DeactivateShort (Schematic Editor)

Disable items and replace with a closed circuit

|                     |                         |             |                                                                                                                  |
|---------------------|-------------------------|-------------|------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Edit>Deactivate (Short) |             |                                                                                                                  |
| <b>Parameters</b>   | <b>Name</b>             | <b>Type</b> | <b>Description</b>                                                                                               |
|                     | <Selections>            | Array       | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)) |
| <b>Return Value</b> | None                    |             |                                                                                                                  |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs.](#)

|                       |                                                                                                     |
|-----------------------|-----------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | DeactivateShort ()                                                                                  |
| <b>Python Example</b> | <pre>oEditor.DeactivateShort(["NAME:Selections", "Selections:=", [ _<br/>"CompInst@R;2;37"]])</pre> |

|                   |                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeactivateShort ()                                                                                           |
| <b>VB Example</b> | <pre>oEditor.DeactivateShort Array("NAME:Selections", "Selections:=", Array( _<br/>"CompInst@R;2;37"))</pre> |

## Delete (Schematic Editor)

Delete items

|                     |              |             |                                                                                                                  |
|---------------------|--------------|-------------|------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | Edit>Delete  |             |                                                                                                                  |
| <b>Parameters</b>   | <b>Name</b>  | <b>Type</b> | <b>Description</b>                                                                                               |
|                     | <Selections> | Array       | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)) |
| <b>Return Value</b> | None         |             |                                                                                                                  |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                           |
|-----------------------|-------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | Delete()                                                                                  |
| <b>Python Example</b> | <pre>oEditor.Delete (["NAME:Selections", "Selections:=",<br/>["CompInst@R;10;93"]])</pre> |

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Delete()                                                                                          |
| <b>VB Example</b> | <pre>oEditor.Delete Array("NAME:Selections", "Selections:=",<br/>Array("CompInst@R;10;93"))</pre> |

## DeletePage (Schematic Editor)

To reduce the number of pages in the schematic

|                     |                       |      |             |
|---------------------|-----------------------|------|-------------|
| <b>UI Access</b>    | Schematic>Remove Page |      |             |
| <b>Parameters</b>   | Name                  | Type | Description |
|                     | None                  |      |             |
| <b>Return Value</b> | None                  |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                             |
|-----------------------|---------------------------------------------|
| <b>Python Syntax</b>  | DeletePage(<pagenum>, <pagenum>*)           |
| <b>Python Example</b> | <pre>num = oEditor.DeletePage([1, 4])</pre> |

|                   |                                                                     |
|-------------------|---------------------------------------------------------------------|
| <b>VB Syntax</b>  | DeletePage Array(<pagenum>[, <pagenum>]*)                           |
| <b>VB Example</b> | <pre>Dim oEditor<br/>Set num = oEditor.DeletePage Array(1, 4)</pre> |

ElectricRuleCheck (Schematic Editor)

Check Electric Rule

|              |                               |       |                                                                                                                                                                                                                                                                                                                                                                                           |
|--------------|-------------------------------|-------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | Schematic>Electric Rule Check |       |                                                                                                                                                                                                                                                                                                                                                                                           |
| Parameters   | Name                          | Type  | Description                                                                                                                                                                                                                                                                                                                                                                               |
|              | <ElectricRules>               | Array | Array(string, ...), bool<br><br>Array<br><br>Array of strings containing names of electric rules to perform.<br><br>Valid names are:<br><br>"PinRule": Checks for components with unconnected pins<br><br>"OutputPinRule": Checks for nets with more than one output pin<br><br>bool<br><br>True if this should check all subcircuits<br><br>False if this should just check this circuit |
| Return Value | None                          |       |                                                                                                                                                                                                                                                                                                                                                                                           |

Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).



|                       |                                                                                                  |
|-----------------------|--------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ElectricRuleCheck()                                                                              |
| <b>Python Example</b> | <pre>oEditor.ElectricRuleCheck (["PinRule",<br/>"OutputPinRule", "OverlapCompRule"], True)</pre> |

|                   |                                                                                                     |
|-------------------|-----------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | ElectricRuleCheck()                                                                                 |
| <b>VB Example</b> | <pre>oEditor.ElectricRuleCheck Array("PinRule",<br/>"OutputPinRule", "OverlapCompRule"), True</pre> |

## ExportImage (Schematic Editor)

To export a picture for a specified page of the current design to a file. The image size can also be specified. The filename extension determines the type of image exported.

|                     |             |             |                                                                                                                                               |
|---------------------|-------------|-------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | None.       |             |                                                                                                                                               |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b>                                                                                                                            |
|                     | <filename>  | String      | The name of the file, with format-specific extension. Extensions supported are: bmp, gif, jpg, jpeg, png, tif, tiff.                          |
|                     | <pagenum>   | Integer     | The page number of the current schematic. Page numbers start at 1. If the number is out of range, the current schematic page will be printed. |
|                     | <dx>        | Integer     | The width of the image. If dx is less than 160, 160 will be used for the width.                                                               |
|                     | <dy>        | Integer     | The height of the image. If dy is less than 160, 160 will be used for the height.                                                             |
| <b>Return Value</b> | None        |             |                                                                                                                                               |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                |
|-----------------------|----------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportImage (<filename>, <pagenum>, <dx>, <dy>)                |
| <b>Python Example</b> | <code>oEditor.ExportImage ("c:\mysch.png", 1, 800, 400)</code> |

|                   |                                                              |
|-------------------|--------------------------------------------------------------|
| <b>VB Syntax</b>  | ExportImage (<filename>, <pagenum>, <dx>, <dy>)              |
| <b>VB Example</b> | <code>oEditor.ExportImage "c:\mysch.png", 1, 800, 400</code> |

## ExportNetlist

Exports a netlist solution.

|                     |             |             |                                                                                                    |
|---------------------|-------------|-------------|----------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | None        |             |                                                                                                    |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b>                                                                                 |
|                     | <solution>  | String      | The Circuit solution in the design. For Maxwell Circuit, the solution must be an empty string, "". |
|                     | <fileName>  | String      | File and path to the netlist file. For Maxwell Circuit, the file name must be a SPH file.          |
| <b>Return Value</b> | None.       |             |                                                                                                    |

|                       |                                                            |
|-----------------------|------------------------------------------------------------|
| <b>Python Syntax</b>  | ExportNetlist(<setupName>,<fileName>)                      |
| <b>Python Example</b> | <code>oDesign.ExportNetlist("", "SolutionData.sph")</code> |

|                   |                                           |
|-------------------|-------------------------------------------|
| <b>VB Syntax</b>  | ExportNetlist <solution> <fileName>       |
| <b>VB Example</b> | oDesign.ExportNetlist "" SolutionData.sph |

## FindElements (Schematic Editor)

Select elements based on properties

| UI Access         | Edit>Find Elements |         |                                                                                                                                                                                          |
|-------------------|--------------------|---------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Parameters</b> | Name               | Type    | Description                                                                                                                                                                              |
|                   | <propname>         | String  | The name of the property, or "" to search through all properties                                                                                                                         |
|                   | <propvalue>        | String  | The value of the property, or a substring                                                                                                                                                |
|                   | <criteria>         | Integer | 0 = Contains<br>1 = Does not contain<br>2 = Exact match                                                                                                                                  |
|                   | <types>            | Integer | 2 - components only<br>4 - graphic objects only<br>8 - wires only<br>16 - ports only<br>Add the numbers to look through more types - 30 is a common number, searching through everything |
|                   | <match>            | Boolean | True = match all of the properties specified<br>False = match any of the properties specified                                                                                            |
|                   | <case>             | Boolean | True = case-sensitive<br>False = not case-sensitive                                                                                                                                      |
|                   | <sub>              | Boolean | True = search subcircuits as well as current schematic                                                                                                                                   |

|              |                                                      |         |                                         |
|--------------|------------------------------------------------------|---------|-----------------------------------------|
|              |                                                      |         | False = only search current schematic   |
|              | <only>                                               | Boolean | True = search within current selections |
|              |                                                      |         | False = search schematic(s)             |
| Return Value | Array of strings, the names of the elements selected |         |                                         |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                |                                                                                                                                                                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | FindElements(["NAME:SearchProps", "Prop:=", [<propname>, <propvalue>, <criterion>][, [<propname>, <propvalue>, <criterion>]]*), ["NAME:Parameters", "Filter:=", <types>, "MatchAll:=", <match>, "MatchCase:=", <case>, "SearchSubCkt:=", <sub>, "SearchSelectionOnly:=", <only>]) |
| Python Example | <pre>oEditor.FindElement(<br/>    [<br/>        "NAME:SearchProps",<br/>        "Prop:=", ["CompName", "", 0]<br/>    ],<br/>    [<br/>        "NAME:Parameters",<br/>        "Filter:=", 30,<br/>        "MatchAll:=", False,</pre>                                              |

|  |                                                                                                                  |
|--|------------------------------------------------------------------------------------------------------------------|
|  | <pre> "MatchCase:="          , False, "SearchSubCkt:="        , True, "SearchSelectionOnly:=" , False  ]) </pre> |
|--|------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre> FindElements(Array("NAME:SearchProps", "Prop:=", Array(&lt;propname&gt;, &lt;propvalue&gt;, &lt;criterion&gt;)[, Array (&lt;propname&gt;, &lt;propvalue&gt;, &lt;criterion&gt;)]*), Array("NAME:Parameters", "Filter:=", &lt;types&gt;, "MatchAll:=", &lt;match&gt;, "MatchCase:=", &lt;case&gt;, "SearchSubCkt:=", &lt;sub&gt;, "SearchSelectionOnly:=", &lt;only&gt;) </pre> |
| <b>VB Example</b> | <pre> eltarray = oEditor.FindElements Array("NAME:SearchProps", "Prop:=", Array("InstanceName", "R", 0)), Array("NAME:Parameters", "Filter:=", _2, "MatchAll:=", true, "MatchCase:=", false, "SearchSubCkt:=", false, "SearchSelectionOnly:=", false) </pre>                                                                                                                         |

## FitToBorder [Schematic Editor]

**Use:** To reset the schematic view to just include the page border, if any.

|                   |                 |      |             |
|-------------------|-----------------|------|-------------|
| <b>UI Access</b>  | View>Fit Border |      |             |
| <b>Parameters</b> | Name            | Type | Description |
|                   | None            |      |             |

|                     |      |
|---------------------|------|
| <b>Return Value</b> | None |
|---------------------|------|

|                       |                                    |
|-----------------------|------------------------------------|
| <b>Python Syntax</b>  | FitToBorder()                      |
| <b>Python Example</b> | <code>oEditor.FitToBorder()</code> |

|                   |                                  |
|-------------------|----------------------------------|
| <b>VB Syntax</b>  | FitToBorder                      |
| <b>VB Example</b> | <code>oEditor.FitToBorder</code> |

## GridSetup (Schematic Editor)

Changes the settings on the schematic grid

|                   |                      |             |                                                                                                                                                                                                                                                                                                                                                                                                        |
|-------------------|----------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | Schematic>Grid Setup |             |                                                                                                                                                                                                                                                                                                                                                                                                        |
| <b>Parameters</b> | <b>Name</b>          | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                     |
|                   | <Options>            | Array       | <code>Array("NAME:Options", _<br/>"MajorGrid:=", string, _ // Major grid size with units<br/>"Divisions:=", int, _ // Number of minor grid divisions<br/>"MajorColor:=", int, _ // RGB Color of major grid lines<br/>"MinorColor:=", int, _ // RGB Color of minor grid lines<br/>"ShowGrid:=", bool, _ // Should the grid be shown?<br/>"SnapToGrid:=", bool, _ // Should objects snap to grid?</code> |

|                     |      |  |                                                                                         |
|---------------------|------|--|-----------------------------------------------------------------------------------------|
|                     |      |  | "BackgroundColor:=", int, _ // RGB Color of the background<br>"SaveAsDefault:=", bool)) |
| <b>Return Value</b> | None |  |                                                                                         |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

| Python Syntax         | GridSetup()                                                                                                                                                                                                                                                                                                                      |
|-----------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>oEditor.GridSetup([     "NAME:Options", _     "MajorGrid:=", "10mm", _     "Divisions:=", 10, _     "MajorColor:=", 0x00ff0000, _ # Red     "MinorColor:=", 0x0000ff00, _ # Green     "ShowGrid:=", true, _     "SnapToGrid:=", true, _     "BackgroundColor:=", 0x00ffffff, _ # White     "SaveAsDefault:=", false])</pre> |

| VB Syntax  | GridSetup()                                                                                                                                                                                                                                                                                                           |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oEditor.GridSetup Array(   "NAME:Options", _   "MajorGrid:=", "10mm", _   "Divisions:=", 10, _   "MajorColor:=", 0x00ff0000, _ // Red   "MinorColor:=", 0x0000ff00, _ // Green   "ShowGrid:=", true, _   "SnapToGrid:=", true, _   "BackgroundColor:=", 0x00ffffff, _ // White   "SaveAsDefault:=", false)</pre> |

For example:

```
oEditor.GridSetup Array(
 "NAME:Options", _
 "MajorGrid:=", "10mm", _
 "Divisions:=", 10, _
 "MajorColor:=", 0x00ff0000, _ // Red
 "MinorColor:=", 0x0000ff00, _ // Green
 "ShowGrid:=", true, _
```



```
"SnapToGrid:=", true, _
"BackgroundColor:=", 0x00ffffff, _ // White
"SaveAsDefault:=", false)
```

## FlipHorizontal (Schematic Editor)

Flip items about the horizontal (x) axis

| UI Access    | NA               |       |                                                                                                                                                           |
|--------------|------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name             | Type  | Description                                                                                                                                               |
|              | <Selections>     | Array | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify))                                          |
|              | <FlipParameters> | Array | Array("NAME: FlipParameters", _<br>"Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool) _ // [Opt=1] Should wires staircase |
| Return Value | None             |       |                                                                                                                                                           |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

| Python Syntax  | FlipHorizontal(         |
|----------------|-------------------------|
| Python Example | oEditor.FlipHorizontal( |

|  |                                                                                                                                                                            |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>["NAME:Selections", "Selections:=", [ _<br/>"CompInst@R;2;4", "CompInst@C;1;1"]],<br/>["NAME:FlipParameters",<br/>"Disconnect:=", true, "Rubberband:=", false])</pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | FlipHorizontal(<br>                                                                                                                                                                                                 |
|------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oEditor.FlipHorizontal<br/>Array("NAME:Selections", "Selections:=", Array( _<br/>"CompInst@R;2;4", "CompInst@C;1;1")),<br/>Array("NAME:FlipParameters",<br/>"Disconnect:=", true, "Rubberband:=", false)</pre> |

FlipVertical (Schematic Editor)

Flip items about the vertical (y) axis

| UI Access  | NA               |       |                                                                                                                  |
|------------|------------------|-------|------------------------------------------------------------------------------------------------------------------|
| Parameters | Name             | Type  | Description                                                                                                      |
|            | <Selections>     | Array | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify)) |
|            | <FlipParameters> | Array | Array("NAME:FlipParameters", _                                                                                   |

|                     |      |  |                                                                                                                        |
|---------------------|------|--|------------------------------------------------------------------------------------------------------------------------|
|                     |      |  | "Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool) _ // [Opt=1] Should wires staircase |
| <b>Return Value</b> | None |  |                                                                                                                        |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs.](#)

|                       |                                                                                                                                                                                     |
|-----------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | FlipVertical(                                                                                                                                                                       |
| <b>Python Example</b> | <pre>oEditor.FlipVertical (["NAME:Selections", "Selections:=", ["CompInst@R;2;4", _ "CompInst@C;1;1"]], ["NAME:FlipParameters", "Disconnect:=", true, "Rubberband:=", false])</pre> |

|                   |                                                                                                                                                                                                  |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | FlipVertical(                                                                                                                                                                                    |
| <b>VB Example</b> | <pre>oEditor.FlipVertical Array("NAME:Selections", "Selections:=", Array("CompInst@R;2;4", _ "CompInst@C;1;1")), Array("NAME:FlipParameters", "Disconnect:=", true, "Rubberband:=", false)</pre> |

Move (Schematic Editor)

Move items

|              |                  |       |                                                                                                                                                                                                                                                      |
|--------------|------------------|-------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | NA               |       |                                                                                                                                                                                                                                                      |
| Parameters   | Name             | Type  | Description                                                                                                                                                                                                                                          |
|              | <Selections>     | Array | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify))                                                                                                                                     |
|              | <MoveParameters> | Array | Array("NAME:MoveParameters", _<br>"xdelta:=", double _ // X distance to move<br>"ydelta:=", double _ // Y distance to move<br>"Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool) _ // [Opt=1] Should wires staircase |
| Return Value | None             |       |                                                                                                                                                                                                                                                      |

Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                |                                                                     |
|----------------|---------------------------------------------------------------------|
| Python Syntax  | Move()                                                              |
| Python Example | <code>oEditor.Move (["NAME:elements", "circle_0", "rect_2"],</code> |

|  |                               |
|--|-------------------------------|
|  | [0.0165613577023499, -0.001]) |
|--|-------------------------------|

|                   |                                                                                                             |
|-------------------|-------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Move()                                                                                                      |
| <b>VB Example</b> | <pre>oEditor.Move Array("NAME:elements", "circle_0", "rect_2"),<br/>Array(0.0165613577023499, -0.001)</pre> |

## NameNets (Schematic Editor)

Change names of nets on the schematic

|                     |                           |      |             |
|---------------------|---------------------------|------|-------------|
| <b>UI Access</b>    | Schematic>Auto-Name Wires |      |             |
| <b>Parameters</b>   | Name                      | Type | Description |
|                     | None                      |      |             |
| <b>Return Value</b> | None                      |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                                                                                                                                                                                  |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <pre>NameNets(<br/>    bool display, // Show Net names or not<br/>    ["NAME:NetNames",_<br/>    "string:=", _ #Old name of the net (e.g. net_1)<br/>    string, _ #New name of the net (e.g. my_net)<br/>    ...]) # Repeat for additional nets to change</pre> |
| <b>Python Example</b> | <pre>oEditor.NameNets (True, ["NAME:NetNames",<br/>    "Wire@net_2;41:1:=", "A[0]"])</pre>                                                                                                                                                                       |

|                   |                                                                                                                                                                                                                                                                           |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre>NameNets(<br/>    bool display, // Show Net names or not<br/>    Array("NAME:NetNames",_<br/>    "string:=", _ // Old name of the net (e.g. net_1)<br/>    string, _ // New name of the net (e.g. my_net)<br/>    ...) // Repeat for additional nets to change</pre> |
| <b>VB Example</b> | <pre>oEditor.NameNets true, Array("NAME:NetNames",<br/>    "Wire@net_2;41:1:=", "A[0]")</pre>                                                                                                                                                                             |

## PageBorders (Schematic Editor)

### Note:

This command has been replaced with the [SetPageData](#) command. Legacy scripts using this command will play back for the first page of a schematic only.

Sets up visible page borders on the schematic

```
PageBorders (
Array("NAME:Options", _
"PageSize:=", _
Array("x:=", string, _ // Width of page border with units
"y:=", string), _ // Height of page border with units
"PageMargins:=", _
Array("x:=", string, _ // Margin Width with units
"y:=", string), _ // Margin Height with units
"ZonesHoriz:=", int, _ // Number of Horizontal zones
"ZonesVert:=", int)) // Number of Vertical zones
```

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

## Pan (Schematic Editor)

Pan Display

|                     |          |      |             |
|---------------------|----------|------|-------------|
| <b>UI Access</b>    | View>Pan |      |             |
| <b>Parameters</b>   | Name     | Type | Description |
|                     | None     |      |             |
| <b>Return Value</b> | None     |      |             |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <pre>Pan([<br/>double, _ // Move the visible area by X meters<br/>double]) // Move the visible area by Y meters</pre> |
| <b>Python Example</b> | <pre>oDefinitionEditor.Pan<br/>([-0.00922653869663931, 0.00686839904222147])</pre>                                    |

|                   |                                                                                                                            |
|-------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre>Pan(Array(<br/>double, _ // Move the visible area by X meters<br/>double)) // Move the visible area by Y meters</pre> |
| <b>VB Example</b> | <pre>oDefinitionEditor.Pan</pre>                                                                                           |



|                                                  |
|--------------------------------------------------|
| Array(-0.00922653869663931, 0.00686839904222147) |
|--------------------------------------------------|

## Paste (Schematic Editor)

Paste copied items

| UI Access    | Edit>Paste |      |             |
|--------------|------------|------|-------------|
| Parameters   | Name       | Type | Description |
|              | None       |      |             |
| Return Value | None       |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

| Python Syntax  | Paste(VARIANT attrs)# specified attributes                                                                                                       |
|----------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.Paste (["NAME:Attributes", "Page:=", 1, "X:=", -0.059131200000001, "Y:=", _ 0.030784799999999, "Angle:=", 0, "Flip:=",false])</pre> |

| VB Syntax  | Paste(VARIANT attrs); // specified attributes                                                      |
|------------|----------------------------------------------------------------------------------------------------|
| VB Example | <pre>oEditor.Paste Array("NAME:Attributes", "Page:=", 1, "X:=", -0.059131200000001, "Y:=", _</pre> |

|                                                   |
|---------------------------------------------------|
| 0.030784799999999, "Angle:=", 0, "Flip:=", false) |
|---------------------------------------------------|

## PasteData [Schematic Editor]

**Use:** Paste graphic object or component properties to another object or component.

**Command:** Edit> Paste Data (or Right-click on graphic object or component and select **Paste Data**)

**Syntax:** PasteData Array("NAME:Selections", "Selections:=", Array(<selections>), "SelectList:=", Array(<select\_prop\_display>, <select\_parameters>), "Exclude:=", Array(<ExcludeList>))

**Return Value:** None.

**Parameters:** Selections

Type: Comma separated component instance list where user wants to paste data

Identifiers for each selected item to be copied, in the form CompInst@<id>, where id is the compname;ID;schematicID for each element.

SelectList

Type: Comma separated Boolean list where:

<select\_prop\_display> true to paste property displays

<select\_parameters> true to paste parameters

<ExcludeList>

Type: Comma separated list of property names to exclude from being pasted.

**Example:** Set oEditor = oDesign.SetActiveEditor("SchematicEditor")

```
oEditor.PasteData Array("NAME:Selections", "Selections:=", Array("CompInst@Cap;2;3"),
"SelectList:=", Array(true, true), "Exclude:=", Array("IC"))
```

## PasteDesign (Schematic Editor)

Paste a design that has already been copied to the clipboard into schematic as a subdesign.

*Syntax:*

```
PasteDesign(
 pasteOption , // see below for values
 Array("NAME:Attributes", _
 "Page:=", int, _ // [Opt=1] Page number (one-based)
 // Note: Page 1 always exists
 "X:=", double, _ // X position of the object
 "Y:=", double, _ // Y position of the object
 "Angle:=", double, _ // Rotation angle (radians)
 "Flip:=", bool) // True if mirrored
)
```

*Return Value:* Component is created and the returned value has the format `CompInst@<CompName>;<compInstID>;<optional schematicID>`

For instance a return value `CompInst@CAP_;4;35` indicates that the component name is `CAP_`, its ID is 5, and its Schematic ID is 35. These details can be found in the Properties window when you select the created component. The component name and ID can be found in the Gen-eral tab and the Schematic ID can be found in the Symbols tab.

