Objects

Aec2dSection AEC 2D section object.

AecAnchor Base class for all AEC anchor objects.

<u>AecAnchorEntToCurve</u> Binds one AEC object to another AEC or AutoCAD object, such as a roof, wall, line, or arc.

<u>AecAnchorEntToGridAssembly</u> Attaches entity objects, such as doors and windows, to grid assemblies, such as curtain walls and window assemblies.

<u>AecAnchorEntToLayoutCell</u> Attaches objects, such as light fixtures, to cells on a 2D layout grid.

<u>AecAnchorEntToLayoutNode</u> Attaches objects, such as columns, to nodes on a layout grid.

<u>AecAnchorEntToLayoutVolume</u> Attaches objects, such as furniture, to volumes on a 3D layout grid.

<u>AecAnchorLeadEntToNode</u> Connects an object to a layout node with a leader between the two objects.

AecAnchorToRef Base class for referencing objects from an anchor.

<u>AecBaseApplication</u> Application object extending AcadApplication to provide support for AEC Applications.

<u>AecBaseDatabase</u> AEC database object extending AcadDatabase to provide support for general AEC objects.

<u>AecBaseDatabasePreferences</u> A derivative of AecDatabasePreferences, providing base desktop database preferences support.

<u>AecBaseDocument</u> An AEC drawing object, extending the AcadDocument to provide support for general AEC objects.

AecBaseDocuments The collection of all AEC drawings open in the

current session.

AecBasePreferences AEC AutoCAD preferences object.

AecBlockRef An instance of an AEC block in a drawing.

AecCamera AEC camera object.

AecCellLayoutTool Base Class for the layout grid objects.

AecClipVol Clip Volume entity defining an extruded region for clipping.

<u>AecClipVolRes</u> Clip Volume Result which captures the resulting graphics for a section or elevation for insertion into the drawing.

AecDatabase AEC Database object extending the Acad Database object.

<u>AecDatabasePreferences</u> Extends the AutoCAD Database Preferences object to include AEC properties and methods.

AecDictionary AEC Dictionary containing a collection of AEC objects.

AecDictRecord An AEC Dictionary item.

<u>AecEditInPlaceProfile</u> In-place edit profile object.

AecEntity Base class for all AEC entities.

<u>AecEntRef</u> A reference to any entity in the drawing, similar to a block reference.

<u>AecGeo</u> Provides location, rotation and anchoring capabilities for AEC entities.

<u>AecGridAssembly</u> Base class for grid assemblies.

<u>AecLayerKey</u> Layer key definition, which specifies the layer properties generated from a particular key.

AecLayerKeys The collection of layer key definitions in a layer key style.

AecLayerKeyStyle A collection of layer keys.

<u>AecLayerKeyStyles</u> The collection of layer key styles in the specified drawing.

AecLayerOverrideSetting A layer key style override setting.

<u>AecLayerOverrideSettings</u> The collection of override settings in a layer key style.

<u>AecLayoutCurve</u> A single dimensional grid with nodes along a single curve.

AecLayoutCurveNode A node in a layout curve.

<u>AecLayoutCurveNodes</u> The collection of nodes in a layout curve.

<u>AecLayoutGrid2D</u> Layout grid entity used to layout any AEC object within a 2D grid.

AecLayoutGrid2DNode A node in a 2D layout grid.

AecLayoutGrid2DNodes The collection of nodes in a 2D layout grid.

AecLayoutGrid2DXNode A node along the X axis of a 2D layout grid.

<u>AecLayoutGrid2DXNodes</u> Collection of nodes along the X axis of a 2D layout grid.

<u>AecLayoutGrid2DYNode</u> A node along the Y axis of a 2D layout grid.

<u>AecLayoutGrid2DYNodes</u> Collection of nodes along the Y axis of a 2D layout grid.

AecLayoutGrid3D A layout volume for arranging objects in a 3D grid.

AecLayoutGrid3DNode A node of a 3D Layout Grid.

AecLayoutGrid3DNodes The collection of nodes in a 3D layout grid.

AecLayoutGrid3DXNode A node along the X axis of a layout volume.

<u>AecLayoutGrid3DXNodes</u> Collection of nodes along the X axis of a layout volume.

AecLayoutGrid3DYNode A node along the Y axis of a layout volume.

<u>AecLayoutGrid3DYNodes</u> Collection of nodes along the Y axis of a layout volume.

<u>AecLayoutGrid3DZNode</u> A node along the Z axis of a layout volume.

<u>AecLayoutGrid3DZNodes</u> Collection of nodes along the Z axis of a layout volume.

<u>AecLayoutNode</u> Provides common properties shared by all types of layout nodes.

AecLayoutNodes Collection of layout nodes.

<u>AecLayoutTool</u> Provides common functionality for layout curve and layout grid objects.

<u>AecMaskBlockRef</u> An instance of a mask block, a two-dimensional block that covers part of an AEC object in a plan (2D) view.

AecMaskBlockStyle Defines the properties of a mask block object.

<u>AecMaskBlockStyles</u> The collection of mask block styles in the specified drawing.

<u>AecMassElement</u> An object that has behaviors based on its shape.

<u>AecMassElementStyle</u> Defines the properties of a mass element object.

<u>AecMassElementStyles</u> The collection of mass element styles in the specified drawing.

<u>AecMassGroup</u> A group of mass elements combined to form a complex shape.

AecMassGroups The collection of mass groups in the specified drawing.

<u>AecMVBlockRef</u> An instance of a block that can have different representations in different view directions.

AecMVBlockStyle Defines the properties of a multi-view block object.

<u>AecMVBlockStyles</u> The collection of multi-view block styles in the specified drawing.

AecObject Provides common functionality for AEC database objects.

AecPolygon An Aec Polygon object.

AecPolygonStyle Defines the properties of an Aec Polygon object.

<u>AecPolygonStyles</u> The collection of Aec Polygon styles in the specified drawing.

<u>AecProfile</u> Defines the properties of a profile style.

<u>AecProfileStyle</u> A custom shape defined by two-dimensional closed polylines.

<u>AecProfileStyles</u> The collection of profile styles in the specified drawing.

<u>AecRing</u> A closed loop of polyline segments, often used as a group inside an AecProfile object.

AecRings The collection of rings in an AecProfile object.

<u>AecSlice</u> Slice of a massing group or any AEC object, typically used to generate floor plates and space boundaries.

<u>AecViewBlock</u> View block used for display representations by a multiview block.

<u>AecViewBlocks</u> A collection of view blocks used for display representations by a multi-view block.

<u>AecVolumeLayoutTool</u> Provides standard functionality for all 3D layout tools.

AEC Base Object Model

Click on an object to view the description.

↑ ↓ 2D Section Object

AEC 2D section object.

VBA object name: Aec2dSection

Create using: AddCustomObject("Aec2dSection")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group. Item

Methods

<u>ArrayPolar</u> Properties

Application

<u>ArrayRectangular</u>

ClipVol

AttachAnchor

Description

Copy

Document

<u>Delete</u>

Handle

GetAnchor

HasExtensionDictionary

<u>GetBoundingBox</u>

Hyperlinks

GetExtensionDictionary

Layer

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u> <u>Events</u>

IntersectWith Lineweight Modified

<u>Mirror</u> <u>Location</u>

Mirror3D Normal

Move ObjectID

ReleaseAnchor ObjectName

<u>Rotate</u> <u>OwnerID</u>

Rotate3D PlotStyleName

<u>ScaleEntity</u> <u>Rotation</u>

<u>SetXData</u> <u>TrueColor</u>

<u>TransformBy</u> <u>Visible</u>

<u>Update</u>

↑ ♦ Anchor Object

Base class for all AEC anchor objects.

VBA object name: AecAnchor

Create using: New AecAnchor

Access via: AecGeo.GetAnchor

You can use AecAnchor as a generic reference to anchor objects.

Application

Methods
Description

AttachEntity
Document
Delete
Handle
GetExtensionDictionary

Methods

Description

Events
Modified

<u>HasExtensionDictionary</u>

<u>GetXData</u>

<u>SetXData</u>

<u>ObjectID</u>

<u>ObjectName</u>

<u>OwnerID</u>

↑ ↓ Anchor Entity to

Curve Object

Binds one AEC object to another AEC or AutoCAD object, such as a roof, wall, line, or arc.

VBA object name: AecAnchorEntToCurve

Create using: New AecAnchorEntToCurve

Access via: AecGeo.GetAnchor

Properties

Application

Description

Document

FlipX

FlinY

<u>ı ııhı</u>

<u>FlipZ</u>

Handle

HasExtensionDictionary

Events

Methods ObjectID

<u>AttachEntity</u> <u>ObjectName</u>

<u>Delete</u> <u>OwnerID</u>

GetExtensionDictionary Reference Modified

<u>GetXData</u> <u>XDistance</u>

<u>SetXData</u> <u>XPositionFrom</u>

XPositionTo

XRotation

YDistance

YPositionFrom

YPositionTo

ZDistance

ZPositionFrom

ZPositionTo

ZRotation

↑ ↓ AecAnchorEntToEnt

Object

Binds one AEC object to another.

VBA object name: AecAnchorEntToEnt

Create using: New AecAnchorEntToEnt

Access via: AecGeo.GetAnchor

Properties

Application

Methods Description

<u>AttachEntity</u> <u>Document</u>

<u>Delete</u> <u>Handle</u> <u>Events</u>

GetExtensionDictionary HasExtensionDictionary Modified

GetXData ObjectID

<u>SetXData</u> <u>ObjectName</u>

<u>OwnerID</u>

<u>Reference</u>

↑ Anchor Entity to Grid

Assembly Object

Attaches entity objects, such as doors and windows, to grid assemblies, such as curtain walls and window assemblies.

VBA object name: AecAnchorEntToGridAssembly

Create using: N/A

Access via: AecGeo.GetAnchor

Properties

AdjustSizing

AllowVariation

Application

BottomOffset

Cell

Description

Document

Methods

<u>FlipX</u>

AttachEntity FlipY

<u>Delete</u>

<u>FlipZ</u>

GetExtensionDictionary Handle

Modified

Events

<u>GetXData</u> <u>HasExtensionDictionary</u>

<u>SetXData</u> <u>LeftOffset</u>

ObjectID

ObjectName

OwnerID

Reference

RightOffset

TopOffset

YAlignment

YOffset

↑ ↓ Anchor Entity to

Layout Cell Object

Attaches objects to cells on a 2D layout grid.

VBA object name: AecAnchorEntToLayoutCell

Create using: New AecAnchorEntToLayoutCell

Access via: AecGeo.GetAnchor

Properties

Application

ApplyResize

CenterOnNode

Description

Document

<u>FlipX</u>

<u>FlipY</u>

Methods FlipZ

<u>AttachEntity</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u> Events

Modified

GetExtensionDictionary Node

GetXData ObjectID

<u>SetXData</u> <u>ObjectName</u>

Offset

OwnerID

Reference

ResizeOffset

<u>UseNodeCS</u>

XRotation

YRotation

ZRotation

↑ ↓ Anchor Entity to

Layout Node Object

Attaches objects, such as columns, to nodes on a layout grid.

VBA object name: AecAnchorEntToLayoutNode

Create using: New AecAnchorEntToLayoutNode

Access via: AecGeo.GetAnchor

Properties

Application

CenterOnNode

Description

Document

<u>FlipX</u>

<u>FlipY</u>

Methods FlipZ

<u>AttachEntity</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u> <u>Events</u>

GetExtensionDictionary Node Modified

GetXData ObjectID

<u>SetXData</u> <u>ObjectName</u>

Offset

OwnerID

Reference

<u>UseNodeCS</u>

XRotation

YRotation

ZRotation

↑ ↓ Anchor Entity to

Layout Volume Object

Attaches objects to volumes on a 3D layout grid.

VBA object name: AecAnchorEntToLayoutVolume

Create using: New AecAnchorEntToLayoutVolume

Access via: AecGeo.GetAnchor

Properties

Application

ApplyResize

CenterOnNode

Description

Document

<u>FlipX</u>

<u>FlipY</u>

Methods FlipZ

<u>AttachEntity</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u> Events

Modified

GetExtensionDictionary Node

GetXData ObjectID

<u>SetXData</u> <u>ObjectName</u>

Offset

OwnerID

Reference

ResizeOffset

<u>UseNodeCS</u>

XRotation

YRotation

ZRotation



AecAnchorExtendedTagToEnt Object

Attaches a tag to an object to display schedule data.

VBA object name: AecAnchorExtendedTagToEnt

Create using: New AecAnchorExtendedTagToEnt

Access via: AecGeo.GetAnchor

This object is available if the drawing contains any MVBlock styles.

	Properties	
	<u>Application</u>	
	<u>Description</u>	
Methods	Document	
<u>AttachEntity</u>		
Delete	<u>ForceHorizontal</u>	

<u>Handle</u>

<u>GetExtensionDictionary</u>

HasExtensionDictionary

<u>GetXData</u>

<u>ObjectID</u>

<u>SetXData</u>

ObjectName

<u>OwnerID</u>

Reference

↑ ↓ Anchor Leader Entity

to Node Object

Connects an object to a layout node with a leader between the two objects.

VBA object name: AecAnchorLeadEntToNode

Create using: New AecAnchorLeadEntToNode

Access via: AecGeo.GetAnchor

Properties

AngleFromNode

Application

CenterOnNode

Description

Document

<u> POGUITICITE</u>

<u>FlipX</u>

FlipY

FlipZ

Methods

Handle

<u>AttachEntity</u> <u>HasExtensionDictionary</u>

<u>Delete</u> <u>LeaderExtension1</u>

Events

Modified

GetExtensionDictionary LeaderExtension2

<u>GetXData</u> <u>Node</u>

<u>SetXData</u> <u>ObjectID</u>

ObjectName

Offset

OwnerID

Reference

<u>UseNodeCS</u>

XRotation

YRotation

ZRotation

↑ ♦ Anchor to Reference

Object

Base class for referencing objects from an anchor.

VBA object name: AecAnchorToRef

Create using: New AecAnchorToRef

Access via: AecGeo.GetAnchor

Properties

Application

Methods Description

<u>AttachEntity</u> <u>Document</u>

<u>Delete</u> <u>Handle</u> <u>Events</u>

GetExtensionDictionary HasExtensionDictionary Modified

GetXData ObjectID

<u>SetXData</u> <u>ObjectName</u>

<u>OwnerID</u>

<u>Reference</u>

↑ ↓ Aec Base Application

Object

Application object extending AcadApplication to provide support for AEC Applications.

VBA object name: AecBaseApplication

For VB:

GetObject("Acad.Application") or

CreateObject("Acad.Application") then

Create using: GetInterfaceObject("AecBase.Application")

For VBA:

not applicable. The application is always available.

Access via: Application Property

The properties associated with the AecBaseApplication object reflect the properties of the main application window. The methods control the loading or listing of the currently loaded external applications and interface objects.

The active document (Autodesk Architectural Desktop drawing) can be accessed using the <u>ActiveDocument</u> property. See the <u>AcadApplication</u> object in the AutoCAD ActiveX and VBA Reference for information regarding the Methods, Properties and Events provided through this object.

Pro	perties
110	pei ues

Methods ActiveDocument

Eval

Application

GetAcadState

Caption

GetInterfaceObject

Documents

Init

FullName

ListArx

Height

LoadArx

LocaleId

LoadDVB

MenuBar

Quit

MenuGroups

RunMacro

Name

UnloadArx

Path

UnloadDVB

Preferences

Update

StatusId

ZoomAll

VBE

ZoomCenter

Version

ZoomExtents

Visible

ZoomPickWindow

ZoomPrevious Width

ZoomScaled WindowLeft

ZoomWindow WindowState

WindowTop

↑ ↓ Aec Base Database

Object

AEC database object extending AcadDatabase to provide support for general AEC objects.

VBA object name: AecBaseDatabase

Create using: N/A

Access via: ModelSpace.Item

Note:All properties and methods of the AecBaseDatabase object are available through the <u>AecBaseDocument</u> object.

Properties

Blocks

Dictionaries

DimStyles

DisplayConfigurations

DisplayRepresentations

DisplaySets

ElevationModelSpace

ElevationPaperSpace

Groups

LayerKeyStyles

Layers

Methods Layouts

CopyObjects Limits

HandleToObject Linetypes

<u>Init</u> <u>MaskBlockStyles</u>

ObjectIdToObject MassElementStyles

MassGroups

ModelSpace

MVBlockStyles

<u>PaperSpace</u>

PlotConfigurations

PolygonStyles

Preferences

ProfileStyles

RegisteredApplications

<u>TextStyles</u>

<u>UserCoordinateSystems</u>

Viewports

<u>Views</u>

↑ ↓ Aec Base Database

Preferences Object

A derivative of AecDatabasePreferences, providing base desktop database preferences support.

VBA object name: AecBaseDatabasePreferences

Create using: N/A

Access via: AecBaseDatabase.Preferences

AecBaseDocument.Preferences

Properties

<u>AllowLongSymbolNames</u>

AlwaysImportLayerStandard

AngularAzimuth

<u>AngularDisplayFormat</u>

AngularPrecision

Application

AreaDisplayUnit

AreaPrecision

AreaSuffix

BasePoint

BasePointNE

BlockBasedLayerOffBehavior

ContourLinesPerSurface

CoordinatePrecision

CreateDimscaleOverride

Database

DatabaseScale

DisplaySilhouette

ElevationPrecision

FacetDeviation

FacetMaxiumum

LayerFile

LayerStandard

LinearDisplayFormat

Methods

LinearPrecision

ConvertToCurrentAreaDisplay

LinearUnit

ConvertToCurrentVolumeDisplay

<u>Init</u>

Lineweight

SaveAsDefault

<u>MaxActiveViewports</u>

LineWeightDisplay

MeasurementUnit

NorthRotation

ObjectSortByPlotting

ObjectSortByPSOutput

ObjectSortByRedraws

ObjectSortByRegens

ObjectSortBySelection

ObjectSortBySnap

OLELaunch

ProjectName

RenderSmoothness

ScaleOnInsert

SegmentPerPolyline

SolidFill

TextFrameDisplay

TextHeight

VerticalScale

<u>v Ci tiodiCodio</u>

VolumeDisplayUnit

VolumePrecision

VolumeSuffix

XRefEdit

XRefLayerVisibility

<u>XrefOverlaysUseOwnDisplayConfig</u>

↑ ↓ Aec Base Document

Object

An AEC drawing object, extending the AcadDocument to provide support for general AEC objects.

Note: This object is actually a component of the AecXUIBase object model, which represents user interface objects.

VBA object name: AecBaseDocument

Documents.Add

Create using: Documents.Open

Document.New

Access via: Documents.Item

Application.ActiveDocument

The <u>Preferences</u> property provides access to drawing settings.

The active document can be accessed through the ActiveDocument property of the AecBaseApplication object, after initializing

AecBaseApplication. For example:

Dim doc As AecBaseDocument Dim app As New AecBaseApplication

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

Properties

Active

<u>ActiveDimStyle</u>

ActiveLayer

ActiveLayout

<u>ActiveLinetype</u>

ActivePViewport

ActiveSelectionSet

ActiveSpace

<u>ActiveTextStyle</u>

ActiveUCS

ActiveViewport

Application

Blocks

Database

Dictionaries

DimStyles

DisplayConfigurations

DisplayRepresentations

Methods

<u>Activate</u> <u>DisplaySets</u>

<u>AuditInfo</u> <u>ElevationModelSpace</u>

<u>Close</u> <u>ElevationPaperSpace</u>

<u>CopyObjects</u> <u>FullName</u>

EndUndoMark Groups

<u>Export</u> <u>Height</u>

<u>GetVariable</u> <u>HWND</u>

HandleToObject LayerKeyStyles

<u>Import</u> <u>Layers</u>

Init Layouts

LoadShapeFile Limits

<u>New</u> <u>Linetypes</u>

ObjectIdToObject MaskBlockStyles

<u>Open</u> <u>MassElementStyles</u>

<u>PurgeAll</u> <u>MassGroups</u>

Regen ModelSpace

Save MSpace

<u>SaveAs</u> <u>MVBlockStyles</u>

SendCommand Name

<u>SetVariable</u> <u>ObjectSnapMode</u>

StartUndoMark PaperSpace

Wblock Path

PickfirstSelectionSet

<u>Plot</u>

PlotConfigurations

PolygonStyles

Preferences

ProfileStyles

ReadOnly

RegisteredApplications

Saved

SelectionSets

<u>TextStyles</u>

<u>UserCoordinateSystems</u>

Utility

<u>Viewports</u>

Views

Width

<u>WindowState</u>

WindowTitle

↑ ↓ Aec Base Documents

Collection

The collection of all AEC drawings open in the current session.

Note: This object is actually a component of the AecXUIBase object model, which represents user interface objects.

VBA object name: AecBaseDocuments

Create using: N/A

Access via: AecBaseApplication.Documents

Methods

Add

Close Properties

Application

<u>Init</u>

Count

Item

<u>Open</u>

↑ ↓ Aec Base Preferences

Object

AEC AutoCAD preferences object.

VBA object name: AecBasePreferences

Create using: N/A

Access via: AecBaseApplication.Preferences

The AecBasePreferences object holds all the options stored external to the drawing, such as User Preferences. The options are stored in separate objects, grouped by category. Use the properties of the AecBasePreferences object to access each category.

Options that reside in the drawing can be accessed through the <u>AecBaseDatabasePreferences</u> object.

Currently, AecBasePreferences does not provide access to AEC-specific options; it is equivalent to AcadPreferences. Refer to <u>AcadPreferences</u> in the AutoCAD ActiveX and VBA Reference for information about this

object's methods and properties.

Properties

Application

Display

Drafting

Files

Methods OpenSave

Init

Output

Profiles

Selection

System

User

↑ ↓ AEC Block Object

An instance of an AEC block in a drawing.

VBA object name: AecBlockRef

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

AecBlockRef is the base object for the AecMVBlockRef and AecMaskBlockRef objects.

	Properties
Methods	Application
<u>ArrayPolar</u>	Color

<u>ArrayRectangular</u> <u>Description</u>

<u>AttachAnchor</u> <u>Document</u>

<u>Copy</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u>

GetAnchor Hyperlinks

<u>GetBoundingBox</u> <u>Layer</u>

GetExtensionDictionary Linetype

<u>GetXData</u> <u>LinetypeScale</u>

<u>Highlight</u> <u>Lineweight</u>

IntersectWith Location Modified

Events

<u>Mirror</u> <u>Normal</u>

Mirror3D ObjectID

<u>Move</u> <u>ObjectName</u>

<u>ReleaseAnchor</u> <u>OwnerID</u>

<u>PlotStyleName</u>

Rotate3D Rotation

ScaleEntity ScaleX

SetXData ScaleY

<u>TransformBy</u> <u>ScaleZ</u>

<u>Update</u> <u>TrueColor</u>

Visible

↑ ↓ Camera Object

AEC camera object.

VBA object name: AecCamera

Create using: AddCustomObject("AecCamera")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

	Properties
Methods	<u>Application</u>
<u>ArrayPolar</u>	• •
ArrayRectangular	Color
,	<u>Description</u>
<u>AttachAnchor</u>	Document

Copy

Handle

Delete

HasExtensionDictionary

GetAnchor

Hyperlinks

<u>GetBoundingBox</u>

Layer

GetExtensionDictionary

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u>

<u>IntersectWith</u> <u>Lineweight</u>

Events

Modified

Mirror Location

Mirror3D Name

Move Normal

ReleaseAnchor ObjectID

Rotate ObjectName

Rotate3D OwnerID

<u>ScaleEntity</u> <u>PlotStyleName</u>

<u>SetXData</u> <u>Rotation</u>

<u>TransformBy</u> <u>TrueColor</u>

<u>Update</u> <u>Visible</u>

Zoom



Object

Base Class for the layout grid objects.

VBA object name: AecCellLayoutTool

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

This is the base class for all rectangular cell layout tools, including AecLayoutGrid2D and AecLayoutGrid3d. You can use AecCellLayoutTool as a generic reference to layout grid objects.

Methods

ArrayPolar

Droportice

<u>ArrayRectangular</u>

r robernes

AttachAnchor

Application

Color

<u>ClosestNode</u>

Description

Copy

Document

<u>Delete</u>

<u>GetAnchor</u>

<u>Handle</u>

<u>GetBoundingBox</u>

HasExtensionDictionary

GetExtensionDictionary Hyperlinks

GetXData

Layer

Highlight

Linetype

IntersectWith

LinetypeScale

Events

Modified

Mirror

<u>Lineweight</u>

Mirror3D

Location

Move

Normal

<u>NodeLocation</u>

ObjectID

ReleaseAnchor

<u>ObjectName</u>

Rotate

OwnerID

Rotate3D

<u>PlotStyleName</u>

ScaleEntity

Rotation

SetXData

TrueColor

TransformBy

Visible

<u>Update</u>

↑ Use Clip Volume Object

Clip Volume entity defining an extruded region for clipping.