*Parameters:* `pasteOption` should be one of the following:

0 to link to the existing design that was copied

1 to create a new copy of the design that was copied but keep the original layers of the design being copied

2 to create a new copy of the design and merge the layers of the design being copied into the design receiving the copy

*VB Example:*

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
oDesktop.RestoreWindow
Set oProject = oDesktop.SetActiveProject("SimpleCircuitPaste")
oProject.CopyDesign "D"
Set oDesign = oProject.SetActiveDesign("A")
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
oEditor.PasteDesign 1, Array("NAME:Attributes", "Page:=", 1, "X:=", 0.0762, "Y:=", _
0.0635, "Angle:=", 0, "Flip:=", false)
```

## PushExcitations

Allows access to computed excitations for transient and linear frequency solutions. The script command can be accessed from three locations.

|                   |                                                                                                                                                                                                     |             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | <ul style="list-style-type: none"> <li>Layout Editor</li> <li>Schematic Editor</li> <li>Select a Nexxim solution in a 3D Layout design, right click, and choose <b>Push Excitations</b>.</li> </ul> |             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>Parameters</b> | <b>Name</b>                                                                                                                                                                                         | <b>Type</b> | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|                   | <Reference Designation>                                                                                                                                                                             | String      | "<refdes>"<br><br>This argument is empty when excitations are pushed from a Nexxim solution.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |
|                   | <PushExcitations Parameters>                                                                                                                                                                        | Array       | <p>This parameter changes depending on whether the excitations comes from a transient or linear frequency solution. The keyword "transient:=" indicates to AEDT which solution generated the excitations. If "transient:=" is present, "CalcThevenin =" and its value are ignored.</p> <p>For a linear frequency solution, use this array:</p> <pre>Array("NAME:options",       "CalcThevenin =" , &lt;true false&gt;,       "Sol:=", "&lt;solution name&gt;")</pre> <p>If CalcThevenin is true, Thevenin's equivalent is calculated. Parameters for the linear frequency solution do not include a Freq argument, so all frequencies from the solution are used.</p> <p>For a transient solution, use:</p> <pre>Array("NAME:options",       "transient:=", Array(</pre> |

|              |      |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|--------------|------|--|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              |      |  | <pre>"start:=", &lt;start time&gt;, "stop:=", &lt;stop time&gt;, "maxHarmonics:=", &lt;max harmonics&gt;, "winType:=", &lt;window&gt;, ["widthPct:=", &lt;width percentage&gt;], ["kaiser:=", &lt;Kaiser value&gt;], ["correctCoherentGain:=", true]], "Sol:=", "&lt;solution name&gt;")</pre> <p>winType can have the following values:</p> <ul style="list-style-type: none"><li>• Rectangular</li><li>• Bartlett</li><li>• Blackman</li><li>• Hamming</li><li>• Hanning</li><li>• Kaiser</li><li>• Welch</li><li>• Weber</li><li>• Lanczos</li></ul> |
| Return Value | None |  |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre>PushExcitations "&lt;refdes&gt;", Array("NAME:options", "transient:=", ["CalcThevenin =", &lt;true&gt;], Array("start:=", &lt;start time&gt;, "stop:=", &lt;stop time&gt;, "maxHarmonics:=", &lt;max harmonics&gt;, "winType:=", &lt;window&gt;, ["widthPct:=", &lt;width percentage&gt;,"kaiser:=", &lt;Kaier value&gt;," ["correctCoherentGain:=", true]), "Sol:=", "&lt;solution name&gt;")</pre>                                                                                                                                                                                                                          |
| <b>VB Example</b> | <p>For a transient solution:</p> <pre>Set oEditor = oDesign.SetActiveEditor("Layout") oEditor.PushExcitations "U3", Array("NAME:options", _ "transient:=", Array("start:=", 0, "stop:=", 5E-005, _ "maxHarmonics:=", 100, "winType:=", "Rectangular", _ "widthPct:=", 100, "kaiser:=", 0, "correctCoherentGain:=", _ true), "Sol:=", "Transient")  Set oDesign = oProject.SetActiveEditor("Design1") oDesign.PushExcitations "", Array("NAME:options", _ "transient:=", Array("start:=", 0, "stop:=", 1E-08, _ "maxHarmonics:=", 100, "winType:=", "Hamming", _ "widthPct:=", 100, "kaiser:=", 0, "correctCoherentGain:=", _</pre> |

|  |                                                                                                                                                                                                                                                        |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>true), "Sol:=", "Transient Setup 1")  For a linear frequency solution:  Set oEditor = oDesign.SetActiveEditor("SchematicEditor") oEditor.PushExcitations "S1", Array("NAME:options", _ "CalcThevenin:=", false, "Sol:=", "LinearFrequency")</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

Rotate (Schematic Editor)

Rotate items

| UI Access    | Draw>Rotate        |       |                                                                                                                                                                                                                                                           |
|--------------|--------------------|-------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name               | Type  | Description                                                                                                                                                                                                                                               |
|              | <Selections>       | Array | Array("NAME:Selections", _<br>"Page:=", page number, _ // [Opt=1] Page number<br>"Selections:=", IDs to modify))                                                                                                                                          |
|              | <RotateParameters> | Array | Array("NAME:RotateParameters", _<br>"Disconnect:=", bool _ // [Opt=0] Should wires disconnect<br>"Rubberband:=", bool _ // [Opt=1] Should wires staircase<br>"Angle:=", double) // [Opt=/2] Angle to rotate<br>Note: Rotation occurs around center of ids |
| Return Value | None               |       |                                                                                                                                                                                                                                                           |



**Note:**

When you record a script, the schematic editor's Rotate command shows the rotation angle for some older scripts in radians, while newer scripts show rotation angle in degrees.

```
oEditor.Rotate Array("NAME:Selections", "Selections:=", _
Array("CompInst@RES_1;1:1"), Array("NAME:RotateParameters", _
"Angle:=", 1.5707963267949, "Disconnect:=", false, "Rubberband:=", false)
```

If "Degrees=xxx" is not shown, the rotation must be scaled.

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs.](#)

| Python Syntax  | Rotate()                                                                                                                                                                                                      |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oEditor.Rotate (["NAME:Selections", "Selections:=",<br/>["CompInst@R;2;4", "CompInst@C;1;1"]],<br/>["NAME:RotateParameters", "Degrees:=", 90, "Disconnect:=", _<br/>false, "Rubberband:=", false])</pre> |

|                   |                                                                                                                                                                                                                            |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | Rotate()                                                                                                                                                                                                                   |
| <b>VB Example</b> | <pre>oEditor.Rotate Array("NAME:Selections", "Selections:=",<br/>Array("CompInst@R;2;4", "CompInst@C;1;1")),<br/>Array("NAME:RotateParameters", "Degrees:=", 90, "Disconnect:=", _<br/>false, "Rubberband:=", false)</pre> |

### SelectAll (Schematic Editor)

Select all elements on the given page.

|                     |                 |      |             |
|---------------------|-----------------|------|-------------|
| <b>UI Access</b>    | Edit>Select All |      |             |
| <b>Parameters</b>   | Name            | Type | Description |
|                     | None            |      |             |
| <b>Return Value</b> | None            |      |             |

#### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                                                                                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <p>SelectAll(int pageNum) # Page number</p> <p>The first page number is 1. If an invalid page number is passed in (i.e. 0, -1, etc), SelectAll will use the active page on the currently active view. If there is no active view, the first page, page 1, is used.</p> |
| <b>Python Example</b> | <pre>oEditor.SelectAll(1)</pre>                                                                                                                                                                                                                                        |

|                   |                                                                                                                                                                                                                                                                               |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SelectAll(int pageNum) // Page number<br><br>The first page number is 1. If an invalid page number is passed in (i.e. 0, -1, etc), <code>SelectAll</code> will use the active page on the currently active view. If there is no active view, the first page, page 1, is used. |
| <b>VB Example</b> | <code>oEditor.SelectAll 1</code>                                                                                                                                                                                                                                              |

### SelectPage (Schematic Editor)

Select a page in the UI

|                     |      |      |             |
|---------------------|------|------|-------------|
| <b>UI Access</b>    | NA   |      |             |
| <b>Parameters</b>   | Name | Type | Description |
|                     | None |      |             |
| <b>Return Value</b> | None |      |             |

#### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                    |
|-----------------------|------------------------------------|
| <b>Python Syntax</b>  | SelectPage(int page) # Page number |
| <b>Python Example</b> | <code>oEditor.SelectPage(1)</code> |

|                  |                                     |
|------------------|-------------------------------------|
| <b>VB Syntax</b> | SelectPage(int page) // Page number |
|------------------|-------------------------------------|

|            |                      |
|------------|----------------------|
| VB Example | oEditor.SelectPage 1 |
|------------|----------------------|

SetPageData [Schematic Editor]

Note:

This command replaces the [PageBorders](#) command.

To add or remove page border, title block and page property data from the specified page. Title blocks will not be shown if the page border type is "None". Any number of title blocks may be specified, as well as any number of page properties. Some of the page properties are special and their values will be updated automatically as appropriate, such as Project, ProjectPath, Design, Number and Date.

| UI Access  | Schematic>Page Borders and Title Blocks... |                        |                                                                                                      |
|------------|--------------------------------------------|------------------------|------------------------------------------------------------------------------------------------------|
| Parameters | Name                                       | Type                   | Description                                                                                          |
|            | <bordertype>                               | String                 | One of: "None", "Outline", "ANSI", "ISO", "DIN"                                                      |
|            | <sizeX>                                    | Double<br>(with units) | Size of X axis                                                                                       |
|            | <sizeY>                                    | Double<br>(with units) | Size of Y axis                                                                                       |
|            | <marginX>                                  | Double<br>(with units) | Margin of X axis                                                                                     |
|            | <marginY>                                  | Double<br>(with units) | Margin of Y axis                                                                                     |
|            | <zonesH>                                   | Integer                |                                                                                                      |
|            | <zonesV>                                   | Integer                |                                                                                                      |
|            | <symbolName>                               | String                 | The name of the title block symbol.                                                                  |
|            | <libName>                                  | String                 | The name of the library (with path from library directory root) that contains the title block symbol |

|                     |                               |        |                                              |
|---------------------|-------------------------------|--------|----------------------------------------------|
|                     | <code>&lt;Hsnap&gt;</code>    | String | One of: "Left", "Right", "Center", "None"    |
|                     | <code>&lt;Vsnap&gt;</code>    | String | One of "Top", "Bottom", "Center", "None"     |
|                     | <code>&lt;rotation&gt;</code> | String | One of "0", "90", "270"                      |
|                     | <code>&lt;propName&gt;</code> | String | The name of a page property followed by ":=" |
|                     | <code>&lt;propVal&gt;</code>  | String | The value of a page property                 |
| <b>Return Value</b> | None                          |        |                                              |

|                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <pre> SetPageData (&lt;pagenum&gt;, ["NAME:Border", "BorderType:=", &lt;bordertype&gt;, "PageSize:=", ["x:=", &lt;sizeX&gt;, "y:=", &lt;sizeY&gt;], "PageMargins:=", ["x:=", &lt;marginX&gt;, "y:=", &lt;marginY&gt;], "ZonesHoriz:=", &lt;zonesH&gt;, "ZonesVert:=", &lt;zonesV&gt;], ["NAME:TitleBlocks"[, "TitleBlock:=", [&lt;symbolName&gt;, &lt;libName&gt;, &lt;Hsnap&gt;, &lt;Vsnap&gt;, &lt;rotation&gt;, -1]*], ["NAME:PageProps"[, &lt;propName&gt;, &lt;propVal&gt;]*]) </pre> |
| <b>Python Example</b> | <pre> oEditor.SetPageData (1, ["NAME:Border", "BorderType:=", "ANSI", "PageSize:=", ["x:=", "11in", "y:=", "8.5in"], "PageMargins:=", ["x:=", "0.38in", "y:=", "0.25in"], "ZonesHoriz:=", 2, "ZonesVert:=", 2], ["NAME:TitleBlocks", "TitleBlock:=", ["TitleBlk1", "TitleBlocks", "Right", "Bottom", "0", -1]], ["NAME:PageProps", </pre>                                                                                                                                                  |

|  |                                                                                                                                                                                                              |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"Title:=", "Design Title", "Author:=", "I. M. User", "Number:=", "1", "Project:=", "Project8", "ProjectPath:=", "e:/projects/", "Design:=", "TwinBuilder1", "Date:=", "9:53:11 PM Jan 27, 2016"])</pre> |
|--|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                   |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre>SetPageData (&lt;pagenum&gt;, Array("NAME:Border", "BorderType:=", &lt;bordertype&gt;, "PageSize:=", Array("x:=", &lt;sizeX&gt;, "y:=", &lt;sizeY&gt;), "PageMargins:=", Array("x:=", &lt;marginX&gt;, "y:=", &lt;marginY&gt;),"ZonesHoriz:=", &lt;zonesH&gt;, "ZonesVert:=", &lt;zonesV&gt;), Array("NAME:TitleBlocks"[, "TitleBlock:=", Array(&lt;symbolName&gt;, &lt;libName&gt;, &lt;Hsnap&gt;, &lt;Vsnap&gt;, &lt;rotation&gt;, -1)]*), Array("NAME:PageProps"[, &lt;propName&gt;, &lt;propVal&gt;]*)</pre> |
| <b>VB Example</b> | <pre>Dim oEditor oEditor.SetPageData 1, Array("NAME:Border", "BorderType:=", "ANSI", "PageSize:=", Array("x:=", "11in", "y:=", "8.5in"), "PageMargins:=", Array("x:=", "0.38in", "y:=", "0.25in"), "ZonesHoriz:=", 2, "ZonesVert:=", 2), Array("NAME:TitleBlocks", "TitleBlock:=", Array("TitleBlk1",</pre>                                                                                                                                                                                                           |

|  |                                                                                                                                                                                                                                                                                  |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre>"TitleBlocks", "Right", "Bottom", "0", -1)), Array("NAME:PageProps", "Title:=", "Design Title", "Author:=", "I. M. User", "Number:=", "1", "Project:=", "Project8", "ProjectPath:=", "e:/projects/", "Design:=", "TwinBuilder1", "Date:=", "9:53:11 PM Jan 27, 2016")</pre> |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## SendToBack

Send the selected object to the back of the view

|                     |                   |        |                             |
|---------------------|-------------------|--------|-----------------------------|
| <b>UI Access</b>    | Draw>Send To Back |        |                             |
| <b>Parameters</b>   | Name              | Type   | Description                 |
|                     | <Object>          | String | Object to send to the back. |
| <b>Return Value</b> | None              |        |                             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#)

|                       |                                                                                             |
|-----------------------|---------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SendToBack(["NAME:Selections", "Selections:=", [<Object>, <Object>, ...]])                  |
| <b>Python Example</b> | <pre>oDefinitionEditor.SendToBack(["NAME:Selections", "Selections:=", ["SchObj@10"]])</pre> |

|                  |                                                                                      |
|------------------|--------------------------------------------------------------------------------------|
| <b>VB Syntax</b> | SendToBack Array("NAME:Selections", "Selections:=", Array (<Object>, <Object>, ...)) |
|------------------|--------------------------------------------------------------------------------------|

|                   |                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>oDefinitionEditor.SendToBack Array("NAME:Selections",<br/>"Selections:=", Array( "SchObj@10"))</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------|

**ShowVariableBlock (Schematic Editor)**

*Use:* Control display of Variable Text Block in Schematic Editor.

*Command:* Schematic Ribbon > Settings > Design Variables

*Syntax:* ShowVariableBlock("True")      // Display Text Block  
          ShowVariableBlock("False")     // Hide Text Block

*Return Value:* None.

*Parameters:* True // Display Text Block  
              False // Hide Text Block

|                       |                                                                 |
|-----------------------|-----------------------------------------------------------------|
| <b>Python Syntax</b>  | ShowVariableBlock ()                                            |
| <b>Python Example</b> | <pre>oEditor ShowVariableBlock<br/>[<br/>  ("True")<br/>]</pre> |

**SortComponents (Schematic Editor)**

Sorts all components, or a block of components, using the specified method

|                  |                           |
|------------------|---------------------------|
| <b>UI Access</b> | Schematic>Sort Components |
|------------------|---------------------------|



| <b>Parameters</b>   | Name   | Type  | Description                                                                                                                                                                                                                                                                                                                    |
|---------------------|--------|-------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                     | <Type> | Array | <pre>Array(type, _ // string specifying sort type method)) // string specifying sort method type // string argument specifying sort type: "All Components" or "Blocks" // An empty string corresponds to "All Components" method // string argument specifying sort method: "By Name", "Left to Right", or "Signal Flow"</pre> |
| <b>Return Value</b> | None   |       |                                                                                                                                                                                                                                                                                                                                |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                        |
|-----------------------|----------------------------------------|
| <b>Python Syntax</b>  | SortComponents ()                      |
| <b>Python Example</b> | <code>oEditor.SortComponents ()</code> |

|                   |                                     |
|-------------------|-------------------------------------|
| <b>VB Syntax</b>  | SortComponents ()                   |
| <b>VB Example</b> | <code>oEditor.SortComponents</code> |

**Wire (Schematic Editor)**

*Use:* Wire specified pairs of pins in two selected component instances.

*Command:* Select two component instances in the Schematic Editor, then right click on one and choose **Wire**.

Syntax:

```
Wire (string firstComponent, _
string firstListOfPins, _// list of pins separated by space
string secondComponent, _
string secondListOfPins) //second list of pins separated by space
```

Return Value: None

```
VB Example: oEditor.Wire "CompInst@etch_sline_s20_z70_1;10;76", "A1 A2", "Com-
pInst@CX0805MRNPO9BB220;1;1", "B1 B2"
```

ZoomArea (Schematic Editor)

Zoom Area

|              |                |       |                                                                                                                                                                                                                                                                |
|--------------|----------------|-------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| UI Access    | View>Zoom Area |       |                                                                                                                                                                                                                                                                |
| Parameters   | Name           | Type  | Description                                                                                                                                                                                                                                                    |
|              | <Size>         | Array | Array(double, _<br><br>//Base X value of the area to zoom into double),<br><br>_ // Base Y value of the area to zoom into<br><br>Array(double, _ // X size of the area to zoom<br><br>double)) // Y Size of the area to zoom<br><br>NOTE: All values in meters |
| Return Value | None           |       |                                                                                                                                                                                                                                                                |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ZoomArea()                                                                                                            |
| <b>Python Example</b> | <pre>oEditor.ZoomArea ([0.00307569800375911, 0.021101611633739],<br/>[-0.050433530091516, -0.0220260723256825])</pre> |

|                       |                                                                                                                       |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | ZoomArea()                                                                                                            |
| <b>Python Example</b> | <pre>oEditor.ZoomArea ([0.00307569800375911, 0.021101611633739],<br/>[-0.050433530091516, -0.0220260723256825])</pre> |

## ZoomIn (Schematic Editor)

Zoom into the schematic

|                     |              |      |             |
|---------------------|--------------|------|-------------|
| <b>UI Access</b>    | View>Zoom In |      |             |
| <b>Parameters</b>   | Name         | Type | Description |
|                     | None         |      |             |
| <b>Return Value</b> | None         |      |             |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                               |
|-----------------------|-------------------------------|
| <b>Python Syntax</b>  | ZoomIn()                      |
| <b>Python Example</b> | <code>oEditor.ZoomIn()</code> |

|                   |                             |
|-------------------|-----------------------------|
| <b>VB Syntax</b>  | ZoomIn()                    |
| <b>VB Example</b> | <code>oEditor.ZoomIn</code> |

## ZoomOut (Schematic Editor)

Zoom out from the schematic

|                     |               |      |             |
|---------------------|---------------|------|-------------|
| <b>UI Access</b>    | View>Zoom Out |      |             |
| <b>Parameters</b>   | Name          | Type | Description |
|                     | None          |      |             |
| <b>Return Value</b> | None          |      |             |

**Note:**

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                |
|-----------------------|--------------------------------|
| <b>Python Syntax</b>  | ZoomOut()                      |
| <b>Python Example</b> | <code>oEditor.ZoomOut()</code> |

|                   |                              |
|-------------------|------------------------------|
| <b>VB Syntax</b>  | ZoomOut()                    |
| <b>VB Example</b> | <code>oEditor.ZoomOut</code> |

## ZoomPrevious (Schematic Editor)

Restore the zoom to the previous settings

|                     |      |      |             |
|---------------------|------|------|-------------|
| <b>UI Access</b>    | NA   |      |             |
| <b>Parameters</b>   | Name | Type | Description |
|                     | None |      |             |
| <b>Return Value</b> | None |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                     |
|-----------------------|-------------------------------------|
| <b>Python Syntax</b>  | ZoomPrevious()                      |
| <b>Python Example</b> | <code>oEditor.ZoomPrevious()</code> |

|                   |                                   |
|-------------------|-----------------------------------|
| <b>VB Syntax</b>  | ZoomPrevious()                    |
| <b>VB Example</b> | <code>oEditor.ZoomPrevious</code> |

## ZoomToFit (Schematic Editor)

Set the Zoom to Fit

|                     |                  |      |             |
|---------------------|------------------|------|-------------|
| <b>UI Access</b>    | View>Fit Drawing |      |             |
| <b>Parameters</b>   | Name             | Type | Description |
|                     | None             |      |             |
| <b>Return Value</b> | None             |      |             |

### Note:

For more information about the component name, schematic IDs, and formats, see the section [Editor Scripting IDs](#).

|                       |                                  |
|-----------------------|----------------------------------|
| <b>Python Syntax</b>  | ZoomToFit()                      |
| <b>Python Example</b> | <code>oEditor.ZoomToFit()</code> |

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Syntax</b>  | ZoomToFit()                    |
| <b>VB Example</b> | <code>oEditor.ZoomToFit</code> |

# Property Method List

This section presents the property script methods that are available.

[ChangeProperty \(Schematic Editor\)](#)

[GetEvaluatedPropertyValue \(Schematic Editor\)](#)

[GetProperties \(Schematic Editor\)](#)

[GetPropertyAttribute \[Schematic Editor\]](#)

[GetPropertyValue \(Schematic Editor\)](#)

[SetPropertyValue \(Schematic Editor\)](#)

## ChangeProperty (Schematic Editor)

Change a property

|              |       |      |             |
|--------------|-------|------|-------------|
| UI Access    | NA    |      |             |
| Parameters   | Name  | Type | Description |
|              | Name  | Type | Description |
| Return Value | Value |      |             |

|                |                                                                                                                                                                                         |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | ChangeProperty()                                                                                                                                                                        |
| Python Example | <pre>oEditor.ChangeProperty(["NAME:AllTabs",<br/>["NAME:ComponentTab", ["NAME:PropServers", _<br/>"Wire@net_1;14:1"], ["NAME:ChangedProps",<br/>["NAME:NetName", "ExtraText:=", _</pre> |

|  |                                                                      |
|--|----------------------------------------------------------------------|
|  | <code>"net_10", "Name:=", "net_10", "SplitWires:=", false]]])</code> |
|--|----------------------------------------------------------------------|

| VB Syntax  | ChangeProperty()                                                                                                                                                                                                                                                                     |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>oEditor.ChangeProperty Array("NAME:AllTabs",<br/>Array("NAME:ComponentTab", Array("NAME:PropServers", _<br/>    "Wire@net_1;14:1"), Array("NAME:ChangedProps",<br/>Array("NAME:NetName", "ExtraText:=", _<br/>    "net_10", "Name:=", "net_10", "SplitWires:=", false))))</pre> |

**GetEvaluatedPropertyValue (Schematic Editor)**

Get value.

*Command:* None.

*Syntax:* GetEvaluatedPropertyValue<tabDescription, componentID, propName>

*Return Value:* Evaluated value of variable property in double format.

*Parameters:* tabDescription = name of property tab where property is found  
componentID = id of component instance where property is found,  
in the format: "Complnst@<name of component type>;<id of complnstance>"  
propName = name of the variable property

*VB Example:*

```
dim info
```



```
evalValue = oEditor.GetEvaluatedPropertyValue("PassedParameterTab", "CompInst@CAP_3", "C")
```

**Notes:**

1. This function is only available with the schematic editor.
2. Calling this function on non-numeric properties (e.g. Text properties) returns 0.

## GetProperties (Schematic Editor)

To get the names of properties for a specified tab and element.

| UI Access    | NA                                   |        |                                                                                                                  |
|--------------|--------------------------------------|--------|------------------------------------------------------------------------------------------------------------------|
| Parameters   | Name                                 | Type   | Description                                                                                                      |
|              | <tab>                                | String | One of six choices, "PassedParameterTab", "ComponentTab", "Component", "BaseElementTab", "Quantities", "Signals" |
|              | <objectId>                           | String | Instance ID of object with the property                                                                          |
| Return Value | Array of strings, the property names |        |                                                                                                                  |

| Python Syntax  | GetProperties(<tab>, <objectId>)                                                                                                                                                                                                               |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>comparray = oEditor.GetAllComponents() for i in comparray:     instval = oEditor.GetPropertyvalue         ("PassedParameterTab", i, "InstanceName")     proparray = oEditor.GetProperties ("Quantities", i)     for j in proparray:</pre> |

|  |                                                                                                                                                     |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------|
|  | <pre> direction = oEditor.GetPropertyAttribute ("Quantities",i, j, "Direction")  AddInfoMessage(str(j) + " = (direction) " + str(direction)) </pre> |
|--|-----------------------------------------------------------------------------------------------------------------------------------------------------|

| VB Syntax  | GetProperties(<tab>, <objectId>)                                                                                                                                                                                                                                                                                                                                                              |
|------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> comparray = oEditor.GetAllComponents for each i in comparray     instval = oEditor.GetPropertyValue ("PassedParameterTab",i,"InstanceName")     proparray = oEditor.GetProperties ("Quantities",i)     for each j in proparray         direction = oEditor.GetPropertyAttribute ("Quantities",i, j, "Direction")          MsgBox j + " = (direction) " + direction     next next </pre> |

## GetPropertyAttribute [Schematic Editor]

To find the value of an attribute of a specified property.

|                     |                                  |             |                                                                                                                  |
|---------------------|----------------------------------|-------------|------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | NA                               |             |                                                                                                                  |
| <b>Parameters</b>   | <b>Name</b>                      | <b>Type</b> | <b>Description</b>                                                                                               |
|                     | <tab>                            | String      | One of six choices, "PassedParameterTab", "ComponentTab", "Component", "BaseElementTab", "Quantities", "Signals" |
|                     | <objectId>                       | String      | Instance ID of object with the property                                                                          |
|                     | <propName>                       | String      | The name of the property                                                                                         |
|                     | <attribute>                      | String      | Currently, only the "Direction" attribute is supported. Return values will be one of "In", "Out" or "InOut"      |
| <b>Return Value</b> | A string, the attribute's value. |             |                                                                                                                  |

|                       |                                                                                                                                                                                                                                                                                                                                                                                                |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetPropertyAttribute(<tab>, <objectId>, <propName>, <attribute>)                                                                                                                                                                                                                                                                                                                               |
| <b>Python Example</b> | <pre> comparray = oEditor.GetAllComponents() for i in comparray:     instval = oEditor.GetPropertyValue         ("PassedParameterTab",i,"InstanceName")     proparray = oEditor.GetProperties ("Quantities",i)     for j in proparray:         direction = oEditor.GetPropertyAttribute             ("Quantities",i, j, "Direction")         AddInfoMessage(str(j)+ " = (direction) " + </pre> |

|  |                              |
|--|------------------------------|
|  | <code>str(direction))</code> |
|--|------------------------------|

|                   |                                                                                                                                                                                                                                                                                                                                                                   |
|-------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <code>GetPropertyAttribute(&lt;tab&gt;, &lt;objectId&gt;, &lt;propName&gt;, &lt;attribute&gt;)</code>                                                                                                                                                                                                                                                             |
| <b>VB Example</b> | <pre>comparray = oEditor.GetAllComponents for each i in comparray  instval = oEditor.GetPropertyValue ("PassedParameterTab",i,"InstanceName")  proparray = oEditor.GetProperties ("Quantities",i)  for each j in proparray  direction = oEditor.GetPropertyAttribute("Quantities",i, j, "Direction")   MsgBox j + " = (direction) " + direction  next  next</pre> |

**GetPropertyValue (Schematic Editor)**

Get a property

|            |                                                                        |             |  |      |      |             |
|------------|------------------------------------------------------------------------|-------------|--|------|------|-------------|
| UI Access  | NA                                                                     |             |  |      |      |             |
| Parameters | <table><tr><td>Name</td><td>Type</td><td>Description</td></tr></table> |             |  | Name | Type | Description |
| Name       | Type                                                                   | Description |  |      |      |             |

|                     |            |        |                       |
|---------------------|------------|--------|-----------------------|
|                     | <tab>      | String | Tab with the property |
|                     | <item>     | String | Name of the object    |
|                     | <propname> | String | Name of the property  |
|                     | <value>    | String | Value of the property |
| <b>Return Value</b> | None       |        |                       |

|                       |                                                   |
|-----------------------|---------------------------------------------------|
| <b>Python Syntax</b>  | GetPropertyValue(<tab>,<item>,<propname>,<value>) |
| <b>Python Example</b> | None                                              |

|                   |                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetPropertyValue(<tab>,<item>,<propname>,<value>)                                                                                                      |
| <b>VB Example</b> | Please refer to the examples that can be found in C:\Program Files\...[Ansys Installation]...\Examples\Twin Builder\Applications\Scripting\Ex6_MathMod |

## SetPropertyValue (Schematic Editor)

Set a property.

|                     |            |        |                               |
|---------------------|------------|--------|-------------------------------|
| <b>UI Access</b>    | NA         |        |                               |
| <b>Parameters</b>   | Name       | Type   | Description                   |
|                     | <tab>      | String | Tab with the property         |
|                     | <item>     | String | Name of the objects           |
|                     | <propname> | String | Name of the property          |
|                     | <value>    | String | The new value of the property |
| <b>Return Value</b> | None       |        |                               |

|                       |                            |
|-----------------------|----------------------------|
| <b>Python Syntax</b>  | SetPropertyValue(<params>) |
| <b>Python Example</b> | None                       |

|                   |                                                                                                                                                        |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetPropertyValue(<params>)                                                                                                                             |
| <b>VB Example</b> | Please refer to the examples that can be found in C:\Program Files\...[Ansys Installation]...\Examples\Twin Builder\Applications\Scripting\Ex6_MathMod |

## Information Method List

This section presents the information methods that are available.

[GetCompInstanceFromRefDes \(Schematic Editor\)](#)

[GetComponentInfo \(Schematic Editor\)](#)

[GetComponentPins \(Schematic Editor\)](#)

[GetComponentPinInfo \(Schematic Editor\)](#)

[GetComponentPinLocation \(Schematic Editor\)](#)

[GetEditorName \(Schematic Editor\)](#)

[GetNetConnections \(Schematic Editor\)](#)

[GetNumPages \(Schematic Editor\)](#)

[GetPortInfo \(Schematic Editor\)](#)

[GetSelections \(Schematic Editor\)](#)

[GetSignals \(Schematic Editor\)](#)**GetAllPorts**

Get all ports in a design.

|                     |                     |      |             |
|---------------------|---------------------|------|-------------|
| <b>UI Access</b>    | NA                  |      |             |
| <b>Parameters</b>   | Name                | Type | Description |
|                     | None                |      |             |
| <b>Return Value</b> | Ports in the design |      |             |

|                       |                                    |  |  |
|-----------------------|------------------------------------|--|--|
| <b>Python Syntax</b>  | GetAllPorts()                      |  |  |
| <b>Python Example</b> | <code>oEditor.GetAllPorts()</code> |  |  |

|                   |                                                                                           |  |  |
|-------------------|-------------------------------------------------------------------------------------------|--|--|
| <b>VB Syntax</b>  | GetAllPorts                                                                               |  |  |
| <b>VB Example</b> | <pre>Set oEditor = oDesign.SetActiveEditor("SchematicEditor") oEditor.GetAllPorts()</pre> |  |  |

**GetCompInstanceFromRefDes (Schematic Editor)**

Informational

|                   |      |      |             |
|-------------------|------|------|-------------|
| <b>UI Access</b>  | NA   |      |             |
| <b>Parameters</b> | Name | Type | Description |
|                   |      |      |             |

|                     |                                                                                         |                       |        |                       |
|---------------------|-----------------------------------------------------------------------------------------|-----------------------|--------|-----------------------|
|                     | <table><tr><td>Component</td><td>String</td><td>Name of the Component</td></tr></table> | Component             | String | Name of the Component |
| Component           | String                                                                                  | Name of the Component |        |                       |
| <b>Return Value</b> | Returns IDispatch for CompInstance                                                      |                       |        |                       |

|                       |                                                                                       |
|-----------------------|---------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <pre>GetCompInstanceFromRefDes(string)<br/># string is refDes for the component</pre> |
| <b>Python Example</b> | <pre>GetCompInstanceFromRefDes ("R1 ")</pre>                                          |

|                   |                                                                                        |
|-------------------|----------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <pre>GetCompInstanceFromRefDes(string)<br/>// string is refDes for the component</pre> |
| <b>VB Example</b> | <pre>GetCompInstanceFromRefDes ("R1 ")</pre>                                           |

## GetComponentInfo (Schematic Editor)

Informational

|              |                                                             |        |                                                                                   |
|--------------|-------------------------------------------------------------|--------|-----------------------------------------------------------------------------------|
| UI Access    | NA                                                          |        |                                                                                   |
| Parameters   | Name                                                        | Type   | Description                                                                       |
|              | <compID>                                                    | String | The Schematic component’s ID, obtained from GetSelections or through FindElements |
| Return Value | Array of Strings, as follows:<br><br>"Page= <i>number</i> " |        |                                                                                   |



|  |                                                                                                                                                                                                                                                                                                                  |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | "ComponentName= <i>string</i> "<br>"GateCount= <i>number</i> "<br>"LocationX= <i>number</i> "<br>"LocationY= <i>number</i> "<br>"BBoxLLx= <i>number</i> "<br>"BBoxLLy= <i>number</i> "<br>"BBoxURx= <i>number</i> "<br>"BBoxURy= <i>number</i> "<br>"Angle= <i>number</i> " (in degrees)<br>"Flip=true or false" |
|--|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

| Python Syntax  | GetComponentInfo(<compID>)                                                                                                                                                            |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>For i in selectionArray:     compArray = oEditor.GetComponentInfo(i)     For Each k in compArray:         AddInfoMessage(str(k) +             "Got with GetComponentInfo")</pre> |

| VB Syntax | GetComponentInfo<compID> |
|-----------|--------------------------|
|-----------|--------------------------|

|                   |                                                                                                                                                                                                              |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>selectionArray = oEditor.GetSelections For Each i in selectionArray     compArray = oEditor.GetComponentInfo(i)     For Each k in compArray         MsgBox k, 0, "GetComponentInfo"     Next Next</pre> |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**GetComponentPins (Schematic Editor)**

Informational

|                     |                                                                   |        |                                                                                    |
|---------------------|-------------------------------------------------------------------|--------|------------------------------------------------------------------------------------|
| <b>UI Access</b>    | NA                                                                |        |                                                                                    |
| <b>Parameters</b>   | Name                                                              | Type   | Description                                                                        |
|                     | <compID>                                                          | String | The Schematic component's ID, obtained from GetSelections or through FindElements. |
| <b>Return Value</b> | Array of strings, which are the names of all the component's pins |        |                                                                                    |

|                       |                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetComponentPins(<compID>)                                                                                             |
| <b>Python Example</b> | <pre>selectionArray = oEditor.GetSelections() for i in selectionArray     pinArray = oEditor.GetComponentPins(i)</pre> |

|  |                                                                     |
|--|---------------------------------------------------------------------|
|  | <pre> for j in pinArray     AddInfoMessage("Pin: " + str(j)) </pre> |
|--|---------------------------------------------------------------------|

| VB Syntax         | GetComponentPins<compID>                                                                                                                                                                                     |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre> selectionArray = oEditor.GetSelections for Each i in selectionArray     pinArray = oEditor.GetComponentPins(i)     for each j in pinArray         MsgBox j, 0, "GetComponentPins"     Next Next </pre> |

### GetComponentPinInfo (Schematic Editor)

Informational - gets details about a pin on a schematic component.

| UI Access           | NA                                                             |        |                              |
|---------------------|----------------------------------------------------------------|--------|------------------------------|
| <b>Parameters</b>   | Name                                                           | Type   | Description                  |
|                     | <compID>                                                       | String | The Schematic component's ID |
|                     | <pinName>                                                      | String | The pin name                 |
| <b>Return Value</b> | String values, as follows:<br><br>"X=number"<br><br>"Y=number" |        |                              |

|  |                                                                                                                                                                                              |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | "Angle=number" (in degrees)<br><br>"Flip=true or false"<br><br>"WireId=netID or empty string" (netID is the identifier of the schematic net, and can be used in other informational methods) |
|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                       |                                                               |
|-----------------------|---------------------------------------------------------------|
| <b>Python Syntax</b>  | GetComponentPinInfo( <compID>, <pinName>)                     |
| <b>Python Example</b> | <pre>info = oEditor.GetComponentPinInfo<br/>("1", "N1")</pre> |

|                   |                                                               |
|-------------------|---------------------------------------------------------------|
| <b>VB Syntax</b>  | GetComponentPinInfo <compID>, <pinName>                       |
| <b>VB Example</b> | <pre>info = oEditor.GetComponentPinInfo<br/>("1", "N1")</pre> |

## GetComponentPinLocation [Schematic Editor]

Informational - gets the X or Y location of a pin on a schematic component. Invoke once with TRUE for "XorY" to get the X value of the pin's location in the schematic, and once with FALSE to get the Y value.

|                   |          |        |                              |
|-------------------|----------|--------|------------------------------|
| <b>UI Access</b>  | NA       |        |                              |
| <b>Parameters</b> | Name     | Type   | Description                  |
|                   | <compID> | String | The Schematic component's ID |

|                     |                                                                      |         |                                                |
|---------------------|----------------------------------------------------------------------|---------|------------------------------------------------|
|                     | <pinName>                                                            | String  | The pin name                                   |
|                     | <XorY>                                                               | Boolean | TRUE gets the X value; FALSE gets the Y value. |
| <b>Return Value</b> | String values, as follows:<br><i>"X=number"</i><br><i>"Y=number"</i> |         |                                                |

|                       |                                                                        |
|-----------------------|------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetComponentPinLocation <compID>, <pinname>, <XorY>                    |
| <b>Python Example</b> | <pre>varx = oEditor.GetComponentPinLocation (compid, "N1", TRUE)</pre> |

|                   |                                                                        |
|-------------------|------------------------------------------------------------------------|
| <b>VB Syntax</b>  | GetComponentPinLocation <compID>, <pinname>, <XorY>                    |
| <b>VB Example</b> | <pre>varx = oEditor.GetComponentPinLocation (compid, "N1", TRUE)</pre> |

## GetEditorName (Schematic Editor)

Get the name of the schematic editor.

|                     |                                                                       |
|---------------------|-----------------------------------------------------------------------|
| <b>UI Access</b>    | N/A                                                                   |
| <b>Parameters</b>   | None                                                                  |
| <b>Return Value</b> | String containing the name of the schematic editor. "SchematicEditor" |

|                       |                                      |
|-----------------------|--------------------------------------|
| <b>Python Syntax</b>  | GetEditorName()                      |
| <b>Python Example</b> | <code>oEditor.GetEditorName()</code> |

|                   |                                                                |
|-------------------|----------------------------------------------------------------|
| <b>VB Syntax</b>  | GetEditorName( [out, retval] string)     // name of the editor |
| <b>VB Example</b> | <pre>dim info         info = oEditor.GetEditorName</pre>       |

### GetNetConnections (Schematic Editor)

*Use:* Informational.

*Command:* None.

*Syntax:* GetNetConnections(<netName>)

*Return Value:* Array of strings containing the connections for the net identified by the

netName argument. Strings in the array are in one of four formats:

"IPort portID x y pinName"

"GPort portID x y pinName"

"PagePort portID x y pinName"

"Component componentName compID x y pinName"

where *IPort*, *GPort*, *PagePort*, and *Component* are key words

that indicate what the rest of the string specifies:

*IPort* indicates that *portID* specifies the name of the interface port,

*GPort* indicates that *portID* specifies the name of the global port,

*PagePort* indicates that *portID* specifies the name of the page port,  
*Component* indicates that *compID* specifies the component instance identifier,  
*x* and *y* are the connection point,  
*pinName* is the name of the connected pin,  
*componentName* is the name of the component.

*Parameters:* <netName>

Type: String

The name of the net. Specifying "0" as the netName will return all objects connected to ground including all ground ports.

*VB Example:* netArray = oEditor.GetNetConnections("net\_1")

## GetNumPages [Schematic Editor]

To determine how many pages there are in the schematic.

|                     |                                                |      |             |
|---------------------|------------------------------------------------|------|-------------|
| <b>UI Access</b>    | NA                                             |      |             |
| <b>Parameters</b>   | Name                                           | Type | Description |
|                     | None                                           |      |             |
| <b>Return Value</b> | Integer, the number of pages in the schematic. |      |             |

|                       |                              |
|-----------------------|------------------------------|
| <b>Python Syntax</b>  | GetNumPages()                |
| <b>Python Example</b> | num = oEditor. GetNumPages() |

|  |  |
|--|--|
|  |  |
|--|--|

|                   |                                |
|-------------------|--------------------------------|
| <b>VB Syntax</b>  | GetNumPages                    |
| <b>VB Example</b> | Set num = oEditor. GetNumPages |

**GetPortInfo (Schematic Editor)**

Informational – get details about a port on a schematic component.

|                     |                                                                                                                                                                                                                                                                                                                                                                                                 |             |                                                                               |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------|-------------------------------------------------------------------------------|
| <b>UI Access</b>    | NA                                                                                                                                                                                                                                                                                                                                                                                              |             |                                                                               |
| <b>Parameters</b>   | <b>Name</b>                                                                                                                                                                                                                                                                                                                                                                                     | <b>Type</b> | <b>Description</b>                                                            |
|                     | <portID>                                                                                                                                                                                                                                                                                                                                                                                        | String      | The Schematic port's ID, obtained from GetSelections or through FindElements. |
| <b>Return Value</b> | Array of strings, as follows:<br><br>"Name=string" // Name of the port.<br><br>"X=val" // Connection point X location.<br><br>"Y=val" // Connection point Y location.<br><br>"Angle=val" // Component angle.<br><br>"Flip=true or false" // Whether the component is flipped or not.<br><br>"WireId=string" // The identifier of the schematic net, can be used in other informational methods. |             |                                                                               |



| Python Syntax  | GetPortInfo(<portID>)                                                                                                                                                                                                                                                                                 |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre> selectionArray = oEditor.GetSelections()                                  for i in selectionArray:                                 info = oEditor.GetPortInfo(i)                                 for j in info:                                 AddInfoMessage(str("GetPortInfo: " + j)) </pre> |

| VB Syntax  | GetPortInfo<portID>                                                                                                                                                                                                                                                                                                                                                            |
|------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre> selectionArray = oEditor.GetSelections                                  For Each i in selectionArray                                 info = oEditor.GetPortInfo(i)                                 for each j in info                                 MsgBox j, mbokonly, "GetPortInfo"                                 Next                                 Next </pre> |

### GetSelections (Schematic Editor)

Returns an array of currently selected objects.

|            |       |
|------------|-------|
| UI Access  | N/A   |
| Parameters | None. |

|                     |                             |
|---------------------|-----------------------------|
| <b>Return Value</b> | Array containing object IDs |
|---------------------|-----------------------------|

|                       |                                      |
|-----------------------|--------------------------------------|
| <b>Python Syntax</b>  | GetSelections()                      |
| <b>Python Example</b> | <code>oEditor.GetSelections()</code> |

|                   |                                    |
|-------------------|------------------------------------|
| <b>VB Syntax</b>  | GetSelections                      |
| <b>VB Example</b> | <code>oEditor.GetSelections</code> |

## GetSignals (Schematic Editor)

Informational

|                     |                                                                                                                                                                                                                                                                                        |        |                                                                                                                    |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------|--------------------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>    | NA                                                                                                                                                                                                                                                                                     |        |                                                                                                                    |
| <b>Parameters</b>   | Name                                                                                                                                                                                                                                                                                   | Type   | Description                                                                                                        |
|                     | <wirename>                                                                                                                                                                                                                                                                             | String | The wire's name, as seen in the UI, obtained in script through parsing return data from <code>GetWireInfo</code> . |
| <b>Return Value</b> | Array of strings which contains the signal names in the <i>wirename</i> argument. The <i>wirename</i> "Net1" would return an array with a single string "Net1". The <i>wirename</i> "WW[0-3],A,B" would return an array with six strings "WW[0]", "WW[1]", "WW[2]", "WW[3]", "A", :B". |        |                                                                                                                    |

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | GetSignals(<wirename>)                                |
| <b>Python Example</b> | <code>selectionArray = oEditor.GetSelections()</code> |

```
for i in selectionArray:

 netArray3 = oEditor.GetWireInfo(i)
 for n in netArray3:

 wname = n[0:9]

 if (wname == "WireName:"):
 nn = len(n)
 wn = n[nn-10:]
 AddInfoMessage("Wirename, parsed from ID :" + str(wn))
 sigs = oEditor.GetSignals(wn)
 for ll in sigs:
 AddInfoMessage("Element in array
 from GetSignals :" + str(ll))
```

| VB Syntax  | GetSignals(<wirename>)                                                                                             |
|------------|--------------------------------------------------------------------------------------------------------------------|
| VB Example | <pre>selectionArray = oEditor.GetSelections For Each i in selectionArray  netArray3 = oEditor.GetWireInfo(i)</pre> |

```
for each n in netArray3

 wname = Left(n, 9)

 If (wname = "WireName:") Then

 nn = Len(n)

 wn = Right(n, nn - 10)

 MsgBox wn, 0, "Wirename, parsed from ID"

 sigs = oEditor.GetSignals(wn)

 for each ll in sigs
 MsgBox ll, 0, "Element in array from GetSignals"
 next
 End If
next
next
```

### GetWireConnections (Schematic Editor)

Informational.

|                     |                                                                                                                                                                                                                                                                                                                                                                            |      |             |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|-------------|
| <b>UI Access</b>    | NA                                                                                                                                                                                                                                                                                                                                                                         |      |             |
| <b>Parameters</b>   | Name                                                                                                                                                                                                                                                                                                                                                                       | Type | Description |
|                     | None                                                                                                                                                                                                                                                                                                                                                                       |      |             |
| <b>Return Value</b> | <p>Array of strings containing the connections for the wire identified by the wireID argument.</p> <p>The strings in the array are in one of four formats:</p> <p>"InterfacePort=portname portID x,y"</p> <p>"GlobalPort=portname portID x,y"</p> <p>"PagePort=portname portID x,y"</p> <p>"ComponentPin=compId pinname x,y"</p> <p>where x,y is the connection point.</p> |      |             |

|                       |                                                                                                                                                                                                                                                     |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | GetWireConnections(<wireID>)                                                                                                                                                                                                                        |
| <b>Python Example</b> | <pre> selectionArray = oEditor.GetSelections() for i in selectionArray:     AddInfoMessage("Selected element" + i)     netArray = oEditor.GetWireConnections(i)     for m in netArray:         AddInfoMessage("GetWireConnections" + str(m)) </pre> |

|                  |                              |
|------------------|------------------------------|
| <b>VB Syntax</b> | GetWireConnections(<wireID>) |
|------------------|------------------------------|

|                   |                                                                                                                                                                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>selectionArray = oEditor.GetSelections For Each i in selectionArray     MsgBox i, 0, "Selected element"     netArray = oEditor.GetWireConnections(i)     for each m in netArray         MsgBox m, mbokonly, "GetWireConnections"     Next Next</pre> |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

**GetWireInfo (Schematic Editor)**

Informational.

|                   |             |             |                                                                                                          |
|-------------------|-------------|-------------|----------------------------------------------------------------------------------------------------------|
| <b>UI Access</b>  | NA          |             |                                                                                                          |
| <b>Parameters</b> | <b>Name</b> | <b>Type</b> | <b>Description</b>                                                                                       |
|                   | <wire id>   | String      | The Schematic wire's ID, obtained from <code>GetSelections</code> or through <code>FindElements</code> . |

|                     |                                                                                                                                                                                                                                                                                                                                                                                           |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Return Value</b> | <p>Array of strings containing the wire info in the wireID argument.</p> <p>The strings in the array are:</p> <p>"Page=number"</p> <p>"WireName=string"</p> <p>"SegmentCount=number"</p> <p>"IPortCount=number"</p> <p>"GPortCount=number"</p> <p>"PagePortCount=number"</p> <p>"PinCount=number"</p> <p>"BBoxLLx=val"</p> <p>"BBoxLLy=val"</p> <p>"BBoxURx=val"</p> <p>"BBoxURy=val"</p> |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                       |                                                                                                                                                                                             |
|-----------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>GetWireInfo(&lt;wireID&gt;)</code>                                                                                                                                                    |
| <b>Python Example</b> | <pre>selectionArray = oEditor.GetSelections() for i in selectionArray:     AddInfoMessage("Selected element" + str(i))     netArray2 = oEditor.GetWireInfo(i)     for n in netArray2:</pre> |

|  |                                        |
|--|----------------------------------------|
|  | AddInfoMessage("GetWireInfo" + str(n)) |
|--|----------------------------------------|

|            |                                                                                                                                                                                                                                        |
|------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| VB Syntax  | GetWireInfo(<wireID>)                                                                                                                                                                                                                  |
| VB Example | <pre>selectionArray = oEditor.GetSelections For Each i in selectionArray     MsgBox i, 0, "Selected element"     netArray3 = oEditor.GetWireInfo(i)     for each n in netArray3         MsgBox n, 0, "GetWireInfo"     Next Next</pre> |

GetWireSegments (Schematic Editor)

Informational.

|            |          |        |                                                                              |
|------------|----------|--------|------------------------------------------------------------------------------|
| UI Access  | NA       |        |                                                                              |
| Parameters | Name     | Type   | Description                                                                  |
|            | <wireID> | String | The Schematic wire’s ID, obtained from GetSelections or through FindElements |



|                     |                                                                                                                                                                                                                                                             |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Return Value</b> | <p>Array of strings containing the wire segments in the wireID argument. The strings in the array are:</p> <p>"Segment=val,val val,val number" the "val" quantities are segment endpoints x1,y1 x2,y2</p> <p>and the "number" is a schematic segment ID</p> |
|---------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                       |                                                                                                                                                                                                                                                      |
|-----------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <b>GetWireSegments (&lt;wireID&gt;)</b>                                                                                                                                                                                                              |
| <b>Python Example</b> | <pre> selectionArray = oEditor.GetSelections() for i in selectionArray:     AddInfoMessage("Selected element" + str(i))     netArray2 = oEditor.GetWireSegments(i)     for n in netArray2:         AddInfoMessage("GetWireSegments" + str(n)) </pre> |

|                   |                                                                                                                                                                                             |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | <b>GetWireSegments (&lt;wireID&gt;)</b>                                                                                                                                                     |
| <b>VB Example</b> | <pre> selectionArray = oEditor.GetSelections For Each i in selectionArray     MsgBox i, 0, "Selected element"     netArray2 = oEditor.GetWireSegments(i)     for each n in netArray2 </pre> |

|  |                                                                           |
|--|---------------------------------------------------------------------------|
|  | <div>MsgBox n, 0, "GetWireSegments"</div> <div>Next</div> <div>Next</div> |
|--|---------------------------------------------------------------------------|

## 27 - Definition Editor Script Commands

The Definition Editor controls the use of materials and scripts in a project. Symbol editor script commands and footprint editor script commands are accessed using the Definition Editor.

```
Set oDefinitionEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", "MySymbol")
```

```
Set oDefinitionEditor = oProject.SetActiveDefinitionEditor("FootprintEditor", "MyFootprint")
```

**The topics for this section include:**

**Also refer to the following subtopics:**

[Symbol Editor Scripts](#)

### Symbol Editor Scripts

Symbol editor script commands are accessed with a definition editor.