VBA object name: AecClipVol

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

This is the base class for ADT building elevation and section line objects.

Properties

Angle1

Methods Angle 2

<u>ArrayPolar</u> <u>Application</u>

<u>ArrayRectangular</u> <u>Color</u>

<u>AttachAnchor</u> <u>Description</u>

<u>ClosestNode</u> <u>Document</u>

<u>Copy</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u>

<u>GetAnchor</u> <u>Height</u>

<u>GetBoundingBox</u> <u>Hyperlinks</u>

GetExtensionDictionary Layer

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u>

<u>IntersectWith</u> <u>Lineweight</u> Modified

Events

<u>Mirror</u> <u>Location</u>

<u>Mirror3D</u> <u>LowerExtension</u>

Move Normal

NodeLocation ObjectID

ReleaseAnchor ObjectName

<u>Rotate</u> <u>OwnerID</u>

<u>Rotate3D</u> <u>PlotStyleName</u>

ScaleEntity Rotation

SetXData Side1

<u>TransformBy</u> <u>Side2</u>

<u>Update</u> <u>TrueColor</u>

<u>UseModelExtents</u>

<u>Visible</u>



Object

Clip Volume Result which captures the resulting graphics for a section or elevation for insertion into the drawing.

VBA object name: AecClipVolRes

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

This is the base class for the ADT section line object.

Methods

<u>ArrayPolar</u>

Properties

Application

ArrayRectangular

Color

AttachAnchor

Description

Copy

Document

<u>Delete</u>

Handle

GetAnchor

<u>GetBoundingBox</u>

<u>HasExtensionDictionary</u>

GetExtensionDictionary Hyperlinks

GetXData

Layer

Highlight

Linetype

Events

IntersectWith

<u>LinetypeScale</u>

Modified

Mirror

<u>Lineweight</u>

Mirror3D

Location

Move

Normal

ReleaseAnchor

ObjectID

Rotate

ObjectName

Rotate3D

OwnerID

ScaleEntity

PlotStyleName

SetXData

Rotation

TransformBy

TrueColor

Update

Visible

↑ ↓ Aec Database Object

AEC Database object extending the Acad Database object.

VBA object name: AecDatabase

Create using: N/A

Access via: ModelSpace.Item

Note: Access all properties and methods of the AecDatabase object through the <u>AecBaseDocument</u> object.

Properties

Blocks

Dictionaries

DimStyles

ElevationModelSpace

Flouration Dance Change

<u>Ειεναιιυπταμεί ομαίε</u>

Groups

Layers

Methods

Layouts

CopyObjects

Limits

HandleToObject Linetypes

<u>Init</u> <u>ModelSpace</u>

ObjectIdToObject PaperSpace

PlotConfigurations

Preferences

RegisteredApplications

<u>TextStyles</u>

<u>UserCoordinateSystems</u>

Viewports

Views

↑ ↓ Aec Database

Preferences Object

Extends the AutoCAD Database Preferences object to include AEC properties and methods.

VBA object name: AecDatabasePreferences

Create using: N/A

Access via: ModelSpace.Item

Note: Use <u>AecBaseDatabasePreferences</u> instead of AecDatabasePreferences.

Properties

<u>AllowLongSymbolNames</u>

Application

ContourLinesPerSurface

DisplaySilhouette

Lineweight

LineWeightDisplay

MaxActiveViewports

ObjectSortByPlotting

Methods ObjectSortByPSOutput

<u>Init</u> <u>ObjectSortByRedraws</u>

<u>ObjectSortByRegens</u>

ObjectSortBySelection

ObjectSortBySnap

OLELaunch

RenderSmoothness

<u>SegmentPerPolyline</u>

SolidFill

TextFrameDisplay

XRefEdit

XRefLayerVisibility

↑ ↓ Aec Dictionary Object

AEC Dictionary containing a collection of AEC objects.

VBA object name: AecDictionary

Create using: N/A

Access via: N/A

AecDictionary is the base class for AEC style collections, such as AecLayerKeyStyles and AecMaskBlockStyles. These collections can be accessed from the AecBaseDocument object.

	Properties	
Methods	<u>Application</u>	
<u>Delete</u>	Count	
GetExtensionDictionary		
GetXData	Document	Frants
	Handle	Events

<u>HasExtensionDictionary Modified</u>

Remove ObjectID

Rename ObjectName

<u>SetXData</u> <u>OwnerID</u>

↑ ↓ Aec Dictionary Record

Object

An AEC Dictionary item.

VBA object name: AecDictRecord

Create using: N/A

Access via: N/A

This is the base class for all AEC Style objects, such as AecLayerKeyStyle and AecMaskBlockStyle.

Properties

AlternateName

Application

Methods Description

<u>Delete</u> <u>Document</u>

GetExtensionDictionary Handle

Events

Modified

<u>GetXData</u> <u>HasExtensionDictionary</u>

<u>SetXData</u> <u>Name</u>

ObjectID

ObjectName

<u>OwnerID</u>

↑ ■ AecDisplayComponent

Object

An entity or hatch component of a the AecDisplayComponents collection.

VBA object name: AecDisplayComponent

Create using: N/A

Access via: AecDisplayComponents.Item

AecDisplayComponent is a base class. The derived classes are AecDisplayComponentEntity or AecDisplayComponentHatch. Check the object type returned by the AecDisplayComponents.Item for the underlying object.

Properties

Index

Methods Name Events

<u>Owner</u>

 \uparrow

AecDisplayComponentEntity Object

The entity properties for the display component.

VBA object name: AecDisplayComponentEntity

Create using: N/A

Access via: AecDisplayComponents.Item

AecDisplayComponentEntity is a derived class. AecDisplayComponent is the base class. Check the type returned by the AecDisplayComponents.Item method for the underlying object.

Properties

AllowByMaterial

ByMaterial

Index

<u>Layer</u>

<u>Linetype</u>

Methods Linetypescale

Events

LineWeight

<u>Name</u>

<u>Owner</u>

<u>PlotStyleName</u>

<u>TrueColor</u>

<u>Visible</u>



AecDisplayComponentHatch Object

The hatch properties for the display component.

VBA object name: AecDisplayComponentHatch

Create using: N/A

Access via: AecDisplayComponents.Item

AecDisplayComponentHatch is a derived class. AecDisplayComponent is the base class. Check the type returned by the AecDisplayComponents.Item method for the underlying object.

Properties

<u>Angle</u>

DoubleHatch

HatchType

<u>Index</u>

<u>Name</u>

Methods Owner

Events

<u>PatternName</u>

<u>Scale</u>

Spacing

<u>UseAngleOfObject</u>

XOffset

YOffset



AecDisplayComponents Collection

The collection of display components in a display representation.

VBA object name: AecDisplayComponents

Create using: N/A

Access via: AecDisplayProperties.DisplayComponents

The display components collection contains AecDisplayComponentEntity and AecDisplayComponentHatch objects.

Methods
Count
Item
Owner



AecDisplayConfiguration Object

A display configuration.

VBA object name: AecDisplayConfiguration

Create using: N/A

Access via: AecDisplayConfiguration.Item

Properties

<u>AlternameName</u>

CutPlaneDisplayAbove

CutPlaneDisplayBelow

Methods CutPlaneHeight

Events

DisplaySet

Modified

FixedViewDirection

<u>Name</u>

<u>UseFixedViewDirection</u>



AecDisplayConfigurations Object

A collection of display configurations.

VBA object name: AecDisplayConfigurations

Create using: N/A

Access via: AecBaseDatabase.DisplayConfigurations

Aec Base Document. Display Configurations

Methods

Add

Has Properties

Import ActiveConfiguration Events

Modified

Item Count

Remove

.

Rename

△ △ AecDisplayProperties

Object

A collecton of display properties for a display representation.

VBA object name: AecDisplayProperties

Create using: N/A

Access via: AecDisplayRepresentation.DefaultDisplayProperties

AecDisplayRepresentation.OverrideDisplayProperties

Use the AecDisplayProperties collection to access the material properties, entity component properties and hatch component properties of the display representation.

AecDisplayProperties is a base class. The derived class is AecDisplayPropertiesMaterial. Check the type returned by the DefaultDisplayProperties method and OverrideDisplayProperties method for the underlying object.

Properties

Methods DisplayComponents Events



AecDisplayPropertiesMaterial Object

The material display properties for a display representation.

VBA object name: AecDisplayPropertiesMaterial

Create using: N/A

Access via: AecDisplayRepresentation.DefaultDisplayProperties

Aec Display Representation. Override Display Properties

AecDisplayPropertiesMaterial is a derived class. AecDisplayProperties is the base class. Check the type returned by the DefaultDisplayProperties method and OverrideDisplayProperties method for the underlying object.

Properties

DisplayComponents

DisplayHiddenLinework

ExcludeFrom2dSectionShrinkwrap

$\underline{Sectioned Body Rendering Material Name}$

<u>SectionRenderingMaterialName</u>

<u>SurfaceHatchPlacement</u>

Methods

<u>SurfaceRenderingMaterialName</u>

<u>SurfaceRenderMaterialMapping</u>

<u>MergeCommonMaterials</u>



Events

AecDisplayRepresentation Object

A display representation.

VBA object name: AecDisplayRepresentation

Create using: AecDisplayRepresentation.Duplicate

Access via: AecDisplayRepresentations.Item

Name

Properties

Methods AlternameName

<u>AddOverride</u> <u>ClassName</u>

DeleteOverride DefaultDisplayProperties Modified

<u>Duplicate</u> <u>DisplayName</u>

<u>OverrideDisplayProperties</u>



AecDisplayRepresentations Object

The collection of display representations.

VBA object name: AecDisplayRepresentations

Create using: N/A

AecDisplaySet.DisplayRepresentations

Access via: AecBaseDatabase.DisplayRepresentations

AecBaseDocument.DisplayRepresentations

Methods

Append Properties

Has Count Events

<u>Item</u> <u>Owner</u>

Remove



Object

In-place edit profile object.

VBA object name: AecEditInPlaceProfile

Create using: AddCustomObject("AecEditInPlaceProfile")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

	Properties
Methods	<u>Application</u>
<u>ArrayPolar</u>	• •
<u>ArrayRectangular</u>	Description
Arrayrvectarigular	Document
Attach Anchor	

<u> ALLACHIANICHUI</u>

Handle

Copy

HasExtensionDictionary

Events

<u>Delete</u>

Hyperlinks

GetAnchor

InsertionPoint

<u>GetBoundingBox</u>

GetExtensionDictionary Layer

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u>

<u>IntersectWith</u> <u>Lineweight</u> <u>Modified</u>

Mirror Location

Mirror3D Normal

Move ObjectID

<u>ReleaseAnchor</u> <u>ObjectName</u>

<u>Rotate</u> <u>OwnerID</u>

Rotate3D PlotStyleName

ScaleEntity Profile

<u>SetXData</u> <u>ProfileStyleName</u>

<u>TransformBy</u> <u>Rotation</u>

<u>Update</u> <u>TrueColor</u>

Visible

↑ ↓ Aec Entity Object

Base class for all AEC entities.

VBA object name: AecEntity

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Although AecEntity can be used as a generic reference to any AEC object, it is better to use <u>AecGeo</u> as a generic AEC object reference. AecGeo is the greatest common dominator for all AEC objects.

Methods	
. 51	Properties
<u>ArrayPolar</u>	
	Application
ArrayRectangular	

ArrayNeciangulai

Color

Copy

Description

<u>Delete</u>

Document

 $\underline{\mathsf{GetBoundingBox}}$

Handle

GetExtensionDictionary

HasExtensionDictionary

Events

<u>GetXData</u> <u>Hyperlinks</u>

<u>Highlight</u> <u>Layer</u>

<u>IntersectWith</u> <u>Linetype</u> <u>Modified</u>

<u>Mirror</u> <u>LinetypeScale</u>

Mirror3D <u>Lineweight</u>

Move ObjectID

<u>Rotate</u> <u>ObjectName</u>

Rotate3D OwnerID

<u>ScaleEntity</u> <u>PlotStyleName</u>

<u>SetXData</u> <u>TrueColor</u>

<u>TransformBy</u> <u>Visible</u>

<u>Update</u>

↑ ↓ Aec Entity Reference

Object

A reference to any entity in the drawing, similar to a block reference.

VBA object name: AecEntRef

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Properties

Application

Methods Color

<u>ArrayPolar</u> <u>Description</u>

<u>ArrayRectangular</u> <u>Document</u>

<u>AttachAnchor</u> <u>Handle</u>

<u>Copy</u> <u>HasExtensionDictionary</u>

<u>Delete</u> <u>Hyperlinks</u>

<u>GetAnchor</u> <u>Layer</u>

<u>GetBoundingBox</u> <u>Linetype</u>

<u>GetExtensionDictionary</u> <u>LinetypeScale</u>

GetXData Lineweight

Highlight Location Events

Modified

<u>IntersectWith</u> <u>Normal</u>

Mirror ObjectID

Mirror3D ObjectName

Move Offset

<u>ReleaseAnchor</u> <u>OwnerID</u>

<u>Rotate</u> <u>PlotStyleName</u>

Rotation Rotation

ScaleEntity ScaleX

<u>SetXData</u> <u>ScaleY</u>

<u>TransformBy</u> <u>ScaleZ</u>

<u>Update</u> <u>TrueColor</u>

<u>UseOffset</u>

<u>Visible</u>

↑ ↓ AecGeo Object

Provides location, rotation and anchoring capabilities for AEC entities.

VBA object name: AecGeo

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Use AecGeo as a generic reference to AEC objects.

Methods

<u>ArrayPolar</u> Properties

<u>Application</u>

<u>ArrayRectangular</u>

Color

AttachAnchor

- 1 (1)

Description

Сору

Document

Delete

Handle

GetAnchor

<u>HasExtensionDictionary</u>

<u>GetBoundingBox</u>

Hyperlinks

GetExtensionDictionary Layer

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u>

Events

<u>IntersectWith</u> <u>Lineweight</u>

Modified

Mirror Location

Mirror3D Normal

Move ObjectID

<u>ReleaseAnchor</u> <u>ObjectName</u>

<u>Rotate</u> <u>OwnerID</u>

Rotate3D PlotStyleName

<u>ScaleEntity</u> <u>Rotation</u>

<u>SetXData</u> <u>TrueColor</u>

<u>TransformBy</u> <u>Visible</u>

Update

Base class for grid assemblies.

VBA object name: AecGridAssembly

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via:

Block.Item

SelectionSet.Item

Group.Item

You can use AecGridAssembly as a generic reference to grids, but no properties or methods specific to grids are currently exposed through this object.

Methods		
Arroy Dolor		
<u>ArrayPolar</u>		

ArrayRectangular

Properties

Application

AttachAnchor

Color

ClosestNode

Description

Copy

Document

Delete Handle

GetAnchor HasExtensionDictionary

<u>GetBoundingBox</u> **Hyperlinks**

GetExtensionDictionary Layer

GetXData **Linetype**

LinetypeScale Highlight

Events

Modified

IntersectWith Lineweight

Mirror **Location**

Mirror3D Normal

Move ObjectID

NodeLocation ObjectName

ReleaseAnchor **OwnerID**

Rotate **PlotStyleName**

Rotation Rotate3D

TrueColor ScaleEntity

SetXData Visible <u>TransformBy</u>

<u>Update</u>

↑ Layer Key Object

Layer key definition, which specifies the layer properties generated from a particular key.

VBA object name: AecLayerKey

Create using: AecLayerKeys.Add

Access via: AecLayerKeys.Item

Properties

Color

Description

<u>Index</u>

Layer

LayerKeyStyle

Methods

None <u>Linetype</u>

LineWeight

<u>Name</u>

<u>PlotStyleName</u>

<u>Plottable</u>

Removeable

1 ■ Layer Keys Collection

Object

The collection of layer key definitions in a layer key style.

VBA object name: AecLayerKeys

Create using: N/A

Access via: AecLayerKeyStyle.Keys

Methods

<u>Add</u> Properties

Count

<u>Item</u>

LayerKeyStyle

Remove

↑ Layer Key Style Object

A collection of layer keys.

VBA object name: AecLayerKeyStyle

Create using: N/A

Access via: AecLayerKeyStyles.Item

Properties

AlternateName

Application

Description

Methods

Document

Delete

Handle

ExpandKey

<u> LAPAHAINO y</u>

HasExtensionDictionary Events

<u>GenerateLayer</u>

<u>Keys</u> <u>Modified</u>

<u>GetExtensionDictionary</u>

Name

GetXData

ObjectID

<u>SetXData</u>

ObjectName

OverridesEnabled

<u>OverrideSettings</u>

OwnerID

↑ Layer Key Styles

Modified

Collection

The collection of layer key styles in the specified drawing.

VBA object name: AecLayerKeyStyles

Create using: N/A

Access via: AecBaseDatabase.LayerKeyStyles

MethodsPropertiesAddApplicationDeleteCount

GetExtensionDictionary Document

GetXData Handle Events

<u>Has</u> <u>HasExtensionDictionary</u>

<u>Item</u> <u>ObjectID</u>

Remove ObjectName

<u>SetXData</u> <u>OwnerID</u>

1 ■ Layer Key Override

Setting Object

A layer key style override setting.

VBA object name: AecLayerOverrideSetting

Create using: N/A

Access via: AecLayerOverrideSettings.Item

Properties

<u>Index</u>

Methods LayerKeyStyle

None

Name

Value

↑ Layer Key Style

Override Settings Collection

The collection of override settings in a layer key style.

VBA object name: AecLayerOverrideSettings

Create using: N/A

Access via: AecLayerKeyStyle.OverrideSettings

Properties

Methods

Count

<u>Item</u>

<u>LayerKeyStyle</u>



A single dimensional grid with nodes along a single curve.

VBA object name: AecLayoutCurve

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group. Item

You cannot create a new layout curve through ActiveX, and many properties of this object are not yet exposed.

Properties

Application

Color

<u>ArrayPolar</u> <u>Count</u>

<u>ArrayRectangular</u> <u>Description</u>

<u>AttachAnchor</u> <u>Document</u>

<u>ClosestNode</u> <u>EndOffset</u>

<u>Copy</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u>

<u>GetAnchor</u> <u>Hyperlinks</u>

<u>GetBoundingBox</u> <u>Layer</u>

GetExtensionDictionary Linetype

<u>GetXData</u> <u>LinetypeScale</u>

<u>Highlight</u> <u>Lineweight</u> <u>Events</u>

IntersectWith Location <u>Modified</u>

<u>Mirror</u> <u>Nodes</u>

Mirror3D Normal

Move ObjectID

NodeLocation ObjectName

ReleaseAnchor OwnerID

<u>Rotate</u> <u>PlotStyleName</u>

Rotate3D Rotation

ScaleEntity Spacing

<u>SetXData</u> <u>StartOffset</u>

<u>TransformBy</u> <u>TrueColor</u>

<u>Update</u> <u>Type</u>

<u>Value</u>

<u>Visible</u>

↑ Layout Curve Node

Object

A node in a layout curve.

VBA object name: AecLayoutCurveNode

Create using: N/A

Access via: AecLayoutCurveNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods

None LayoutCurve

1 Layout Curve Nodes

Collection

The collection of nodes in a layout curve.

VBA object name: AecLayoutCurveNodes

Create using: N/A

Access via: AecLayoutCurve.Nodes

You can access the members of the collection, but they currently contain no useful information.

 $\frac{\text{Methods}}{\text{Count}} = \frac{\text{Events}}{\text{None}}$

1 ■ Layout Grid 2D Object

Layout grid entity used to layout any AEC object within a 2D grid.

VBA object name: AecLayoutGrid2D

Create using: AddCustomObject("AecLayoutGrid2D")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Properties

Angle

Application

BayAngle

Boundary

Color

Depth

Description

Document

Methods Handle

<u>ArrayPolar</u> <u>HasExtensionDictionary</u>

<u>ArrayRectangular</u> <u>Hyperlinks</u>

<u>AttachAnchor</u> <u>InsideRadius</u>

<u>Copy</u> <u>Layer</u>

<u>ClosestNode</u> <u>Linetype</u>

<u>Delete</u> <u>LinetypeScale</u>

GetAnchor Lineweight

GetBoundingBox Location

GetExtensionDictionary Normal

GetXData ObjectID

Highlight ObjectName Events

IntersectWith OwnerID Modified

Mirror PlotStyleName

Mirror3D Rotation

1400

<u>Niove</u> <u>Shape</u>

NodeLocation <u>TrueColor</u>

ReleaseAnchor Visible

Rotate Width

Rotate3D XCount

ScaleEntity XEndOffset

<u>SetXData</u> <u>XNodes</u>

<u>TransformBy</u> <u>XSpacing</u>

<u>Update</u> <u>XStartOffset</u>

<u>XType</u>

YCount

YEndOffset

YNodes

YSpacing

YStartOffset

<u>YType</u>

ZCount

1 ■ Layout Grid 2D Node

Object

A node in a 2D layout grid.

VBA object name: AecLayoutGrid2DNode

Create using: N/A

Access via: AecLayoutGrid2DNodes.Item

(But see note below.)

This object is not currently accessible. You can access 2D node objects through <u>AecLayoutGrid2DXNode</u> and <u>AecLayoutGrid2DYNode</u>, but most of their properties are not exposed.

Properties

Index

Methods Events

None LayoutGrid2D None

↑ Layout Grid 2D Nodes

Collection

The collection of nodes in a 2D layout grid.

VBA object name: AecLayoutGrid2DNodes

Create using: N/A

Access via: N/A

This collection is not accessible. Use <u>AecLayoutGrid2DXNodes</u> and <u>AecLayoutGrid2DYNodes</u> instead.

 $\frac{\text{Methods}}{\text{Count}} = \frac{\text{Events}}{\text{Item}}$ None $\frac{\text{LayoutGrid2D}}{\text{LayoutGrid2D}}$

↑ Layout Grid 2D X

Node Object

A node along the X axis of a 2D layout grid.

VBA object name: AecLayoutGrid2DXNode

Create using: N/A

Access via: AecLayoutGrid2DXNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods Events
None LayoutGrid2D None



Nodes Collection

Collection of nodes along the X axis of a 2D layout grid.

VBA object name: AecLayoutGrid2DXNodes

Create using: N/A

Access via: AecLayoutGrid2D.XNodes

You can access the members of this collection, but they contain no useful information.

 $\frac{\text{Methods}}{\text{Count}} = \frac{\text{Events}}{\text{Item}}$ None $\frac{\text{LayoutGrid2D}}{\text{LayoutGrid2D}}$

1 Layout Grid 2D Y

Node Object

A node along the Y axis of a 2D layout grid.

VBA object name: AecLayoutGrid2DYNode

Create using: N/A

Access via: AecLayoutGrid2DYNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods Events
None LayoutGrid2D None

↑ Layout Grid 2D Y

Nodes Collection

Collection of nodes along the Y axis of a 2D layout grid.

VBA object name: AecLayoutGrid2DYNodes

Create using: N/A

Access via: AecLayoutGrid2D.YNodes

You can access the members of this collection, but they contain no useful information.

 $\frac{\text{Methods}}{\text{Count}} = \frac{\text{Events}}{\text{Item}}$ None $\frac{\text{LayoutGrid2D}}{\text{LayoutGrid2D}}$

1 Layout Grid 3D Object

A layout volume for arranging objects in a 3D grid.

VBA object name: AecLayoutGrid3D

Create using: AddCustomObject("AecLayoutGrid3D")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Properties

Application

Color

Depth

Description

Document

Handle

HasExtensionDictionary

Height

Methods Hyperlinks

<u>ArrayPolar</u> <u>Layer</u>

<u>ArrayRectangular</u> <u>Linetype</u>

<u>AttachAnchor</u> <u>LinetypeScale</u>

<u>ClosestNode</u> <u>Lineweight</u>

<u>Copy</u> <u>Location</u>

<u>Delete</u> <u>Normal</u>

GetAnchor ObjectID

<u>GetBoundingBox</u> <u>ObjectName</u>

GetExtensionDictionary OwnerID

GetXData PlotStyleName

Highlight Rotation Events

IntersectWith TrueColor Modified

<u>Mirror</u> <u>Visible</u>

Mirror3D Width

1400

<u>XCount</u>

NodeLocation XEndOffset

ReleaseAnchor XNodes

Rotate XSpacing

Rotate3D XStartOffset

ScaleEntity XType

<u>SetXData</u> <u>YCount</u>

<u>TransformBy</u> <u>YEndOffset</u>

<u>Update</u> <u>YNodes</u>

YSpacing

YStartOffset

<u>YType</u>

ZCount

ZEndOffset

ZNodes

ZSpacing

ZStartOffset

ZType

1 Layout Grid 3D Node

Object

A node of a 3D Layout Grid.