```
Set oDefinitionEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", "MySymbol")
```

The symbol editor script commands are listed below.

[AddLevel \(Symbol Editor\)](#)

[AlignHorizontal \(Symbol Editor\)](#)

[AlignVertical \(Symbol Editor\)](#)

"BringToFront (Symbol Editor)" on page 27-6

[ChangeProperty \(Symbol Editor\)](#)

[CloseEditor \(Symbol Editor\)](#)

[CreateArc \(Symbol Editor\)](#)

[CreateCircle \(Symbol Editor\)](#)

[CreateImage \(Symbol Editor\)](#)

[CreateLine \(Symbol Editor\)](#)

[CreatePin \(Symbol Editor\)](#)

[CreatePolygon \(Symbol Editor\)](#)

[CreateRectangle \(Symbol Editor\)](#)

[CreateText \(Symbol Editor\)](#)

[Cut \(Symbol Editor\)](#)

[Delete \(Symbol Editor\)](#)

[DisplayPinNames \(Symbol Editor\)](#)

[ExportFile \(Symbol Editor\)](#)

[FlipHorizontal \(Symbol Editor\)](#)

[FlipVertical \(Symbol Editor\)](#)

[GetProperties \(Symbol Editor\)](#)

[GetPropertyValue \(Symbol Editor\)](#)

[GetVisibleLevels \(Symbol Editor\)](#)

[GridSetup \(Symbol Editor\)](#)

[Move \(Symbol Editor\)](#)

[Pan \(Symbol Editor\)](#)

[Paste \(Symbol Editor\)](#)

[Redo \(Symbol Editor\)](#)

[RemoveLevels \(Symbol Editor\)](#)

[RemovePort \(Symbol Editor\)](#)

[Rotate \(Symbol Editor\)](#)

[Save \(Symbol Editor\)](#)

[SelectAll \(Symbol Editor\)](#)

[SendToBack \(Symbol Editor\)](#)

[SetActiveLevel \(Symbol Editor\)](#)

[SetInitialLevels \(Symbol Editor\)](#)

[SetPropertyValue \(Symbol Editor\)](#)

[SetVisibleLevels \(Symbol Editor\)](#)

[ToggleLevel \(Symbol Editor\)](#)

[Undo \(Symbol Editor\)](#)

[ZoomArea \(Symbol Editor\)](#)

[ZoomIn \(Symbol Editor\)](#)

[ZoomOut \(Symbol Editor\)](#)

[ZoomPrevious \(Symbol Editor\)](#)

[ZoomToFit \(Symbol Editor\)](#)

## **AddLevel (Symbol Editor)**

To add a graphics level to the symbol.

*Command:* **Add Level** in the **View>Levels** dialog.

**Syntax:** AddLevel <level>

**Return Value:** Integer, the previous active level. The current active level is the newly added level.

**Parameters:** <level>

Type: Integer

The new level, an integer greater than 1.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
"Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oldLevel = oSymbolEditor.AddLevel(4)
```

### **AlignHorizontal (Symbol Editor)**

Align items horizontally

**Syntax:** AlignHorizontal(  
Array("NAME:Selections", \_  
"Page:=", page number, \_ // [Opt=1] Page number  
"Selections:=", IDs to modify)),  
Array("NAME:AlignParameters", \_  
"Disconnect:=", bool \_ // [Opt=0] Should wires disconnect  
"Rubberband:=", bool) \_ // [Opt=1] Should wires staircase  
// Note: Alignment occurs relative to the first item in ids

|                       |                                                                                                                                                                                                                                         |
|-----------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>AlignHorizontal(["NAME:Selections", "Selections:=", [&lt;selections&gt;]], 0)</code>                                                                                                                                              |
| <b>Python Example</b> | <pre>oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _ "Simplorer Elements\basic Elements\Circuit\PassiveElements:R")  oSymbolEditor.AlignHorizontal (["NAME:Selections", _ "Selections:=", ["SchObj@11"]], 0)</pre> |

### AlignVertical (Symbol Editor)

Align items vertically

*Syntax:* AlignVertical(

```
Array("NAME:Selections", _
"Page:=", page number, _ // [Opt=1] Page number
"Selections:=", IDs to modify)),
Array("NAME:AlignParameters", _
"Disconnect:=", bool _ // [Opt=0] Should wires disconnect
"Rubberband:=", bool) _ // [Opt=1] Should wires staircase
// Note: Alignment occurs relative
to the first item in ids
```

|                       |                                                                                          |
|-----------------------|------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | <code>AlignVertical(["NAME:Selections", "Selections:=", [&lt;selections&gt;]], 0)</code> |
| <b>Python Example</b> | <pre>oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _</pre>          |

|                                                                                                                                                                  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>"Simplorer Elements\basic Elements\Circuit\PassiveElements:R")<br/>oSymbolEditor.AlignVertical (["NAME:Selections","Selections:=", ["SchObj@11"]], 0)</pre> |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------|

## BringToFront (Symbol Editor)

Changes the drawing for the symbol so that the specified objects are drawn on top of other overlapping objects.

**Command:** Draw > Bring To Front

**Syntax:** BringToFront Array("NAME:Selections", "Selections:=", Array (<Object>, <Object>, ...))

**Return Value:** None.

**Parameters:** <string><Object>

<string> // object to bring to the front

*Example:*

```
oDefinitionEditor.BringToFront Array("NAME:Selections", "Selections:=", Array("SchObj@10"))
```

## ChangeProperty (Symbol Editor)

**Use:** Changes to properties are scripted using the **ChangeProperty** command. This command can be executed by the **oEditor** to change editor properties, by the **oDesign** to change design level properties, and by the **oProject** to change project level properties. The command can be used to create, edit, and/or remove properties. In Circuit, only Variable and Separator properties can be deleted.

**Command:** None

**Syntax:** ChangeProperty Array("Name:AllTabs", <PropTabArray>, <PropTabArray>, ...)

**Return Value:** None

**Parameters:** <PropTabArray>

Array("Name:<PropTab>,"



```
<PropServersArray>,
<NewPropArray>,
<ChangedPropsArray>,
<DeletedPropsArray>)
```

```
<PropServersArray>
Array("Name:PropServers", <PropServer>,
<PropServer>, ...)
```

```
<NewPropArray>
Array("Name:NewProp", <PropDataArray>,
<PropDataArray>,...)
```

```
<ChangedPropsArray>
Array("Name:ChangedProps",<PropDataArray>,
<PropDataArray>, ...)
```

```
<DeletedPropsArray>
Array("Name:DeletedProps", <PropName>,
<PropName>, ...)
```

OR (for PropDisplay deletions only)

```
Array("Name:DeletedProps",<PropDataArray>,
<PropDataArray>, ...)
```

```
<PropDataArray>
Array("NAME:<PropName>",
"PropType:=", <PropType>,
"NewName:=", <string>,
"Description:=", <string>,
"Callback:=", <string>,
"NewRowPosition:=", <int>,
"ReadOnly:=", <bool>,
"Hidden:=", <bool>,
<PropTypeSpecificArgs>)OR (for PropDisplays only)
Array("Name:<PropName>",<PropDisplayData>)
```

```
<PropDisplayData>
for adding, changing, deleting PropDisplays
<PropDisplayAttributes>
for changing PropDisplays only
```

<PropDisplayNewAttributes>

<PropDisplayAttributes>

Layer & Location only used for PropDisplays in layout

For adding PropDisplays, this will add a single PropDisplay with attributes as shown; if an attribute is missing, a default value will be assigned. Adding PropDisplay to schematic with attributes that are identical to one already existing there will fail without an error message.

For deleting PropDisplays, these attributes are used to identify an existing PropDisplay to delete. If there doesn't exist a PropDisplay that matches the given attributes, then nothing will be deleted. If multiple PropDisplays match the given attributes, then all of them will be deleted. If an attribute is missing, then all PropDisplays match that missing attribute. For example, if Layer is missing, then PropDisplays on all layers that match the remaining given attributes will be deleted.

For changing PropDisplays, these attributes are used to identify an existing PropDisplay to change. If no PropDisplay matching the attributes is found, no changes will be made. If multiple PropDisplays match the attributes, all of them will be changed. If an attribute is missing, it matches all PropDisplays. For example, to change the format of PropDisplays that are on the bottom, but have any layer, style or format to show the name only, this command should have Location set to "Bottom" and all other attributes omitted.

"Format:=", <PropDisplayType>,

"Location:=", <PropDisplayLocation>,

"Layer:=", <string>,

"Style:=", <string>

<PropDisplayNewAttributes>

NewLayer & NewLocation only used for PropDisplays in layout

For changing PropDisplays, these attributes are used to identify which attributes to change and what the new value is. If the attribute should not be changed, the corresponding entry should be omitted.

"NewName:=", <string>,

"NewFormat:=", <PropDisplayType>,

"NewLocation:=", <PropDisplayLocation>,

"NewLayer:=", <string>,

"NewStyle:=", <string>

<PropDisplayType>

Type: string

Identifies the format of PropDisplay.

"Name"

"Value"

"NameAndValue"

"EvaluatedValue"

"NameAndEvaluatedValue"

<PropDisplayLocation>

Type: string

Identifies where PropDisplay is located with respect to object

"Left"

"Top"

"Right"

"Bottom"

"Custom"

<PropType>

Type: string

Identifies the type of property when a new property is added. In Circuit, only separator properties and variable properties can be added.

"SeparatorProp"

"VariableProp"

"TextProp"

"NumberProp"

"ValueProp"

"CheckboxProp"

"MenuProp"

"PointProp"

"VPointProp"

"ButtonProp"

NewName

Specify the new name of a property if the property's name is being edited. In Circuit, the name can only be changed for separators and variables.

### Description

Specify a description of the property. In Circuit, the description can only be changed for separators and variables.

### Callback

Specify the name of the script callback to be run when the property value is changed.

### NewRowPosition

Used to reorder rows in the **Property** dialog box. In Circuit, this only applies to the **Project>Project Variables** panel and the **Designer>DesignProperties** panel. Specify the new zero-based row index of the variable or separator.

### ReadOnly

Used to mark a property as "read only" so it can not be modified. In Circuit, this flag can only be set for variables and separators.

### Hidden

Used to hide a property so it can not be viewed outside of the **Property** dialog box. In Circuit, this flag can only be set for variables and separators.

### <PropTypeSpecificArgs>

**SeparatorProp:** no arguments

TextProp: "Value:=", <string>

NumberProp: "Value:=", <double>

ValueProp: "Value:=", <value>

CheckboxProp: "Value:=", <bool>

MenuProp: "Value:=", <string>

PointProp "X:=", <double>, "Y:=", <double>

VPointProp: "X:=", <value>, "Y:=", <value>

Material Button: "Material:=", <string>

Color Button: "R:=", <int>, "G:=", <int>, "B:=", <int>

Transparency Button: "Value:=", <double>

<PropTypeSpecificArgs> for MenuProps

Syntax for NewProps array: "AllChoices:=",

<"choice1,choice2,..."> or <Array("choice1" "choice2", ... )>,

"Value:=", <string>

Syntax for ChangedProps array: "Value:=", <string>

<PropTypeSpecificArgs> for VariableProps

Syntax:

**"Value:=", <value>, <OptimizationFlagsArray>,**

**<TuningFlagsArray>, <SensitivityFlagsArray>,  
<StatisticsFlagsArray>**

Parameters:

**<OptimizationFlagsArray>**  
**Array("NAME:Optimization",**  
"Included:=", <bool>,  
"Min:=", <value>,  
"Max:=", <value>)

**<TuningFlagsArray>**  
Array("NAME:Tuning",  
"Included:=", <bool>,  
"Step:=", <value>,  
"Min:=", <value>,  
"Max:=", <value>)

**<SensitivityFlagsArray>**  
Array("NAME:Sensitivity",  
"Included:=", <bool>,  
"Min:=", <value>,



"Max:=", <value>,  
"IDisp:=", <value> )

<StatisticsFlagsArray>  
Array("NAME:Statistical",  
"Included:=", <bool>,  
"Dist:=", <Distribution>,  
"StdD:=", <value>,  
"Min:=", <value>,  
"Max:=", <value>,  
"Tol:=", <string>)

<Distribution>  
Type: string  
Value should be **"Gaussian"** or **"Uniform"**

StdD  
Standard deviation.

Min  
Low cut-off for the distribution.

Max

High cut-off for the distribution.

Tol

Tolerance for uniform distributions. Format is "<int>%".

Example: "20%".

*VB Example:*

Adding a new project level variable "\$width":

```
oProject.ChangeProperty Array("NAME:AllTabs",_
Array("NAME:ProjectVariableTab",_
Array("NAME:PropServers", "ProjectVariables"),_
Array("NAME:NewProps",_
Array("NAME:$width",_
"PropType:=", "VariableProp",_
"Value:=", "3mm",_
"Description:=", "my new variable"))))
```

*VB Example:* Deleting the design level variable "height":

```
oDesign.ChangeProperty Array("NAME:AllTabs", _
Array("NAME:LocalVariableTab", _
Array("NAME:PropServers", "DefinitionParameters"), _
Array("NAME:DeletedProps", "height"))
```

Changing a property's value. If the following command were executed, then the value of the property "XSize" of the PropServer

"Box1:CreateBox:1" on the "Geometry3DCmdTab" tab would be changed. (oEditor is the Geometry3D editor in Circuit.)

```
oEditor.ChangeProperty Array("NAME:AllTabs", _
Array("NAME:Geometry3DCmdTab", _
Array("NAME:PropServers", "Box1:CreateBox:1"), _
Array("NAME:ChangedProps", _
Array("NAME:XSize", "Value:=", "1.4mil"))))
```

*VB Example:* Changing a property's value. If the following command were executed, then the values of Callback and L on the PassedParameterTab would be changed.

```
oEditor.ChangeProperty Array("NAME:AllTabs", _
Array("NAME:PassedParameterTab", _
Array("NAME:PropServers", "CHOKE2"), _
Array("NAME:ChangedProps", _
Array("NAME:L", "Callback:=", "ac", "OverridingDef:=", true), _
```

```
Array("NAME:L", "Value:=", "lnH"))))
```

*VB Example:* Changing the Company Name, Design Name, the background color, and the Axis scaling in a Report.

```
Set oProject = oDesktop.SetActiveProject("wgcombiner")
Set oDesign = oProject.SetActiveDesign("CircuitDesign2")
Set oModule = oDesign.GetModule("ReportSetup")

oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Header", _ Array("NAME:PropServers", "XY
Plot1:Header"), _
Array("NAME:ChangedProps", Array("NAME:Company Name", _
"Value:=", "My Company"))))

oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Header", _ Array("NAME:PropServers", "XY
Plot1:Header"), _
Array("NAME:ChangedProps", Array("NAME:Design Name", _
"Value:=", "WG Combiner"))))

oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:General", _ Array("NAME:PropServers", "XY
Plot1:General"), _
Array("NAME:ChangedProps", Array("NAME:Back Color", _
"R:=", 128, "G:=", 255, "B:=", 255))))

oModule.ChangeProperty Array("NAME:AllTabs", Array("NAME:Axis", _ Array("NAME:PropServers", "XY
Plot1:AxisX"), _
```

```
Array("NAME:ChangedProps", Array("NAME:Axis Scaling", _
"Value:=", "Log"))))
```

*VB Example:* Changing a property's value. Note that the AllChoices parameter is only used when the MenuProp is being added. Also note that either a string of choices separated by commas or an Array("choice1", "choice2", "choice3") works for the AllChoices parameter.

```
Set oEditor = oDesign.SetActiveEditor("SchematicEditor")
oEditor.ChangeProperty Array("NAME:AllTabs", Array("NAME:PassedParameterTab", Array
("NAME:PropServers", _
"CompInst@CAP_;2"), Array("NAME:NewProps", Array("NAME:xxxx", "PropType:=", _
"MenuProp", "AllChoices:=", Array("aa", "bb", "cc", "dd"), "Value:=", "bb"))))
```

**CloseEditor (Symbol Editor)**

Closes the specified editor.

*Command:* None

*Syntax:* CloseEditor (VARIANT ptDelta)

*Return Value:* None

|                |                                 |
|----------------|---------------------------------|
| Python Syntax  | CloseEditor()                   |
| Python Example | oDefinitionEditor.CloseEditor() |

## Copy (Symbol Editor)

Copy items for pasting.

**Command:** Edit > Copy

**Syntax:** Copy Array("NAME:Selections", "Selections:=", Array(<selections>))

**Return Value:** None.

**Parameters:** <selections>

Type: Comma-separated list of strings

Identifiers for each selected item to be copied, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

*Example:* Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
```

```
"Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```

```
oSymbolEditor.Copy Array("NAME:Selections", "Selections:=", Array("SchObj@11", "SchObj@34"))
```

## CreateArc (Symbol Editor)

Draws an arc in the symbol editor

**Command:** None

**Syntax:** CreateArc

**Return Value:** None

*VB Example:*

```
oDefinitionEditor.CreateArc Array("NAME:ArcData", _
```

```
"X:=", -0.004318, "Y:=", -0.00127, _
"Radius:=", 0.00297299377732279, _
"StartAng:=", 1.9195673303788, _
"EndAng:=", 3.32144615338227, _
"Id:=", 10), _
Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreateArc()                                                                                                                                                                                                                                                                      |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreateArc<br/>(["NAME:ArcData", _<br/>"X:=", -0.004318, "Y:=", -0.00127, _<br/>"Radius:=", 0.00297299377732279, _<br/>"StartAng:=", 1.9195673303788, _<br/>"EndAng:=", 3.32144615338227, _<br/>"Id:=", 10], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |

CreateCircle (Symbol Editor)

Draws a circle in the symbol editor  
Command: None

*Syntax:* CreateCircle

*Return Value:* None

*VB Example:*

```
oDefinitionEditor.CreateCircle Array("NAME:CircleData", _
"X:=", -0.004572, "Y:=", -0.000508, _
"Radius:=", 0.001778, "Id:=", 12), _
Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreateCircle()                                                                                                                                                                                |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreateCircle<br/>(["NAME:CircleData", _<br/>"X:=", -0.004572, "Y:=", -0.000508, _<br/>"Radius:=", 0.001778, "Id:=", 12], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |

**CreateImage (Symbol Editor)**

Create picture from a BMP or GIF file.

*Command:***Draw>Image**

*Syntax:* CreateImage(

*Return Value:* None.



```
Parameters: Array("NAME:ImageData",
 "X1:=", double, // the image rectangle left X value, in meters
 "Y1:=", double, // the image rectangle upper Y value, in meters
 "X2:=", double, // the image rectangle right X value, in meters
 "Y2:=", double, // the image rectangle lower Y value, in meters
 "File:=", string, // the name of the BMP or GIF file
 "DisplayBorder:=", bool, // true or false, whether to display the image border
 "LineWidth:=", double, // the width of the image border, in meters
 "Color:=", int, // the RGB value of the image border color
 "Id:=", int), // Id for this item
 Array("NAME:Attributes",
 "Page:=", int) // [Opt=1] The page number (one-based)
```

```
Example: oDefinitionEditor.CreateImage Array("NAME:ImageData", "X1:=", 0.00229684174875881,
"Y1:=", _ 0.00100608189091032, "X2:=", 0.00890589813779289, "Y2:=", -0.00432967005619977,
"File:=", "C:/Documents and Settings/user/Desktop/logo.gif", "DisplayBorder:=", false,
"LineWidth:=", 0, "Color:=", 0, "Id:=", 22), Array("NAME:Attributes", "Page:=", 1)
```

## CreateLine (Symbol Editor)

Draws a line in the symbol editor

**Command:** None

**Syntax:** CreateLine

*VB Example:*

```
oDefinitionEditor.CreateLine Array("NAME:LineData", _Return Value: None
```

```
"Points:=", Array("(0.001778, -0.001270)", _
"(0.004572, 0.002794)", "(0.003048, 0.003556)"), _
"Id:=", 14), Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreateLine()                                                                                                                                                                                                                           |
|----------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreateLine<br/>(["NAME:LineData", _Return Value: None<br/>"Points:=", [ "(0.001778, -0.001270)",_<br/>"(0.004572, 0.002794)", "(0.003048, 0.003556)"],_<br/>"Id:=", 14], ["NAME:Attributes", "Page:=",1])</pre> |

CreatePin (Symbol Editor)

Draws a pin in the symbol editor

Command: None

Syntax: CreatePin

Return Value: None

VB Example:

```
oDefinitionEditor.CreatePin Array("NAME:PinData", _
"Name:=", "newpin3"), Array("NAME:PinParams", _
"X:=", -0.00762, "Y:=", 0.00254, "Angle:=", 0, "Flip:=", false)
```

| Python Syntax  | CreatePin()                                                                                                                                                                         |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreatePin (["NAME:PinData", _<br/>"Name:=", "newpin3"], ["NAME:PinParams",_<br/>"X:=", -0.00762, "Y:=", 0.00254, "Angle:=", _<br/>0, "Flip:=", false])</pre> |

## CreatePolygon (Symbol Editor)

Draws a polygon in the symbol editor

*Command:* None

*Syntax:* CreatePolygon

*Return Value:* None

*VB Example:*

```
oDefinitionEditor.CreatePolygon Array("NAME:PolygonData", _
"Points:=", Array("(0.004826, -0.000508)", "(0.003048, -0.003048)", _
"(0.006858, -0.003302)"), "Id:=", 16), _
Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreatePolygon()                                                                                               |
|----------------|---------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreatePolygon (["NAME:PolygonData", _<br/>"X1:=", -0.001524, "Y1:=", 0.000508, _</pre> |

|  |                                                                                                      |
|--|------------------------------------------------------------------------------------------------------|
|  | <pre>"X2:=", 0.002286, "Y2:=", -0.002032, "Id:=", 18], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |
|--|------------------------------------------------------------------------------------------------------|

CreateRectangle (Symbol Editor)

Draws a rectangle in the symbol editor

Command: None

Syntax: CreateRectangle

Return Value: None

VB Example:

```
oDefinitionEditor.CreateRectangle Array("NAME:RectData", _
"X1:=", -0.001524, "Y1:=", 0.000508, _
"X2:=", 0.002286, "Y2:=", -0.002032, "Id:=", 18), _
Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreateRectangle()                                                                                                                                                                                          |
|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreateRectangle (["NAME:RectData", _<br/>"X1:=", -0.001524, "Y1:=", 0.000508, _<br/>"X2:=", 0.002286, "Y2:=", -0.002032, "Id:=", 18], _<br/>["NAME:Attributes", "Page:=", 1])</pre> |

## CreateText (Symbol Editor)

Draws text in the symbol editor

*Command:* None

*Syntax:* CreateText

*Return Value:* None

*VB Example:*

```
oDefinitionEditor.CreateText Array("NAME:TextData", _
"X:=", 0.001524, "Y:=", 0.00127, _
"Text:=", "My text", "Id:=", 20), Array("NAME:Attributes", "Page:=", 1)
```

| Python Syntax  | CreateText()                                                                                                                                                                |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.CreateText(<br/>["NAME:TextData", "X:=", 0.001524, "Y:=",<br/>0.00127, "Text:=", "My text", "Id:=", 20],<br/>["NAME:Attributes", "Page:=", 1])</pre> |

## Cut (Symbol Editor)

Cut page selection.

*Command:* None

*Syntax:* Cut(PageNumber, Selections)

*Return Value:* None

*VB Example:*

```
oDefinitionEditor.CreateText Array("NAME:Selections", _
"Page:=", page number, _ // [Opt=1] Page number
"Selections:=", IDs to modify)) 1)
```

|                |                                                                                                                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | Cut(["NAME:Selections", "Selections:=", [<selections>]])                                                                                                                                                                          |
| Python Example | <pre>oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _ "Simplorer Elements\basic Elements\Circuit\PassiveElements:R") oSymbolEditor.Cut (["NAME:Selections", "Selections:=", ["SchObj@11", "SchObj@3"]])</pre> |

**Delete (Symbol Editor)**

Delete items - remove from the symbol.

*Command:* **Edit>Delete**

*Syntax:* Delete Array("NAME:Selections", "Selections:=", Array(<selections>))

*Return Value:* None.

*Parameters:* <selections>

    Type: Comma-separated list of strings

    Identifiers for each selected item to be deleted, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

*Example:* Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
"Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```

```
oSymbolEditor.Delete Array("NAME:Selections", "Selections:=", Array("SchObj@11", "SchObj@34"))
```

### **DisplayPinNames (Symbol Editor)**

To turn on or off the display of pin names.

*Command:*None.

*Syntax:* DisplayPinNames <onoff>

*Return Value:* None.

*Parameters:* <onoff>

Type: Boolean

TRUE or FALSE to turn on or off the display of pin names.

*Example:* Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", "Simplorer Elements\basic
Elements\Circuit\Passive Elements:R")
```

```
oSymbolEditor.DisplayPinNames TRUE
```

### **ExportFile (Symbol Editor)**

Export file of symbol.

*Command:***Symbol>Export File**

*Syntax:* ExportFile <type>, <filename>

*Return Value:* None.

*Parameters:* <type>

Type: String

Type of file to export - only "EMF" is supported currently.

<filename>

Type: String

Path, name and extension for the file to be saved.

```
Example: Dim oSymbolEditor
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.ExportFile "EMF", "c:\sym.emf"
```

### **FlipHorizontal (Symbol Editor)**

To flip the specified objects about the horizontal (x) axis.