VBA object name: AecLayoutGrid3DNode

Create using: N/A

Access via: AecLayoutGrid3DNodes.Item

(But see note below.)

This object is not currently accessible. You can access 3D node objects through <u>AecLayoutGrid3DXNode</u> and <u>AecLayoutGrid3DYNode</u>, but most of their properties are not exposed.

Properties

Index

Methods Events

None LayoutGrid3D None

1 Layout Grid 3D Nodes

Collection

The collection of nodes in a 3D layout grid.

VBA object name: AecLayoutGrid3DNodes

Create using: N/A

Access via: N/A

This collection is not accessible. Use <u>AecLayoutGrid3DXNodes</u> and <u>AecLayoutGrid3DYNodes</u> instead.

 $\frac{\text{Methods}}{\text{Count}} = \frac{\text{Events}}{\text{Item}}$ None $\frac{\text{LayoutGrid3D}}{\text{LayoutGrid3D}}$

1 Layout Grid 3D X

Node Object

A node along the X axis of a layout volume.

VBA object name: AecLayoutGrid3DXNode

Create using: N/A

Access via: AecLayoutGrid3DXNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods Events
None LayoutGrid3D None



Nodes Collection

Collection of nodes along the X axis of a layout volume.

VBA object name: AecLayoutGrid3DXNodes

Create using: N/A

Access via: AecLayoutGrid3D.XNodes

You can access the members of this collection, but they contain no useful information.

Properties

Methods

Count

<u>Item</u>

LayoutGrid3D

1 Layout Grid 3D Y

Node Object

A node along the Y axis of a layout volume.

VBA object name: AecLayoutGrid3DYNode

Create using: N/A

Access via: AecLayoutGrid3DYNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods Events
None LayoutGrid3D None



Nodes Collection

Collection of nodes along the Y axis of a layout volume.

VBA object name: AecLayoutGrid3DYNodes

Create using: N/A

Access via: AecLayoutGrid3D.YNodes

You can access the members of this collection, but they contain no useful information.

Properties

Methods

Count

<u>Item</u>

LayoutGrid3D

1 Layout Grid 3D Z

Node Object

A node along the Z axis of a layout volume.

VBA object name: AecLayoutGrid3DZNode

Create using: N/A

Access via: AecLayoutGrid3DZNodes.Item

Most node properties have not been exposed through ActiveX.

Properties

<u>Index</u>

Methods

None LayoutGrid3D



Nodes Collection

Collection of nodes along the Z axis of a layout volume.

VBA object name: AecLayoutGrid3DZNodes

Create using: N/A

Access via: AecLayoutGrid3D.ZNodes

You can access the members of this collection, but they contain no useful information.

Properties

Methods

Count

<u>Item</u>

LayoutGrid3D

↑ Layout Node Object

Provides common properties shared by all types of layout nodes.

VBA object name: AecLayoutNode

Create using: N/A

Access via: AecLayoutNodes.Item

Methods Properties Events
None Spacing None

↑ ↓ Layout Nodes

Collection

Collection of layout nodes.

VBA object name: AecLayoutNodes

Create using: N/A

Access via: AecLayoutCurve.Nodes

Methods Properties

<u>Item</u> <u>Count</u>



Provides common functionality for layout curve and layout grid objects.

VBA object name: AecLayoutTool

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

AecLayoutTool can be used as a generic reference to layout objects.

Methods

ArrayPolar

ArrayRectangular

Properties

Application

AttachAnchor

Color

ClosestNode

Description

Copy

Document

Delete

Handle

GetAnchor

HasExtensionDictionary

Modified

<u>GetBoundingBox</u> <u>Hyperlinks</u>

GetExtensionDictionary Layer

<u>GetXData</u> <u>Linetype</u>

Highlight LinetypeScale Events

IntersectWith Lineweight

Mirror Location

Mirror3D Normal

Move ObjectID

NodeLocation ObjectName

ReleaseAnchor OwnerID

<u>Rotate</u> <u>PlotStyleName</u>

Rotation Rotation

<u>ScaleEntity</u> <u>TrueColor</u>

<u>SetXData</u> <u>Visible</u>

TransformBy

Update

↑ Wask Block Object

An instance of a mask block, a two-dimensional block that covers part of an AEC object in a plan (2D) view.

VBA object name: AecMaskBlockRef

Create using: Modelspace.AddCustomObject("AecMaskBlockRef")

Model Space. Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Properties

Application

Color

Methods CutProfile

<u>ArrayPolar</u>

Description

<u>ArrayRectangular</u>
<u>Document</u>

<u>AttachAnchor</u>

<u>Handle</u> <u>Copy</u>

<u>HasExtensionDictionary</u> <u>Delete</u>

GetAnchor Hyperlinks

<u>GetBoundingBox</u> <u>Layer</u>

<u>GetExtensionDictionary</u> <u>Linetype</u>

GetXData LinetypeScale

Highlight Lineweight Events

<u>IntersectWith</u> <u>Location</u> <u>Modified</u>

<u>Mirror</u> <u>Normal</u>

Mirror3D ObjectID

<u>Move</u> <u>ObjectName</u>

ReleaseAnchor OwnerID

<u>Rotate</u> <u>PlotStyleName</u>

Rotation Rotation

ScaleEntity ScaleX

<u>SetXData</u> <u>ScaleY</u>

<u>TransformBy</u> <u>ScaleZ</u>

<u>Update</u> <u>Style</u>

<u>StyleName</u>

<u>TrueColor</u>

<u>Visible</u>



Object

Defines the properties of a mask block object.

VBA object name: AecMaskBlockStyle

Create using: AecMaskBlockStyles.Add("Name")

Access via: AecMaskBlockStyles.Item

Most mask block style properties are not yet exposed through ActiveX.

Properties

AlternateName

Application

Methods Description

<u>Delete</u> <u>Document</u>

Events

GetExtensionDictionary Handle

LVCIICS

Modified

GetXData HasExtensionDictionary

<u>SetXData</u> <u>Name</u>

ObjectID

ObjectName

<u>OwnerID</u>



Collection

The collection of mask block styles in the specified drawing.

VBA object name: AecMaskBlockStyles

Create using: N/A

Access via: AecBaseDatabase.MaskBlocks

Use of this collection is mostly limited to listing the name and description of the styles.

Methods	Properties				
Add	Application				
<u>Delete</u>	Count				
GetExtensionDictionary Document					
0-1/0-1-	Events				

<u>GetXData</u> <u>Handle</u>

Modified

<u>Has</u> <u>HasExtensionDictionary</u>

<u>Item</u> <u>ObjectID</u>

Remove ObjectName

<u>SetXData</u> <u>OwnerID</u>

↑ Wass Element Object

An object that has behaviors based on its shape.

VBA object name: AecMassElement

Create using: AddCustomObject("AecMassElement")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group. Item

Properties

Application

<u>Color</u>

Depth

Description

Deviation

Document

Methods

Handle

<u>ArrayPolar</u>

HasExtensionDictionary

<u>ArrayRectangular</u>

Height

AttachAnchor

Hyperlinks

Copy

Layer

<u>Delete</u>

Linetype

ExportFreeForm

LinetypeScale

GetAnchor

Lineweight

 $\underline{\mathsf{GetBoundingBox}}$

Location

GetExtensionDictionary

MassGroup

GetXData

MassGroupName

Highlight

Normal

Modified

Events

<u>ImportFreeForm</u>

<u>ObjectID</u>

<u>IntersectWith</u>

ObjectName

Mirror

Operation

Mirror3D

OwnerID

<u>Move</u>

PlotStyleName

ReleaseAnchor

<u>Rotate</u> <u>Profile</u>

Rotate3D ProfileStyle

<u>ScaleEntity</u> <u>ProfileStyleName</u>

<u>SetXData</u> <u>Radius</u>

<u>TransformBy</u> <u>Rise</u>

<u>Update</u> <u>Rotation</u>

<u>Style</u>

StyleName

Subtype

<u>TrueColor</u>

<u>Type</u>

<u>Visible</u>

<u>Volume</u>

Width

↑ Wass Element Style

Object

Methods

Defines the properties of a mass element object.

VBA object name: AecMassElementStyle

Create using: AecMassElementStyles.Add("Name")

Access via: AecMassElementStyles.Item

Properties

<u>AlternateName</u>

Application

<u>Description</u>

<u>Delete</u> <u>Document</u>

GetExtensionDictionary Handle

Events

Modified

IVIUGIIIUG

<u>GetXData</u> <u>HasExtensionDictionary</u>

<u>SetXData</u> <u>Name</u>

<u>ObjectID</u>

ObjectName

<u>OwnerID</u>



Collection

The collection of mass element styles in the specified drawing.

VBA object name: AecMassElementStyles

Create using: N/A

Access via: AecBaseDatabase.MassElementStyles

Methods

<u>Add</u>

Application

Delete

Count

<u>GetExtensionDictionary</u>

Document

GetXData

<u>Handle</u>

<u>Has</u> <u>Modified</u>

HasExtensionDictionary

I IMOLACCIONO I DIOCIONAL Y

<u>Item</u>

<u>ObjectID</u>

Remove

ObjectName

Rename

<u>OwnerID</u>

<u>SetXData</u>

↑ Wass Group Object

A group of mass elements combined to form a complex shape.

VBA object name: AecMassGroup

Create using: AecMassGroups.Add

Access via: AecMassGroups.Item

Properties

Application

Methods Color

<u>ArrayPolar</u> <u>Description</u>

<u>ArrayRectangular</u> <u>Document</u>

<u>AttachAnchor</u> <u>Handle</u>

<u>Copy</u> <u>HasExtensionDictionary</u>

<u>Delete</u> <u>Hyperlinks</u>

<u>ExportFreeForm</u> <u>Layer</u>

GetAnchor Linetype

GetBoundingBox LinetypeScale

GetExtensionDictionary Lineweight

GetXData Location

Highlight MassGroup Modified

Events

<u>IntersectWith</u> <u>MassGroupName</u>

<u>Mirror</u> <u>Name</u>

Mirror3D Normal

Move ObjectID

ReleaseAnchor ObjectName

<u>Rotate</u> <u>Operation</u>

Rotate3D OwnerID

ScaleEntity PlotStyleName

SetXData Rotation

<u>TransformBy</u> <u>SupportsAnchoring</u>

<u>Update</u> <u>TrueColor</u>

Visible

Volume



Collection

The collection of mass groups in the specified drawing.

VBA object name: AecMassGroups

Create using: N/A

Access via: AecBaseDatabase.MassGroups

Properties

Methods

Count

<u>ltem</u>

Database



AecMVBlockDisplayRepDef Object

A display representation definition for a multi-view block style.

VBA object name: AecMVBlockDisplayRepDef

Create using: N/A

Access via: AecMVBlockDisplayRepDefs.Item

Properties

DisplayName

Methods Index

Events

Owner

ViewBlockDefs



AecMVBlockDisplayRepDefs Collection

The collection of display representation definitions for a multi-view block style.

VBA object name: AecMVBlockDisplayRepDefs

Create using: N/A

Access via: AecMVBlockStyle.DisplayRepDefs

Methods
Count
Item
Owner



Object

An instance of a block that can have different representations in different view directions.

VBA object name: AecMVBlockRef

Create using: AddCustomObject("AecMVBlockRef")

Model Space. Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group. Item

Properties

Application

Color

Methods

<u>Description</u>

ArrayPolar

Document

<u>ArrayRectangular</u>

Handle

AttachAnchor

HasExtensionDictionary

Copy

Mirror

Hyperlinks

ObjectName

<u>Delete</u> <u>Layer</u>

GetAnchor Linetype

<u>GetBoundingBox</u> <u>LinetypeScale</u>

GetExtensionDictionary Lineweight

GetXData Location

<u>Highlight</u> <u>Normal</u>

Events

Modified

IntersectWith ObjectID

Mirror3D OwnerID

Move PlotStyleName

ReleaseAnchor Rotation

Rotate ScaleX

Rotate3D ScaleY

ScaleEntity ScaleZ

<u>SetXData</u> <u>Style</u>

<u>TransformBy</u> <u>StyleName</u>

<u>Update</u> <u>TrueColor</u>

<u>ViewBlocks</u>

<u>Visible</u>



Style Object

Defines the properties of a multi-view block object.

VBA object name: AecMVBlockStyle

Create using: AecMVBlockStyles.Add("Name")

Access via: AecMVBlockStyles.Item

Most multi-view block style properties are not yet exposed through ActiveX.

Properties

AlternateName

Application

Description

DisplayRepDefs

Mathada

WICHIUUS . . .

<u>Delete</u> <u>Document</u>

GetExtensionDictionary Handle

<u>Modified</u>

GetXData HasExtensionDictionary

<u>SetXData</u> <u>InterferenceBlockName</u>

Name

ObjectID

ObjectName

OwnerID



Styles Collection

The collection of multi-view block styles in the specified drawing.

VBA object name: AecMVBlockStyles

Create using: N/A

Access via: AecBaseDatabase.MVBlockStyles

Use of this collection is mostly limited to listing the name and description of the styles.

Methods	Properties	
Add	Application	
<u>Delete</u>	Count	
GetExtensionDictionary Document		
<u>GetXData</u>	<u>Handle</u>	Events

Modified

<u>Has</u> <u>HasExtensionDictionary</u>

<u>Item</u> <u>ObjectID</u>

Remove ObjectName

<u>SetXData</u> <u>OwnerID</u>

↑ AecObject Object

Provides common functionality for AEC database objects.

VBA object name: AecObject

Create using: N/A

Mathada

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

Although AecObject can be used as a generic reference to AEC objects, it is better to use <u>AecGeo</u> as a generic reference. AecGeo is the greatest common dominator for all AEC objects.

Properties

Application

Description

vienions

<u>Delete</u> <u>Document</u>

GetExtensionDictionary Handle

Events

Modified

GetXData HasExtensionDictionary

<u>SetXData</u> <u>ObjectID</u>

ObjectName

OwnerID

↑ AEC Polygon Object

An Aec Polygon object.

VBA object name: AecPolygon

Create using: Modelspace.AddCustomObject("AecPolygon")

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group. Item

You can modify existing AECPolygon objects. Although you can create AECPolygons, you cannot control the number or length of its sides, so it's not very useful.

Properties

Application

ArrayPolar

Color

<u>ArrayRectangular</u> <u>Description</u>

<u>AttachAnchor</u> <u>Document</u>

<u>Copy</u> <u>Handle</u>

<u>Delete</u> <u>HasExtensionDictionary</u>

<u>GetAnchor</u> <u>Hyperlinks</u>

GetBoundingBox Layer

<u>GetExtensionDictionary</u> <u>Linetype</u>

<u>GetXData</u> <u>LinetypeScale</u>

<u>Highlight</u> <u>Lineweight</u>

IntersectWith Location

Events

Modified

<u>Mirror</u> <u>Normal</u>

Mirror3D ObjectID

<u>Move</u> <u>ObjectName</u>

ReleaseAnchor OwnerID

<u>Rotate</u> <u>PlotStyleName</u>

Rotate3D Profile

<u>ScaleEntity</u> <u>Rotation</u>

<u>SetXData</u> <u>Style</u>

<u>TransformBy</u> <u>StyleName</u>

<u>Update</u> <u>TrueColor</u>

<u>Visible</u>



Object

Defines the properties of an Aec Polygon object.

VBA object name: AecPolygonStyle

Create using: AecPolygonStyles.Add("Name")

Access via: AecPolygonStyles.Item

Properties

<u>AlternateName</u>

Application

Description

Methods <u>Document</u>

<u>Delete</u> <u>EdgeWidth</u>

<u>GetExtensionDictionary</u> <u>EdgeWidthJustify</u>

Events

Modified

<u>GetXData</u> <u>Handle</u>

<u>SetXData</u> <u>HasExtensionDictionary</u>

<u>Name</u>

<u>ObjectID</u>

ObjectName

<u>OwnerID</u>



Collection

The collection of Aec Polygon styles in the specified drawing.

VBA object name: AecPolygonStyles

Create using: N/A

Access via: AecBaseDatabase.PolygonStyles

Methods Properties

Add Application

<u>Delete</u> <u>Count</u>

GetExtensionDictionary Document

GetXData Handle Events

Modified

<u>Has Extension Dictionary</u>

<u>Item</u> <u>ObjectID</u>

Remove ObjectName

<u>SetXData</u> <u>OwnerID</u>

↑ Profile Object

Defines the properties of a profile style.

VBA object name: AecProfile

Create using: New AecProfile

Access via: AecProfileStyle.Profile

Properties

Methods Area

Add Bounds

<u>Clear</u> <u>Centroid</u>

CopyFrom Coordinates

Intersect Rings

ScaleXY Perimeter

Subtract SelfIntersects

<u>Valid</u>

↑ Profile Style Object

A custom shape defined by two-dimensional closed polylines.

VBA object name: AecProfileStyle

Create using: AecProfileStyles.Add("Name")

Access via: AecProfileStyles.Item

Note: Profiles are inserted into drawings as polylines, not as AEC objects.

	Properties
	<u>AlternateName</u>
	Application
	Description
Methods	<u>Document</u>
<u>Delete</u>	Llandla

<u>naiiuit</u>

Events

GetExtensionDictionary

HasExtensionDictionary Modified

GetXData

Name

<u>SetXData</u>

<u>ObjectID</u>

ObjectName

<u>OwnerID</u>

Profile



Modified

Collection

The collection of profile styles in the specified drawing.

VBA object name: AecProfileStyles

Create using: N/A

Access via: AecBaseDatabase.ProfileStyles

MethodsPropertiesAddApplicationDeleteCount

GetExtensionDictionary Document

GetXData Handle Events

<u>HasExtensionDictionary</u>

<u>Item</u> <u>ObjectID</u>

Remove ObjectName

<u>SetXData</u> <u>OwnerID</u>

↑ ↓ Aec Ring Object

A closed loop of polyline segments, often used as a group inside an AecProfile object.

VBA object name: AecRing

Create using: AecRings.Add

Access via: AecRings.Item

Methods Properties

<u>Clear</u> <u>Coordinates</u>

<u>Contains</u> <u>Index</u>

CopyFrom Profile

FromPoints Valid

FromPolyline Void

↑ ↓ Aec Rings Collection

The collection of rings in an AecProfile object.

VBA object name: AecRings

Create using: N/A

Access via: AecProfile.Rings

Methods

<u>Properties</u>

<u>Count</u>

<u>ltem</u>

<u>Profile</u>

Remove

↑ Slice Object

Slice of a massing group or any AEC object, typically used to generate floor plates and space boundaries.

VBA object name: AecSlice

Create using: AddCustomObject("AecSlice")

Model Space. Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

	Properties
Methods	Application
<u>ArrayPolar</u>	<u>Application</u>
<u>Allayr Olal</u>	Area
<u>ArrayRectangular</u>	<u> Med</u>
- uracy reocarigatar	Color
<u>AttachAnchor</u>	

D-----

Description

ClosestNode

Document

Copy

Handle

<u>Delete</u>

HasExtensionDictionary

GetAnchor

<u>GetBoundingBox</u> <u>Hyperlinks</u>

GetExtensionDictionary Layer

<u>GetXData</u> <u>Linetype</u>

<u>Highlight</u> <u>LinetypeScale</u>

Events

<u>IntersectWith</u> <u>Lineweight</u>

Modified

Mirror Location

Mirror3D Normal

Move ObjectID

NodeLocation ObjectName

ReleaseAnchor OwnerID

Rotate PlaneDepth

Rotate3D PlaneWidth

<u>ScaleEntity</u> <u>PlotStyleName</u>

<u>SetXData</u> <u>Profile</u>

<u>TransformBy</u> <u>Rotation</u>

<u>Update</u> <u>TrueColor</u>

<u>Visible</u>

↑ ↓ Aec View Block Object

View block used for display representations by a multi-view block.

VBA object name: AecViewBlock

Create using: N/A

Access via: AecViewBlocks.Item

Properties

<u>Index</u>

MVBlockRef

Methods Name Events
None None

None XOffset

YOffset

ZOffset

↑ ↓ Aec View Blocks

Collection

A collection of view blocks used for display representations by a multiview block.

VBA object name: AecViewBlocks

Create using: N/A

Access via: AecMVBlockRef.ViewBlocks

You cannot add new view blocks.

Properties

Methods

Count

<u>Item</u>

MVBlockRef

↑ ↓ AecViewBlockDef

Object

A view block definition for a multi-view block display representation definition.

VBA object name: AecViewBlockDef

Create using: N/A

Access via: AecViewBlockDefs.Item

Properties

Block

Methods Index

Events

<u>Owner</u>

ViewDirection

↑ AecViewBlockDefs

Collection

The collection of view block definitions for a multi-view block display representation definition.

VBA object name: AecViewBlockDefs

Create using: N/A

Access via: AecMVBlockDisplayRepDef.ViewBlockDefs

Methods

<u>ltem</u>

<u>Properties</u>

Count

Events

Owner

Remove



Object

Provides standard functionality for all 3D layout tools.

VBA object name: AecVolumeLayoutTool

Create using: N/A

ModelSpace.Item

PaperSpace.Item

Access via: Block.Item

SelectionSet.Item

Group.Item

AecVolumeLayoutTool can be used as a generic reference to 3D layout (volume) grid objects, but you would normally use AecLayoutGrid3D.

Methods

ArrayPolar

ArravRectangular

Properties

Application

AttachAnchor

Color

ClosestNode

Description

Copy

Document

Delete

GetAnchor Handle

<u>GetBoundingBox</u> <u>HasExtensionDictionary</u>

GetExtensionDictionary Hyperlinks

<u>GetXData</u>

<u>Highlight</u> <u>Linetype</u>

IntersectWith LinetypeScale Modified

Events

Layer

<u>Mirror</u> <u>Lineweight</u>

Mirror3D Location

<u>Move</u> <u>Normal</u>

NodeLocation ObjectID

ReleaseAnchor ObjectName

<u>Rotate</u> <u>OwnerID</u>

Rotate3D PlotStyleName

<u>ScaleEntity</u> <u>Rotation</u>

<u>SetXData</u> <u>TrueColor</u>

<u>TransformBy</u> <u>Visible</u>

<u>Update</u>

All Drawing Objects

All of these objects can be defined as an AcadEntity object.

AecAnchorEntToGridAssembly	AecAnchor AecAnchor	<u>AecAnchorEntTo</u>
<u>AecAnchorEntToLayoutCell</u>	<u>AecAnchorEntToLayoutNode</u>	<u>AecAnchorEntTo</u>
<u>AecAnchorLeadEntToNode</u>	<u>AecAnchorToRef</u>	<u>AecBlockRef</u>
<u>AecCamera</u>	<u>AecCellLayoutTool</u>	<u>AecClipVol</u>
<u>AecClipVolRes</u>	AecDictionary	<u>AecDictRecord</u>
<u>AecEntity</u>	<u>AecEntRef</u>	<u>AecGeo</u>
<u>AecGridAssembly</u>	AecLayoutGrid2D	AecLayoutGrid3I
<u>AecLayerKeyStyle</u>	<u>AecLayerKeyStyles</u>	<u>AecLayoutTool</u>
<u>AecMaskBlockRef</u>	<u>AecMaskBlockStyle</u>	<u>AecMaskBlockSt</u>
<u>AecMassElement</u>	<u>AecMassGroup</u>	<u>AecMVBlockSty</u>
<u>AecMVBlockStyles</u>	<u>AecObject</u>	<u>AecPolygon</u>
<u>AecPolygonStyle</u>	<u>AecPolygonStyles</u>	<u>AecProfileStyle</u>
<u>AecProfileStyles</u>	<u>AecSlice</u>	<u>AecVolumeLayou</u>

Methods

| image |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| image |
| image | image | image | image | image | image | | | | |

Α

Add Adds an object to a collection, or adds two AecProfile objects.

<u>ArrayPolar</u> Creates a polar array of objects given a NumberOfObjects, AngleToFill, and CenterPoint.

ArrayRectangular Creates a 2D or 3D rectangular array of objects.

AttachAnchor Attaches an anchor to an AEC Entity

AttachEntity Anchors one AecGeo object to another.