**Command:**Draw>Flip Horizontal

**Syntax:** FlipHorizontal Array("NAME:Selections", "Selections:=", Array(<selections>)), 0

**Return Value:** None.

**Parameters:** <selections>

Type: Comma-separated list of strings

Identifiers for each selected item to be flipped, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

```
Example: Dim oSymbolEditor
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.FlipHorizontal Array("NAME:Selections", "Selections:=", Array("SchObj@11")), 0
```



## FlipVertical (Symbol Editor)

To flip the specified objects about the horizontal (x) axis.

**Command:** Draw>Flip Vertical

**Syntax:** FlipVertical Array("NAME:Selections", "Selections:=", Array(<selections>)), 0

**Return Value:** None.

**Parameters:** <selections>

Type: Comma-separated list of strings

Identifiers for each selected item to be flipped, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.FlipVertical Array("NAME:Selections", "Selections:=", Array("SchObj@11")), 0
```

## GetProperties (Symbol Editor)

Gets a list of all the properties belonging to a specific **PropServer** and **PropTab**. This can be executed by the **oProject**, **oDesign**, or **oEditor** objects.

**Command:** None

**Syntax:** GetProperties( <PropTab>, <PropServer> )

**Return Value:** Variant array of strings – the names of the properties belonging to the prop server.

**VB Example:** Dim all\_props

```
all_props = oDesign.GetProperties("BaseElementTab", _
 "rect_1")
```

## GetPropertyValue (Symbol Editor)

Gets the value of a single property. This can be executed by the **oProject**, **oDesign**, or **oEditor** objects.

*Command:* None

*Syntax:* GetPropertyValue(<PropTab>, <PropServer>, <PropName>)

*Return Value:* String representing the property value.

```
VB Example: value_string = _
oEditor.GetPropertyValue("BaseElementTab", _
"rect_1", "Name")
```

## GetVisibleLevels (Symbol Editor)

To determine the levels in the symbol.

*Command:*None.

*Syntax:* GetVisibleLevels

*Return Value:* An array of integers, which are the levels defined for the particular symbol.

*Parameters:* None.

```
Example: Dim oSymbolEditor
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
"Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
levels = oSymbolEditor.GetVisibleLevels
```

## GridSetup (Symbol Editor)

Changes the settings on the symbol grid.

**Command:** **Symbol>Grid Setup**

**Syntax:** GridSetup(

**Return Value:** None.

*Parameters:* Array("NAME:Options", \_  
    "MajorGrid=", string, \_ // Major grid size with units  
    "Divisions=", int, \_ // Number of minor grid divisions  
    "MajorColor=", int, \_ // RGB Color of major grid lines  
    "MinorColor=", int, \_ // RGB Color of minor grid lines  
    "ShowGrid=", bool, \_ // Should the grid be shown?  
    "SnapToGrid=", bool, \_ // Should objects snap to grid?  
    "BackgroundColor=", int, \_ // RGB Color of the background  
    "SaveAsDefault=", bool)) // Should these settings be the default for new schematics

*Example:* oEditor.GridSetup Array(  
    "NAME:Options",  
    "MajorGrid=", "10mm",  
    "Divisions=", 10,  
    "MajorColor=", 0x00ff0000, // Red  
    "MinorColor=", 0x0000ff00, // Green  
    "ShowGrid=", true,

```
"SnapToGrid:=", true,
"BackgroundColor:=", 0x00ffffff, // White
"SaveAsDefault:=", false)
```

### Move (Symbol Editor)

To move the specified objects in X and Y.

*Command:*None.

*Syntax:* Move Array("NAME:Selections", "Selections:=", Array(<selections>)), Array("NAME:FlipParameters", "xdelta:=", <xval>, "ydelta:=", <yval>)

*Return Value:* None.

*Parameters:* <selections>

Type: Comma-separated list of strings

Identifiers for each selected item to be moved, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

<xval>

Type: Real number

The X delta to move.

<yval>

Type: Real number

The Y delta to move.

*Example:* Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",
```

```
"Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor. Move Array("NAME:Selections", "Selections:=", Array("SchObj@34")),
Array("NAME:FlipParameters", "xdelta:=", -0.00254, "ydelta:=", 0.00254)
```

## Pan (Symbol Editor)

Pans the display.

| UI Access    | View > Pan.     |                  |                                                                  |
|--------------|-----------------|------------------|------------------------------------------------------------------|
| Parameters   | Name            | Type             | Description                                                      |
|              | <argSelections> | Array of doubles | First double pans along x-axis; second double pans along y-axis. |
| Return Value | None.           |                  |                                                                  |

| Python Syntax  | Pan (<argSelections>)                                                         |
|----------------|-------------------------------------------------------------------------------|
| Python Example | <pre>oDefinitionEditor.Pan([-0.00922653869663931, 0.00686839904222147])</pre> |

| VB Syntax  | Pan <argSelections>                                                                   |
|------------|---------------------------------------------------------------------------------------|
| VB Example | <pre>oDefinitionEditor.Pan<br/>Array(-0.00922653869663931, 0.00686839904222147)</pre> |

## Paste (Symbol Editor)

To paste copied objects to a specified X and Y location.

**Command:** Edit>Paste

**Syntax:** Paste Array("NAME:Attributes", "X:=", <xval>, "Y:=",<yval>)

**Return Value:** None.

**Parameters:** <xval>

Type: Real number

The X placement value.

<yval>

Type: Real number

The Y placement value.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.Paste Array("NAME:Attributes", "X:=", -0.00254, "Y:=", 0.00254)
```

## Redo (Symbol Editor)

Redo the last operation

**Command:** Click **Edit>Redo**

**Syntax:** Redo

**Return Value:** None

**Parameters:** None

**VB Example:** oDesign.Redo

## RemoveLevels (Symbol Editor)

Removes one or more graphics levels from a symbol.

|                     |                                                                                                                     |                   |                           |
|---------------------|---------------------------------------------------------------------------------------------------------------------|-------------------|---------------------------|
| <b>UI Access</b>    | <b>Tools &gt; Edit Libraries &gt; Symbols.</b> From Symbol editor, click <b>View &gt; Levels &gt; Remove Level.</b> |                   |                           |
| <b>Parameters</b>   | Name                                                                                                                | Type              | Description               |
|                     | <levels>                                                                                                            | Array of integers | The levels to be deleted. |
| <b>Return Value</b> | None.                                                                                                               |                   |                           |

|                       |                                                 |
|-----------------------|-------------------------------------------------|
| <b>Python Syntax</b>  | RemoveLevels (<levels>)                         |
| <b>Python Example</b> | <code>oSymbolEditor.RemoveLevels ([3,4])</code> |

|                   |                                                                                                                                                                                                                                      |
|-------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | RemoveLevels <levels>                                                                                                                                                                                                                |
| <b>VB Example</b> | <pre> Dim oSymbolEditor  Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", -     "Simplorer Elements\basic Elements\Circuit\Passive     Elements:R")  oldLevel = oSymbolEditor.RemoveLevels (Array(3,4)) </pre> |

## RemovePort (Symbol Editor)

Selected pins are changed to vias. Selected edge terminal ports are removed.

*Syntax:* RemovePort <NAME:elements", <object\_name> ... // objects to be removed

*Return Value:* None.

*Parameters:* <object\_name>

Type: <String>

*VB Example:* oEditor.RemovePort Array("NAME:elements", "via\_195", "Port1")

### Rotate (Symbol Editor)

To rotate the specified objects about their common center.

*Command:***Draw>Rotate**

*Syntax:* Rotate Array("NAME:Selections", "Selections:=", Array(<selections>)), Array("NAME:FlipParameters", <angle>)

*Return Value:* None.

*Parameters:* <selections>

Type: Comma-separated list of strings

Identifiers for each selected item to be rotated, in the form SchObj@<id>, where id is the value of the SchematicID property for each element.

<angle>

Type: A comma-separated string and real number

Either "Angle:=", <rad>, where <rad> is the angle in radians, or "Degrees:=", <deg>, where <deg> is the angle in degrees.

*Example:* Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```



```
oSymbolEditor.Rotate Array("NAME:Selections", "Selections:=", Array("SchObj@11")),
Array("NAME:FlipParameters", "Angle:=", 1.5707963267949)
```

### **Save (Symbol Editor)**

Saves any changes in the current symbol to the project.

**Command:**Symbol>Update Project

**Syntax:** Save

**Return Value:** None.

**Parameters:** None.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.Save
```

### **SelectAll (Symbol Editor)**

Select all elements in the symbol editor.

**Command:** None.

**Syntax:** SelectAll()

**Parameters:** None

**VB Example:** Dim removedDefs

```
removedDefs = oDefinitionEditor.SelectAll()
```

|                       |                                                       |
|-----------------------|-------------------------------------------------------|
| <b>Python Syntax</b>  | SelectAll()                                           |
| <b>Python Example</b> | <code>removedDefs = oSymbolManager.SelectAll()</code> |

### SendToBack (Symbol Editor)

Changes the drawing for the symbol so that the specified objects are drawn behind other overlapping objects.

*Command:* Draw > Send To Back

*Syntax:* SendToBack Array("NAME:Selections", "Selections:=", Array(<Object>, <Object>, ...))

*Return Value:* None

*Parameters:* <Object>

<string> // object to send to the back

*VB Example:* oDefinitionEditor.SendToBack Array("NAME:Selections", "Selections:=", Array("SchObj@10"))

|                       |                                                                                               |
|-----------------------|-----------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | SendToBack(["NAME:Selections", "Selections:=", [<Object>, <Object>, ...]])                    |
| <b>Python Example</b> | <code>oDefinitionEditor.SendToBack(["NAME:Selections", "Selections:=", ["SchObj@10"]])</code> |

### SetActiveLevel (Symbol Editor)

To set the active graphics level in the symbol editor.

*Command:* "Current" radio button for a level in the **View>Levels** dialog.

*Syntax:* SetActiveLevel <level>

*Return Value:* None.

Parameters: <level>

Type: Integer

The new active level.

```
Example: Dim oSymbolEditor
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oldLevel = oSymbolEditor.SetActiveLevel (4)
```

SetInitialLevels (Symbol Editor)

Sets one or more graphics levels as the default visible levels for a symbol. The levels submitted do not need to contain 1 or 0, which are always visible.

|              |                                                                                                                             |                   |                            |
|--------------|-----------------------------------------------------------------------------------------------------------------------------|-------------------|----------------------------|
| UI Access    | Tools > Edit Libraries > Symbols. From Symbol editor, click View > Levels. Select the Default check box for desired levels. |                   |                            |
| Parameters   | Name                                                                                                                        | Type              | Description                |
|              | </levels>                                                                                                                   | Array of integers | The levels to be selected. |
| Return Value | None.                                                                                                                       |                   |                            |

|                |                                        |
|----------------|----------------------------------------|
| Python Syntax  | SetInitialLevels (</levels>)           |
| Python Example | oSymbolEditor.SetInitialLevels ([3,4]) |

|           |                            |
|-----------|----------------------------|
| VB Syntax | SetInitialLevels </levels> |
|-----------|----------------------------|

|                   |                                                                                                                                                                                                                                       |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Example</b> | <pre>Dim oSymbolEditor  Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", -     "Simplorer Elements\basic Elements\Circuit\Passive     Elements:R")  oldLevel = oSymbolEditor.SetInitialLevels(Array(3,4))</pre> |
|-------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

### SetPropertyValue (Symbol Editor)

Set a property.

**Command:** None.

**Syntax:** SetPropertyValue(tab, item, propname, value)

*Parameters:* None

*VB Example:* Dim removedDefs

```
removedDefs = oDefinitionEditor.SetPropertyValue(string tab, // Tab with the property
string item, // Name of the object
string propname, // Name of the property
string value) // The new value of the prop
```

### SetVisibleLevels (Symbol Editor)

Makes one or more graphics levels visible in the symbol editor.

|                     |                                                                                                                                                                |                   |                                |
|---------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|--------------------------------|
| <b>UI Access</b>    | <b>Tools &gt; Edit Libraries &gt; Symbols.</b> From Symbol editor, click <b>View &gt; Levels</b> . Select the <b>Visible</b> check box for the desired levels. |                   |                                |
| <b>Parameters</b>   | Name                                                                                                                                                           | Type              | Description                    |
|                     | <levels>                                                                                                                                                       | Array of integers | The levels to be made visible. |
| <b>Return Value</b> | None.                                                                                                                                                          |                   |                                |

|                       |                                                    |
|-----------------------|----------------------------------------------------|
| <b>Python Syntax</b>  | SetVisibleLevels (<levels>)                        |
| <b>Python Example</b> | <code>oSymbolEditor.SetVisibleLevels([3,4])</code> |

|                   |                                                                                                                                                                                                                                         |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>VB Syntax</b>  | SetVisibleLevels <levels>                                                                                                                                                                                                               |
| <b>VB Example</b> | <pre> Dim oSymbolEditor  Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", -     "Simplorer Elements\basic Elements\Circuit\Passive     Elements:R")  oldLevel = oSymbolEditor.SetVisibleLevels(Array(3,4)) </pre> |

## ToggleLevel (Symbol Editor)

To toggle display of the graphics level in the symbol editor.

**Command:** "Visible" checkbox for a level in the **View>Levels** dialog.

**Syntax:** ToggleLevel <level>

*Return Value:* None.

*Parameters:* <level>

Type: Integer

The level to turn on or off.

*Example:* Dim oSymbolEditor  
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", \_  
    "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")  
oSymbolEditor.ToggleLevel (2)

**Undo (Symbol Editor)**

Undo the last operation

*Command:* Edit>Undo

*Syntax:* Undo

*Return Value:* None

*Parameters:* None

*VB Example:* oDesign.Undo

|                |                                                                                                                                                   |
|----------------|---------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | Undo()                                                                                                                                            |
| Python Example | <pre>oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",_<br/>"Simplorer Elements\basic Elements\Circuit\PassiveElements:R")</pre> |

|  |                                   |
|--|-----------------------------------|
|  | <code>oSymbolEditor.Undo()</code> |
|--|-----------------------------------|

## ZoomArea (Symbol Editor)

To change the current graphics view area.

**Command:** View>Zoom Area

**Syntax:** ZoomArea Array(<xpt>, <ypt>), Array(<xdelta>, <ydelta>)

**Return Value:** None.

**Parameters:** <xpt>

Type: Real number

The X value of the zoom area box, in meters.

<ypt>

Type: Real number

The X value of the zoom area box, in meters.

<xdelta>

Type: Real number

The width of the zoom area box, in meters.

<ydelta>

Type: Real number

The height of the zoom area box, in meters.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor", _
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```

```
oSymbolEditor.ZoomArea Array(0, 0), Array(0.004, 0.004)
```

### **ZoomIn (Symbol Editor)**

To zoom in a standard amount to the current graphics view area.

**Command:**View>Zoom In

**Syntax:** ZoomIn

**Return Value:** None.

**Parameters:** None.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
oSymbolEditor.ZoomIn
```

### **ZoomOut (Symbol Editor)**

To zoom out a standard amount to the current graphics view area.

**Command:**View>Zoom Out

**Syntax:** ZoomOut

**Return Value:** None.

**Parameters:** None.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```



```
oSymbolEditor.ZoomOut
```

### ZoomPrevious (Symbol Editor)

To zoom the current graphics view area to the previous extents.

**Command:** View>Zoom Previous

**Syntax:** ZoomPrevious

**Return Value:** None.

**Parameters:** None.

**Example:** Dim oSymbolEditor

```
Set oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",
```

```
 "Simplorer Elements\basic Elements\Circuit\Passive Elements:R")
```

```
oSymbolEditor.ZoomPrevious
```

### ZoomToFit (Symbol Editor)

Set the current symbol zoom to fit the contents of the currently visible page

**Command:** None

**Syntax:** ZoomToFit()

**Return Value:** None

| Python Syntax  | ZoomToFit()                                                                                                                                                             |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Python Example | <pre>oSymbolEditor = oProject.SetActiveDefinitionEditor("SymbolEditor",_ "Simplorer Elements\basic Elements\Circuit\PassiveElements:R") oSymbolEditor.ZoomToFit()</pre> |

This page intentionally  
left blank.

## 28 - Core Global Script Context Commands

To run these commands:

```
import CoreGlobalScriptContextFunctions

CoreGlobalScriptContextFunctions.[CommandName]
```

The following are general script commands recognized by the **CoreGlobalScriptContextFunctions** object:

- [AddErrorMessage](#)
- [AddFatalMessage](#)
- [AddInfoMessage](#)
- [AddWarningMessage](#)
- [LogDebug](#)
- [LogError](#)

### AddErrorMessage

Adds an error message to the **Message Manager** window. AddErrorMessage is a function of CoreGlobalScriptContextFunctions.

|              |           |        |                |
|--------------|-----------|--------|----------------|
| UI Access    | N/A       |        |                |
| Parameters   | Name      | Type   | Description    |
|              | <message> | String | Error message. |
| Return Value | None.     |        |                |

|                |                                                                                                                                   |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------|
| Python Syntax  | AddErrorMessage(<message>)                                                                                                        |
| Python Example | <pre>import CoreGlobalScriptContextFunctions<br/><br/>CoreGlobalScriptContextFunctions.AddErrorMessage('My error message.')</pre> |

|            |                                        |
|------------|----------------------------------------|
| VB Syntax  | N/A                                    |
| VB Example | N/A. Script cannot be run in VBScript. |

## AddFatalMessage

Adds a fatal error message to the **Message Manager** window. AddFatalMessage is a function of CoreGlobalScriptContextFunctions.

|                     |             |             |                    |
|---------------------|-------------|-------------|--------------------|
| <b>UI Access</b>    | N/A         |             |                    |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b> |
|                     | <message>   | String      | Error message.     |
| <b>Return Value</b> | None.       |             |                    |

|                       |                                                                                                                                |
|-----------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddFatalMessage(<message>)                                                                                                     |
| <b>Python Example</b> | <pre>import CoreGlobalScriptContextFunctions CoreGlobalScriptContextFunctions.AddFatalMessage('My fatal error message.')</pre> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | N/A                                    |
| <b>VB Example</b> | N/A. Script cannot be run in VBScript. |

## AddInfoMessage

Adds an informational message to the **Message Manager** window. AddInfoMessage is a function of CoreGlobalScriptContextFunctions.

|                     |             |             |                        |
|---------------------|-------------|-------------|------------------------|
| <b>UI Access</b>    | N/A         |             |                        |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b>     |
|                     | <message>   | String      | Informational message. |
| <b>Return Value</b> | None.       |             |                        |

|                       |                                                                                                                |
|-----------------------|----------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddInfoMessage(<message>)                                                                                      |
| <b>Python Example</b> | <pre>import CoreGlobalScriptContextFunctions CoreGlobalScriptContextFunctions.AddInfoMessage('My info.')</pre> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | N/A                                    |
| <b>VB Example</b> | N/A. Script cannot be run in VBScript. |

## AddWarningMessage

Adds a warning message to the **Message Manager** window. AddWarningMessage is a function of CoreGlobalScriptContextFunctions.

|                     |             |             |                    |
|---------------------|-------------|-------------|--------------------|
| <b>UI Access</b>    | N/A         |             |                    |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b> |
|                     | <message>   | String      | Warning message.   |
| <b>Return Value</b> | None.       |             |                    |

|                       |                                                                                                                        |
|-----------------------|------------------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | AddWarningMessge(<message>)                                                                                            |
| <b>Python Example</b> | <pre>import CoreGlobalScriptContextFunctions CoreGlobalScriptContextFunctions.AddWarningMessage('My warn- ing.')</pre> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | N/A                                    |
| <b>VB Example</b> | N/A. Script cannot be run in VBScript. |

## LogDebug

Adds a debug line to the log specified at **Tools > Debug Logging**. LogDebug is a function of CoreGlobalScriptContextFunctions.

|                     |             |             |                    |
|---------------------|-------------|-------------|--------------------|
| <b>UI Access</b>    | N/A         |             |                    |
| <b>Parameters</b>   | <b>Name</b> | <b>Type</b> | <b>Description</b> |
|                     | <message>   | String      | Debug message.     |
| <b>Return Value</b> | None.       |             |                    |

|                      |                     |
|----------------------|---------------------|
| <b>Python Syntax</b> | LogDebug(<message>) |
|----------------------|---------------------|

|                       |                                                                                                                    |
|-----------------------|--------------------------------------------------------------------------------------------------------------------|
| <b>Python Example</b> | <pre>import CoreGlobalScriptContextFunctions  CoreGlobalScriptContextFunctions.LogDebug('My debug message.')</pre> |
|-----------------------|--------------------------------------------------------------------------------------------------------------------|

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | N/A                                    |
| <b>VB Example</b> | N/A. Script cannot be run in VBScript. |

## LogError

Adds an error line to the log specified at **Tools > Debug Logging**. LogError is a function of CoreGlobalScriptContextFunctions.

|                     |         |        |               |
|---------------------|---------|--------|---------------|
| <b>UI Access</b>    | N/A     |        |               |
| <b>Parameters</b>   | Name    | Type   | Description   |
|                     | <error> | String | Error to log. |
| <b>Return Value</b> | None.   |        |               |

|                       |                                                                                                            |
|-----------------------|------------------------------------------------------------------------------------------------------------|
| <b>Python Syntax</b>  | LogError(<error>)                                                                                          |
| <b>Python Example</b> | <pre>import CoreGlobalScriptContextFunctions  CoreGlobalScriptContextFunctions.LogError('My error.')</pre> |

|                   |                                        |
|-------------------|----------------------------------------|
| <b>VB Syntax</b>  | N/A                                    |
| <b>VB Example</b> | N/A. Script cannot be run in VBScript. |

## 29 - Example Scripts

This section contains [VBScript](#) and [IronPython](#) example scripts.

### VBScript Example Scripts

VBScript Examples:

- [Variable Helix Script](#)
- [HFSS Data Export Script](#)
- [ExampleGetLayeredImpedanceBoundaryPropertyNamesandValues.htm](#)

#### Variable Helix Script

This sample script creates a tapered helix. Tapering helices is not supported from the interface.

The script includes comment lines, which are preceded by an apostrophe ( ' ) and offer explanations for each subsequent line or lines.

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
Set oProject = oDesktop.GetActiveProject()
Set oDesign = oProject.GetActiveDesign()
Set oEditor = oDesign.SetActiveEditor("3D Modeler")

' Declare the arrays and variables needed for building the polyline.
,

Dim points(), segments()
Dim NumPoints, R(2), P(2), PointsPerTurn, Turns, Units
,
```

```

 ' Establish the constant Pi.

Pi = 4*Atn(1)

 ' Retrieve the variable helix parameters from the user.

 ' Start with the input for unit selection.
 ,

Units = InputBox("Select the units:"&Chr(13)& _
 "(cm,mm,um,in,mil)", "Variable Helix", "mil", 50, 50)
 ,

 ' Check to make sure it is a valid unit.
 ,

Select Case Units
 Case "m"
Units = ""
 Case "cm"
 Case "mm"
 Case "um"
 Case "in"
 Case "mil"
 Case Else
MsgBox("Invalid Units - defaults to m")
Units = ""
End Select
 ,

 ' Obtain the other user-defined parameters.
 ,

Turns = InputBox("Select the number of turns (must be
 integer):", "Variable Helix", 2, 50, 50)
PointsPerTurn = InputBox("Select the points per turn:", _
 "Variable Helix", 16, 50, 50)
R(0) = InputBox("Select the initial Radius: ", _

```



```

 "Variable Helix",10,50,50)
R(1) = InputBox("Select the final Radius: ", _
 "Variable Helix",10,50,50)
P(0) = InputBox("Select the initial Pitch: ", _
 "Variable Helix", 4,50,50)
P(1) = InputBox("Select the final Pitch: ", _
 "Variable Helix", 4,50,50)
NumPoints = Turns*PointsPerTurn
,

 ' Initialize the points and segments arrays.
 ,

Redim points (NumPoints+1)
Redim segments (NumPoints)
points(0) = "NAME:PolylinePoints"
segments(0) = "NAME:PolylineSegments"
,

 ' Build the Point and Segment Arrays needed in the HFSS polyline call.
 ,

For n = 1 To (NumPoints+1)
 Angle = (n-1)*2*Pi/PointsPerTurn
 Radius = R(0) + ((n-1)/NumPoints)*(R(1)-R(0))
 Pitch = P(0) + ((n-1)/NumPoints)*(P(1)-P(0))
 Rise = (n-1)*Pitch/PointsPerTurn

 XValue = cstr(Radius*cos(Angle)) & Units
 YValue = cstr(Radius*sin(Angle)) & Units
 ZValue = cstr(Rise) & Units
 points(n) = Array("NAME:PLPoint", "X:=", XValue, "Y:=", _
YValue, "Z:=", ZValue)
,

```

```
' Create the line segments between each of the pairs of points.
,

If n<=NumPoints Then
segments(n) = Array("NAME:PLSegment", "SegmentType:=", _
"Line", "StartIndex:=", (n-1), "NoOfPoints:=", 2)
End If
Next
,

' Create the polyline.
,

oEditor.CreatePolyline _
Array("NAME:PolylineParameters", "IsPolylineCovered:=", true, _
"IsPolylineClosed:=", false, points, segments), _
Array("NAME:Attributes", "Name:=", "Line_Helix", "Flags:=", _
"", "Color:=", "(132 132 193)", "Transparency:=", 0.4, _
"PartCoordinateSystem:=", "Global", "MaterialName:=", _
"vacuum", "SolveInside:=", true)
,

' Create the helix cross-section.
,

oEditor.CreateCircle _
Array("NAME:CircleParameters", "IsCovered:=", true, "XCenter:=", _
cstr(R(0))&Units, "YCenter:=", 0, "ZCenter:=", 0, "Radius:=", _
"1"&Units, "WhichAxis:=", "Y"), _
Array("NAME:Attributes", "Name:=", "Circle_Helix", "Flags:=", _
"", "Color:=", "(132 132 193)", "Transparency:=", 0.4, _
"PartCoordinateSystem:=", "Global", "MaterialName:=", "vacuum", _
"SolveInside:=", true)
,

' Sweep the cross-section along the path.
```

```
,
oEditor.SweepAlongPath _
 Array("NAME:Selections", "Selections:=", _
 "Circle_Helix,Line_Helix"),
 Array("NAME:PathSweepParameters", "DraftAngle:=", "0deg", _
 "DraftType:=", "Round", "TwistAngle:=", "0deg")
```

## Data Export Script

The following sample script demonstrates how to export data and save it to a file. The output data in the example script is in 3 columns. The first column is frequency in GHz, the second is the Real part of S11, and the third is the Img part of S11. It uses a tab-delimited format. The output is done using output variables.