В

C

Clear Clears AecRings and AecProfiles of segments or rings.

ClosestNode Returns the index of the node closest to a given point.

Contains Tests to see if a ring is inside another ring.

<u>ConvertToCurrentAreaDisplay</u> Converts an area value using the LinearUnit and AreaDisplayUnit settings.

<u>ConvertToCurrentVolumeDisplay</u> Converts a volume value using the LinearUnit and VolumeDisplayUnit settings.

Copy Duplicates the given object to the same location.

CopyFrom Copies AecRings and AecProfiles.

CopyObjects Duplicates multiple objects (deep cloning).

D

Delete Deletes the specified object.

Ε

ExpandKey Calculates and returns the Layer Name that would be generated given the supplied 'Key'.

ExportFreeForm Exports a Free Form representation to a file.

F

FromPoints Makes a ring from 2D Points.

<u>FromPolyline</u> Makes a ring from a 2D Polyline.

G

GenerateLayer Generate the layer record, given a Layer Key.

<u>GetAnchor</u> Returns the anchor attached to an object.

GetBoundingBox Gets two points of a box enclosing the specified object.

<u>GetExtensionDictionary</u> Gets the extension dictionary associated with an object.

GetXdata Gets the extended data (XData) associated with an object.

Н

HandleToObject Gets the object that corresponds to the given handle.

<u>Has</u> Determines if an object is a member of the specified collection.

Highlight Sets the highlight status for the given object, or for all objects in a given selection set.

ImportFreeForm Imports a Free Form representation from a file.

<u>Init</u> Initialization function which must be called before the AEC database object can be accessed

Intersect Boolean Intersects an AecProfile with the current AecProfile.

<u>IntersectWith</u> Gets the points where one object intersects another object in the drawing.

<u>Item</u> Returns a member object of a collection.

J

K

L

M

Mirror Creates a mirror image copy of a planar object around an axis.

Mirror3d Creates a mirror image of the given object about a plane.

Move Moves an object along a vector.

Ν

NodeLocation Returns the location of the node at a given index.

0

ObjectIDToObject Gets the object that corresponds to the given object ID.

P

Q

R

ReleaseAnchor Removes an anchor from an object.

Remove Removes an item from a collection.

Rename Renames an item in an AecDictionary.

Rotate Rotates an object around a base point.

Rotate3d Rotates an object around a 3D axis. Point1 and Point2 define the line that becomes the axis of rotation.

S

SaveAsDefault Saves the DatabasePreferences to the registry.

ScaleEntity Scales an object equally in the

<u>ScaleXY</u> Scales a profile in the X and Y directions.

SetXdata Sets the extended data (XData) associated with an object.

Subtract Subtracts an AecProfile from the current AecProfile.

Т

<u>TransformBy</u> Moves, scales, or rotates an object given a 4x4 transformation matrix.

U

<u>Update</u> Updates the object in the drawing window.

V

W

X

Υ

Ζ

Add Method

Adds an object to a collection, or adds two AecProfile objects.

See Also | Example

Signatures: Overview

Layer Keys,
Layer Key Styles,
Mask Block Styles,
Mass Element Styles,
Multi-view Block Styles,
Polygon Styles,
Profile Styles,
Display Configurations

Display Configurations,

Display Sets

RetVal = collection.Add(Name)

Profiles

object.Add(profile)

Rings

RetVal = collection.Add()

ViewBlockDefs

RetVal = collection.Add(AcadBlock)

Signatures: Detail

RetVal = collection.Add(Name)

collection

AecLayerKeys, AecLayerKeyStyles,

AecMassElementStyles, AecMaskBlockStyles,

AecMVBlockStyles, AecPolygonStyles,

AecProfileStyles, AecDisplayConfigurations,

AecDisplaySets, AecViewBlockDefs

The collection the object is being added to.

Name

String; input-only

Name of the object to be added to the collection.

RetVal

AecLayerKey, AecLayerKeyStyle,

AecMaskBlockStyle, AecMassElementStyle,

AecMVBlockStyle, AecPolygonStyle,

AecProfileStyle, AecDisplayConfigurations,

AecDisplaySets, AecViewBlockDefs

The newly added object.

object.Add(profile)

object

<u>AecProfile</u>

The profile you are adding a new profile object to.

profile

AecProfile; input-only

AecProfile to add to the first AecProfile

RetVal = collection.Add()

collection

AecRings

The collection you are adding a new ring definition

to.

AecRing

The newly added object.

RetVal = collection.Add(AcadBlock)

collection

AecViewBlockDefs

The collection you are adding a new block to.

AcadBlock

AcadBlock object; input-only

The block to add to the collection.

RetVal

AecViewBlockDef

The newly created view block definition object.

Append Method

Appends an existing display representation to the collection.

See Also | Example

Signature

object.Append DisplayRepresentation

Object

<u>AecDisplayRepresentations</u> The object or objects this method applies to.

DisplayRepresentation

<u>AecDisplayRepresentation</u>; input-only The display representation to add to the collection.

Remarks

This method applies to a display representations collection in a display set.

ArrayPolar Method

Creates a polar array of objects given a NumberOfObjects, AngleToFill, and CenterPoint.

See Also | Example

Signature

RetVal = object.ArrayPolar (NumberOfObjects, AngleToFill, CenterPoint)

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

Number Of Objects

Integer; input-only

The number of objects to be created in the polar array. This must be a positive integer greater than 1.

AngleToFill

Double; input-only

The angle to fill in radians. A positive value specifies

counterclockwise rotation. A negative value

specifies clockwise rotation. An error is returned for

an angle that equals 0.

CenterPoint

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the center

point for the polar array.

RetVal

Variant Array (array of objects) The array of new objects.

Remarks

AutoCAD determines the distance from the array's center point to a reference point on the last object selected. The reference point used depends on the type of object previously selected. AutoCAD uses the center point of a circle or arc, the insertion point of a block or shape, the start point of text, and one endpoint of a line or trace.



Polar array with NumberOfObjects = 5, AngleToFill = 180, CenterPoint = 0,0,0.

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

ArrayRectangular Method

Creates a 2D or 3D rectangular array of objects.

See Also | Example

Signature

RetVal = object.ArrayRectangular (NumberOfRows, NumberOfColumns, NumberOfLevels, DistBetweenRows, DistBetweenColumns, DistBetweenLevels)

Object

AecBlockRef, AecCamera, AecCellLayoutTool,
AecClipVol, AecClipVolRes, AecEntity, AecEntRef,
AecGeo, AecGridAssembly, AecLayoutCurve,
AecLayoutGrid2D, AecLayoutGrid3D,
AecLayoutTool, AecMaskBlockRef,
AecMassElement, AecMassGroup,
AecMVBlockRef, AecPolygon, AecSlice,
AecVolumeLayoutTool
The object or objects this method applies to.

NumberOfRows

Integer; input-only

The number of rows in the rectangular array. This must be a positive number. If this number is 1, then NumberOfColumns must be greater than 1.

NumberOfColumns

Integer; input-only

The number of columns in the rectangular array. This must be a positive number. If this number is 1, then NumberOfRows must be greater than 1.

NumberOfLevels

Integer; input-only

The number of levels in a 3D array.

DistBetweenRows

Double; input-only

The distance between the rows. If the distance between rows is a positive number, rows are added upward from the base entity. If the distance is a negative number, rows are added downward.

DistBetweenColumns

Double; input-only

The distance between the columns. If the distance between columns is a positive number, columns are added to the right of the base entity. If the distance is a negative number, columns are added to the left.

DistBetweenLevels

Double; input-only

The distance between the array levels. If the distance between levels is a positive number, levels are added in the positive direction from the base entity. If the distance is a negative number, levels are added in the negative direction.

RetVal

Variant Array (array of objects)
The array of newly created objects.

Remarks

For a 2D array, specify the NumberOfRows, NumberOfColumns, DistBetweenRow, and DistBetweenColumns. For creating a 3D array, specify the NumberOfLevels and DistBetweenLevels as well.

A rectangular array is constructed by replicating the object in the selection set the appropriate number of times. If you define one row, you must specify more than one column and vice versa.

The object in the selection set is assumed to be in the lower left-hand corner, and the array is generated up and to the right. If the distance between rows is a negative number, rows are added downward. If the

distance between columns is a negative number, the columns are added to the left.

AutoCAD builds the rectangular array along a baseline defined by the current snap rotation angle.



Rectangular array with NumberOfRows = 4, NumberOfColumns = 3, DistBetweenRows = a, DistBetweenColumns = b. The base entity is represented in blue.

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

AttachAnchor Method

Attaches an anchor to an AEC Entity.

See Also | Example

Signature

object.AttachAnchor anchor

object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntRef, AecGeo,

AecGridAssembly, AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this method applies to.

anchor

AecAnchor; The anchor that will be attached This can be any of the AEC Anchor objects.

AttachEntity Method

Anchors an AEC entity to an AutoCAD or AEC entity.

See Also | Example

Signature

object.AttachEntity entity

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToLayoutCell, AecAnchorEntToLayoutNode, AecAnchorEntToLayoutVolume, AecAnchorExtendedTagToEnt

AecAnchorLeadEntToNode, AecAnchorToRef

The object this method applies to.

entity

AecEntity; input-only

The AEC entity to be attached to the anchored

object.

Remarks

Although <u>AecAnchorEntToGridAssembly</u> inherits the AttachEntity method from AecAnchor, it is disabled.

Clear Method

Clears AecRings and AecProfiles of segments or rings.

See Also | Example

Signature

object.Clear()
 object

AecRing, AecProfile
The object to be cleared.

ClosestNode Method

Returns the index of the node closest to a given point.

See Also | Example

Signature

RetVal = object.ClosestNode(Point)

Object

AecCellLayoutTool, AecClipVol, AecGridAssembly,

AecLayoutCurve, AecLayoutGrid2D,

AecLayoutGrid3D, AecLayoutTool, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this method applies to.

Point

Variant (three-element array of doubles); input-only

A 3D point near a layout node.

RetVal

Long

The index of the closest node.

Remarks

The node index is also returned by the Node property of AecAnchorEntToLayoutNode.

Contains Method

Tests to see if a ring is inside another ring.

See Also | Example

Signature

RetVal = object.Contains(Ring)

object

AecRing The object this method applies to.

Ring

AecRing; input-only The AecRing to check.

RetVal

Boolean

TRUE: The ring is inside. FALSE: The ring is not inside.

ConvertToCurrentAreaDisplay Method

Converts an area value using the LinearUnit and AreaDisplayUnit settings.

See Also | Example

Signature

RetVal = object.ConvertToCurrentAreaDisplay(Area)

object

DatabasePreferences The object this method

applies to.

Area

Double; input-only The area to convert.

RetVal

Double

The resulting area.

ConvertToCurrentVolumeDisplay Method

Converts a volume value using the LinearUnit and VolumeDisplayUnit settings.

See Also | Example

Signature

RetVal = object.ConvertToCurrentVolumeDisplay(Volume)

object

DatabasePreferences The object this method

applies to.

Volume

Double; input-only

The volume to convert.

RetVal

Double

The resulting volume.

Copy Method

Duplicates the given object to the same location.

See Also | Example

Signature

RetVal = object.Copy

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

AecVolumeLayoutTool The object to be copied.

RetVal

Object

The newly created duplicate object.

Remarks

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

CopyFrom Method

Copies AecRings and AecProfiles.

See Also | Example

Signature

object.CopyFrom SourceObject

object

AecProfile, AecRing

The object being copied to.

SourceObject

The source object to be copied; input only

When copying a Profile, the source object must be an AecProfile object.

When copying a Ring, the source object must be an <u>AecRing</u> object.

CopyObjects Method

Duplicates multiple objects (deep cloning).

See Also | Example

Signature

RetVal = object.CopyObjects(Objects[, Owner][, IDPairs])

Object

AecBaseDatabase, AecDatabase

The object or objects this method applies to.

Objects

Variant (array of objects); input-only

The array of primary objects to be copied. All the objects must have the same owner, and the owner must belong to the database or document that is

calling this method.

Owner

Variant (a single object); input-only; optional

The new owner for the copied objects. If no owner is specified, the objects will be created with the same

owner as the objects in the Objects array.

IDPairs

Variant (array of IDPair objects); input-output;

optional

Information on what happened during the copy and

translation process. Input: an empty variant.

Output: an array of IDPair objects.

RetVal

Variant (array of objects)

An array of newly created duplicate objects. Only

primary objects are returned in this array. For more information on what occured during the CopyObjects operation, or a list of objects owned by primary objects that were also copied, consult the IDPairs array.

Remarks

To copy objects to another open drawing, set the Owner parameter to the other drawing's model space.

During the CopyObjects operation, objects that are owned or referenced by the primary objects in the Objects parameter will also be copied.

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

Delete Method

Deletes the specified object.

See Also | Example

Signature

object.Delete

object

The object to be deleted.

ExpandKey Method

Calculates and returns the Layer Name that would be generated given the supplied 'Key'.

See Also | Example

Signature

RetVal = object.ExpandKey(key)

object

AecLayerKeyStyle The object this method applies

to.

key

String; input-only The key to convert.

RetVal

String; Layer Name

Remarks

If the Key is not valid, an empty string is returned.

ExportFreeForm Method

Exports a free form representation to a file.

See Also | Example

Signatures: Overview

Mass Elements

object.ExportFreeForm("FileName"[,

CenterAtOrigin])

Mass Groups

object.ExportFreeForm("FileName")

Signatures: Detail

object. ExportFreeForm ("FileName" [, CenterAtOrigin])

object

<u>AecMassElement</u> The object being exported.

FileName

String; input-only

The name of the file to export the free form

representation to.

CenterAtOrigin

Boolean; optional

Determines whether the original coordinates of the

boundary representation are stored, or the

coordinates are translated to WCS.

TRUE: Use the original coordinates of the boundary representation.

FALSE: Translate the vertices of the boundary representation to WCS.

The default is TRUE.

object.ExportFreeForm("FileName")

object

AecMassGroup

The object being exported.

FileName

String; input-only

The name of the file to export the free form

representation to.

Remarks

If no path is specified for *FileName*, the file is created in the current drawing's directory. If you do not specify a file extension, none is added.

FromPoints Method

Makes a ring from 2D Points.

See Also | Example

Signature

object.FromPoints(points)

object

AecRing The object this method applies to.

points

Variant(n-element array of doubles); input-only

List of 2D points.

Remarks

The input for this method takes a variant array of doubles representing 2D points.

FromPolyline Method

Makes a ring from a 2D Polyline.

See Also | Example

Signature

object.FromPolyline(polyline)

object

AecRing The object this method applies to.

polyline

AcadLWPolyline; input-only

2D polyline to copy the graphics from.

GenerateLayer Method

Generate the layer record, given a Layer Key.

See Also | Example

Signature

RetVal = object.GenerateLayer(Key)

object

AecLayerKeyStyle The object this method applies

to.

Key

String; input-only

RetVal

<u>AcadLayer</u>

The AcadLayer will be created if necessary, using the settings from the Layer Standard for Name,

Color, Linetype, etc.

GetAnchor Method

Returns the anchor attached to an object.

See Also | Example

Signature

RetVal = object.GetAnchor()

object

<u>AecBlockRef</u>, <u>AecCamera</u>, <u>AecCellLayoutTool</u>, <u>AecClipVol</u>, <u>AecClipVolRes</u>, <u>AecEntRef</u>, <u>AecGeo</u>,

AecGridAssembly, AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object the anchor is attached to.

RetVal

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

<u>AecAnchorEntToLayoutCell</u>, <u>AecAnchorEntToLayoutNode</u>,

<u>AecAnchorEntToLayoutVolume</u>,

AecAnchorLeadEntToNode, AecAnchorToRef The

anchor attached to the object.

Remarks

Use an AecAnchor object as a generic return argument. (The example code for this method demonstrates how to access an anchor attached to any AEC object.)

GetBoundingBox Method

Gets two points of a box enclosing the specified object.

See Also | Example

Signature

object.GetBoundingBox MinPoint, MaxPoint

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

MinPoint

Variant (three-element array of doubles); output-only The 3D WCS coordinates specifying the minimum point of the object's bounding box.

MaxPoint

Variant (three-element array of doubles); output-only The 3D WCS coordinates specifying the maximum point of the object's bounding box.

Remarks

The corners are returned in WCS coordinates with the box edges parallel to the WCS X, Y, and Z axes.

MaxPoint

MinPoint

GetExtensionDictionary Method

Gets the extension dictionary associated with an object.

See Also | Example

Signature

RetVal = object.GetExtensionDictionary

Object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

<u>AecAnchorEntToLayoutNode</u>,

AecAnchorEntToLayoutVolume,

AecAnchorExtendedTagToEnt

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

<u>AecLayerKeyStyles</u>, <u>AecLayoutCurve</u>,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

RetVal

<u>Dictionary</u>; output-only

The extension dictionary for the object.

Remarks

If an object does not have an extension dictionary, this method will create a new extension dictionary for that object and return it in the return value.

You can query an object to see if it has an extension dictionary by using the <u>HasExtensionDictionary</u> property.

GetXData Method

Gets the extended data (XData) associated with an object.

See Also | Example

Signature

object.GetXData AppName, XDataType, XDataValue

Object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

AecAnchorEntToLayoutVolume,

AecAnchorLeadEntToNode.

AecAnchorExtendedTagToEnt AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

<u>AecLayoutTool</u>, <u>AecMaskBlockRef</u>,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

AppName

String; input-only

A NULL string will return all the data attached to the object, regardless of the application that created it.

Supplying an application name will return only the data that was created by the specified application.

XDataType

Variant (array of shorts); output-only

XDataValue

Variant (array of variants); output-only

Remarks

Extended data can be used to store instance-specific data created by custom applications. This data can be added to any object. This data follows the object's definition data, and is maintained in the order that it was saved into the document. (AutoCAD maintains this information, but doesn't use it.)

HandleToObject Method

Gets the object that corresponds to the given handle.

See Also | Example

Signature

RetVal = object.HandleToObject(Handle)

Object

AecBaseDatabase, AecDatabase

The object or objects this method applies to.

Handle

String; input-only

The handle of the object to return.

RetVal

Object

The object that corresponds to the given handle.

Remarks

This method can only return objects in the current document.

Has Method

Determines if an object is a member of the specified collection.

See Also | Example

Signature

RetVal = object.Has("Name")

object

<u>AecLayerKeyStyles</u>, <u>AecMaskBlockStyles</u>, <u>AecMVBlockStyles</u>, <u>AecPolygonStyles</u>,

AecProfileStyles, AecDisplayRepresentations

The collection this method applies to.

Name

String; input-only

The name of the object to check for.

RetVal

Boolean

TRUE: Collection contains the object.

FALSE: Collection does not contain the object.

Highlight Method

Sets the highlight status for the given object, or for all objects in a given selection set.

See Also | Example

Signature

object.Highlight HighlightFlag

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

HighlightFlag

Boolean; input-only

TRUE: The object is highlighted.

FALSE: The existing highlight is removed from the object.

Remarks

Once the highlight flag for an object has been set, a call to the <u>Update</u> or <u>Regen</u> method is required to view the change.

Note that this function does not return the current highlight status of an object.

ImportFreeForm Method

Imports a free form representation from a file.

See Also | Example

Signature

object.ImportFreeForm("FileName"[, CenterAtLocation])

object

<u>AecMassElement</u> The object being imported to. This must be a free form mass element (<u>Type</u> = aecMassElementTypeFreeForm).

FileName

String; input-only The name of the file to import the free form representation from.

CenterAtLocation

Boolean; optional.

Controls whether the the vertices of the imported boundary representation are translated to the insertion point of the mass element.

TRUE: Translate the boundary representation to the

mass element insertion point.

FALSE: Use the coordinates saved with the

exported boundary representation.

The default is FALSE.

Remarks

If no path is specified for *FileName*, the current drawing's directory is searched.

Init Method

Initialization function which must be called before the AEC database object can be accessed

See Also | Example

Signature

object.Init(AcadDatabaseObject)

object

<u>AecBaseDatabase</u>, <u>AecBaseDatabasePreferences</u> The object this method applies to.

A cad Database Object

AcadDatabase; input-only The database.

Intersect Method

Boolean Intersects an AecProfile with the current AecProfile.

See Also | Example

Signature

object.Intersect(Profile)

object

AecProfile The object this method applies to.

Profile

AecProfile; input-only

The profile to intersect with the current profile.

IntersectWith Method

Gets the points where one object intersects another object in the drawing.

See Also | Example

Signature

RetVal = object.IntersectWith(IntersectObject, ExtendOption)

Object

<u>AecBlockRef</u>, <u>AecCamera</u>, <u>AecCellLayoutTool</u>, <u>AecClipVol</u>, <u>AecClipVolRes</u>, <u>AecEntity</u>, <u>AecEntRef</u>,

AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

IntersectObject

Object, input-only;

This can be any drawing entity object.

Extend Option

AcExtendOption enum; input-only

This option specifies if one or the other, both, or none of the entities are to be extended in order to

attempt an intersection.

acExtendNone Does not extend either

object.

acExtendThisEntity Extends the base object.

acExtendOtherEntity

Extends the object passed

as an argument.

acExtendBoth

Extends both objects.

RetVal

Variant (array of doubles)
The array of points where one object intersects another object in the drawing.

Remarks

If the two objects do not intersect, no data is returned. You can request the point of intersection that would occur if one or both of the objects were extended to meet the other. For example, suppose a drawing contains the three lines shown in the illustration below. Line1 is the base object from which this method was called and line3 is the object passed as a parameter. If the ExtendOption passed is acExtendThisEntity, point A is returned as the point where line1 would intersect line3 if line1 were extended. If the ExtendOption is acExtendOtherEntity, no data is returned because even if line3 were extended, it would not intersect line1.

If the intersection type is acExtendBothEntities and line2 is passed as the parameter entity, point B is returned. If the ExtendOption is acExtendNone and line2 is the parameter entity, no data is returned.

line3

line2

line1

Item Method

Returns a member object of a collection.

See Also | Example

Signature

RetVal = object.Item(Index)

object

All Collections, AecDisplayConfigurations, AecDisplaySets, AecDisplayRepresentations,

AecDisplayComponents,

AecMVBlockDisplayRepDefs, AecViewBlockDefs,

<u>AecsConnectionNodes</u>

The collection this method applies to.

Index

Variant; input-only

The index location in the collection for the member

item to query.

The index must be either an integer or a string. If an integer, the index must be between 0 and N-1, where N is the number of objects in the collection.

RetVal

Object;

The object at the given index location in the collection.

Remarks

This method supports case-sensitive string based iteration. For example, you could reference the Multi-View Block definition named "Fixture1" through the following statement:

Set mvBlock = cMvBlocks.Item("Fixture1")

Mirror Method

Creates a mirror image copy of a planar object around an axis.

See Also | Example

Signature

RetVal = object.Mirror(Point1, Point2)

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

Point1

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the first point of

the mirror axis.

Point2

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the second

point of the mirror axis.

RetVal

Mirrored object

This can be any drawing entity object.

Remarks

The two points specified as parameters become the endpoints of a line around which the base object is reflected.

Selected object

Point1 and Point2 specifying the mirror axis

Mirrored object

This method places the reflected image into the drawing and retains the original object. To remove the original object, use the Delete method.

You can mirror a Viewport object in paper space, although doing so has no affect on its model space view or on model space objects.

AutoCAD checks to see if the object to be copied owns any other object. If it does, it performs a copy on those objects as well. The process continues until all owned objects have been copied.

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

Mirror3D Method

Creates a mirror image of the given object about a plane.

See Also | Example

Signature

RetVal = object.Mirror3D(Point1, Point2, Point3)

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayoutCurve,

<u>AecLayoutGrid2D</u>, <u>AecLayoutGrid3D</u>, <u>AecLayoutTool</u>, <u>AecMaskBlockRef</u>, <u>AecMassElement</u>, <u>AecMassGroup</u>,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

Point1

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the first point of

the mirror plane.

Point2

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the second

point of the mirror plane.

Point3

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the third point

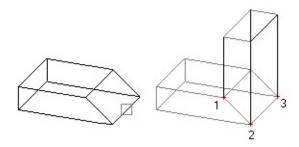
of the mirror plane.

RetVal

Mirrored object

This object can be any drawing entity object.

Remarks



Object mirrored about a plane defined by three points

AutoCAD checks to see if the object to be copied owns any other object. If it does, it performs a copy on those objects as well. The process continues until all owned objects have been copied.

NOTE You cannot execute this method while simultaneously iterating through a collection. An iteration will open the work space for a read-only operation, while this method attempts to perform a read-write operation. Complete any iteration before you call this method.

Move Method

Moves an object along a vector.

See Also | Example

Signature

object.Move Point1, Point2

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

Point1

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the first point of the move vector.

Point2

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the second point of the move vector.

Remarks

The two points you specify define a displacement vector indicating how far the given object is to be moved and in what direction.

Given object with two points indicated Moved object

NodeLocation Method

Returns the location of the node at a given index.

See Also | Example

Signature

RetVal = object.NodeLocation(Node)

Object

AecCellLayoutTool, AecClipVol, AecGridAssembly,

AecLayoutCurve, AecLayoutGrid2D,

AecLayoutGrid3D, AecLayoutTool, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this method applies to.

Node

Long; input-only Index of the node.

RetVal

Variant (three-element array of doubles)

The location of the node.

Remarks

The node index is returned by the Node property of AecAnchorEntToLayoutNode.

ObjectIDToObject Method

Gets the object that corresponds to the given object ID.

See Also | Example

Signature

RetVal = object.ObjectIDToObject(ID)

Object

AecBaseDatabase, AecDatabase

The object or objects this method applies to.

ID

Long; input-only

The object ID of the object to return.

RetVal

Object

The object that corresponds to the given object ID.

Remarks

This method can only return objects in the current document.

ReleaseAnchor Method

Removes an anchor from an object.

See Also | Example

Signature

object.ReleaseAnchor()

object

AecBlockRef, AecCamera, AecCellLayoutTool,
AecClipVol, AecClipVolRes, AecEntRef, AecGeo,
AecGridAssembly, AecLayoutCurve,
AecLayoutGrid2D, AecLayoutGrid3D,
AecLayoutTool, AecMaskBlockRef,
AecMassElement, AecMassGroup,
AecMVBlockRef, AecPolygon, AecSlice,
AecVolumeLayoutTool

The object this method applies to.

Remove Method

Removes an item from a collection.

See Also | Example

Signatures: Overview

AecDictionary, Layer Key Styles, Mask Block Styles, Mass Element Styles, Multi-view Block Styles, Polygon Styles, Profile Styles

collection.Remove(Name)

Layer Keys, Rings

collection.Remove(Index)

ViewBlockDefs

collection.Remove(AcadBlock)

Signatures: Detail

object.Remove (Name)

object

<u>AecDictionary</u>, <u>AecLayerKeyStyles</u>, <u>AecMaskBlockStyles</u>, <u>AecMassElementStyles</u>, <u>AecMVBlockStyles</u>, <u>AecPolygonStyles</u>, AecProfileStyles, AecDisplayRepresentations,

AecViewBlockDefs

The collection this method applies to.

Name

String; input-only

The name of the item to be removed.

object.Remove (Index)

object

AecLayerKeys, AecRings

The collection this method applies to.

Index

Long; input-only

The index number value of the item to remove from

the collection.

object. Remove (A cad Block)

object

AecViewBlockDefs

The collection you are removing a block from.

AcadBlock

AcadBlock object; input-only

The block to remove from the collection.

Rename Method

Renames an item in an AecDictionary.

See Also | Example

Signature

RetVal = object.Rename(OrigName, NewName)

Object

AecDictionary, AecLayerKeyStyles,

AecMaskBlockStyles, AecMassElementStyles,

AecMVBlockStyles, AecPolygonStyles,

AecProfileStyles

The object this method applies to.

OrigName

String; input-only

The original name of the item.

NewName

String; input-only

The new name of the item.

Remarks

Objects will maintain their reference to a style even if it has been renamed.

Rotate Method

Rotates an object around a base point.

See Also | Example

Signature

object.Rotate BasePoint, RotationAngle

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

BasePoint

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the point through which the axis of rotation is defined as parallel to the *Z* axis of the UCS.

RotationAngle

Double; input-only

The angle in radians to rotate the object. This angle determines how far an object rotates around the base point relative to its current location.

Remarks

Rotate3D Method

Rotates an object around a 3D axis. Point1 and Point2 define the line that becomes the axis of rotation.

See Also | Example

Signature

object.Rotate3D Point1, Point2, RotationAngle

Object

<u>AecBlockRef</u>, <u>AecCamera</u>, <u>AecCellLayoutTool</u>, <u>AecClipVol</u>, <u>AecClipVolRes</u>, <u>AecEntity</u>, <u>AecEntRef</u>,

AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

Point1

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the first point of

the axis line.

Point2

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the second

point of the axis line.

RotationAngle

Double; input-only

The angle in radians to rotate the object about the

selected axis.

Remarks

SaveAsDefault Method

Saves the DatabasePreferences to the registry.

See Also | Example

Signature

object.SaveAsDefault()

object

<u>AecBaseDatabasePreferences</u> The object this method applies to.

ScaleEntity Method

Scales an object equally in the X, Y, and Z directions.

See Also | Example

Signature

object.ScaleEntity BasePoint, ScaleFactor

Object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

BasePoint

Variant (three-element array of doubles); input-only The 3D WCS coordinates specifying the base point.

ScaleFactor

Double; input-only

The factor by which to scale the object. The dimensions of the object are multiplied by the scale factor. A scale factor greater than 1 enlarges the object. A scale factor between 0 and 1 reduces the object. The scale factor must be greater than 0.0.

Remarks



ScaleFactor = 0.5 and ScaleFactor = 2

ScaleXY Method

Scales a profile in the X and Y directions.

See Also | Example

Signature

object.ScaleXY (X, Y)

object

AecProfile The object this method applies to.

X

Double; input-only

The X factor to scale by.

Y

Double; input-only

The Y factor to scale by.

SetXData Method

Sets the extended data (XData) associated with an object.

See Also | Example

Signature

object.SetXData XDataType, XData

Object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

AecAnchorEntToLayoutVolume,

AecAnchorExtendedTagToEnt

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

AecPolygonStyles, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this method applies to.

XDataType

Variant (array of short); input-only

See Extended Data in the *DXF Reference* for the extended data group codes to specify for this argument.

XData

Array of Variant; input-only

Remarks

Extended data can be used for storing instance-specific data created by custom applications. This data can be added to any entity. This data follows the entity's definition data and is maintained in the order in which it was saved into the document. (AutoCAD maintains this information but doesn't use it.)

Subtract Method

Subtracts an AecProfile from the current AecProfile.

See Also | Example

Signature

object.Subtract(Profile)

object

AecProfile The object this method applies to.

Profile

AecProfile; input-only The profile to subtract.

TransformBy Method

Moves, scales, or rotates an object given a 4x4 transformation matrix.

See Also | Example

Signature

object.TransformBy TransformationMatrix

Object

AecBlockRef, AecCamera, AecCellLayoutTool,
AecClipVol, AecClipVolRes, AecEntity, AecEntRef,
AecGeo, AecGridAssembly, AecLayoutCurve,
AecLayoutGrid2D, AecLayoutGrid3D,
AecLayoutTool, AecMaskBlockRef,
AecMassElement, AecMassGroup,
AecMVBlockRef, AecPolygon, AecSlice,
AecVolumeLayoutTool
The object or objects this method applies to.

TransformationMatrix

Variant (4x4 array of doubles); input-only A 4x4 matrix specifying the transformation to perform.

Remarks

The following table demonstrates the transformation matrix configuration, where R = Rotation, and T = Translation:

R00 R01 R02 T0

R10 R11 R12 T1

R20 R21 R22 T2

0 0 0 1

This method will return an error if the transformation matrix is not correct.

Sample transformation matrices are provided in the example code for this method.

Update Method

Updates the object in the drawing window.

See Also | Example

Signature

object.Update

Object

AecBlockRef, AecCamera, AecCellLayoutTool,
AecClipVol, AecClipVolRes, AecEntity, AecEntRef,
AecGeo, AecGridAssembly, AecLayoutCurve,
AecLayoutGrid2D, AecLayoutGrid3D,
AecLayoutTool, AecMaskBlockRef,
AecMassElement, AecMassGroup,
AecMVBlockRef, AecPolygon, AecSlice,
AecVolumeLayoutTool
The object or objects this method applies to.

Properties

| image |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| image |
| image | image | image | image | image | image | | | | |

Α

<u>AdjustSizing</u> Specifies if the size of the anchored object is adjusted to match the size of the grid cell.

<u>AllowLongSymbolNames</u> Determines if symbol names may include extended character sets, or more than 31 characters.

AllowVariation Specifies if the anchored object is allowed to vary.

AlternateName The AlternateName of the Style.

<u>AlwaysImportLayerStandard</u> Specifies whether to always import a newer layer standard.

Angle Specifies the angle of the layout grid.

<u>Angle1</u> Deviation of cut plane at starting edge where zero is perpendicular to the clip line.

<u>Angle2</u> Deviation of cut plane at ending edge where zero is perpendicular to the clip Line.

<u>AngleFromNode</u> Specifies the angle from the anchored entity to the node.

AngularAzimuth The format used to display angles.

<u>AngularDisplayFormat</u> Specifies the unit used when angular measurements are displayed.

Angular Precision Specifies the angular precision for the drawing.

<u>Application</u> Gets the Application object.

<u>ApplyResize</u> Specifies if the anchored object is resized when the layout cell is resized.

Area Gets the area of an object.

AreaDisplayUnit The unit used when area measurements are displayed.

<u>AreaPrecision</u> The precision to which area measurements are displayed.

AreaSuffix The suffix displayed after an area unit in the drawing.

В

BasePoint Gets and sets the base point for the drawing as an XYZ.

BasePointNE Gets and sets the base point for the drawing as an ENZ.

BayAngle Specifies the bay angle of the layout grid.

<u>BlockBasedLayerOffBehavior</u> Immediate and independent layer on/off control of display components.

Blocks Gets the Blocks collection for the drawing.

<u>BottomOffset</u> Specifies the bottom offset of the anchored object to the grid it is attached to.

Boundary Specifies the boundary object attached to the grid.

Bounds Gets the upper left and lower right corners of the profile.

C

Cell Specifies the grid cell to anchor the entity to.

<u>CenterOnNode</u> Specifies if the anchored object is centered on the layout node.

<u>Centroid</u> Gets the center of area or mass for a profile shape.

Color Specifies the object's color.

<u>ContourlinesPerSurface</u> Specifies the number of contour lines (isolines) per surface on objects.

<u>CoordinatePrecision</u> Specifies the precision to which coordinate measurements are displayed.

Coordinates Returns the vertices of a ring or profile.

Count Gets the number of items in the collection.

<u>CreateDimscaleOverride</u> Automatically create a dimscale override.

D

<u>Database</u> Gets the database in which the object belongs.

<u>DatabaseScale</u> Specifies the database scale at which the horizontal axis of the drawing is displayed.

<u>Depth</u> Gets or sets the size of the grid or mass element in its relative Y direction.

Description Gets or sets text describing the AEC Object.

Dictionaries Gets the Dictionaries collection for the document.

DimStyles Gets the DimStyles collection for the document.

<u>DisplaySilhouette</u> Controls if silhouette curves of solid objects are displayed in Wireframe mode.

Document Gets the document (drawing) in which the object belongs.

Ε

EdgeWidth Specifies the edge width of the polygon style.

<u>EdgeWidthJustify</u> Specifies whether edges are displayed inside, outside, or centered on the AEC Polygon vertices.

<u>ElevationModelSpace</u> Specifies the elevation setting in the model space.

<u>ElevationPaperSpace</u> Specifies the elevation setting in the paper space.

<u>ElevationPrecision</u> Specifies the precision to which elevation measurements are displayed in the drawing.

<u>EndOffset</u> Specifies the offset distance from the end point of the curve, which may be used to reduce the effective layout curve length.

F

<u>FacetDeviation</u> Specifies the allowed deviation when facetting curved lines and surfaces.

<u>FacetMaximum</u> Specifies the maximum number of facets used for a circle.

<u>FlipX</u> Determines if the anchored object is flipped about the X axis of the curve it is anchored to.

<u>FlipY</u> Determines if the anchored object is flipped about the Y axis of the curve it is anchored to.

<u>FlipZ</u> Determines if the anchored object is flipped about the Z axis of the curve it is anchored to.

G

Groups Gets the Groups collection for the document.

Н

<u>Handle</u> Gets the handle of an object.

<u>HasExtensionDictionary</u> Determines if the object has an extension dictionary associated with it.

<u>Height</u> Specifies the height of the object.

Hyperlinks Gets the Hyperlinks collection for an entity.

<u>Index</u> Returns the position of the item within its collection.

<u>InsertionPoint</u> Specifies the insertion point of the profile.

<u>InsideRadius</u> Specifies the inside radius of the layout grid.

J

K

Keys Returns the collection of layer keys in a layer key style.

L

<u>Layer</u> Specifies the layer the object is on.

<u>LayerFile</u> Specifies the name of the file containing default layer standard and key style definitions.

LayerKeyStyle Specifies the layer key style for the object.

<u>LayerKeyStyles</u> Returns the collection of layer key styles in the current drawing.

<u>Layers</u> Gets the Layers collection for the document.

<u>LayerStandard</u> The layer standard to apply to the drawing.

<u>LayoutCurve</u> Provides access to layout curve geometry.

<u>LayoutGrid2D</u> Identifies the 2D layout grid that owns the object.

<u>LayoutGrid3D</u> Identifies the 3D layout grid that owns the object.

<u>Layouts</u> Gets the Layouts collection for the document.

<u>LeaderExtension1</u> Specifies the first leader extension.

<u>LeaderExtension2</u> Specifies the second leader extension.

<u>LeftOffset</u> Specifies the left offset for the anchored object.

Limits Specifies the drawing limits.

<u>LinearDisplayFormat</u> Specifies the format used to display linear measurement values.

<u>LinearPrecision</u> Specifies the linear precision for the drawing.

<u>LinearUnit</u> Specifies the unit used to display linear values.

<u>LineType</u> Specifies the linetype of the object.

<u>Linetypes</u> Gets the Linetypes collection for the document.

<u>LinetypeScale</u> Specifies the linetype scale of an entity.

<u>LineWeight</u> Specifies the lineweight of the object.

<u>LineweightDisplay</u> Specifies whether lineweights are displayed in model space for the drawing.

<u>Location</u> Specifies the insertion position of the object.

<u>LowerExtension</u> Specifies the distance of the lower cut plane below the clip volume position.

M

<u>MaskBlockStyles</u> Returns the collection of mask block styles in the specified drawing document or database.

<u>MassElementStyles</u> Returns the collection of mass element styles in the specified drawing database.

MassGroup Gets or sets the parent mass group of a mass element or mass group.

<u>MassGroupName</u> Specifies the mass group that the mass element or group is currently attached to.

MassGroups Returns the collection of mass groups in a document or database.

MaxActiveViewports Specifies the maximum number of active viewports.

MeasurementUnit Identifies the system of units used to display measurements in the drawing.

ModelSpace Gets the ModelSpace collection for the document.

MVBlockRef Gets or sets a multi-view block reference for a view block or collection of view blocks.

<u>MVBlockStyles</u> Returns the collection of multi-view block styles in the specified document or database.

Ν

Name Returns the name of the object.

Node Specifies the layout node the anchor attaches an object to.

Nodes Returns the collection of nodes on the layout curve.

Normal Specifies the normal vector of the AEC object.

NorthRotation Specifies the angle aligned to north within a drawing.

0

ObjectID Gets the object ID of the object.

ObjectName Gets the AutoCAD class name of the object.

ObjectSortByPlotting Toggles sorting of drawing objects by plotting order.

<u>ObjectSortByPSOutput</u> Toggles sorting of drawing objects by PostScript output order.

<u>ObjectSortByRedraws</u> Toggles sorting of drawing objects by redraw order.

<u>ObjectSortByRegens</u> Toggles sorting of drawing objects by regeneration order.

<u>ObjectSortBySelection</u> Toggles sorting of drawing objects by object selection.

ObjectSortBySnap Toggles sorting of drawing objects by object snap.

Offset Specifies the offset from the layout node.

<u>OLELaunch</u> Determines whether to launch the parent application when plotting OLE objects.

Operation Specifies how objects are combined.

<u>OverridesEnabled</u> Specifies whether layer overrides are applied to the style when a layer is generated from a key.

<u>OverrideSettings</u> Returns the collection of layer key override settings for a layer key style.

OwnerID Gets the object ID of the owner (parent) object.

P

<u>PaperSpace</u> Gets the PaperSpace collection for the document.

<u>PlotConfigurations</u> Gets the PlotConfigurations collection for the document.

PlotStyleName Specifies the plotstyle name for the object.

Plottable Specifies whether the layer generated from the key is plottable.

<u>PolygonStyles</u> Returns the collection of polygon styles in the specified drawing database.

<u>Preferences</u> Returns the Preferences object for the specified drawing.

<u>Profile</u> Returns or sets the object's profile.

<u>ProfileStyle</u> Specifies the profile style of the mass element.

ProfileStyleName Specifies the name of the mass element's profile style.

<u>ProfileStyles</u> Returns the collection of profiles in the specified drawing database.

<u>ProjectName</u> Specifies the name of the project that the drawing belongs to.

Q

R

Radius Specifies the radius of curved mass elements.

Reference Specifies the object being anchored to.

RegisteredApplications Gets the RegisteredApplications collection for the document.

Removeable Specifies whether you can remove the layer key from the layer key style.

<u>RenderSmoothness</u> Specifies the smoothness of shaded, rendered, and hidden line-removed objects.

ResizeOffset Specifies the offset of the anchored object to the cell boundary when the cell is resized.

RightOffset Specifies the right offset for the anchored object.

Rings Returns the collection of rings that make up the profile.

Rise Specifies the rise of gabled mass elements.

Rotation Specifies the rotation reference angle for the object.

S

ScaleOnInsert Specifies whether objects are automatically scaled when

inserted into the drawing.

<u>ScaleX</u> Specifies the X direction scale factor of the inserted block or entity.

<u>ScaleY</u> Specifies the Y direction scale factor of the inserted block or entity.

<u>ScaleZ</u> Specifies the Z direction scale factor of the inserted block or entity.

<u>SegmentPerPolyline</u> Specifies the number of line segments to be generated for each polyline curve.

<u>SelfIntersects</u> Determines if the profile is self-intersecting.

Shape Specifies the shape of the layout grid.

<u>Side1</u> Specifies the length of the cut plane along the starting edge.

<u>Side2</u> Specifies the length of the cut plane along the ending edge.

<u>SolidFill</u> Specifies if multilines, traces, solids, all hatches (including solid-fill) and wide polylines are filled in.

<u>Spacing</u> Specifies the spacing between nodes or for an individual node on a layout tool.

<u>StartOffset</u> Specifies the offset distance from the start point of the curve, which may be used to reduce the effective layout curve length.

Style Specifies the style of the object.

<u>StyleName</u> Specifies the name of the style that applies to the object.

Т

<u>TextFrameDisplay</u> Toggles the display of frames for text objects instead of displaying the text itself.

<u>TextHeight</u> Specifies the default height of text in the drawing.

<u>TextStyles</u> Gets the TextStyles collection for the document.

<u>TopOffset</u> Specify the top offset for the anchored object.

<u>TrueColor</u> Returns the true color of the object.

Type For layout curves, specifies how nodes are specified along the curve; for mass elements, specifies the shape of the mass element.

U

<u>UseModelExtents</u> Determines if the height and lower extension are set automatically using model extents.

<u>UseNodeCS</u> Specifies whether to use the coordinate system of the layout node.

<u>UserCoordinateSystems</u> Gets the UCSs collection for the document.

V

Valid Tests the validity of the object.

<u>Value</u> For layer overrides, specifies the value of the layer key style override; for layout curves, specifies either the spacing between nodes or the number of nodes on the layout curve.

<u>VerticalScale</u> Specifies the scale at which the vertical axis of the drawing is displayed (in Land Development Desktop only).

<u>ViewBlocks</u> Returns the collection of view blocks used for display representations by the multi-view block reference.

<u>Viewports</u> Gets the Viewports collection for the document.

<u>Views</u> Gets the Views collection for the document.

<u>Visible</u> Specifies whether the object or application is visible.

<u>Void</u> Specifies if a ring produces a void in a profile.

<u>Volume</u> Returns the volume of the mass element or mass group.

<u>VolumeDisplayUnit</u> Specifies the units used to display volume measurements.

<u>VolumePrecision</u> Specifies the number of decimal places displayed for volume measurements.

<u>VolumeSuffix</u> The suffix string added when volume measurements are displayed in the drawing.

W

Width Specifies the width of the object.

WindowState Specifies the state of the application or document window.

WindowTitle Gets the title of the document window.

X

XCount Specifies the number of nodes along the X direction.

XDistance Specifies the distance along the curve's X axis to the anchored object.

<u>XEndOffset</u> Specifies the offset distance from the end point along the X direction grid curve, which may be used to reduce the grid size.

XNodes Returns the collection of nodes along the X direction of the layout grid.

XOffset Specifies the offset in the X direction of the view block within the multi-view block.

XPositionFrom Specifies the position on the curve from which the XDistance to an anchored object is measured.

<u>XPositionTo</u> Specifies the position on the anchored object to measure XDistance to.

XRefEdit Determines if the current drawing can be edited in place when being referenced by another user.

XRefLayerVisibility Determines the visibility of xref-dependent layers and specifies if nested xref path changes are saved.

<u>XrefOverlaysUseOwnDisplayConfig</u> When used as an Xref Overlay, use this drawing's default display configuration initially.

XRotation Specifies the rotation of the anchored object around the X axis.

XSpacing Specifies the spacing between nodes along the X direction of the layout grid.

XStartOffset Specifies the offset distance from the start point along the X direction grid curve, which may be used to reduce the grid size.

XType Specifies the rule controlling node spacing along the X direction.

Υ

<u>YAlignment</u> Specifies the alignment of the anchored object in relation to the baseline of the assembly.

YCount Specifies the number of nodes along the y direction.

<u>YDistance</u> Specifies the distance along the curve's Y axis to the anchored object.

<u>YEndOffset</u> Specifies the offset distance from the end point along the Y direction grid curve, which may be used to reduce the grid size.

<u>YNodes</u> Returns the collection of nodes along the Y direction of the layout grid.

YOffset Specifies the Y offset of the anchored object, or the offset in the Y direction of the view block within the multi-view block.

<u>YPositionFrom</u> Specifies the position on the curve from which the YDistance to an anchored object is measured.

<u>YPositionTo</u> Specifies the position on the anchored object to measure YDistance to.

YRotation Specifies the rotation of the anchored object around the layout node's Y axis.

YSpacing Specifies the spacing between nodes along the Y direction of the layout grid.

<u>YStartOffset</u> Specifies the offset distance from the start point along the Y direction grid curve, which may be used to reduce the grid size.

YType Specifies the rule controlling node spacing along the Y direction.

Ζ

ZCount Specifies the number of nodes along the Z direction.

ZDistance Specifies the distance above the curve to the anchored object.

<u>ZEndOffset</u> Specifies the offset distance from the end point along the Z direction grid curve, which may be used to reduce the grid size.

<u>ZNodes</u> Returns the collection of nodes along the Z direction of the 3D layout grid.

<u>ZOffset</u> Specifies the offset in the Z direction of the view block within the multi-view block.

Zoom Specifies the focal length of the camera, in millimeters.

<u>ZPositionFrom</u> Specifies the position on the curve from which the ZDistance to an anchored object is measured.

<u>ZPositionTo</u> Specifies the position on the anchored object to measure ZDistance to.

ZRotation Specifies the rotation of the anchored object around the Z axis.

ZSpacing Specifies the spacing between nodes along the Z direction of the layout grid.

<u>ZStartOffset</u> Specifies the offset distance from the start point along the Z direction grid curve, which may be used to reduce the grid size.

<u>ZType</u> Specifies the rule controlling node spacing along the Z direction.

ActiveSet Property

Returns the active display set for the specified view port.

See Also | Example

Signature

RetVal = object.ActiveSet(ViewPort)

object

AecDisplaySets The object this property applies to.

ViewPort

AcadObject; input-only

The AcadViewport or AcadPViewport that the

display configuration applies to.

RetVal

AecDisplaySet

The active display set for the specified viewport.

AdjustSizing Property

Specifies whether you can override the default offset values of the anchored object, thus changing its size.

See Also | Example

Signature

object.AdjustSizing

object

AecAnchorEntToGridAssembly The object this

property applies to.

AdjustSizing

Boolean; read-write

TRUE: Offsets can be modified. FALSE: Offsets cannot be modified.

AllowByMaterial Property

Determines if the ByMaterial property should be applied.

See Also | Example

Signature

object.AllowByMaterial

object

AecDisplayComponentEntity The object this

property applies to.