The frequency sweep data must be entered correctly. If it is incorrect, the script will request a freq point that does not exist and execution will stop.

The script includes comment lines, which are preceded by an apostrophe ( ') and offer explanations for each subsequent line or lines.

```
Dim oAnsoftApp
Dim oDesktop
Dim oProject
Dim oDesign
Dim oEditor
Dim oModule

Set oAnsoftApp = CreateObject("Ansoft.ElectronicsDesktop")
Set oDesktop = oAnsoftApp.GetAppDesktop()
set oProject = oDesktop.GetActiveProject
set oDesign = oProject.GetActiveDesign()
Dim oFS,oFile,x,y,z,path,range,
Dim arr2,del_f,freq,cfreq,val,temp,stn,stw,i,line
 ,
 ' Input the desired file name.
 ,

path = inputbox("Input the file name" &chr(13) & _
"Note: If you do not specify a path the file will " & _
```

```
"be placed in the script directory", _
"File", "C:\hfss_export.txt", 50, 50)
'
' If the user clicks Cancel the path will be blank, in which
' case the script should just exit.
If path <>"" then
'
' Create the file, open it for data entry, and output the
' column labels.
'
Set oFS = CreateObject("Scripting.FileSystemObject")
Set ofile = oFS.CreateTextFile (path)
line = "Freq" & chr(9) & "RE(S11)" & chr(9) & "IMG(S11)"
ofile.WriteLine line
'
' Input the needed freq, solution, and sweep data and clean it
' up.
'
msgbox("For the following input make sure it matches " & _
"the frequencies defined in your sweep")
range = inputbox("Input the range of frequencies in GHz " & _
"and number of points", _
"Frequency", "8,12,10", 50, 50)
'
' The following 2 lines define the 2 output variables.
'
oDesign.CreateOutputVariable "re_S", "re(S(port1,port1))"
oDesign.CreateOutputVariable "im_S", "im(S(port1,port1))"
arr = split (range, ",")
arr(0) = Trim(arr(0))
arr(1) = Trim(arr(1))
```

```
arr(2) = Trim(arr(2))
if cint(arr(2)) <> 1 then
del_f = (arr(1)-arr(0))/(arr(2)-1)
else
del_f = 0
end if

temp = InputBox("Input the Setup and Sweep number to use:"_
& chr(13) & "(e.g. input 1,2 for Setup1 and Sweep2)", _
"Solution Data","1,1",50,50)
arr2 = split(temp,",")
stn = arr2(0)
swn = arr2(1)
stn = Trim(stn)
swn = Trim(swn)

'
' Loop through the freq points.
'

for i=1 to arr(2) step 1
freq = arr(0) + (cint(i)-1)*del_f
x=freq
cfreq="Freq='" & freq & "Ghz'"

'
' Get the values of the output variables for the desired freq.
'

val = oDesign.GetOutputVariableValue("re_S","Setup" & _
stn & " :Sweep" & swn,cfreq, "")
y = val
val = oDesign.GetOutputVariableValue("im_S","Setup" & _
stn & " : Sweep" & swn,cfreq, "")
z = val
```

```
'
' Create the line of text to send to the file and write it to
the file.
'

line = x & chr(9) & y & chr(9) & z
ofile.WriteLine line
Next

'
' Delete the 2 output variables before finishing.
'

oDesign.DeleteOutputVariable "re_S"
oDesign.DeleteOutputVariable "im_S"

'
' Close the file.
'

ofile.close
End if
```

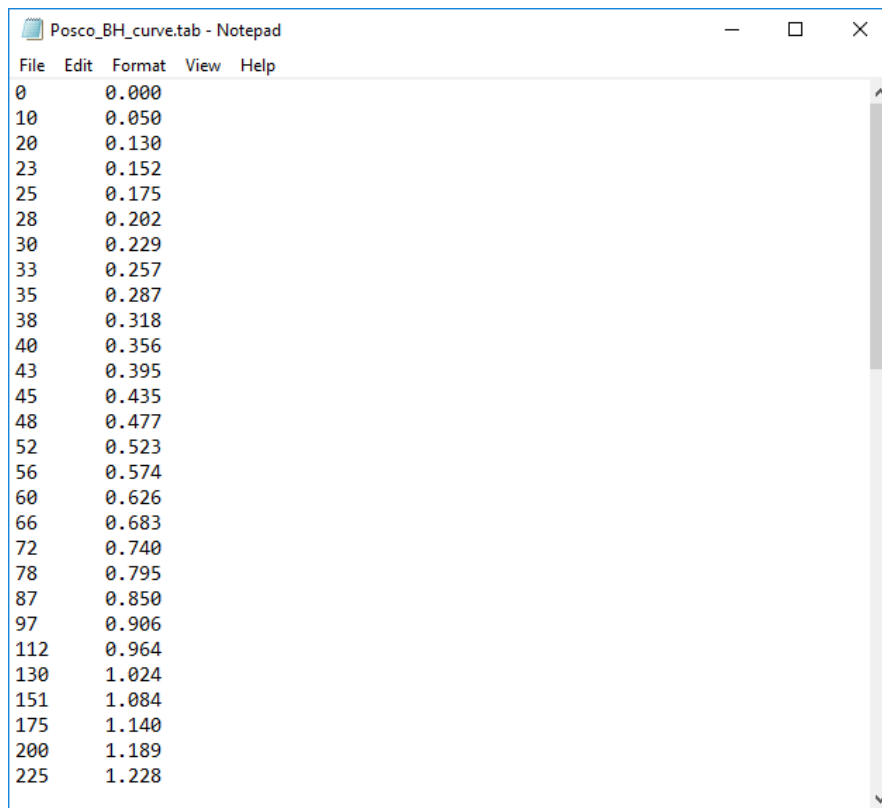
## IronPython Example Scripts

IronPython Examples:

- [BH Coordinates Python Script](#)
- [Equation Based Curve Python Script](#)

### BH Coordinates Python Script

This sample Python script adds a material ("Posco 35PN250") with BH coordinates from a specified file (Posco\_BH\_curve.tab):



|     |       |
|-----|-------|
| 0   | 0.000 |
| 10  | 0.050 |
| 20  | 0.130 |
| 23  | 0.152 |
| 25  | 0.175 |
| 28  | 0.202 |
| 30  | 0.229 |
| 33  | 0.257 |
| 35  | 0.287 |
| 38  | 0.318 |
| 40  | 0.356 |
| 43  | 0.395 |
| 45  | 0.435 |
| 48  | 0.477 |
| 52  | 0.523 |
| 56  | 0.574 |
| 60  | 0.626 |
| 66  | 0.683 |
| 72  | 0.740 |
| 78  | 0.795 |
| 87  | 0.850 |
| 97  | 0.906 |
| 112 | 0.964 |
| 130 | 1.024 |
| 151 | 1.084 |
| 175 | 1.140 |
| 200 | 1.189 |
| 225 | 1.228 |

To recreate this tab file, paste [the text below the script](#) into a text editor and save as Posco\_BH\_curve.tab.

The script itself includes comment lines, which are preceded by # and offer explanations for the subsequent line(s).

### Script Contents

```
specify path to the file with BH coordinates
path_to_file = r"D:\Posco_BH_curve.tab"

specify name of the material
material_name = "Posco 35PN250"

oProject = oDesktop.GetActiveProject()
oDefinitionManager = oProject.GetDefinitionManager()

""" create list with B and H points to pass to AddMaterial command
bh_coordinates list is a three dimensional array: 1D: name, 2D:
Coordinate list, 3D: BH point"""

bh_coordinates = ["NAME:BHCoordinates", ["NAME:DimUnits", "", ""]]
```

```
with open(path_to_file) as input_file:
 for line in input_file:
 h, b = line.split()

 bh_coordinates.append(["NAME:Coordinate", [
 "NAME:CoordPoint",
 float(h),
 float(b)
]
])

create a new magnetic material with BH curve
oDefinitionManager.AddMaterial(
[
 "NAME:" + material_name,
 "CoordinateSystemType:=", "Cartesian",
 "BulkOrSurfaceType:=" , 1,
 [
 "NAME:PhysicsTypes",
 "set:=" , ["Electromagnetic"]
],
 [
 "NAME:permeability",
 "property_type:=" , "nonlinear",
 "BTypeForSingleCurve:=" , "normal",
 "HUnit:=" , "A_per_meter", # unit can be specified as variable
 "BUnit:=" , "tesla", # unit can be specified as variable
 "IsTemperatureDependent:=", False,
 bh_coordinates,
 [
 "NAME:Temperatures"
```



```
]
],
 "conductivity:=" , "1818181.82",
 [
 "NAME:magnetic_coercivity",
 "property_type:=" , "VectorProperty",
 "Magnitude:=" , "0A_per_meter",
 "DirComp1:=" , "1",
 "DirComp2:=" , "0",
 "DirComp3:=" , "0"
]
])
```

**Posco\_BH\_Curve.tab Contents**

|    |       |
|----|-------|
| 0  | 0.000 |
| 10 | 0.050 |
| 20 | 0.130 |
| 23 | 0.152 |
| 25 | 0.175 |
| 28 | 0.202 |
| 30 | 0.229 |
| 33 | 0.257 |
| 35 | 0.287 |
| 38 | 0.318 |
| 40 | 0.356 |
| 43 | 0.395 |
| 45 | 0.435 |
| 48 | 0.477 |
| 52 | 0.523 |
| 56 | 0.574 |
| 60 | 0.626 |

|      |       |
|------|-------|
| 66   | 0.683 |
| 72   | 0.740 |
| 78   | 0.795 |
| 87   | 0.850 |
| 97   | 0.906 |
| 112  | 0.964 |
| 130  | 1.024 |
| 151  | 1.084 |
| 175  | 1.140 |
| 200  | 1.189 |
| 225  | 1.228 |
| 252  | 1.258 |
| 282  | 1.283 |
| 317  | 1.304 |
| 357  | 1.324 |
| 402  | 1.343 |
| 450  | 1.360 |
| 500  | 1.375 |
| 550  | 1.387 |
| 602  | 1.398 |
| 657  | 1.407 |
| 717  | 1.417 |
| 784  | 1.426 |
| 867  | 1.437 |
| 974  | 1.448 |
| 1117 | 1.461 |
| 1299 | 1.478 |
| 1515 | 1.495 |
| 1752 | 1.513 |
| 2000 | 1.529 |

|       |       |
|-------|-------|
| 2253  | 1.543 |
| 2521  | 1.557 |
| 2820  | 1.570 |
| 3167  | 1.584 |
| 3570  | 1.600 |
| 4021  | 1.617 |
| 4503  | 1.634 |
| 5000  | 1.652 |
| 5503  | 1.669 |
| 6021  | 1.685 |
| 6570  | 1.701 |
| 7167  | 1.717 |
| 7839  | 1.733 |
| 8667  | 1.749 |
| 9745  | 1.769 |
| 11167 | 1.793 |
| 12992 | 1.820 |
| 15146 | 1.851 |
| 17518 | 1.882 |
| 20000 | 1.909 |
| 22552 | 1.931 |
| 25417 | 1.948 |
| 28906 | 1.961 |
| 33333 | 1.971 |
| 38932 | 1.981 |

## Equation Based Curve Python Script

This sample Python script creates an equation based curve that produces a helix.

```
from math import pi, sin, cos
```

```
oProject = oDesktop.GetActiveProject()
```

```
oDesign = oProject.GetActiveDesign()
oEditor = oDesign.SetActiveEditor("3D Modeler")

Start_t = 0
End_t = pi*2

Npoint = 128
Nsection = Npoint-1

d_t = (End_t-Start_t)/Nsection

for n in range(1,Nsection):

 P1 = Start_t+d_t*(n-1)
 P2 = P1+d_t
 X_t1 = cos(P1*6)
 Y_t1 = sin(P1*6)
 Z_t1 = P1
 X_t2 = cos(P2*6)
 Y_t2 = sin(P2*6)
 Z_t2 = P2

oEditor.CreatePolyline(
[
 "NAME:PolylineParameters",
 "IsPolylineCovered:=" , True,
 "IsPolylineClosed:=" , False,
 [
 "NAME:PolylinePoints",
```

```
[
 "NAME:PLPoint",
 "X:=" , '1mm*' + str(X_t1),
 "Y:=" , '1mm*' + str(Y_t1),
 "Z:=" , '1mm*' + str(Z_t1)
],
[
 "NAME:PLPoint",
 "X:=" , '1mm*' + str(X_t2),
 "Y:=" , '1mm*' + str(Y_t2),
 "Z:=" , '1mm*' + str(Z_t2)
]
],
[
 "NAME:PolylineSegments",
 [
 "NAME:PLSegment",
 "SegmentType:=" , "Line",
 "StartIndex:=" , 0,
 "NoOfPoints:=" , 2
]
],
[
 "NAME:PolylineXSection",
 "XSectionType:=" , "None",
 "XSectionOrient:=" , "Auto",
 "XSectionWidth:=" , "0mm",
 "XSectionTopWidth:=" , "0mm",
 "XSectionHeight:=" , "0mm",
 "XSectionNumSegments:=" , "0",
```

```
"XSectionBendType:=" , "Corner"
]
],
[
 "NAME:Attributes",
 "Name:=" , "Polyline"+str(n),
 "Flags:=" , "",
 "Color:=" , "(132 132 193)",
 "Transparency:=" , 0,
 "PartCoordinateSystem:=", "Global",
 "UDMId:=" , "",
 "MaterialValue:=" , "\"vacuum\"",
 "SurfaceMaterialValue:=", "\"\"",
 "SolveInside:=" , True,
 "IsMaterialEditable:=" , True,
 "UseMaterialAppearance:=", False,
 "IsLightweight:=" , False
])
```

# Index

## 3

### 3D Modeler editor commands 10-1

AssignMaterial 10-142  
Chamfer 10-144  
ChangeProperty 6-8, 10-262  
Connect 10-146  
Copy 10-120  
CoverLines 10-147  
CoverSurfaces 10-148  
CreateBondwire 10-10  
CreateBox 10-13  
CreateCircle 10-16  
CreateCone 10-19  
CreateCutplane 10-21  
CreateCylinder 10-23  
CreateEllipse 10-26  
CreateEntityList 10-149  
CreateEquationCurve 10-29  
CreateEquationSurface 10-32  
CreateFaceCS 10-151, 10-157  
CreateGroup 10-156  
CreateHelix 10-35  
CreateObjectFromdges 10-164  
CreateObjectFromFaces 10-165  
CreatePoint 10-37

CreatePolyline 10-47  
CreateRectangle 10-52  
CreateRegion 10-55  
CreateRegularPolygon 10-61  
CreateRegularPolyhedron 10-58  
CreateRelativeCS 10-167  
CreateSphere 10-64  
CreateSpiral 10-67  
CreateTorus 10-69  
CreateUserDefinedModel 10-71  
CreateUserDefinedPart 10-39, 10-79  
Delete 10-269  
DeleteEmptyGroups 10-169  
DeleteLastOperation 10-170  
DeletePolylinePoint 10-120  
DetachFaces 10-171  
DuplicateAlongLine 10-122  
DuplicateAroundAxi 10-124  
DuplicateMirror 10-127  
Edit3dComponent 10-89  
EditEntityList 10-172  
EditFaceCS 10-174  
EditObjectCS 10-180  
EditPolyline 10-92

|                                       |                               |
|---------------------------------------|-------------------------------|
| EditRelativeCS 10-187                 | GetVertexIDsFromObject 10-287 |
| Export 10-189                         | GetVertexPosition 10-288      |
| ExportModelImageToFile 10-191         | HealObject 10-198             |
| Fillet 10-194                         | Import 10-205                 |
| FlattenGroup 10-196                   | ImportDXF 10-208              |
| GenerateHistory 10-197                | ImportGDSII 10-212            |
| Get3DComponentParameters<br>10-96     | Insert3DComponent 10-99       |
| GetActiveCoordinateSystem 10-<br>203  | InsertNativeComponent 10-101  |
| GetBodyNamesByPosition 10-<br>270     | Intersect 10-216              |
| GetCoordinateSystems 10-204           | Mirror 10-130                 |
| GetEdgeByPosition 10-271              | Move 10-132                   |
| GetEdgeIDsfromFace 10-273             | MoveCSToEnd 10-217            |
| GetEdgeIDsfromObject 10-274           | MoveEntityToGroup 10-218      |
| GetFaceArea 10-274                    | MoveFaces 10-219              |
| GetFaceByPosition 10-276              | OffsetFaces 10-134            |
| GetFaceCenter 10-275                  | PageSetup 10-290              |
| GetFaceIDs 10-277                     | ProjectSheet 10-222           |
| GetGeometryModelerMode 10-<br>278     | RenamePart 10-292             |
| GetModelBoundingBox 10-279            | Rotate 10-136                 |
| GetProperties 6-38, 8-25              | Scale 10-137                  |
| GetPropertyValue 6-39, 8-26,<br>12-62 | Section 10-224                |
| GetPropEvaluatedValue 6-34            | SeparateBody 10-225           |
| GetPropSIValue 6-36                   | SetModelUnits 10-226          |
| GetUser Position 10-285               | SetWCS 10-228                 |
| GetVertexIDsFromEdge 10-286           | Split 10-229                  |
| GetVertexIDsFromFace 10-287           | Subtract 10-232               |
|                                       | SweepAlongPath 10-107         |
|                                       | SweepAlongVector 10-109       |



- SweepAroundAxis 10-111
- SweepFacesAlongNormal 10-113, 10-113, 10-234, 10-234
- Sweep-FacesAlongNormalWithAttributes 10-115
- ThickenSheet 10-235
- UncoverFaces 10-237
- Ungroup 10-240
- Unite 10-239
- UpdateComponentDefinition 10-118
- WrapSheet 10-241
- 3D Modeler editor object commands
  - GetNumObjects 10-284
  - GetObjectIDByName 10-279
  - GetObjectName 10-280
  - GetObjectNameByFaceID 10-281
- A**
- Aborting Scripts 1-11
- Add 7-48, 7-131, 7-165, 7-193, 7-238
- AddAddSolverOnDemandModel 7-76
- AddAssignmentToBoundary 13-3
- AddCartesianLimitLine 12-4
- AddCartesianXMarker 12-6
- AddCartesianYMarker 12-7
- AddDataset 6-4
- AddDefinitionFromBlock 10-245
- AddDefinitionFromLibFile 10-250
- AddDeltaMarker 12-8
- AddDiffPair 24-4
- AddDynamicNPortData 7-67
- AddLevel 27-3
- AddMarker 12-9
- AddMarker[Fields Reporter] 18-3
- AddMarkerToPlot 18-2
- AddMaterial 7-2, 7-19
- AddMessage 4-5
- AddNote 12-10
- AddNPortData 7-70
- AddPinGrounds 26-48
- AddPinIPorts 26-51
- AddPinPageConnectors 26-52
- AddReduceOp 21-2
- AddTerminalsToWinding 13-49
- AddTraceCharacteristics 12-12
- AddTraces 12-13
- AlignHorizontal 26-53, 27-4
- AlignVertical 26-54, 27-5
- Analysis module commands
  - Analyze 9-6
  - CopySetup 15-3, 16-7
  - DeleteSetups 15-3
  - EditSetup 15-4
  - ExportIcepak 15-18
  - ExportSolnData 15-18

- GetSetupCount 15-19
- GetSetups 15-20
- GetSweepCount 15-21
- GetSweeps 15-21
- InsertSetup [Maxwell] 15-22
- PasteSetup 15-42
- PasteSweep 15-43
- RenameSetup 15-43
- ResetAllToTimeZero 15-44
- ResetSetupToTimeZero 15-44
- ResetToTimeZero 9-42
- RevertAllToInitial 15-45
- RevertAllToInitialTemperature 15-45
- RevertSetupToInitial 15-46
- RevertSetupToInitialTemperature 15-47
- Analysis Setup Module Script Commands 15-1
- Analyze
  - Analysis module command 9-6
- AnalyzeAll 9-7
- AnalyzeDistributed 9-7
- Ansoft Application Object commands 3-1
  - GetAppDesktop 3-2
- ApplyMeshOps 9-5
- arithmetic operators 1-6
- array variables 1-4
- AssignBand B
  - AssignCharge 13-50
  - AssignCoilGroup 13-50
  - AssignCoilTerminal 13-52
  - AssignCoilTerminalGroup 13-52
  - AssignCurrent 13-54
  - AssignCurrentDensity 13-55
  - AssignCurrentDensityGroup 13-56
  - AssignCurrentDensityTerminal 13-56
  - AssignCurrentDensityTerminalGroup 13-56, 13-57
  - AssignCurrentGroup 13-57
  - AssignCylindricalGapOp 14-5
  - AssignCylindricalHField 13-19
  - AssignDependent 13-20
  - AssignFloating 13-57
  - AssignFluxTangential 13-23
  - AssignForce 21-4
  - AssignImpedance 13-23
  - AssignIndependent 13-24
  - AssignInsulating 13-26
  - AssignLayoutForce 21-5
  - AssignLengthOp 14-12
  - AssignMaterial 10-142
  - AssignMatrix 21-7
  - AssignModelResolutionOp 14-15
  - AssignRadiation 13-26
  - AssignResistiveSheet 13-27
  - AssignSink 13-58
  - AssignSkinDepthLayerSetting 14-17

AssignSkinDepthOp 14-18  
AssignSymmetry 13-29  
AssignTangentialHField 13-30  
AssignThinLayer 13-30  
AssignTouching 13-33  
AssignTrueSurfOp 14-19  
AssignVectorPotential 13-34  
AssignVoltage 13-58  
AssignVoltageAPhi 13-59  
AssignVoltageDrop 13-60  
AssignVoltageDropGroup 13-61  
AssignVoltageGroup 13-61  
AssignVolumeChargeDensity 13-61  
AssignWindingGroup 13-62  
AssignZeroTangentialHField 13-36

## **B**

Boundary/Excitation module commands  
AddAssignmentToBoundary 13-3  
AddTerminalsToWinding 13-49  
AssignCharge 13-50  
AssignCoilGroup 13-50  
AssignCoilTerminal 13-52  
AssignCoilTerminalGroup 13-52  
AssignCurrentDensity 13-55  
AssignCurrentDensityGroup 13-56

AssignCurrentDensityTerminal 13-56  
AssignCurrentDensityTerminalGroup 13-56, 13-57  
AssignCurrentGroup 13-57  
AssignCylindricalHField 13-19  
AssignDependent 13-20  
AssignFloating 13-57  
AssignImpedance 13-23  
AssignIndependent 13-24  
AssignInsulating 13-26  
AssignRadiation 13-26  
AssignSink 13-58  
AssignSymmetry 13-29  
AssignTangentialHField 13-30  
AssignVoltage 13-58  
AssignVoltageDrop 13-60  
AssignVoltageDropGroup 13-61  
AssignVoltageGroup 13-61  
AssignVolumeChargeDensity 13-61  
AssignWindingGroup 13-62  
AssignZeroTangentialHField 13-36  
DeleteAllBoundaries 13-4  
DeleteAllExcitations 13-5  
DeleteBoundaries 13-5  
EditCharge 13-63  
EditCoilTerminal 13-63  
EditCurrentDensity 13-65