AllowByMaterial

Boolean; read-write

Remarks

This is an advanced setting that should rarely need to be changed.

AllowVariation Property

Specifies if the anchored object is allowed to vary.

See Also | Example

Signature

object.AllowVariation

object

AecAnchorEntToGridAssembly The object this

property applies to.

AllowVariation

Boolean; read-write TRUE: Allow variation.

FALSE: Don't allow variation.

AlternateName Property

The AlternateName of the Style.

See Also | Example

Signature

object.AlternateName

object

AecMaskBlockStyle, AecLayerKeyStyle, AecDictRecord, AecMVBlockStyle, AecPolygonStyle, AecProfileStyle

The object this property applies to.

AlternateName

String; read-write

The AlternateName of the Style.

AlwaysImportLayerStandard Property

Specifies whether to always import a newer layer standard.

See Also | Example

Signature

object.AlwaysImportLayerStandard

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

AlwaysImportLayerStandard

Boolean; read-write Specifies whether to always import a newer layer standard.

Angle Property

Specifies the angle of the layout grid, or the angle of the hatch pattern for a display component.

See Also | Example

Signature

object.Angle

object

AecDisplayComponentHatch, AecLayoutGrid2D

The object this property applies to.

Angle

ACAD_ANGLE; read-write

The angle of the layout grid, or the angle of the

hatch pattern for a display component.

Angle1 Property

Deviation of cut plane at starting edge where zero is perpendicular to the clip line.

See Also | Example

Signature

object.Angle1

object

AecClipVol The object this property applies to.

Angle1

Double (radians); read-write

Remarks

The angle is specified in radians.

Angle2 Property

Deviation of cut plane at ending edge where zero is perpendicular to the clip Line.

See Also | Example

Signature

object.Angle2

object

AecClipVol The object this property applies to.

Angle2

Double (radians); read-write

Remarks

The angle is specified in radians.

AngleFromNode Property

Specifies the angle from the anchored entity to the node.

See Also | Example

Signature

object.AngleFromNode

object

<u>AecAnchorLeadEntToNode</u> The object this property

applies to.

AngleFromNode

Double; read-write

The angle, in radians, from the anchored entity to

the node.

Angular Azimuth Property

The format used to display angles.

See Also | Example

Signature

object.AngularAzimuth

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

AngularAzimuth

Long; read-write

0: Use Bearings

1: Use North Azimuths

2: Use South Azimuths

Angular Display Format Property

Specifies the unit used when angular measurements are displayed.

See Also | Example

Signature

object.AngularDisplayFormat

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

AngularDisplayFormat

Long; read-write

- 0: Use Degrees
- 1: Use Degrees/Minutes/Seconds
- 2: Use Grads
- 3: Use Radians
- 4: Use Surveyor

Remarks

When you type degrees at AutoCAD command prompts, type them in the decimal format indicated (DD.MMSS). Use a period (.) between the degree value and the minutes and seconds. For example, to enter 67°45'15" type the value as 67.4515. If you use bearings, then type the bearing quadrant first, and then the angle in degrees.

AngularPrecision Property

Specifies the angular precision for the drawing.

See Also | Example

Signature

object.AngularPrecision

object

<u>AecBaseDatabasePreferences</u> The object this

property applies to.

AngularPrecision

Long; read-write

The number of decimals to the right of the decimal

point.

Remarks

This is used to display minutes and seconds.

The precision settings are used only for labeling and listing values, not actual computations. AutoCAD commands always calculate all numbers up to the highest internal precision.

Application Property

Gets the Application object.

See Also | Example

Signature

object.Application

object

All objects.

The object or objects this property applies to.

Application

Application object; read-only

Remarks

The Application object represents the application's frame controls and path settings, and provides the means to navigate down the object hierarchy.

ApplyResize Property

Specifies if the anchored object is resized when the layout cell is resized.

See Also | Example

Signature

object.ApplyResize

object

AecAnchorEntToLayoutVolume,

<u>AecAnchorToLayoutCell</u>

The object this property applies to.

ApplyResize

Boolean; read-write TRUE: Apply resize.

FALSE: Don't apply resize.

Remarks

If you want to set the <u>ResizeOffset</u> property, you must first set ApplyResize to True in order for the specified offset to take affect.

Area Property

Gets the area of an object.

See Also | Example

Signature

object.Area

object

AecProfile, AecSlice

The object this property applies to.

Area

Double; read-only

The area enclosed by the object.

AreaDisplayUnit Property

The unit used when area measurements are displayed.

See Also | Example

Signature

object.AreaDisplayUnit

object

AecBaseDatabasePreferences The object this

property applies to.

AreaDisplayUnit

AecBuiltInUnit enum; read-write

AreaPrecision Property

The precision to which area measurements are displayed.

See Also | Example

Signature

object.AreaPrecision

object

AecBaseDatabasePreferences The object this

property applies to.

AreaPrecision

Long; read-write

The display precision used for areas.

Remarks

The precision settings are used only for labeling and listing values, not actual computations.

AreaSuffix Property

The suffix displayed after an area unit in the drawing.

See Also | Example

Signature

object.AreaSuffix

object

AecBaseDatabasePreferences The object this

property applies to.

AreaSuffix

String; read-write

The suffix displayed following the area unit.

Remarks

The default value is "Sq.Ft." or "m2", depending on the value of the MeasurementUnit property. The value must be no longer than 255 characters in length.

BasePoint Property

Gets and sets the base point for the drawing as an XYZ.

See Also | Example

Signature

object.BasePoint

object

AecBaseDatabasePreferences The object this

property applies to.

BasePoint

Variant (3 element array of doubles); read-write

The base point in WCS X, Y, Z.

Remarks

Set this value to indicate the WCS X, Y, Z that corresponds to the value of the <u>BasePointNE</u> property. In the default coordinate system X,Y is 0,0 and the Northing/Easting is 0,0.

BasePointNE Property

Gets and sets the base point for the drawing as an ENZ.

See Also | Example

Signature

object.BasePointNE

object

AecBaseDatabasePreferences The object this

property applies to.

BasePointNE

Variant (3 element array of doubles); read-write

The base point in WCS X, Y, Z.

Remarks

Set this value to indicate the Easting, Northing and elevation that corresponds to the value of the <u>BasePoint</u> property. In the default coordinate system, X,Y is 0,0 and the Northing/Easting is 0,0.

BayAngle Property

Specifies the bay angle of the layout grid.

See Also | Example

Signature

object.BayAngle

object

<u>AecLayoutGrid2D</u> The object this property applies

to.

BayAngle

ACAD_ANGLE; read-write

The bay angle of the layout grid.

Block Property

Gets or sets the AcadBlock for the view.

See Also | Example

Signature

object.Block

object

AecViewBlockDef The object this property applies

to.

Block

AcadBlock; read-write

BlockBasedLayerOffBehavior Property

Immediate and independent layer on/off control of display components.

See Also | Example

Signature

object.BlockBasedLayerOffBehavior

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

Block Based Layer Off Behavior

Boolean; read-write Immediate and independent layer on/off control of display components.

Blocks Property

Gets the Blocks collection for the drawing.

See Also | Example

Signature

object.Blocks

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Blocks

Blocks collection; read-only

The Blocks collection for the drawing.

BottomOffset Property

Specifies the bottom offset of the anchored object.

See Also | Example

Signature

object.BottomOffset

object

<u>AecAnchorEntToGridAssembly</u> The anchor object

this property applies to.

BottomOffset

Double; read-write

The bottom offset of the anchored object. (See

Remarks.)

Remarks

To change the BottomOffset value, you must first set <u>AdjustSizing</u> to True. Modifying BottomOffset when AdjustSizing is False does not result in an error or warning, but it has no effect.

Boundary Property

Specifies the boundary object attached to the grid.

See Also | Example

Signature

object.Boundary

object

AecLayoutGrid2D The object this property applies

to.

Boundary

AcadEntity; read-write

The boundary object attached to the grid.

Bounds Property

Gets the upper left and lower right corners of the profile.

See Also | Example

Signature

object.Bounds

object

AecProfile The object this property applies to.

Bounds

Variant (array of 4 doubles); read-only Top left / Bottom right corner of profile.

ByMaterial Property

Specifies if the display properties are determined by the material assigned to the display component.

See Also | Example

Signature

object.ByMaterial

object

AecDisplayComponentEntity The object this

property applies to.

ByMaterial

Boolean; read-write

Cell Property

Specifies the grid cell to anchor the entity to.

See Also | Example

Signature

object.Cell

object

AecAnchorEntToGridAssembly The object this

property applies to.

Cell

Long; read-write

The index of the grid cell the object is anchored to.

CenterOnNode Property

Specifies if the anchored object is centered on the layout node.

See Also | Example

Signature

object.CenterOnNode

object

AecAnchorEntToLayoutVolume, AecAnchorLeadEntToNode, AecAnchorEntToLayoutNode, AecAnchorEntToLayoutCell

The object this property applies to.

CenterOnNode

Boolean; read-write TRUE: Center on node.

FALSE: Don't center on node.

Centroid Property

Gets the center of area or mass for a profile shape.

See Also | Example

Signature

object.Centroid

object

AecProfile The object this property applies to.

Centroid

Variant (two-element array of doubles); read-only

Centroid point of the profile.

ClassName Property

The class name of the object.

See Also | Example

Signature

object.ClassName

object

AecDisplayRepresentation The object this property

applies to.

ClassName

String; read-write

ClipVol Property

Specifies the clip volume of the 2d section.

See Also | Example

Signature

object.ClipVol

object

Aec2dSection The object this property applies to.

ClipVol

AecClipVol; read-write

ContourLinesPerSurface Property

Specifies the number of contour lines (isolines) per surface on objects.

See Also | Example

Signature

object.ContourLinesPerSurface

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ContourLinesPerSurface

Integer; read-write The valid range is 0 to 2047.

System variables

The value of this property is stored in the **ISOLINES** system variable.

Remarks

The initial value for this property is 4.



A torus with ContourLinesPerSurface set to 4 A torus with ContourLinesPerSurface set to 20

CoordinatePrecision Property

Specifies the precision to which coordinate measurements are displayed.

See Also | Example

Signature

object.CoordinatePrecision

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

CoordinatePrecision

Long; read-write

The number of decimals to the right of the decimal point.

Remarks

This is used to display all Northing/Easting coordinate information.

The precision settings are used only for labeling and listing values, not actual computations.

Coordinates Property

Returns the vertices of a ring or profile.

See Also | Example

Signature

object.Coordinates

object

AecRing, AecProfile

The object this property applies to.

Coordinates

Variant (n-element array of doubles);

read-only

List of 2D points.

The vertices of this ring or profile.

Count Property

Gets the number of items in a collection, or the number of nodes in a layout curve.

See Also | Example

Signature

object.Count

object

All Collections, AecLayoutCurve, AecDisplayRepresentations, AecDisplayComponents,

AecMVBlockDisplayRepDefs, AecViewBlockDefs,

<u>AecsConnectionNodes</u>

The object this property applies to.

Count

Long; read-only

The number of items.

CreateDimscaleOverride Property

Automatically create a dimscale override.

See Also | Example

Signature

object.CreateDimscaleOverride

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

CreateDimscaleOverride

Boolean; read-write Automatically create a dimscale override.

CutPlaneDisplayAbove Property

Gets or sets the upper range above the cut plane to display objects.

See Also | Example

Signature

object. Cut Plane Display Above

object

<u>AecDisplayConfiguration</u> The object this property applies to.

CutPlaneDisplayAbove

Double; read-write

CutPlaneDisplayBelow Property

Gets or sets the lower range below the cut plane to display objects.

See Also | Example

Signature

object. Cut Plane Display Below

object

<u>AecDisplayConfiguration</u> The object this property applies to.

CutPlaneDisplayBelow

Double; read-write

CutPlaneHeight Property

Gets or sets the height of the cut plane.

See Also | Example

Signature

object.CutPlaneHeight

object

AecDisplayConfiguration The object this property

applies to.

CutPlaneHeight

Double; read-write

CutProfile Property

Returns the cut profile for the object.

See Also | Example

Signature

object.CutProfile

object

AecMaskBlockRef The object this property applies

to.

CutProfile

AecProfile; read-only

Database Property

Gets the database in which the object belongs.

See Also | Example

Signature

object.Database

object

<u>AecBaseDatabasePreferences</u>, <u>AecMassGroups</u>

The object or objects this property applies to.

Database

AcadDatabase object; read-only

The Database object that contains the object.

DatabaseScale Property

Specifies the database scale at which the horizontal axis of the drawing is displayed.

See Also | Example

Signature

object.DatabaseScale

object

<u>AecBaseDatabasePreferences</u> The object this

property applies to.

DatabaseScale

Double; read-write

The database (horizontal) scale. For example, for a

1:200 scale, type 200.

Remarks

The database (horizontal) scale controls the size of annotation placed in the drawing, including text, scaled blocks and special lines. It does not affect the line lengths or point coordinates because they are always defined in real world coordinates, not to any scale. Neither does it affect any design data in the drawing or project files.

If you change the DatabaseScale property in the middle of a drawing session, then any annotation added subsequently is scaled accordingly.

Depth Property

Gets or sets the size of the grid or mass element in its relative Y direction.

See Also | Example

Signature

object.Depth

object

AecLayoutGrid2D, AecLayoutGrid3D,

AecMassElement

The object this property applies to.

Depth

Double; read-write

The size of the object in its relative Y direction; see

Remarks.

Remarks

For radial 2D layout grids, Depth is an angle, in radians.

DefaultDisplayProperties Property

The collection of default display properties for the display representation.

See Also | Example

Signature

object. Default Display Properties

object

<u>AecDisplayRepresentation</u> The object this property applies to.

DefaultDisplayProperties

AecDisplayProperties; read-write

Description Property

Gets or sets text describing the AEC Object.

See Also | Example

Signature

object.Description

object

<u>AecAnchorExtendedTagToEnt</u> <u>AecObject</u> The object

this property applies to.

Description

String; read-write

Description of the AEC Object.

Deviation Property

Specifies the deviation of the mass element.

See Also | Example

Signature

object.Deviation

object

AecMassElement The object this property applies

to.

Deviation

Double; read-write

Dictionaries Property

Gets the Dictionaries collection for the document.

See Also | Example

Signature

object.Dictionaries

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Dictionaries

Dictionaries collection; read-only

The Dictionaries collection for the document.

DimStyles Property

Gets the DimStyles collection for the document.

See Also | Example

Signature

object.DimStyles

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

DimStyles

AcadDimStyles collection; read-only

The DimStyles collection for the document.

DisplayComponents Property

Gets the collection of display components for the display representation.

See Also | Example

Signature

object.DisplayComponents

object

<u>AecDisplayProperties</u> The object this property applies to.

DisplayComponents

AecDisplayComponents; read-write

Remarks

The display components collection contains AecDisplayComponentEntity and AecDisplayComponentHatch objects.

DisplayConfigurations Property

Returns the display configurations collection.

See Also | Example

Signature

object.DisplayConfigurations

object

<u>AecBaseDatabase</u>, <u>AecBaseDocument</u> The object this property applies to.

DisplayConfigurations

AecDisplayConfigurations; read-write

DisplayHiddenLinework Property

Specifies whether hidden linework is displayed.

See Also | Example

Signature

object. Display Hidden Linework

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

DisplayHiddenLinework

Boolean; read-write

DisplayName Property

The name displayed for the object.

See Also | Example

Signature

object.DisplayName

object

AecDisplayRepresentation, AecMVBlockDisplayRepDef

The object this property applies to.

DisplayName

String; read-write

DisplayRepDefs Property

Returns the display representation definitions for the multi-view block style.

See Also | Example

Signature

object.DisplayRepDefs

object

<u>AecMVBlockStyle</u> The object this property applies to.

MVBlock Display Rep Defs

AecMVBlockDisplayRepDefs; read-write

DisplayRepresentations Property

Gets the display representations collection.

See Also | Example

Signature

object.DisplayRepresentations

object

AecDisplaySet, AecBaseDatabase, AecBaseDocument
The object this property applies to.

Display Representations

AecDisplayRepresentations; read-write

Remarks

You can append to or remove from the DisplayRepresentations collection when accessed from the DisplaySet.

DisplaySet Property

Gets or sets the the display set for a given view direction.

See Also | Example

Signature

RetVal = object.DisplaySet(ViewDir)

object

AecDisplayConfiguration The object this property

applies to.

ViewDir

AecViewDirection; input-only

The view direction.

RetVal

AecDisplaySet

The display set that applies used for the view

direction.

DisplaySets Property

Returns the collection of display sets.

See Also | Example

Signature

object.DisplaySets

object

AecBaseDatabase, AecBaseDocument

The object this property applies to.

DisplaySets

AecDisplaySets; read-write

DisplaySilhouette Property

Controls if silhouette curves of solid objects are displayed in Wireframe mode.

See Also | Example

Signature

object.DisplaySilhouette

object

AecBaseDatabasePreferences The object or

objects this property applies to.

DisplaySilhouette

Boolean; read-write

TRUE: Display only silhouette lines for solid objects.

FALSE: Display isolines for solid objects.

System variables

The value of this property is stored in the **DISPSILH** system variable.

Remarks

The initial value for this property is FALSE.

Document Property

Gets the document (drawing) in which the object belongs.

See Also | Example

Signature

object.Document

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

<u>AecAnchorEntToLayoutNode</u>,

AecAnchorEntToLayoutVolume,

<u>AecAnchorExtendedTagToEnt</u>

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

Document

Document object; read-only

The document (drawing) that contains the object.

DoubleHatch Property

Specifies if double hatching is used.

See Also | Example

Signature

object.DoubleHatch

object

AecDisplayComponentHatch The object this

property applies to.

DoubleHatch

Boolean; read-write

Remarks

The DoubleHatch property applies when the HatchType is set to aecHatchTypeUserDefined.

EdgeWidth Property

Specifies the edge width of the polygon style.

See Also | Example

Signature

object.EdgeWidth

object

<u>AecPolygonStyle</u> The object this property applies to.

EdgeWidth

Double; read-write

The edge width of the polygon style.

EdgeWidthJustify Property

Specifies whether edges are displayed inside, outside, or centered on the AEC Polygon vertices.

See Also | Example

Signature

object.EdgeWidthJustify

object

AecPolygonStyle The object this property applies to.

EdgeWidthJustify

AecPolygonStyleEdgeWidthJustification enum;

read-write

The edge justification of the polygon style.

ElevationModelSpace Property

Specifies the elevation setting in the model space.

See Also | Example

Signature

object. Elevation Model Space

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

ElevationModelSpace

Double; read-write The elevation setting for model space.

Remarks

The current elevation is the Z value that is used whenever a 3D point is expected but only the X and Y values are supplied. The current elevation is maintained separately in model space and paper space.

ElevationPaperSpace Property

Specifies the elevation setting in the paper space.

See Also | Example

Signature

object.ElevationPaperSpace

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

ElevationPaperSpace

Double; read-write The elevation for paper space.

Remarks

The current elevation is the Z value that is used whenever a 3D point is expected but only the X and Y values are supplied. The current elevation is maintained separately in model space and paper space.

ElevationPrecision Property

Specifies the precision to which elevation measurements are displayed in the drawing.

See Also | Example

Signature

object.ElevationPrecision

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

ElevationPrecision

Long; read-write

The number of decimals to the right of the decimal

point.

Remarks

The precision settings are used only for labeling and listing values, not actual computations.

EndOffset Property

Specifies the offset distance from the end point of the curve, which may be used to reduce the effective layout curve length.

See Also | Example

Signature

object.EndOffset

object

AecLayoutCurve The object this property applies to.

EndOffset

Double; read-write

Offset distance from end point of the curve.

ExcludeFrom2dSectionShrinkwrap Property

Specifies whether the material is excluded from the shrinkwrap of a 2D section.

See Also | Example

Signature

object. Exclude From 2d Section Shrinkwrap

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

ExcludeFrom2dSectionShrinkwrap

Boolean; read-write

FacetDeviation Property

Specifies the allowed deviation when facetting curved lines and surfaces.

See Also | Example

Signature

object.FacetDeviation

object

AecBaseDatabasePreferences The object this

property applies to.

FacetDeviation

Double; read-write

The facet deviation of AEC 3D Objects.

System variables

The value of this property is stored in the FACETDEV system variable.

Remarks

Facet deviation is similar to FacetRes for ACIS bodies.

FacetMaximum Property

Specifies the maximum number of facets used for a circle.

See Also | Example

Signature

object.FacetMaximum

object

AecBaseDatabasePreferences The object this

property applies to.

FacetMaximum

Integer; read-write

The maximum number of facets used for a circle.

FixedViewDirection Property

Gets or sets a fixed view direction for a display configuration.

See Also | Example

Signature

object.FixedViewDirection

object

<u>AecDisplayConfiguration</u> The object this property applies to.

FixedViewDirection

AecFixedViewDirection; read-write

FlipX Property

Determines if the anchored object is flipped about the X axis of the curve it is anchored to.

See Also | Example

Signature

object.FlipX

object

AecAnchorEntToCurve,

AecAnchorEntToGridAssembly, AecAnchorEntToLayoutCell, AecAnchorEntToLayoutNode, AecAnchorEntToLayoutVolume,

<u>AecAnchorLeadEntToNode</u>

The object this property applies to.

FlipX

Boolean; read-write

TRUE: Anchored object is flipped around the X axis. FALSE: Anchored object is not flipped around the X

axis.

FlipY Property

Determines if the anchored object is flipped about the Y axis of the curve it is anchored to.

See Also | Example

Signature

object.FlipY

object

AecAnchorEntToCurve,

AecAnchorEntToGridAssembly, AecAnchorEntToLayoutCell, AecAnchorEntToLayoutNode,

<u>AecAnchorEntToLayoutVolume</u>, <u>AecAnchorLeadEntToNode</u>

The object this property applies to.

FlipY

Boolean; read-write

TRUE: Anchored object is flipped around the Y axis. FALSE: Anchored object is not flipped around the Y

axis.

FlipZ Property

Determines if the anchored object is flipped about the Z axis of the curve it is anchored to.

See Also | Example

Signature

object.FlipZ

object

AecAnchorEntToCurve,

<u>AecAnchorEntToGridAssembly</u>, <u>AecAnchorEntToLayoutCell</u>,

AecAnchorEntToLayoutNode,

<u>AecAnchorEntToLayoutVolume</u>,

<u>AecAnchorLeadEntToNode</u>

The object this property applies to.

FlipZ

Boolean; read-write

TRUE: Anchored object is flipped around the Z axis. FALSE: Anchored object is not flipped around the Z

axis.

Groups Property

Gets the Groups collection for the document.

See Also | Example

Signature

object.Groups

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Groups

AcadGroups collection; read-only

The Groups collection for the document.

Handle Property

Gets the handle of an object.

See Also | Example

Signature

object.Handle

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

<u>AecAnchorEntToLayoutNode</u>,

<u>AecAnchorEntToLayoutVolume</u>,

<u>AecAnchorExtendedTagToEnt</u>

<u>AecAnchorLeadEntToNode</u>, <u>AecAnchorToRef</u>,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

<u>AecDictRecord</u>, <u>AecEntity</u>, <u>AecEntRef</u>, <u>AecGeo</u>,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

<u>AecMassElement</u>, <u>AecMassGroup</u>,

AecMVBlockRef, AecMVBlockStyle,

<u>AecMVBlockStyles</u>, <u>AecObject</u>, <u>AecPolygon</u>,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

Handle

String; read-only

The handle of the entity.

Remarks

An object ID and a unique handle are the two ways of referencing an object. A handle is persistent (stays the same) in a drawing for the lifetime of the object.

In general, use a handle unless you plan to work with certain ObjectARX functions that require an object ID.

HasExtensionDictionary Property

Determines if the object has an extension dictionary associated with it.

See Also | Example

Signature

object.HasExtensionDictionary

Object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

<u>AecAnchorEntToLayoutVolume</u>,

AecAnchorExtendedTagToEnt

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

<u>AecPolygonStyle</u>, <u>AecPolygonStyles</u>,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

Has Extension Dictionary

Boolean; read-only

TRUE: The object has an extension dictionary associated

with it.

FALSE: The object does not have an extension dictionary associated with it.

Remarks

You can create an extension dictionary for an object, or query an existing extension dictionary by using the <u>GetExtensionDictionary</u> method.

HatchType Property

Gets or sets the hatch type for the display component.

See Also | Example

Signature

object.HatchType

object

AecDisplayComponentHatch The object this

property applies to.

HatchType

AecHatchType; read-write

Height Property

Specifies the height of the object.

See Also | Example

Signature

object.Height

object

AecClipVol, AecLayoutGrid3D, AecMassElement

The object this property applies to.

Height

Double; read-write

The relative height of the object.

Hyperlinks Property

Gets the Hyperlinks collection for an entity.

See Also | Example

Signature

object.Hyperlinks

object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

AecVolumeLayoutTool

The object or objects this property applies to.

Hyperlinks

AcadHyperlinks; read-only

The Hyperlinks collection for the entity.

Index Property

Returns the position of the item within its collection.

See Also | Example

Signature

object.Index

object

AecRing AecLayerOverrideSetting,

AecLayoutCurveNode, AecLayoutGrid2DNode, AecLayoutGrid2DXNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DXNode, AecLayoutGrid3DXNode, AecLayoutGrid3DZNode,

AecViewBlock, AecLayerKey,

AecDisplayComponent,

AecMVBlockDisplayRepDef, AecViewBlockDef,

<u>AecsConnectionNode</u>

The object this property applies to.

Index

Long, except for AecLayerOverrideSetting, which is

a String; read-only

The position of the object within its collection.

Remarks

AecLayerOverrideSetting is indexed randomly, by name.

InsertionPoint Property

Specifies the insertion point of the profile.

See Also | Example

Signature

object.InsertionPoint

object

AecEditInPlaceProfile The object this property

applies to.

InsertionPoint

Variant (three-element array of doubles); read-write

The insertion point of the profile.

InsideRadius Property

Specifies the inside radius of the layout grid.

See Also | Example

Signature

object.InsideRadius

object

AecLayoutGrid2D The object this property applies

to.

InsideRadius

Double; read-write

The inside radius of the layout grid.

InterferenceBlockName Property

Specifies the name of the interference block for the Multi-view block.

See Also | Example

Signature

object. Interference Block Name

object

<u>AecMVBlockStyle</u> The object this property applies to.

InterferenceBlockName

String; read-write

Keys Property

Returns the collection of layer keys in a layer key style.

See Also | Example

Signature

object.Keys

object

AecLayerKeyStyle The object this property applies

to.

Keys

AecLayerKeys; read-only

Returns the collection of AecLayerKeys.

Layer Property

Specifies the layer the object is on.

See Also | Example

Signature

object.Layer

object

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes,

AecDisplayComponentEntity, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayerKey,

AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement,

AecMassGroup, AecMVBlockRef, AecPolygon,

AecSlice, AecVolumeLayoutTool
The object this property applies to.

Layer

String; read-write

The name of the object's layer.

LayerFile Property

Specifies the name of the file containing default layer standard and key style definitions.

See Also | Example

Signature

object.LayerFile

object

AecBaseDatabasePreferences The object this

property applies to.

LayerFile

String; read-write

The name of the layer file.

Remarks

The layer file is stored as a global setting in the registry, and not on a perdrawing basis. An entry is maintained for each <u>MeasurementUnit</u>.

LayerKeyStyle Property

Specifies the layer key style for the object.

See Also | Example

Signature

object.LayerKeyStyle

object

AecLayerKey, AecLayerKeys, AecLayerOverrideSetting, AecLayerOverrideSettings

The object this property applies to.

LayerKeyStyle

AecLayerKeyStyle; read-write Layer key style of the object.

Layers Property

Gets the Layers collection for the document.

See Also | Example

Signature

object.Layers

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Layers

AcadLayers collection; read-only

The Layers collection for the document.

LayerStandard Property

The layer standard to apply to the drawing.

See Also | Example

Signature

object.LayerStandard

object

AecBaseDatabasePreferences The object this

property applies to.

LayerStandard

String; read-write

The name of the layer standard.

Remarks

The layer standards are stored on a per drawing basis in the dictionary.

LayoutCurve Property

Provides access to layout curve geometry.

See Also | Example

Signature

object.LayoutCurve

object

<u>AecLayoutCurveNode</u>, <u>AecLayoutCurveNodes</u>

The object this property applies to.

LayoutCurve

AecLayoutCurve; read-write

A layout curve object.

LayoutGrid2D Property

Identifies the 2D layout grid that owns the object.

See Also | Example

Signature

object.LayoutGrid2D

object

AecLayoutGrid2DNode, AecLayoutGrid2DNodes,

<u>AecLayoutGrid2DXNode</u>, <u>AecLayoutGrid2DXNodes</u>,

AecLayoutGrid2DYNode, AecLayoutGrid2DYNodes

The object this property applies to.

LayoutGrid2D

AecLayoutGrid2D; read-write

The layout grid that owns the layout grid node or

collection of nodes.

LayoutGrid3D Property

Identifies the 3D layout grid that owns the object.

See Also | Example

Signature

object.LayoutGrid3D

object

AecLayoutGrid3DNodes, AecLayoutGrid3DNodes,

AecLayoutGrid3DXNode, AecLayoutGrid3DXNodes, AecLayoutGrid3DYNode,

AecLayoutGrid3DYNodes, AecLayoutGrid3DZNode,

<u>AecLayoutGrid3DZNodes</u>

The object this property applies to.

LayoutGrid3D

AecLayoutGrid3D; read-write

The 3D layout grid that owns the layout grid node or

collection of nodes.

Layouts Property

Gets the Layouts collection for the document.

See Also | Example

Signature

object.Layouts

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Layouts

AcadLayouts object; read-only

The Layouts collection for the document.

LeaderExtension1 Property

Specifies the first leader extension.

See Also | Example

Signature

object. Leader Extension 1

object

<u>AecAnchorLeadEntToNode</u> The object this property

applies to.

LeaderExtension1

Double; read-write

The first leader extension.

LeaderExtension2 Property

Specifies the second leader extension.

See Also | Example

Signature

object. Leader Extension 2

object

<u>AecAnchorLeadEntToNode</u> The object this property

applies to.

LeaderExtension2

Double; read-write

The second leader extension.

LeftOffset Property

Specifies the left offset for the anchored object.

See Also | Example

Signature

object.LeftOffset

object

AecAnchorEntToGridAssembly The object this

property applies to.

LeftOffset

Double; read-write

The left offset for the anchored object. (See

Remarks.)

Remarks

To change the LeftOffset value, you must first set <u>AdjustSizing</u> to True. Modifying LeftOffset when AdjustSizing is False does not result in an error or warning, but it has no effect.

Limits Property

Specifies the drawing limits.

See Also | Example

Signature

object.Limits

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Limits

Variant (array of doubles); read-write An array of four values. The first pair of values define the *X* and *Y* coordinates of the lower-left limit, the second pair of values define the *X* and *Y* coordinates of the upper-right limit.

System variables

The lower-left limit controls the <u>LIMMIN</u> system variable. The upper-right limit controls the <u>LIMMAX</u> system variable. The <u>LIMCHECK</u> system variable turns limits checking on and off for the current space.

Remarks

The drawing limits are two-dimensional points in the World Coordinate System (WCS) that represent a lower-left and an upper-right limit. You cannot impose limits on the Z direction.

The drawing limits also govern the portion of the drawing covered by the visible grid and determine the minimum area that the <u>ZoomAll</u> method displays.

LinearDisplayFormat Property

Specifies the format used to display linear measurement values.

See Also | Example

Signature

object.LinearDisplayFormat

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

LinearDisplayFormat

Long; read-write The format used to display linear measurement values. Can be one of the following:

- 1: Scientific
- 2: Decimal
- 3: Engineering
- 4: Architectural
- 5: Fractional

LinearPrecision Property

Specifies the linear precision for the drawing.

See Also | Example

Signature

object.LinearPrecision

object

AecBaseDatabasePreferences The object this

property applies to.

LinearPrecision

Long; read-write

The number of decimals to the right of the decimal

point.

Remarks

The precision settings are used only for labeling and listing values, not actual computations.

LinearUnit Property

Specifies the unit used to display linear values.

See Also | Example

Signature

object.LinearUnit

object

AecBaseDatabasePreferences The object this

property applies to.

LinearUnits

AecBuiltInUnit enum; read-write

The unit used to display linear values. Specify one

of the following:

aecUnitInch: Inches

aecUnitFoot: Feet

aecUnitMillimeter: Millimeters

aecUnitCentimeter: Centimeters

aecUnitDecimeter: Decimeters

aecUnitMeter: Meters

Remarks

Use this property to specify whether you want to use feet or meters in your drawing.

The linear units that you select must be the same for all drawings in a project. You cannot mix foot-based drawings with meter-based drawings in the same project.

Linetype Property

Specifies the linetype of the object.

See Also | Example

Signature

object.Linetype

object

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes,

AecDisplayComponentEntity, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayerKey,

AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement,

AecMassGroup, AecMVBlockRef, AecPolygon,

AecSlice, AecVolumeLayoutTool
The object this property applies to.

Linetype

String; read-write

The name of the linetype of the object.

Linetypes Property

Gets the Linetypes collection for the document.

See Also | Example

Signature

object.Linetypes

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Linetypes

<u>AcadLineTypes</u> collection; read-only

The Linetypes collection for the document.

LinetypeScale Property

Specifies the linetype scale of an entity.

See Also | Example

Signature

object.LinetypeScale

object

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes,

AecDisplayComponentEntity, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

LinetypeScale

Double; read-write

This value must be a positive real number. The

default is 1.0.

Remarks

The linetype scale of an object specifies the relative length of dash-dot linetypes per drawing unit.

Linetype scale = 1.0

Linetype scale = 0.5

Linetype scale = 0.25

LineWeight Property

Specifies the lineweight of the object.

See Also | Example

Signature

object.LineWeight

object

AecBaseDatabasePreferences, AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecDatabasePreferences,

AecDisplayComponentEntity, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayerKey,

AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement,

AecMassGroup, AecMVBlockRef, AecPolygon,

AecSlice, AecVolumeLayoutTool
The object this property applies to.

Line Weight

AcLineWeight or ACAD_LWEIGHT enum; read-write

Lineweight of the object.

LineweightDisplay Property

Specifies whether lineweights are displayed in model space for the drawing.

See Also | Example

Signature

object.LineweightDisplay

object

<u>AecBaseDatabasePreferences</u> The object or

objects this property applies to.

LineweightDisplay

Boolean; read-write

TRUE: Lineweights are displayed in model space.

FALSE: Lineweights are not displayed in model space.

Remarks

The initial value for this property is TRUE.

Regeneration time increases with lineweights that are represented by more than one pixel. Set this property to FALSE if performance slows down when working with large lineweights.

Location Property

Specifies the insertion position of the object.

See Also | Example

Signature

object.Location

object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntRef, AecGeo,

AecGridAssembly, AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this property applies to.

Location

Variant (three-element array of doubles); read-write

Insertion position of the object.

LowerExtension Property

Specifies the distance of the lower cut plane below the clip volume position.

See Also | Example

Signature

object.LowerExtension

object

AecClipVol The object this property applies to.

LowerExtension

Double; read-write

Distance of the lower cut plane below clip volume

position.

MassElementStyles Property

Returns the collection of mass element styles in the specified drawing database.

See Also | Example

Signature

object.MassElementStyles

object

<u>AecBaseDatabase</u> The object this property applies to.

MassElementStyles

AecMassElementStyles; read-write The collection of mass element styles.

MassGroup Property

Gets or sets the parent mass group of a mass element or mass group.

See Also | Example

Signature

object.MassGroup

object

<u>AecMassElement</u>, <u>AecMassGroup</u> The object this property applies to.

MassGroup

AecMassGroup; read-write The parent mass group, if any.

MassGroupName Property

Specifies the mass group that the mass element or group is currently attached to.

See Also | Example

Signature

object.MassGroupName

object

<u>AecMassElement</u>, <u>AecMassGroup</u> The object this property applies to.

MassGroupName

String; read-write Specifies the mass group that the mass element or group is currently attached to.

MaxActiveViewports Property

Specifies the maximum number of active viewports.

See Also | Example

Signature

object.MaxActiveViewports

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

MaxActiveViewports

Integer; read-write
The maximum number of active viewports.
2 <= MaxActiveViewports <= 48

System variables

The value of this property is stored in the MAXACTVP system variable.

Remarks

Inactive viewports are blank, so their contents are not regenerated. You can, therefore, improve performance by specifying a low value. The initial value is 48.

MeasurementUnit Property

Identifies the system of units used to display measurements in the drawing.

See Also | Example

Signature

object.MeasurementUnit

object

AecBaseDatabasePreferences The object this

property applies to.

MeasurementUnit

AcMeasurementUnit enum; read-only
The system of units used to display measurements.

Can be one of the following:

acEnglish: English (Imperial) units

acMetric: Metric units

MergeCommonMaterials Property

Specifies whether common materials are merged into one component.

See Also | Example

Signature

object. Merge Common Materials

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

Merge Common Materials

Boolean; read-write

ModelSpace Property

Gets the ModelSpace collection for the document.

See Also | Example

Signature

object.ModelSpace

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

ModelSpace

AcadModelSpace collection; read-only

The ModelSpace collection for the document.

MVBlockRef Property

Gets or sets a multi-view block reference for a view block or collection of view blocks.

See Also | Example

Signature

object.MVBlockRef

object

AecViewBlock, AecViewBlocks

The object this property applies to.

MVBlockRef

AecMVBlockRef; read-write

Reference of a multi-view block inserted into a

drawing.

Name Property

Returns the name of the object.

See Also | Example

Signature

object.Name

object

<u>AecCamera</u>, <u>AecDictRecord</u>, <u>AecLayerKey</u>, <u>AecLayerKeyStyle</u>, <u>AecLayerOverrideSetting</u>,

<u>AecMaskBlockStyle</u>, <u>AecMassGroup</u>, <u>AecMVBlockStyle</u>, <u>AecPolygonStyle</u>,

AecProfileStyle, AecViewBlock,

<u>AecDisplayComponent</u>

The object this property applies to.

Name

String; read-only for all objects except AecCamera.

The name of the object.

Node Property

Specifies the layout node the anchor attaches an object to.

See Also | Example

Signature

object.Node

object

AecAnchorEntToLayoutNode,
AecAnchorLeadEntToNode,
AecAnchorEntToLayoutVolume,
AecAnchorEntToLayoutCell

The anchor object this property applies to.

Node

Long; read-write

Node entity is anchored to. The first node on the

grid is 1.

Nodes Property

Returns the collection of nodes on the layout curve.

See Also | Example

Signature

object.Nodes

object

AecLayoutCurve The object this property applies to.

Nodes

AecLayoutNodes; read-only

The collection of nodes on the curve.

NorthRotation Property

Specifies the angle aligned to north within a drawing.

See Also | Example

Signature

object.NorthRotation

object

AecBaseDatabasePreferences The object this

property applies to.

NorthRotation

Double; read-write

The north rotation angle. The angle is in radians and measured counter-clockwise from the X axis (East).

Remarks

The initial value for this property is 0.

If you change the NorthRotation in an existing drawing then you must move any objects in the drawing to match the new coordinate system.

ObjectID Property

Gets the object ID of the object.

See Also | Example

Signature

object.ObjectID

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

AecAnchorEntToLayoutVolume,

<u>AecAnchorExtendedTagToEnt</u>

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

<u>AecMVBlockStyles</u>, <u>AecObject</u>, <u>AecPolygon</u>,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

ObjectID

Long; read-only

The object ID of an entity.

Remarks

An object ID and a unique handle are both ways of referencing an object.

In general, use a handle unless you plan to work with certain ObjectARX functions that require an object ID.

ObjectName Property

Gets the AutoCAD class name of the object.

See Also | Example

Signature

object.ObjectName

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

<u>AecAnchorEntToLayoutNode</u>,

AecAnchorEntToLayoutVolume,

AecAnchorExtendedTagToEnt

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

<u>AecPolygonStyle</u>, <u>AecPolygonStyles</u>,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

ObjectName

String; read-only

The AutoCAD class name of an entity.

ObjectSortByPlotting Property

Toggles sorting of drawing objects by plotting order.

See Also | Example

Signature

object.ObjectSortByPlotting

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ObjectSortByPlotting

Boolean; read-write

TRUE: Enable sort by plotting order.

FALSE: Disable sort by plotting order.

System variables

The value of this property is stored in the **SORTENTS** system variable.

Remarks

The initial value of this property is TRUE.

ObjectSortByPSOutput Property

Toggles sorting of drawing objects by PostScript output order.

See Also | Example

Signature

object.ObjectSortByPSOutput

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ObjectSortByPSOutput

Boolean; read-write

TRUE: Enable sort by PostScript output order.

FALSE: Disable sort by PostScript output order.

System variables

The value of this property is stored in the <u>SORTENTS</u> system variable.

Remarks

The initial value of this property is TRUE.

ObjectSortByRedraws Property

Toggles sorting of drawing objects by redraw order.

See Also | Example

Signature

object.ObjectSortByRedraws

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ObjectSortByRedraws

Boolean; read-write

TRUE: Enable sort by redraw order.

FALSE: Disable sort by redraw order.

System variables

The value of this property is stored in the **SORTENTS** system variable.

Remarks

The initial value of this property is FALSE.

ObjectSortByRegens Property

Toggles sorting of drawing objects by regeneration order.

See Also | Example

Signature

object.ObjectSortByRegens

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ObjectSortByRegens

Boolean; read-write

TRUE: Enable sort by regeneration order.

FALSE: Disable sort by regeneration order.

System variables

The value of this property is stored in the **SORTENTS** system variable.

Remarks

The initial value of this property is FALSE.

ObjectSortBySelection Property

Toggles sorting of drawing objects by object selection.

See Also | Example

Signature

object.ObjectSortBySelection

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

ObjectSortBySelection

Boolean; read-write

TRUE: Enable sort by object selection.

FALSE: Disable sort by object selection.

System variables

The value of this property is stored in the <u>SORTENTS</u> system variable.

Remarks

The initial value of this property is FALSE.

ObjectSortBySnap Property

Toggles sorting of drawing objects by object snap.

See Also | Example

Signature

object. Object Sort By Snap

object

AecBaseDatabasePreferences The object or

objects this property applies to.

ObjectSortBySnap

Boolean; read-write

TRUE: Enable sort by object snap.

FALSE: Disable sort by object snap.

System variables

The value of this property is stored in the <u>SORTENTS</u> system variable.

Remarks

The initial value of this property is FALSE.

Offset Property

Specifies the offset from the object.

See Also | Example

Signature

object.Offset

object

AecAnchorLeadEntToNode, AecAnchorEntTolayoutCell, AecAnchorEntToLayoutNode,

AecAnchorEntToLayoutVolume, AecEntRef

The object this property applies to.

Offset

Variant; read-write

Offset from the layout node or entity reference.

OLELaunch Property

Determines whether to launch the parent application when plotting OLE objects.

See Also | Example

Signature

object.OLELaunch

object

<u>AecBaseDatabasePreferences</u> The object or

objects this property applies to.

OLELaunch

Boolean; read-write

TRUE: The parent application of an OLE object is

launched to print it.

FALSE: The OLE object is printed from the current

application.

System variables

The value of this property is stored in the **OLESTARTUP** system variable.

Remarks

The initial value of this property is FALSE.

Plotting from the parent application achieves a higher quality plot; however, the speed of the plot is decreased.

Operation Property

Specifies how objects are combined.

See Also | Example

Signature

object.Operation

object

<u>AecMassElement</u>, <u>AecMassGroup</u> The object this property applies to.

Operation

<u>AecMassOperation</u> enum; read-write

Boolean operation: addition, subtraction, or

intersection.

OverrideDisplayProperties Property

The collection of display property overrides for the display representation.

See Also | Example

Signature

object.OverrideDisplayProperties AecObject

object

<u>AecDisplayRepresentation</u> The object this property applies to.

OverrideDisplayProperties

AecDisplayProperties; read-write

AecObject

AecObject; input-only
An object instance or style override.

Remarks

Overrides can be applied by object or by style.

OverridesEnabled Property

Specifies whether layer overrides are applied to the style when a layer is generated from a key.

See Also | Example

Signature

object. Overrides Enabled

object

AecLayerKeyStyle The object this property applies

to.

OverridesEnabled

Boolean; read-write

TRUE: Overrides are applied.

FALSE: Overrides are not applied.

OverrideSettings Property

Returns the collection of layer key override settings for a layer key style.

See Also | Example

Signature

object.OverrideSettings

object

AecLayerKeyStyle The object this property applies

to.

OverrideSettings

<u>AecLayerOverrideSettings</u>; read-only

The collection of layer key override settings.

OwnerID Property

Gets the object ID of the owner (parent) object.

See Also | Example

Signature

object.OwnerID

object

AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

<u>AecAnchorEntToLayoutVolume</u>,

<u>AecAnchorExtendedTagToEnt</u>

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEntity, AecEntRef, AecGeo,

AecGridAssembly, AecLayerKeyStyle,

AecLayerKeyStyles, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D,

AecLayoutTool, AecMaskBlockRef,

AecMaskBlockStyle, AecMaskBlockStyles,

AecMassElement, AecMassGroup,

AecMVBlockRef, AecMVBlockStyle,

AecMVBlockStyles, AecObject, AecPolygon,

AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

The object or objects this property applies to.

OwnerID

Long; read-only

The object ID of an object's owner.

PaperSpace Property

Gets the PaperSpace collection for the document.

See Also | Example

Signature

object.PaperSpace

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

PaperSpace

<u>AcadPaperSpace</u> collection; read-only

The PaperSpace collection for the document.

PatternName Property

Gets or sets the name of the custom hatch pattern.

See Also | Example

Signature

object.PatternName

object

AecDisplayComponentHatch The object this

property applies to.

PatternName

String; read-write

Remarks

The PatternName applies when the HatchType is set to aecHatchTypeCustomDefined.

Perimeter Property

Specifies the perimeter of the profile.

See Also | Example

Signature

object.Perimeter

object

AecProfile The object this property applies to.

Perimeter

Double; read-only

PlaneDepth Property

Specifies the plane depth of the slice.

See Also | Example

Signature

object.PlaneDepth

object

AecSlice The object this property applies to.

PlaneDepth

Double; read-write

PlaneWidth Property

Specifies the plane width of the slice.

See Also | Example

Signature

object.PlaneWidth

object

AecSlice The object this property applies to.

PlaneWidth

Double; read-write

PlotConfigurations Property

Gets the PlotConfigurations collection for the document.

See Also | Example

Signature

object.PlotConfigurations

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

PlotConfigurations

<u>AcadPlotConfigurations</u> collection; read-only

The PlotConfigurations collection for the document.

PlotStyleName Property

Specifies the plotstyle name for the object.

See Also | Example

Signature

object.PlotStyleName

object

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes,

AecDisplayComponentEntity, AecEntity, AecEntRef,

AecGeo, AecGridAssembly, AecLayerKey,

AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement,

AecMassGroup, AecMVBlockRef, AecPolygon,

AecSlice, AecVolumeLayoutTool
The object this property applies to.

PlotStyleName

String; read-write

The name of the object's plotstyle.

Plottable Property

Specifies whether the layer generated from the key is plottable.

See Also | Example

Signature

object.Plottable

object

AecLayerKey The object this property applies to.

Plottable

Boolean; read-write

TRUE: Layer is plottable

FALSE: Layer is not plottable

PolygonStyles Property

Returns the collection of polygon styles in the specified drawing database.

See Also | Example

Signature

object.PolygonStyles

object

AecBaseDatabase The object this property applies

to.

Polygon Styles

AecPolygonStyles; read-only The collection of polygon styles.

Preferences Property

Returns the Preferences object for the specified drawing.

See Also | Example

Signature

object.Preferences

object

<u>AecBaseDatabase</u>, <u>AecDatabase</u> The object this property applies to.

Preferences

AcadDatabasePreferences; read-only The Preferences object for the drawing.

Profile Property

Returns or sets the object's profile.

See Also | Example

Signature

object.Profile

object

<u>AecEditInPlaceProfile</u>, <u>AecPolygon</u>, <u>AecProfileStyle</u>, <u>AecRing</u>, <u>AecRings</u>, <u>AecMassElement</u>, <u>AecSlice</u>

The object this property applies to.

Profile

AecProfile; read-write The object's profile.

ProfileStyle Property

Specifies the profile style of the mass element.

See Also | Example

Signature

object.ProfileStyle

object

AecMassElement The object this property applies

to.

Profile Style

AecProfileStyle; read-write

The profile style of the mass element.

ProfileStyleName Property

Specifies the name of the object's profile style.

See Also | Example

Signature

object.ProfileStyleName

object

AecEditInPlaceProfile, AecMassElement

The objects this property applies to.

ProfileStyleName

String; read-write

The name of the profile style.

ProjectName Property

Specifies the name of the project that the drawing belongs to.

See Also | Example

Signature

object.ProjectName

object

AecBaseDatabasePreferences The object this

property applies to.