- EditCurrentDensityTerminal 13-65
  - EditCylindricalHField 13-36
  - EditDependent 13-36
  - EditExternalCircuit 13-65
  - EditFloating 13-66
  - EditImpedance 13-38
  - EditIndependent 13-38
  - EditInsulating 13-39
  - EditRadiation 13-40
  - EditSink 13-66
  - EditSymmetry 13-42
  - EditTangentialHField 13-43
  - EditVoltage 13-66
  - EditVoltageDrop 13-69
  - EditVolumeChargeDensity 13-69
  - EditWindingGroup 13-70
  - EditZeroTangentialHField 13-47
  - GetBoundaries 13-7
  - GetBoundariesOfType 13-7
  - GetBoundaryAssignment 13-6
  - GetCoreLossEffect 13-8
  - GetCoreLossEffectOnField 13-9
  - GetDisplacementCurrent 13-9
  - GetEddyEffect 13-9
  - GetExcitations 13-10
  - GetExcitationsOfType 13-11
  - GetNumBoundaries 13-11
  - GetNumBoundariesOfType 13-12
  - GetNumExcitations 13-13
  - GetNumExcitationsOfType 13-13
  - ReassignBoundaries 13-3, 13-14
  - ReassignBoundary 13-14
  - RemoveAssignmentFromBoundary 13-15
  - RenameBoundary 13-16
  - ReprioritizeBoundaries 13-17
  - SetCoreLoss 13-70
  - SetEddyEffect 13-70
  - SetMagnetizationCompute 13-71
  - SetMinimumTimeStep 13-73
  - SetPermeabilityLink 13-75
  - boundary/excitation script commands 13-2
  - BreakUDMConnection 10-261
  - BringToFront 7-176, 27-6
- C**
- CalcOp 18-6, 19-5
  - CalcRead (deprecated) 19-6
  - CalcStack 18-6, 19-8
  - CalculatorRead 18-7, 19-7
  - CalculatorWrite 18-8, 19-8
  - Cascade 24-5
  - Chamfer 10-144
  - ChangeGeomSettings 18-12, 19-13
  - ChangeProperty 6-8, 10-262, 12-16, 26-105, 27-6
  - ClcEval 18-13, 19-13

- ClcMaterial 18-17, 19-17
- ClcMaterialValue 18-18
- ClearAllMarkers 12-20
- ClearAllMarkers[Fields Reporter] 18-18
- ClearAllNamedExpr 18-19, 19-18
- ClearAllTraceCharacteristics 12-20
- ClearDiffPairs 24-6
- ClearMessages 2-34, 4-6, 6-15
- ClearSolutionCache 7-76
- Clone 24-7
- CloneMaterial 7-16
- Close 2-35, 6-16, 24-8
- CloseAllWindows 4-10
- CloseEditor 26-57, 27-19
- CloseProject 4-11
- CloseProjectNoForce 4-12
- Combine 24-9
- comparison operators 1-7
- CompInstance Functions 25-2
- CompInstance Script Commands 25-1
- Component Manager Script Commands 7-47
- ComputeCoreLossCoefficients 7-17
- conditional statements
  - If...Then... Else 1-8
  - Select Case 1-8
  - types of 1-8
- ConfigureFluentConductivityCoupling 9-9
- Connect 10-146
- ConstructVariationString 9-10
- conventions
  - scripting help 1-1
- converting data types 1-10
- ConvertToDynamic 7-142
- ConvertToParametric 7-142
- Copy 10-120, 26-57, 27-20
- CopyData 26-58
- CopyDesign 2-36, 6-16
- CopyNamedExprToStack 18-20, 19-18
- CopyReportDefinition 12-23
- CopyReportsData 12-22
- Copyright and Trademark Information 2
- CopySetup
  - Analysis module command 15-3, 16-7
  - Optimetrics module command 15-3, 16-7
- CopySubdesign 26-59
- CopyTraceDefinitions 12-23
- CopyTracesData 12-21
- Core Global Script Context Commands 28-1
- Count 4-12
- CoverLines 10-147
- CoverSurfaces 10-148

|                              |                                    |
|------------------------------|------------------------------------|
| CPython 1-69                 | CreateRegion 10-55                 |
| Create3DComponent 10-6       | CreateRegularPolygon 10-61         |
| Create3DDesign 9-11          | CreateRegularPolyhedron 10-58      |
| CreateArc 27-20              | CreateRelativeCS 10-167            |
| CreateBondwire 10-10         | CreateReport 12-25                 |
| CreateBox 10-13              | CreateReportFromFile 12-24         |
| CreateCircle 10-16, 27-21    | CreateReportFromTemplate 12-35     |
| CreateCone 10-19             | CreateReportofAllQuantities 12-36  |
| CreateCutplane 10-21         | CreateSphere 10-64                 |
| CreateCylinder 10-23         | CreateSpiral 10-67                 |
| CreateEllipse 10-26          | CreateText 27-27                   |
| CreateEntityList 10-149      | CreateTorus 10-69                  |
| CreateEquationCurve 10-29    | CreateUserDefinedModel 10-71       |
| CreateEquationSurface 10-32  | CreateUserDefinedPart 10-39, 10-79 |
| CreateFaceCS 10-151, 10-157  | CreateUserDefinedSolution 23-1     |
| CreateFieldPlot 18-21        | Cut 26-61, 27-27                   |
| CreateGroup 10-156           | CutDesign 2-36, 6-17               |
| CreateHelix 10-35            |                                    |
| CreteImage 27-22             | <b>D</b>                           |
| CreateLine 27-23             | dataset commands                   |
| CreateObjectFromEdges 10-164 | AddDataset 6-4                     |
| CreateObjectFromFaces 10-165 | DeleteDataset 6-18                 |
| CreateOutputVariable 11-1    | EditDataset 6-20                   |
| CreatePage 26-60             | ExportDataset 6-23                 |
| CreatePin 27-24              | ImportDataset 6-43                 |
| CreatePoint 10-37            | DeactivateOpen 26-62               |
| CreatePolygon 27-25          | DeactivateShort 26-63              |
| CreatePolyline 10-47         | Deembed 24-10                      |
| CreateRectangle 10-52, 27-26 | DeembedBack 24-11                  |

- DeembedFront 24-13
- Definition Editor Script
  - Commands 27-1
- Definition Manager commands
  - AddDefinitionFromBlock 10-245
  - AddDefinitionFromLibFile 10-250
  - GetProjectMaterialNames 7-43
- Definition Manager Script Commands 7-1
- Delete 10-269, 26-64, 27-28
- DeleteAllBoundaries 13-4
- DeleteAllExcitations 13-5
- DeleteAllParameters 21-13
- DeleteAllReports 12-38
- DeleteBoundaries 13-5
- DeleteDataset 6-18
- DeleteDesign 2-37, 6-19
- DeleteEmptyGroups 10-169
- DeleteFieldPlot 18-46
- DeleteFieldVariation 9-12
- DeleteFullVariation 9-13
- DeleteLastOperation 10-170
- DeleteLinkedDataVariation 9-14
- DeleteMarker 12-37
- DeleteMarker[Fields Reporter] 18-47
- DeleteMotionSetup A
- DeleteNamedExpr 18-47, 19-19
- DeleteOp 14-2
- DeleteOutputVariable 11-3
- DeleteParameters 21-13
- DeletePolylinePoint 10-120
- DeleteProject 4-14
- DeleteReduceMatrix 21-14
- DeleteReduceOp 21-15
- DeleteReport 12-39
- DeleteSetups 15-3
  - Analysis module command 15-3
  - Optimetrics module command 16-7
- DeleteSolutionVariation 17-7
- Deletetraces 12-39
- DeleteUneditablePlot 18-48
- DeleteUserDefinedSolutions 23-3
- DeleteVariation 9-15
- Design object commands 12-49, 12-50, 12-51, 12-52, 12-53, 12-66
  - AddCartesianLimitLine 12-4
  - AddCartesianXMarker 12-6
  - AddCartesianYMarker 12-7
  - AddDeltaMarker 12-8
  - AddMarker 12-9
  - AddMarkerToPlot 18-2
  - AddNote 12-10
  - AddTraceCharacteristics 12-12
  - AddTraces 12-13
  - AnalyzeAll 9-7
  - AnalyzeDistributed 9-7

|                                      |                                    |
|--------------------------------------|------------------------------------|
| ApplyMeshOps 9-5                     | GetLibraryDirectory 4-25           |
| CalculatorRead 18-7, 19-7            | GetMatchedObjectName 10-283        |
| ClearAllMarkers 12-20                | GetModelUnits 10-284               |
| ClearAllTraceCharacteristics 12-20   | GetModule 9-34                     |
| CloseAllWindows 4-10                 | GetName 9-35, 24-30                |
| CopyReportDefinition 12-23           | GetObjectsByMaterial 10-281        |
| CreateOutputVariable 11-1            | GetObjectsInGroup 10-282           |
| CreateReport 12-25                   | GetOutputVariableValue 11-6        |
| CreateReportFromTemplate 12-35       | GetProjectDirectory 4-32           |
| DeleteAllReports 12-38               | GetProperties 6-38, 8-25           |
| DeleteOutputVariable 11-3            | GetPropertyValue 6-39, 8-26, 12-62 |
| DeleteReport 12-39                   | GetPropEvaluatedValue 6-34         |
| DeleteTraces 12-39                   | GetPropSIValue 6-36                |
| DoesOutputVariableExist 11-3         | GetRegistryInt 4-68                |
| EditOutputVariable 11-4              | GetRegistryString 4-69             |
| ExportConvergence 9-25               | GetSelections 10-285               |
| ExportElementBasedHarmonicForce 9-26 | GetSolutionType 9-38               |
| ExportMeshStats 9-29                 | GetTempDirectory 4-36              |
| ExportPlotImageToFile 12-42, 18-68   | GetVariables 6-42, 8-28            |
| ExportPlotImageWithViewToFile 18-69  | GetVariableValue 6-41, 8-29        |
| ExportProfile 9-30                   | GetVariationVariableValue 9-39     |
| ExportReport 12-43                   | GetVersion 4-37                    |
| ExportToFile 12-46                   | ImportIntoReport 12-72             |
| Fast Transformation 9-57, 9-58       | Is2D 9-39                          |
| GetDisplayType 12-59                 | Is3D 9-40                          |
|                                      | PasteReports 12-77                 |
|                                      | Redo 9-40                          |
|                                      | RenameDesignInstance 9-41          |



|                                           |                                                     |
|-------------------------------------------|-----------------------------------------------------|
| RenameReport 12-79                        | Count 4-12                                          |
| RenameTraces 12-80                        | DeleteProject 4-14                                  |
| RunToolkit 6-53, 9-69                     | EnableAutosave 4-15                                 |
| SavePlotSettingsAsDefault 12-82           | ExportOptionsFiles 4-15                             |
| SetActiveEditor 9-42                      | GetActiveProject 4-16                               |
| SetConductivityThreshold 9-43             | GetAutosaveEnabled 4-17                             |
| SetDesignSettings 9-44                    | GetBuildTimeDateString 4-18                         |
| SetLibraryDirectory 4-60                  | GetChildNames 6-25                                  |
| SetProjectDirectoryVBCommand> 4-61        | GetChildTypes 6-27                                  |
| SetPropertyValue 6-62, 8-30               | GetCustomMenuSet 4-18                               |
| SetTempDirectory 4-62                     | GetDesigns 4-21, 6-30                               |
| SetupYConnection 13-73                    | GetDesktopConfiguration 4-22                        |
| SetVariableValue 6-63, 8-31               | GetDistributedAnalysisMachines 4-23                 |
| Show Layout 9-65, 9-65                    | GetDis-trib-utedAna-lysisMachinesForDesignType 4-23 |
| ShowWindow 10-229                         | GetName 4-29                                        |
| Solve 9-67, 9-70                          | GetProjectList 4-33                                 |
| StopSimLink 9-68                          | GetProjects 4-28, 4-32                              |
| UpdateAllReports 12-82                    | GetPropNames 6-35                                   |
| ValidateDesign 9-70                       | GetSchematicEnvironment 4-34                        |
| Design Object Script Commands 9-1         | LaunchJobMonitor 4-38                               |
| Desktop Commands For Registry Values 4-67 | NewProject 4-39                                     |
| Desktop object commands                   | OpenMultipleProjects 4-41                           |
| AddMessage 4-5                            | OpenProject 4-41                                    |
| ClearMessages 4-6                         | PauseRecording 4-44                                 |
| CloseProject 4-11                         | PauseScript 4-45                                    |
| CloseProjectNoForce 4-12                  | Print 4-45                                          |

- QuitApplication 4-46
- RefreshJobMonitor 4-47
- ResetLogging 4-48
- RestoreWindow 4-49
- ResumeRecording 4-50
- RunProgram 4-51
- RunScript 4-52
- SelectScheduler 4-55
- SetActiveProject 4-56
- SetActiveProjectByPath 4-57
- SetCustomMenuSet 4-58
- SetDesktopConfiguration 4-59
- SetSchematicEnvironment 4-61
- Sleep 4-64
- SubmitJob 4-65
- TileWindows 4-66
- Desktop Object Script  
Commands 4-1
- Desktop Scripting Conventions 1-  
82
- DetachFaces 10-171
- DisableDiffPairs 24-14
- DisplayPinNames 27-29
- DistributedAnalyzeSetup, Opti-  
metrics module command 16-  
8
- DoesNamedExpressionExists 18-  
49
- DoesOutputVariableExist 11-3
- Draw Menu commands 10-4
- DuplicateAlongLine 10-122
- DuplicateAroundAxis 10-124
- DuplicateMirror 10-127
- E**
- Edit 7-77, 7-143, 7-176, 7-216, 7-245
- Edit Menu Commands 10-119
- Edit3DComponent 10-86, 10-89
- EditCharge 13-63
- EditCoilTerminal 13-63
- EditCurrent 13-63
- EditCurrentDensity 13-65
- EditCurrentDensityTerminal 13-65
- EditCylindricalGapOp 14-22
- EditCylindricalHField 13-36
- EditDataset 6-20
- EditDependent 13-36
- EditEntityList 10-172
- EditExternalCircuit 13-65
- EditFaceCS 10-174
- EditFloating 13-66
- EditFluxTangential\_ 13-37
- EditForce 21-16
- EditImpedance 13-38
- EditIndependent 13-38
- EditInsulating 13-39
- EditLayoutForce 21-16
- EditLengthOp 14-27
- EditMatrix 21-18
- EditModelResolutionOp 14-30

- EditMotionSetup E
- EditNotes 9-15
- EditObjectCS 10-180
- Editor object commands
  - SetPropertyValue 6-62, 8-30
- Editor Scripting IDs 26-4
- EditOutputVariable 11-4
- EditPolyline 10-92
- EditRadiation 13-40
- EditReduceOp 21-18
- EditRelativeCS 10-187
- EditResistiveSheet 13-40
- EditSetup
  - Analysis module command 15-4
  - optimization command 16-29
  - parametric command 16-19
  - sensitivity command 16-47
  - statistical command 16-59
- EditSink 13-66
- EditSkinOp 14-31
- EditSolverOnDemandModel 7-104
- EditSurfaceMeshSummaryData 18-49
- EditSymmetry 13-42
- EditTangentialHField 13-43
- EditThinLayer 13-43
- EditTorque 21-19
- EditTouching 13-45
- EditTrueSurfOp 14-33
- EditUserDefinedSolution 23-4
- EditVectorPotential 13-46
- EditVoltage 13-66
- EditVoltageAPhi 13-68
- EditVoltageDrop 13-69
- EditVolumeChargeDensity 13-69
- EditWindingGroup 13-70
- EditWithComps 7-104, 7-143, 7-177, 7-216, 7-252
- EditZeroTangentialHField 13-47
- ElectricRuleCheck 26-66
- EnableAutoSave 4-15
- EnableDiffPairs 24-15
- EnableHarmonicForceCalculation 9-15
- EnterComplex 18-53, 19-20
- EnterComplexVector 18-54, 19-20
- EnterCoord 18-55
- EnterEdge 18-56
- EnterLine 18-57, 19-21
- EnterOutputVar 18-57
- EnterPoint 18-58, 19-22
- EnterQty 18-59, 19-23
- EnterScalar 18-59, 19-24
- EnterScalarFunc 18-60, 19-24
- EnterSurf 18-61, 19-25
- EnterVector 18-61, 19-26
- EnterVectorFunc 18-62, 19-26
- EnterVol 18-63, 19-27

Event Callback Scripting 1-85  
Example Scripts 29-1  
Export 7-116, 7-155, 7-188, 7-234,  
7-259, 10-189  
ExportCircuit 15-13  
ExportCitiFile 24-16  
ExportConvergence 9-25  
ExportDataset 6-23  
ExportDXConfigFile, Optimetrics  
module command 16-9  
ExportFieldPlot 18-63  
ExportFile 27-29  
ExportFullWaveSpice 7-161  
ExportHarmonicTransientForce 9-  
27  
ExportIcepak, Analysis module com-  
mand 15-17  
ExportImageToFile 12-41  
ExportMarkerTable 12-47, 18-64  
ExportMaterial 7-44  
ExportMatlab 24-18  
ExportMesh Stats 9-29  
ExportNetlist 26-68  
ExportOptimetricsProfile, Opti-  
metrics module command 16-  
10  
ExportOptimetricsResults, Opti-  
metrics module command 16-  
11  
ExportOptionsFiles 4-15  
ExportParametricResults, Opti-  
metrics module command 16-  
12

ExportPlotImageToFile 12-42, 18-68  
ExportPlotImageWithViewToFile 18-69  
ExportProfile 9-26, 9-30  
ExportReport 12-43  
ExportScript 7-266  
ExportSolnData, Analysis module com-  
mand 15-18  
ExportSpreadsheet 24-20  
ExportSurfaceMeshSummary 18-70  
ExportToFile 12-45, 18-72  
ExportTouchstone 24-22  
ExportTouchstone2 24-24  
ExportTransientData 17-11  
Extract 24-27

## F

FFTONReport 12-48, 17-12  
Field Calculator commands  
DoesNamedExpressionExists 18-49  
Field Overlay module commands,  
GetFieldPlotNames 18-74  
field overlay script commands 18-1  
Field Overlays module commands  
CalcOp 18-6, 19-5  
CalcStack 18-6, 19-8  
ChangeGeomSettings 18-12, 19-13  
ClcMaterial 18-17, 19-17  
ClearAllNamedExpr 18-19, 19-18  
CopyNamedExprToStack 18-20, 19-18  
CreateFieldPlot 18-21

|                                  |                                    |
|----------------------------------|------------------------------------|
| DeleteFieldPlot 18-46            | ModifyFieldPlot 18-76, 18-82       |
| DeleteNamedExpr 18-47, 19-19     | ReassignFieldPlot 18-86            |
| DeleteUneditablePlot 18-48       | RenameFieldPlot 18-89              |
| EditSurfaceMeshSummaryData 18-49 | RenamePlotFolder 18-90             |
| EnterComplex 18-53, 19-20        | SaveFieldsPlots 18-90              |
| EnterComplexVector 18-54, 19-20  | SetFieldPlotSettings 18-91, 18-95  |
| EnterCoord 18-55                 | SetPlotFolderSettings 18-98        |
| EnterEdge 18-56                  | SetPlotsViewSolutionContext 18-103 |
| EnterLine 18-57, 19-21           | UpdateAllFieldsPlots 18-104        |
| EnterOutputVar 18-57             | UpdateQuantityFieldsPlots 18-105   |
| EnterPoint 18-58, 19-22          | Fields Calculator commands         |
| EnterQty 18-59, 19-23            | AddNamedExpr 18-4, 19-2            |
| EnterScalar 18-59, 19-24         | CalcOp 18-6, 19-5                  |
| EnterScalarFunc 18-60, 19-24     | CalcStack 18-6, 19-8               |
| EnterSurf 18-61, 19-25           | CalculatorWrite 18-8, 19-8         |
| EnterVector 18-61, 19-26         | ChangeGeomSettings 18-12, 19-13    |
| EnterVectorFunc 18-62, 19-26     | ClcEval 18-13, 19-13               |
| EnteVol 18-63, 19-27             | ClcMaterial 18-17, 19-17           |
| ExportFieldPlot 18-63            | ClcMaterialValue 18-18             |
| ExportOnGrid 18-65               | ClearAllNamedExpr 18-19, 19-18     |
| ExportSurfaceMeshSummary 18-70   | CopyNamedExprToStack 18-20, 19-18  |
| GetFieldFolderNames 18-73        | DeleteNamedExpr 18-47, 19-19       |
| GetFieldPlotQuantityName 18-74   | EnterComplex 18-53, 19-20          |
| GetMeshPlotNames 18-75           | EnterComplexVector 18-54, 19-20    |
| HidePolarPlot 18-79              | EnterCoord 18-55                   |
| HideRadiatedPlotOverlay 18-80    | EnterEdge 18-56                    |
|                                  | EnterLine 18-57, 19-21             |

- EnterOutputVar 18-57
  - EnterPoint 18-58, 19-22
  - EnterQty 18-59, 19-23
  - EnterScalar 18-59, 19-24
  - EnterScalarFunc 18-60, 19-24
  - EnterSurf 18-61, 19-25
  - EnterVector 18-61, 19-26
  - EnterVectorFunc 18-62, 19-26
  - EnterVol 18-63, 19-27
  - ExportOnGrid 19-28
  - ExportToFile 19-31
  - Fields Calculator Script  
Commands 19-1
  - Fields reporter module commands
    - AddMarker[Fields Reporter] 18-3
    - ClearAllMarkers[Fields Reporter] 18-18
    - DeleteMarker[Fields Reporter] 18-47
    - ExportMarkerTable 12-47, 18-64
  - Fillet 10-194
  - FindElements 26-69
  - FitToBorder 26-71
  - FlattenGroup 10-196
  - FlipHorizontal 26-75, 27-30
  - FlipVertical 26-76, 27-31
  - For...Next loop 1-9
  - functions
    - VBScript procedures 1-9
- G**
- general method script commands 26-45
  - GenerateHistory 10-197
  - GenerateVariationData Parametric, parametric command 16-21
  - Get3DComponentDefinitionNames 10-97
  - Get3DComponentInstanceNames 10-97
  - Get3DComponentMaterialNames 10-98
  - Get3DComponentMaterialProperties 10-99
  - Get3DComponentParameters 10-96
  - GetActiveCoordinateSystem 10-203
  - GetActiveDesign 2-39, 6-24
  - GetActiveProject 4-16
  - GetAdaptiveSettings 17-13
  - GetAllCategories 12-49
  - GetAllQuantities 12-50
  - GetAllReportNames 12-49
  - GetAllSourceMagnitudes 17-14
  - GetAllSourceModes 17-14
  - GetAllSourcePhases 17-15
  - GetAllSources 17-16
  - GetAntennaParameters 17-16
  - GetArrayVariables 6-24, 8-24
  - GetAutoSaveEnabled 4-17
  - GetAvailableDisplayTypes 12-51
  - GetAvailableReportTypes 12-52
  - GetAvailableSolutions 12-53
  - GetBodyNamesByPosition 10-270

|                                                  |                                                       |
|--------------------------------------------------|-------------------------------------------------------|
| GetBoundaries 13-7                               | GetDis-                                               |
| GetBoundariesOfType 13-7                         | trib-                                                 |
| GetBoundaryAssignment 13-6                       | utedAna-                                              |
| GetBuildDateTimeString 4-18                      | lysisMachinesForDesignType                            |
| GetChildNames 6-25                               | 4-23                                                  |
| GetChildObject Design 6-26                       | GetDocumentNames 22-8                                 |
| GetChildTypes 6-27                               | GetEddyEffect 13-9                                    |
| GetComponentName 25-2                            | GetEdgeByPosition 10-271                              |
| GetComponentPinLocation 26-118                   | GetEdgeIdsFromFace 10-273                             |
| GetCoordinateSystems 10-204                      | GetEdgeIdsFromObject 10-274                           |
| GetCoreLossEffect 13-8                           | GetEditorName [Schematic] 26-119                      |
| GetCoreLossEffectOnField 13-9                    | GetEvaluatedPropertyValue 26-106                      |
| GetData 7-117, 7-126, 7-156, 7-189, 7-235, 7-260 | GetExcitations 13-10                                  |
| GetDataExpressions 12-53                         | GetExcitationsOfType 13-11                            |
| GetDataUnits 12-54                               | GetExeDir 4-24                                        |
| GetDefinitionManager 2-40, 6-28                  | GetFaceArea 10-274                                    |
| GetDependentFiles 2-41, 6-29                     | GetFaceByPosition 10-276                              |
| GetDesign 2-41, 2-41                             | GetFaceCenter 10-275                                  |
| GetDesigns 4-21, 6-30                            | GetFacelDs 10-277                                     |
| GetDesignValidationInfo 9-32                     | GetFieldFolderNames 18-73                             |
| GetDesignVariableNames 12-55                     | GetFieldPlotNames, Field Overlay module command 18-74 |
| GetDesignVariableUnits 12-56                     | GetFieldPlotQuantityName 18-74                        |
| GetDesignVariableValue 12-57                     | GetFieldType 17-17                                    |
| GetDesignVariationKey 12-58                      | GetFrequencies 24-28                                  |
| GetDisplacementCurrent 13-9                      | GetFrequencyCount 24-29                               |
| GetDisplayType 12-59                             | GetGeometryMode 9-33                                  |
| GetDistributedAnalysisMachines 4-23              | GetGeometryModelerMode 10-278                         |
|                                                  | GetImagDataValues 12-60                               |
|                                                  | GetIncludePortPostProcessing 17-18                    |
|                                                  | GetInstanceID 25-3                                    |

|                                                   |                                                |
|---------------------------------------------------|------------------------------------------------|
| GetInstanceName 25-4                              | GetObjectByNameByFaceID 10-281                 |
| GetLayerAndNetOfMeshBody 17-18                    | GetObjectsByMaterial 10-281                    |
| GetLibraryDirectory 4-25                          | GetObjectsIngroup 10-282                       |
| GetMatchedObjectName 10-283                       | GetObjPath 6-33                                |
| GetMeshBodiesOnLayerAndNet 17-21                  | GetOperationNames 14-3                         |
| GetMeshPlotNames 18-75                            | GetOutputVariableValue 11-6                    |
| GetModelBoundingBox 10-279                        | GetParentDesign 25-4                           |
| GetModelUnits 10-284                              | GetPath 2-44, 6-34                             |
| GetModule 9-34                                    | GetPerQuantityPrimarySweepValues 12-61         |
| GetMotionSetupNames G                             | GetPersonalLibDirectory 4-30                   |
| GetMultipactionBreakdown 17-23                    | GetPortCount 24-30                             |
| GetName 2-43, 4-29, 6-32, 9-35, 24-30             | GetPortNumber 24-31                            |
| GetNames 7-118, 7-127, 7-157, 7-190, 7-235, 7-261 | GetPostProcSettings 24-32                      |
| GetNetConnections 26-120                          | GetProcessID 4-31                              |
| GetNetworkDataSolution 17-24                      | GetProjectDirectory 4-32                       |
| GetNetworkDataSolutionDefinition 17-25            | GetProjectList 4-33                            |
| GetNetworkPostprocSetup 17-25                     | GetProjectMaterialNames 7-43                   |
| GetNPortData 7-121                                | GetProjects 4-28, 4-32                         |
| GetNumBoundaries 13-11                            | GetProperties 6-38, 7-129, 8-25, 26-107, 27-31 |
| GetNumBoundariesOfType 13-12                      | GetPropertyValue 6-39, 8-26, 12-62, 27-32      |
| GetNumExcitations 13-13                           | GetPropEvaluatedValue 6-34                     |
| GetNumExcitationsOfType 13-13                     | GetPropHost 25-5, 25-5                         |
| GetNumObjects 10-284                              | GetPropNames 6-35                              |
| GetNumPages 26-121                                | GetPropServerName 25-5                         |
| GetObjectIDByName 10-279                          | GetPropSIValue 6-36                            |
| GetObjectName 10-280                              | GetPropValue Project 6-37                      |
|                                                   | GetRealDataValues 12-64                        |



- GetRegistryInt 4-68
- GetRegistryString 4-69
- GetReportTraceNames 12-65
- GetSelections 9-38, 10-285
- GetSetupCount 15-19
- GetSetupCount, Analysis module  
command 15-19
- GetSetupNames (Optimetrics), Opti-  
metrics module command 16-  
13
- GetSetupNamesByType (Opti-  
metrics), Optimetrics module  
command 16-13
- GetSetups 15-20
  - Analysis module command 15-  
20
- GetSolutionContexts 12-65
- GetSolutionDataPerVariation 12-66
- GetSolutionType 9-38
- GetSolverOnDemandData 7-122
- GetSolverOnDemandModelList 7-  
122
- GetSweepCount 15-21
- GetSweepCount, Analysis module  
command 15-21
- GetSweepNames 12-69
- GetSweeps 15-21
  - Analysis module command 15-  
21
- GetSweepUnits 12-70
- GetSweepValues 12-71
- GetSymmetryMultiplier H
- GetSysLibDirectory 4-35
- GetTempDirectory 4-36
- GetTerminalExcitationType 17-26
- GetTool 4-72
- GetTopDesignList 2-45, 6-41
- GetTopEntryValue 18-78, 19-33
- GetTransientSolveTimes 17-27
- GetUserLibDirectory 4-37
- GetUserPosition 10-285
- GetValidISolutionList 17-27
- GetVariables 6-42, 8-28
- GetVariableValue 6-41, 8-29
- GetVariation 24-33
- GetVariationVariableValue 9-39
- GetVersion 4-37
- GetVertexIDsFromEdge 10-286
- GetVertexIDsFromFace 10-287
- GetVertexIDsFromObject 10-287
- GetVertexPosition 10-288
- GetVisibleLevels 27-32
- GridSetup 26-72, 27-33
- GroupPlotCurvesByGroupingStrategy  
12-72

## H

- HasSameData 24-34
- HealObject 10-198
- HidePolarPlot 18-79
- HideRadiatedPlotOverlay 18-80

hierarchy of variables in HFSS 1-72

## I

Icepak boundary condition commands

ImportIDX 4-87

If...Then... Else statement 1-8

Import (3D Modeler command) 10-205

ImportANF 4-73, 4-75

ImportAutoCAD 4-76

ImportAWRMicrowaveOffice 4-77

ImportDataset 6-43

ImportDXF 10-208

ImportEDB 4-79

ImportExport Tool 4-72

ImportExtracta 4-80

ImportGDSII 4-81, 4-82, 10-212

ImportIDF 4-83, 4-86

ImportIDX 4-87

ImportIntoReport 12-72

ImportIPC 4-90

ImportODB 4-91

ImportSetup, Optimetrics module command 16-14

ImportXFL 4-92

include files

scripts 1-10

indentation, IronPython 1-19

Information Method List 26-112

InitialMeshSettings 14-35

InputBox function 1-11

Insert3DComponent 10-99

InsertDesign 2-45, 6-45

InsertDesignWithWorkflow 2-46, 6-47

InsertNativeComponent 10-101

InsertSetup

Analysis module command 15-22

optimization command 16-35

parametric command 16-21

sensitivity command 16-55

statistical command 16-62

Intersect 10-216

IronPython 1-11, 1-13

indentation in 1-19

Is2D 9-39

Is3D 9-40

IsDataComplex 12-73

IsPerQuantityPrimarySweep 12-74

IsUsed 7-122, 7-129, 7-157, 7-191, 7-236, 7-261

## L

LaunchJobMonitor 4-38

LoadNamedExpressions 18-81, 19-34

LoadSolution 24-36

logical operators 1-7

looping through code

Do ... Loop 1-8

For ... Next 1-8

## M

### material commands

AddDefinitionFromBlock 10-245

AddDefinitionFromLibFile 10-250

AddMaterial 7-2, 7-19

CloneMaterial 7-16

ExportMaterial 7-44

RemoveMaterial 6-50, 7-45

### Materials Scripting Support 2-22

### Maxwell boundary condition commands

AssignCurrent 13-54

AssignFluxTangential\_1 13-23

AssignResistiveSheet 13-27

AssignThinLayer 13-30

AssignTouching 13-33

AssignVectorPotential 13-34

AssignVoltageAPhi\_ 13-59

EditCurrent 13-63

EditFluxTangential 13-37

EditResistiveSheet 13-40

EditThinLayer 13-43

EditTouching 13-45

EditVectorPotential 13-46

EditVoltageAPhi 13-68

### Maxwell definition manager commands

ComputeCoreLossCoefficients 7-17

### Maxwell Draw commands

Create3DComponent 10-6

### Maxwell parameter setup commands

AddReduceOp 21-2

EditReduceOp 21-18

### Mesh Operations module commands

AssignCylindricalGapOp 14-5

AssignLengthOp 14-12

AssignModelResolutionOp 14-15

AssignSkinDepthLayerSetting 14-17

AssignSkinDepthOp 14-18

AssignTrueSurfOp 14-19

DeleteOp 14-2

EditCylindricalGapOp 14-22

EditLengthOp 14-27

EditModelResolutionOp 14-30

EditSkinOp 14-31

EditTrueSurfOp 14-33

GetOperationNames 14-3

InitialMeshSettings 14-35

RenameOp 14-3

### Mesh Operations Module Script Commands 14-1

### mesh operations script commands 14-2

- method format, create schematic objects 26-1
- Microsoft
  - VBScript user's guide 1-11
- Mirror 10-130
- Model Manager Script
  - Commands 7-131
- Modeler Menu Commands 10-139
- ModifyFieldPlot 18-76, 18-82
- ModifyInceptionParameters 18-84
- ModifyLibraries 7-268
- module script commands
  - boundary/excitation 13-2
  - field overlay 18-1
  - general method 26-45
  - mesh operations 14-2
- Module Setup commands
  - GetSymmetryMultiplier H
  - SetSymmetryMultiplier H
- modules in HFSS scripting 1-72
- Motion Setup commands
  - AssignBand B
  - DeleteMotionSetup A
  - EditMotionSetup E
  - GetMotionSetupNames G
  - ReassignMoving B
- Move 10-132, 27-34
- MoveCSToEnd 10-217
- MoveEntityToGroup 10-218
- MoveFaces 10-219

- MovePlotCurveToGroup 12-75
- MovePlotCurveToNewGroup 12-76
- MsgBox function 1-11

## N

- Named Arguments 1-82
- NameNets 26-79
- NdExplorer
  - Script Commands 24-1
- Network Data Explorer Manager Commands 7-160
- NewProject 4-39

## O

- oAnsoftApp object 1-72
- Object Oriented Property Scripting 2-1
- oDesign object 1-72
- oDesktop object 1-72
- oEditor object 1-72
- OffsetFaces 10-134
- oModule object 1-72
- Open 24-37
- OpenAndConvertProject 4-40
- OpenExternalEditor 10-289
- OpenMultipleProjects 4-41
- OpenProject 4-41
- operators
  - arithmetic 1-6
  - categories in VBScript 1-5
  - comparison 1-7

- logical 1-7
  - precedence of 1-5
  - oProject object 1-72
  - Optimetrics module commands
    - DeleteSetups 16-8
    - DistributeAnalyzeSetup 16-8
    - ExportDXConfigFile 16-9
    - ExportOptimetricsProfile 16-10
    - ExportOptimetricsResults 16-11
    - ExportParametricResults 16-12
    - GetSetupNames 16-13
    - GetSetupNamesByType 16-14
    - ImportSetup 16-14
    - RenameSetup 16-16
    - SolveSetup 15-3, 16-7, 16-17, 16-18
  - Optimetrics Module Script Commands 16-1
  - optimization commands
    - EditSetup 16-29
    - InsertSetup 16-35
  - Optimization Script Commands 16-29
  - Other oEditor Commands 10-242
  - output variable commands
    - CreateOutputVariable 11-1
    - DeleteOutputVariable 11-3
    - DoesOutputVariableExist 11-3
    - EditOutputVariable 11-4
    - GetOutputVariableValue 11-6
  - Output Variable Script Commands 11-1
- P**
- PageBorders 26-81
  - PageSetup 10-290
  - Pan 26-81, 27-35
  - Parameter setup commands
    - DeleteReduceMatrix 21-14
    - DeleteReduceOp 21-15
    - RenameReduceMatrix 21-21
    - RenameReduceOp 21-21
  - Parameter Setup commands
    - AssignMatrix 21-7
    - DeleteAllParameters 21-13
    - EditForce 21-16
    - EditMatrix 21-18
    - EditTorque 21-19
    - ReassignParameter 21-20
    - RenameParameter 21-20
  - parametric commands
    - EditSetup 16-19
    - GenerateVariationData Parametric 16-21
    - InsertSetup 16-21
  - Parametric Script Commands 16-19
  - Paste 2-48, 4-43, 4-43, 6-48, 26-83, 27-35
  - Paste (Model Editor) 10-135
  - Paste (Project Object) 2-48, 4-43, 6-48

- PasteData 26-84
- PasteDesign (Schematic Editor) 26-84
- PasteReports 12-77
- PasteSetup
  - Analysis module command 15-42
  - Optimetrics module command 16-16
- PasteSweep 15-43
- PasteTraces 12-78
- PauseRecording 4-44
- PauseScript 4-45
- Print 4-45
- Project object commands
  - AddDataset 6-4
  - AddMaterial 7-2, 7-19
  - ChangeProperty 27-6
  - CloneMaterial 7-16
  - Close 2-35, 6-16
  - CopyDesign 2-36, 6-16
  - CutDesign 2-36, 6-17
  - DeleteDataset 6-18
  - DeleteDesign 2-37, 6-19
  - EditDataset 6-20, 6-23, 6-43
  - ExportMaterial 7-44
  - GetActiveDesign 2-39, 6-24
  - GetArrayVariables 6-24, 8-24
  - GetChildObject Design 6-26
  - GetDependentFiles 2-41, 6-29
  - GetDesign 2-41
  - GetName 2-43, 6-32
  - GetObjPath 6-33
  - GetPath 2-44, 6-34
  - GetProjectMaterialNames 7-43
  - GetProperties 6-38, 8-25, 27-31
  - GetPropertyValue 6-39, 8-26, 12-62, 27-32
  - GetPropEvaluatedValue 6-34
  - GetPropSIValue 6-36
  - GetPropvalue Project 6-37
  - GetTopDesignList 2-45, 6-41
  - GetTopEntryValue 18-78, 19-33
  - GetVariables 6-42, 8-28
  - GetVariableValue 6-41, 8-29
- InsertDesign 2-45, 6-45
- LoadSolution 24-36
- Paste 2-48, 2-48, 4-43, 4-43, 4-43, 6-48, 6-48
- Redo 2-49, 6-49
- RemoveAllUnusedDefinitions 6-49
- RemoveMaterial 6-50, 7-45
- RemoveUnusedDefinitions 6-51, 7-46
- Rename 2-49, 6-52
- RestoreProjectArchive 2-50, 4-48
- Save 2-51, 6-55
- SaveAs 2-52, 6-55
- SaveAsStandAloneProject 2-54, 6-58
- SaveProjectArchive 2-55, 6-58

- SetActiveDesign 2-57, 6-60
- SetPropertyValue 6-62, 8-30
- SetPropValue Project 6-61
- SetVariableValue 6-63, 8-31
- SimulateAll 2-33, 2-57, 6-7, 6-64
- Undo 2-58, 6-65
- UpdateDefinitions 2-59, 6-66
- Project Object Script Commands 2-30, 6-1
- ProjectSheet 10-222
- property commands
  - ChangeProperty 27-6
  - GetArrayVariables 6-24, 8-24
  - GetProjectMaterialNames 7-43
  - GetProperties 6-38, 8-25, 27-31
  - GetPropertyValue 6-39, 8-26, 12-62, 27-32
  - GetPropEvaluatedValue 6-34
  - GetPropSIValue 6-36
  - GetTopEntryValue 18-78, 19-33
  - GetVariables 6-42, 8-28
  - GetVariableValue 6-41, 8-29
  - SetPropertyValue 6-62, 8-30
  - SetVariableValue 6-63, 8-31
- Property Method List 26-105
- Property Script Commands 8-1
  - Conventions uSed in thie Chapter 8-12
  - Object Script Property Function Summary 8-3
- PurgeHistory 10-223
- PushExcitations 26-87
  - Layout Editor 26-87
- Q**
- QuitApplication 4-46
- R**
- ReassignBoundaries 13-3, 13-14
- ReassignBoundary 13-14
- ReassignFieldPlot 18-86
- ReassignMoving B
- ReassignParameter 21-20
- Redo 27-36
  - design-level command 9-40
  - project-level command 2-49, 6-49
- references, for VBScript 1-11
- RefreshJobMonitor 4-47
- ReleaseData 12-78
- Remove 7-123, 7-158, 7-191, 7-236, 7-262
- RemoveAllUnusedDefinitions 6-49
- RemoveAssignmentFromBoundary 13-15
- RemoveLevels 27-37
- RemoveMaterial 6-50, 7-45
- RemovePort 27-37
- RemoveScript 7-267
- RemoveSolverOnDemandModel 7-124

- RemoveUnused 7-125, 7-130, 7-159, 7-192, 7-238, 7-263
- RemoveUnusedDefinitions 6-51, 7-46
- Rename 2-49, 6-52, 24-38
- RenameBoundary 13-16
- RenameDesignInstance 9-41
- RenameFieldPlot 18-89
- RenameOp 14-3
- RenameParameter 21-20
- RenamePart 10-292
- RenamePlotFolder 18-90
- RenameReduceMatrix 21-21
- RenameReduceOp 21-21
- RenameReport 12-79
- RenameSetup 15-43
  - Analysis module command 15-43
  - Optimetrics module command 16-16
- RenameTraces 12-80
- Renormalize 24-39
- Reorder 24-41, 24-42
- Repeating a Statement Until a Condition Becomes True 1-9
- Repeating Statements While a Condition is True 1-9
- Reporter editor commands
  - AddCartesianLimitLine 12-4
  - AddCartesianXMarker 12-6, 12-7
  - AddDeltaMarker 12-8
  - AddMarker 12-9
  - AddMarkerToPlot 18-2
  - AddNote 12-10
  - AddTraceCharacteristics 12-12
  - AddTraces 12-13
  - ChangeProperty 12-16
  - ClearAllMarkers 12-20
  - ClearAllTraceCharacteristics 12-20
  - CopyReportDefinition 12-23
  - CopyReportsData 12-22
  - CopyTraceDefinitions 12-23
  - CopyTracesData 12-21
  - CreateReport 12-25
  - CreateReportFromFile 12-24
  - CreateReportFromTemplate 12-35
  - CreateReportOfAllQuantities 12-36
  - DeleteAllReports 12-38
  - DeleteReport 12-39
  - DeleteTraces 12-39
  - ExportImageToFile 12-41
  - ExportPlotImageToFile 12-42, 18-68, 18-69
  - ExportReport 12-43
  - ExportToFile 12-46, 12-72
  - FFTONReport 12-48, 17-12
  - GetAllCategories 12-49
  - GetAllQuantities 12-50
  - GetAllReportNames 12-49



- GetAvailableDisplayTypes 12-51
  - GetAvailableReportTypes 12-52
  - GetAvailableSolutionss 12-53
  - GetDisplayType 12-59
  - GetReportTraceNames 12-65
  - GetSolutionContexts 12-65, 12-66
  - GroupPlotCurvesByGroupingStrategy 12-72
  - MovePlotCurvestoGroup 12-75
  - MovePlotCurvestoNewGroup 12-76
  - PasteReports 12-77
  - PasteTraces 12-78
  - RenameReport 12-79
  - RenameTraces 12-80
  - ResetPlotSettings 12-81
  - SavePlotSettingsAsDefault 12-82
  - UnGroupPlotCurvesInGroup 12-89
  - UpdateAllReports 12-82
  - UpdateReports 12-83
  - UpdateTraces 12-84
  - UpdateTracesContextAndSweeps 12-86
  - Reporter Editor Script Commands 12-1
  - ReprioritizeBoundaries 13-17
  - Reset 24-43
  - ResetAllToTimeZero 15-44
  - ResetLogging 4-48
  - ResetPlotSettings 12-81
  - ResetSetupToTimeZero 15-44
  - RestoreWindow 4-49
  - RestToTimeZero 9-42
  - ResumeRecording 4-50
  - RevertAllToInitial 15-45
  - RevertSetupToInitial 15-46
  - RevertSetupToInitialTemperature 15-47
  - Rotate 10-136, 26-90, 27-38
  - Running Instance Manager Script Commands 5-1
  - RunProgram 4-51
  - RunScript 4-52
  - RunScriptWithArguments 4-54
  - RunToolkit 6-53, 9-69
- S**
- sample scripts
    - variable helix 29-13
  - Save 2-51, 6-55, 27-39
  - SaveAs 2-52, 6-55
  - SaveAsStandAloneProject 2-54, 6-58
  - SaveFieldsPlots 18-90
  - SaveNamedExpressions 18-94, 19-35
  - SavePlotSettingsAsDefault 12-82
  - Scale 10-137
  - Schematic Scripting 26-1
    - Create Method List 26-5

- Scope and Lifetime of Variables 1-4
- Script and Library Scripts 7-264
  - script commands
    - boundary/excitation 13-2
    - field overlay 18-1
    - general method script commands 26-45
    - mesh operations 14-2
  - Script Commands for Creating and Modifying Excitations in Maxwell 13-48
  - Script Commands for Creating and Modifying Boundaries in Maxwell 13-18
  - Script Commands for Creating and Modifying Mesh Operations 14-4
  - scripts
    - CPython 1-69
    - IronPython 1-11, 1-13
    - overview 1-1
    - pausing 1-77
    - recording 1-78
    - running 1-76
    - stopping 1-77
    - VBScript 1-2
- Section 10-224
- Select Case statement 1-8
- SelectAll 26-92, 27-39
- SelectPage 26-93
- SelectScheduler 4-55
- SendToBack 26-97, 27-40
- sensitivity commands
  - EditSetup 16-47
  - InsertSetup 16-55
- Sensitivity Script Commands 16-47
- SeparateBody 10-225
- SetActiveDefinitionEditor 2-56, 6-59
- SetActiveDesign 2-57, 6-60
- SetActiveEditor 9-42
- SetActiveLevel 27-40
- SetActiveProject 4-56
- SetActiveProjectByPath 4-57
- SetAllPortImpedance 24-44
- SetConductivityThreshold 9-43
- SetCoreLoss 13-70
- SetDesignSettings 9-44
- SetEddyEffect 13-70
- SetFastTransformationForLayoutComponent 9-57, 9-58
- SetFieldPlotSettings 18-91, 18-95
- SetInitialLevels 27-41
- SetLibraryDirectory 4-60
- SetMagnetizationCompute 13-71
- SetMinimumTimeStep 13-73
- SetModelUnits 10-226
- SetPageData 26-94
- SetPermeabilityComputation 13-75
- SetPlotFolderSettings 18-98

- SetPlotsViewSolutionContext 18-103
- SetPortDeembedDistance 24-45
- SetPortImpedance 24-46
- SetPostProcSettings 24-47
- SetProjectDirectory 4-61
- SetPropertyValue 6-62, 8-30, 27-42
- SetPropValue Project 6-61
- SetShowLayoutForLayoutComponent 9-65, 9-65
- SetSymmetryMultiplier H
- SetTempDirectory 4-62
- Setting Numerical Values 1-84
- SetupYConnection 13-73
- SetVariableValue 6-63, 8-31
- SetVisibleLevels 27-42
- SetWCS 10-228
- SGetAppDesktop 3-1
- ShowWindow 10-229
- ShowVariableBlock 26-98
- simple and composite names 1-3
- SimulateAll 2-33, 2-57, 6-7, 6-64
- Sleep 4-64
- Smooth 24-48
- Solutions module commands
  - ConstructVariationString 9-10
  - DeleteFieldVariation 9-12
  - DeleteFullVariation 9-9, 9-13
  - DeleteLinkedDataVariation 9-14
  - DeleteSolutionVariation 17-7
  - ExportTransientData 17-11
  - GetAdaptiveSettings 17-13
  - GetAllSourceMagnitudes 17-14
  - GetAllSourceModes 17-14
  - GetAllSourcePhases 17-15
  - GetAllSources 17-16
  - GetAntennaParameters 17-16
  - GetFieldType 17-17
  - GetIncludePortPostProcessing 17-18
  - GetLayerAndNetOfMeshBody 17-18
  - GetMeshBodiesOnLayerAndNet 17-21
  - GetMultipactionBreakdown 17-23
  - GetNetworkDataSolution 17-24
  - GetNetworkDataSolutionDefinition 17-25
  - GetNetworkPostprocSetup 17-25
  - GetTerminalExcitationType 17-26
  - GetTransientSolveTimes 17-27
  - GetValidISolutionList 17-27
  - TDROnReport 17-28
- Solutions Module commands
  - ExportNetworkData 17-8
- Solutions Module Script Commands 17-1
- Solve 9-67
- SolveAllSetup, Optimetrics module command 16-18

SolveSetup, Optimetrix module  
command 16-17

SortComponents 26-98

Split 10-229

statistical commands

    EditSetup 16-59

    InsertSetup 16-62

Statistical Script Commands 16-59

StopSimLink 9-68

Stretch 24-49

Sub Procedures 1-10

SubmitJob 4-64

Subtract 10-232

SweepAlongPath 10-107

SweepAlongVector 10-109

SweepAroundAxis 10-111

SweepFacesAlongNormal 10-113,  
10-113, 10-234, 10-234

Sweep-  
    FacesAlongNormalWithAt-  
    tributes 10-115

Symbol Editor Scripts 27-1

Symbol Manager Script  
    Commands 7-165

## T

TDROnReport 17-28

Terminate 24-51

ThickenSheet 10-235

ToggleLevel 27-43

## U

UncoverFaces 10-237

Undo 27-44

    project-level command 2-58, 6-65

Ungroup 10-240

UnGroupPlotCurvesInGroup 12-89

Unite 10-239

UpdateAllFieldsPlots 18-104

UpdateAllReports 12-82

UpdateComponentDefinition 10-118

UpdateDefinitions, project-level  
    command 2-59, 6-66

UpdateDynamicLink 7-126

UpdateQuantityFieldsPlots 18-105

UpdateReports 12-83

UpdateTraces 12-84

UpdateTracesContextAndSweeps 12-86

User defined documents module com-  
    mands

    GetDocumentNames 22-8

User defined solution module commands

    CreateUserDefinedSolution 23-1

    DeleteUserDefinedSolutions 23-3

    EditUserDefinedSolution 23-4

User Defined Solutions Commands 23-1

Using a Do Loop 1-9

## V

ValidateDesign 9-70

Variable Naming Conventions 1-4

variables

- array 1-4

- assigning information 1-3

- declaring 1-3

- hierarchy in HFSS 1-72

- used in HFSS scripts 1-72

VBScript 1-2

- Microsoft user's guide 1-11

- operators 1-5

- references 1-11

VBScript Procedures 1-9

## **W**

Wire (Schematic Editor) 26-99

WrapSheet 10-241

## **Z**

ZoomArea 26-100, 27-45

ZoomIn 26-101, 27-46

ZoomOut 26-102, 27-46

ZoomPrevious 26-103, 27-47

ZoomToFit 26-104, 27-47