ProjectName

String; read-write The project name.

Remarks

The project name can be no more than 255 characters in length.

Radius Property

Specifies the radius of curved mass elements.

See Also | Example

Signature

object.Radius

object

AecMassElement The object this property applies

to.

Radius

Double; read-write

The radius of the mass element.

Reference Property

Specifies the object being anchored to.

See Also | Example

Signature

object.Reference

object

<u>AecAnchorEntToLayoutCell</u>, <u>AecAnchorEntToCurve</u>,

AecAnchorLeadEntToNode, AecAnchorEntToGridAssembly, AecAnchorEntToLayoutNode, AecAnchorEntToLayoutVolume,

<u>AecAnchorExtendedTagToEnt</u> <u>AecAnchorToRef</u>

The object this property applies to.

Reference

AcadEntity; read-write, except for

AecAnchorEntToGridAssembly, which is read-only.

The object being anchored to.

RegisteredApplications Property

Gets the RegisteredApplications collection for the document.

See Also | Example

Signature

object. Registered Applications

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

RegisteredApplications

<u>AcadRegisteredApplications</u> collection; read-only The RegisteredApplications collection for the document.

Removeable Property

Specifies whether you can remove the layer key from the layer key style.

See Also | Example

Signature

object.Removeable

object

AecLayerKey The object this property applies to.

Removeable

Boolean; read-only

TRUE: Layer key is removable. FALSE: Layer key is not removable.

RenderSmoothness Property

Specifies the smoothness of shaded, rendered, and hidden line-removed objects.

See Also | Example

Signature

object.RenderSmoothness

object

<u>AecBaseDatabasePreferences</u> The object or

objects this property applies to.

RenderSmoothness

Double; read-write

The valid range is 0.01 to 10.0.

System variables

The value of this property is stored in the **FACETRES** system variable.

Remarks

The initial value for this property is 0.5. To improve performance, set this value to 1 or less when drawing.

ResizeOffset Property

Specifies the offset of the anchored object to the cell boundary when the cell is resized.

See Also | Example

Signature

object.ResizeOffset

object

AecAnchorEntToLayoutCell,
AecAnchorEntToLayoutVolume
The object this property applies to.

ResizeOffset

Double; read-write

The offset from the cell to the anchored object.

Remarks

To change the ResizeOffset value, you must first set <u>ApplyResize</u> to True. Modifying ResizeOffset when ApplyResize is False does not result in an error or warning, but it has no effect.

To specify a distance between the anchored object and the cell, specify a negative value for ResizeOffset. To extend the anchored object beyond the cell edges, specify a positive value.

RightOffset Property

Specifies the right offset for the anchored object.

See Also | Example

Signature

object.RightOffset

object

AecAnchorEntToGridAssembly The object this

property applies to.

RightOffset

Double; read-write

The right offset for the anchored object.

Remarks

To change the RightOffset value, you must first set <u>AdjustSizing</u> to True. Modifying RightOffset when AdjustSizing is False does not result in an error or warning, but it has no effect.

Rings Property

Returns the collection of rings that make up the profile.

See Also | Example

Signature

object.Rings

object

AecProfile The object this property applies to.

Rings

AecRings; read-only

Collection of rings that make up the profile.

Rise Property

Specifies the rise of gabled mass elements.

See Also | Example

Signature

object.Rise

object

AecMassElement The object this property applies

to.

Rise

Double; read-write

The rise of the gabled mass element.

Rotation Property

Specifies the rotation reference angle for the object.

See Also | Example

Signature

object.Rotation

object

AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecEntRef, AecGeo,

AecGridAssembly, AecLayoutCurve, AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool AecMaskBlockRef, AecMassELement, AecMassGroup, AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this property applies to.

Rotation

ACAD_ANGLE; read-write

Rotation reference angle for the object.

Scale Property

Specifies the scale factor applied to the hatch pattern.

See Also | Example

Signature

object.Scale

object

AecDisplayComponentHatch The object this

property applies to.

Scale

Double; read-write

Remarks

The Scale property determines how the hatch pattern is repeated.

ScaleOnInsert Property

Specifies whether objects are automatically scaled when inserted into the drawing.

See Also | Example

Signature

object.ScaleOnInsert

object

AecBaseDatabasePreferences The object this

property applies to.

ScaleOnInsert

Boolean; read-write

TRUE: New objects are scaled automatically.

FALSE: New objects are not scaled automatically.

ScaleX Property

Specifies the X direction scale factor of the inserted block or entity.

See Also | Example

Signature

object.ScaleX

object

AecBlockRef, AecEntRef, AecMaskBlockRef,

AecMVBlockRef

The object this property applies to.

ScaleX

Double; read-write

Scale of the block in the X direction.

ScaleY Property

Specifies the Y direction scale factor of the inserted block or entity.

See Also | Example

Signature

object. Scale Y

object

AecBlockRef, AecEntRef, AecMaskBlockRef,

AecMVBlockRef

The object this property applies to.

ScaleY

Double; read-write

Scale of the block in the Y direction.

ScaleZ Property

Specifies the Z direction scale factor of the inserted block or entity.

See Also | Example

Signature

object. Scale Z

object

AecBlockRef, AecEntRef, AecMaskBlockRef,

<u>AecMVBlockRef</u>

The object this property applies to.

ScaleZ

Double; read-write

Scale of the block in the Z direction.

SectionedBodyRenderingMaterialName Property

Gets or sets the name of the sectioned body rendering material.

See Also | Example

Signature

object. Sectioned Body Rendering Material Name

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

SectionedBodyRenderingMaterialName String; read-write

SectionRenderingMaterialName Property

Gets or sets the name of the section rendering material.

See Also | Example

Signature

object. Section Rendering Material Name

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

SectionRenderingMaterialName
String; read-write

SegmentPerPolyline Property

Specifies the number of line segments to be generated for each polyline curve.

See Also | Example

Signature

object.SegmentPerPolyline

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

SegmentPerPolyline

Integer; read-write

The number of line segments to be generated for

each polyline curve.

0 < SegmentPerPolyline <= 42950

System variables

The value of this property is stored in the <u>SPLINESEGS</u> system variable.

Remarks

The initial value is 8. The higher the number you specify, the greater the performance impact.

SelfIntersects Property

Determines if the profile is self-intersecting.

See Also | Example

Signature

object.SelfIntersects

object

AecProfile The object this property applies to.

SelfIntersects

Boolean; read-only

TRUE: Profile intersects itself.

FALSE: Profile does not intersect itself.

Shape Property

Specifies the shape of the layout grid.

See Also | Example

Signature

object.Shape

object

<u>AecLayoutGrid2D</u> The object this property applies

to.

Shape

AecLayoutShape enum; read-write

The shape of the layout grid.

Side1 Property

Specifies the length of the cut plane along the starting edge.

See Also | Example

Signature

object.Side1

object

AecClipVol The object this property applies to.

Side1

Double; read-write

The length of the cut plane along the starting edge.

Side2 Property

Specifies the length of the cut plane along the ending edge.

See Also | Example

Signature

object.Side2

object

AecClipVol The object this property applies to.

Side2

Double; read-write

Length of cut plane along the ending edge.

SolidFill Property

Specifies if multilines, traces, solids, all hatches (including solid-fill) and wide polylines are filled in.

See Also | Example

Signature

object.SolidFill

object

AecBaseDatabasePreferences The object or

objects this property applies to.

SolidFill

Boolean; read-write

TRUE: Displays solid fills.

FALSE: Does not display solid fills.

System variables

The value of this property is stored in the **FILLMODE** system variable.

Remarks

The initial value of this property is TRUE.

Spacing Property

Specifies the spacing between nodes or for an individual node on a layout tool, or the spacing between repeated hatch patterns for a display component.

See Also | Example

Signature

object.Spacing

object

AecLayoutCurveNode, AecLayoutGrid2DNode, AecLayoutGrid2DNode, AecLayoutGrid2DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutGrid3DNode, AecLayoutNode

The object this property applies to.

Spacing

Double; read-write

The spacing between nodes or for an individual node on a layout tool, or the spacing between repeated hatch patterns for a display component.

StartOffset Property

Specifies the offset distance from the start point of the curve, which may be used to reduce the effective layout curve length.

See Also | Example

Signature

object.StartOffset

object

AecLayoutCurve The object this property applies to.

StartOffset

Double; read-write

The offset distance from the start point of the curve.

Style Property

Specifies the style of the object.

See Also | Example

Signature

object.Style

object

AecMaskBlockRef, AecMassElement,

AecMVBlockRef, AecPolygon

The object this property applies to.

Style

Style; read-write

The style of the object. The type of style depends on

the type of object, as follows:

AecMaskBlockRef <u>AecMaskBlockStyle</u>

AecMassElement <u>AecMassElementStyle</u>

AecMVBlockRef <u>AecMVBlockStyle</u> AecPolygon <u>AecPolygonStyle</u>

StyleName Property

Specifies the name of the style that applies to the object.

See Also | Example

Signature

object.StyleName

object

AecMaskBlockRef, AecMVBlockRef, AecPolygon

The object this property applies to.

StyleName

String; read-write The style name.

SubType Property

Specifies the mass element shape subtype.

See Also | Example

Signature

object.SubType

object

<u>AecMassElement</u> The object this property applies

to.

SubType

<u>AecMassElementSubType</u>; read-only

SupportsAnchoring Property

Specifies whether the mass group supports element anchoring.

See Also | Example

Signature

object.SupportsAnchoring

object

AecMassGroup The object this property applies to.

SupportsAnchoring

Boolean; read-write

Remarks

This property enables enhanced mass group behavior.

SurfaceHatchPlacement Property

Specifies how the surface hatch pattern is placed on the object.

See Also | Example

Signature

object. Surface Hatch Placement

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

SurfaceHatchPlacement

AecSurfaceHatchPlacement; read-write

SurfaceRenderingMaterialName Property

Gets or sets the name of the surface rendering material.

See Also | Example

Signature

object. Surface Rendering Material Name

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

SurfaceRenderingMaterialName

String; read-write

SurfaceRenderMaterialMapping Property

Specifies how the surface render material is mapped to the object.

See Also | Example

Signature

object. Surface Render Material Mapping

object

<u>AecDisplayPropertiesMaterial</u> The object this property applies to.

SurfaceRenderMaterialMapping

AecSurfaceMappingType; read-write

TextFrameDisplay Property

Toggles the display of frames for text objects instead of displaying the text itself.

See Also | Example

Signature

object.TextFrameDisplay

object

<u>AecBaseDatabasePreferences</u> The object or

objects this property applies to.

TextFrameDisplay

Boolean; read-write

TRUE: Displays the frame around the extents of the text,

but does not display the text itself.

FALSE: Displays the full text.

System variables

The value of this property is stored in the **QTEXTMODE** system variable.

Remarks

The initial value for this property is FALSE.

After you enable or disable this option, you must use the <u>Regen</u> method to update the display.

TextHeight Property

Specifies the default height of text in the drawing.

See Also | Example

Signature

object.TextHeight

object

AecBaseDatabasePreferences The object this

property applies to.

TextHeight

Double; read-write

The text height in inches or millimeters.

Remarks

The default value is 0.125 inches, or 3.175 millimeters, depending on the setting of the MeasurementUnit property.

TextStyles Property

Gets the TextStyles collection for the document.

See Also | Example

Signature

object.TextStyles

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

TextStyles

AcadTextStyles collection; read-only

The TextStyles collection for the document.

TopOffset Property

Specify the top offset for the anchored object.

See Also | Example

Signature

object.TopOffset

object

AecAnchorEntToGridAssembly The object this

property applies to.

TopOffset

Double; read-write

The top offset for the anchored object. (See

Remarks.)

Remarks

To change the TopOffset value, you must first set <u>AdjustSizing</u> to True. Modifying TopOffset when AdjustSizing is False does not result in an error or warning, but it has no effect.

Type Property

For layout curves, specifies how nodes are specified along the curve; for mass elements, specifies the shape of the mass element.

See Also | Example

Signature

object.Type

object

AecLayoutCurve, AecMassElement The object this property applies to.

Type

AecLayoutCurve: <u>AecLayoutType</u> enum; read-write Type of node spacing along the curve.

AecMassElement: <u>AecMassElementType</u> enum;

read-write

The Mass Element shape.

UseAngleOfObject Property

Specifies the orientation of the hatch pattern.

See Also | Example

Signature

object. Use Angle Of Object

object

AecDisplayComponentHatch The object this

property applies to.

UseAngleOfObject

Boolean; read-write

TRUE: Orient the hatching to the object, regardless

of the object rotation.

FALSE: Orient the hatching to the World Coordinate

System.

UseFixedViewDirection Property

Specifies if the FixedViewDirection is used for the display configuration.

See Also | Example

Signature

object. Use Fixed View Direction

object

<u>AecDisplayConfiguration</u> The object this property applies to.

UseFixedViewDirection

Boolean; read-write

UseModelExtentsProperty

Determines if the height and lower extension are set automatically using model extents.

See Also | Example

Signature

object.UseModelExtents

object

AecClipVol The object this property applies to.

UseModelExtents

Boolean; read-write

TRUE: Height and Lower Extension are set

automatically

FALSE: Height and Lower Extension are not set

automatically

UseOffset Property

Specifies if the offset value is used.

See Also | Example

Signature

object. Use Off set

object

AecEntRef The object this property applies to.

UseOffset

Boolean; read-write

UserCoordinateSystems Property

Gets the UCSs collection for the document.

See Also | Example

Signature

object. User Coordinate Systems

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

User Coordinate Systems

AcadUCSs collection; read-only

The UCSs collection for the document.

Valid Property

Tests the validity of the object.

See Also | Example

Signature

object.Valid

object

AecProfile, AecRing

The object this property applies to.

Valid

Boolean; read-only TRUE: Object is valid. FALSE: Object is not valid.

Remarks

A ring is valid if the following are true:

- There is more than one segment.
- The segments are continuous.
- The segments are closed.
- No segment intersects itself.

A profile is valid if each ring is closed and does not self-intersect, and the profile is not self intersecting.

Value Property

For layer overrides, specifies the value of the layer key style override; for layout curves, specifies either the spacing between nodes or the number of nodes on the layout curve.

See Also | Example

Signature

object.Value

object

AecLayerOverrideSetting, AecLayoutCurve

The object this property applies to.

Value

AecLayerOverrideSetting: String; read-write The string the layer key override is set to.

AecLayoutCurve: Variant; read-write

Bay spacing or number of even divisions on a layout

curve.

Remarks

AecLayoutCurve:

- If the curve uses manual spacing, Value is zero.
- Check the <u>Type</u> property to determine what Value represents.

VerticalScale Property

Specifies the scale at which the vertical axis of the drawing is displayed (in Land Development Desktop only).

See Also | Example

Signature

object.VerticalScale

object

AecBaseDatabasePreferences The object this

property applies to.

HeadHeight

Double; read-write

The vertical scale. For example, for a 1:200 scale,

enter 200.

Remarks

This is only used in Autodesk Land Development Desktop.

ViewBlockDefs Property

Returns the collection of view block definitions for the multi-view block display representation definition.

See Also | Example

Signature

object. ViewBlockDefs

object

<u>AecMVBlockDisplayRepDef</u> The object this property applies to.

ViewBlockDefs

AecViewBlockDefs; read-write

ViewBlocks Property

Returns the collection of view blocks used for display representations by the multi-view block reference.

See Also | Example

Signature

object.ViewBlocks

object

AecMVBlockRef The object this property applies to.

ViewBlocks

<u>AecViewBlocks</u>; read-only

The collection of view block references, with each multi-view block display representation mapped to an individual AcadBlock.

ViewDirection Property

Specifies the view direction for the view block definition.

See Also | Example

Signature

object.ViewDirection

object

AecViewBlockDef The object this property applies

to.

ViewDirection

AecViewDirection; read-write

Viewports Property

Gets the Viewports collection for the document.

See Also | Example

Signature

object.Viewports

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Viewports

AcadViewports collection; read-only

The Viewports collection for the document.

Views Property

Gets the Views collection for the document.

See Also | Example

Signature

object.Views

object

AecBaseDatabase, AecDatabase

The object or objects this property applies to.

Views

AcadViews collection; read-only

The Views collection for the document.

Visible Property

Specifies whether the object or application is visible.

See Also | Example

Signature

object.Visible

object

AecBaseApplication, AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecDisplayComponentEntity, AecEntity, AecEntRef, AecGeo, AecGridAssembly, AecLayoutCurve,

AecLayoutGrid2D, AecLayoutGrid3D, AecLayoutTool, AecMaskBlockRef, AecMassElement, AecMassGroup,

AecMVBlockRef, AecPolygon, AecSlice,

<u>AecVolumeLayoutTool</u>

The object this property applies to.

Visible

Boolean; read-only

TRUE: The object or application is visible. FALSE: The object or application is not visible.

Remarks

Other factors can also cause an object to be invisible. For example, an object will not be displayed if its layer is off or frozen.

Specifying the application to be invisible allows you to run tasks in the background without having to see the component.

Void Property

Specifies if a ring produces a void in a profile.

See Also | Example

Signature

object.Void

object

AecRing The object this property applies to.

Void

Boolean; read-write TRUE: Ring is void. FALSE: Ring is additive.

Volume Property

Returns the volume of the mass element or mass group.

See Also | Example

Signature

object.Volume

object

AecMassElement, AecMassGroup

The object this property applies to.

Volume

Double; read-only

The volume of the mass element or mass group.

VolumeDisplayUnit Property

Specifies the units used to display volume measurements.

See Also | Example

Signature

object. Volume Display Unit

object

<u>AecBaseDatabasePreferences</u> The object this property applies to.

VolumeDisplayUnit

AecBuiltInUnit enum; read-write

The units used to display volume measurements.

VolumePrecision Property

Specifies the number of decimal places displayed for volume measurements.

See Also | Example

Signature

object.VolumePrecision

object

AecBaseDatabasePreferences The object this

property applies to.

VolumePrecision

Long; read-write

The display precision for volume mesurements.

Remarks

The precision settings are used only for labeling and listing values, not actual computations.

VolumeSuffix Property

The suffix string added when volume measurements are displayed in the drawing.

See Also | Example

Signature

object.VolumeSuffix

object

AecBaseDatabasePreferences The object this

property applies to.

VolumeSuffix

String; read-write

The suffix displayed after the volume measurement.

Remarks

The default value is "Cu.Ft." or "m3", depending on the value of the MeasurementUnit property. The value can be up to 255 characters in length.

Width Property

Specifies the width of the object.

See Also | Example

Signature

object.Width

object

AecLayoutGrid2D, AecLayoutGrid3D,

<u>AecMassElement</u>

The object this property applies to.

Width

Double; read-write

The width of the object.

WindowState Property

Specifies the state of the application or document window.

See Also | Example

Signature

object.WindowState

object

AecBaseDocument The object or objects this

property applies to.

WindowState

acWindowState enum; read-write

Indicates if the application window is minimized or

maximized.

WindowTitle Property

Gets the title of the document window.

See Also | Example

Signature

object.WindowTitle

object

AecBaseDocument The object or objects this

property applies to.

WindowTitle

String; read-only

The window title of the document.

XCount Property

Specifies the number of nodes along the X direction.

See Also | Example

Signature

object.XCount

object

AecLayoutGrid2D, AecLayoutGrid3D

The object this property applies to.

XCount

Long; read-write

The number of nodes along the X direction.

XDistance Property

Specifies the distance along the curve's X axis to the anchored object.

See Also | Example

Signature

object.XDistance

object

AecAnchorEntToCurve The object this property

applies to.

XDistance

Double; read-write

Distance along the curve's X axis to the anchored

object.

XOffset Property

Specifies the offset in the X direction of the view block within the multiview block, or the x offset for the hatch pattern of a display component.

See Also | Example

Signature

object.XOffset

object

AecDisplayComponentHatch, AecViewBlock

The object this property applies to.

XOffset

Double; read-write

The offset in the X direction.

XPositionFrom Property

Specifies the position on the curve from which the XDistance to an anchored object is measured.

See Also | Example

Signature

object.XPositionFrom

object

<u>AecAnchorEntToCurve</u> The object this property

applies to.

XPositionFrom

AecCurvePosition enum; read-write

The position on the curve from which XDistance is

measured.

XPositionTo Property

Specifies the position on the anchored object to measure XDistance to.

See Also | Example

Signature

object.XPositionTo

object

AecAnchorEntToCurve The object this property

applies to.

XPositionTo

AecEdgePosition enum; read-write

The position that **XDistance** measures to.

XRefEdit Property

Determines if the current drawing can be edited in place when being referenced by another user.

See Also | Example

Signature

object.XRefEdit

object

AecBaseDatabasePreferences The object or

objects this property applies to.

XRefEdit

Boolean; read-write

TRUE: Allows in-place reference editing.

FALSE: Does not allow in-place reference editing.

System variables

The value of this property is stored in the XEDIT system variable.

Remarks

The initial value of this property is TRUE.

XRefLayerVisibility Property

Determines the visibility of xref-dependent layers and specifies if nested xref path changes are saved.

See Also | Example

Signature

object.XRefLayerVisibility

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

XRefLayerVisibility

Boolean; read-write

TRUE: Xref-dependent layer changes made in the current drawing take precedence.

FALSE: The layer table, as stored in the reference drawing (xref) takes precedence.

System variables

The value of this property is stored in the <u>VISRETAIN</u> system variable.

Remarks

The initial value of this property is TRUE.

When this property is set to TRUE, layer settings are saved with the current drawing's layer table and persist from session to session. Nested xref path changes are saved with the current drawing and persist from session to session.

When this property is set to FALSE, changes made to xref-dependent layers in the current drawing are valid in the current session only, but are

not saved with the drawing. When the current drawing is reopened, the layer table is reloaded from the reference drawing and the current drawing reflects those settings. The layer settings affected are: On/Off, Freeze/Thaw, Color, and Linetype. This setting also specifies that changes made to the paths of nested xrefs are for the current session only and are not saved with the drawing.

XrefOverlaysUseOwnDisplayConfig Property

When used as an Xref Overlay, use this drawing's default display configuration initially.

See Also | Example

Signature

object. Xref Overlays Use Own Display Config

object

<u>AecBaseDatabasePreferences</u> The object or objects this property applies to.

XrefOverlays Use Own Display Config

Boolean; read-write When used as an Xref Overlay, use this drawing's default display configuration initially.

XRotation Property

Specifies the rotation of the anchored object around the X axis.

See Also | Example

Signature

object.XRotation

object

<u>AecAnchorEntToLayoutCell</u>, <u>AecAnchorEntToLayoutNode</u>, <u>AecAnchorEntToLayoutVolume</u>,

AecAnchorEntToCurve, AecAnchorLeadEntToNode

The object this property applies to.

XRotation

Double; read-write

Rotation of the anchored object around the X axis.

XSpacing Property

Specifies the spacing between nodes along the X direction of the layout grid.

See Also | Example

Signature

object.XSpacing

object

<u>AecLayoutGrid2D</u>, <u>AecLayoutGrid3D</u> The object this property applies to.

XSpacing

ACAD_DISTANCE; read-write

The spacing between nodes along the X direction of

the layout grid.

YAlignment Property

Specifies the alignment of the anchored object in relation to the baseline of the assembly.

See Also | Example

Signature

object.YAlignment

object

<u>AecAnchorEntToGridAssembly</u> The object this

property applies to.

YAlignment

AecInfillAlignment enum; read-write

Alignment of the infill anchored to the assembly.

(See Remarks.)

Remarks

To change the YAlignment value, you must first set <u>AllowVariation</u> to True. Modifying YAlignment when AllowVariation is False does not result in an error or warning, but it has no effect.

YCount Property

Specifies the number of nodes along the Y direction.

See Also | Example

Signature

object.YCount

object

AecLayoutGrid2D, AecLayoutGrid3D

The object this property applies to.

YCount

Long; read-write

The number of nodes along the Y direction.

YDistance Property

Specifies the distance along the curve's Y axis to the anchored object.

See Also | Example

Signature

object.YDistance

object

AecAnchorEntToCurve The object this property

applies to.

YDistance

Double; read-write

Distance along the curve's Y axis to the anchored

object.

YOffset Property

Specifies the Y offset of the anchored object, or the offset in the Y direction of the view block within the multi-view block, or the Y offset for the hatch pattern of a display component.

See Also | Example

Signature

object.YOffset

object

AecAnchorEntToGridAssembly,

AecDisplayComponentHatch, AecViewBlock

The object this property applies to.

YOffset

Double; read-write

The offset in the Y direction.

YPositionFrom Property

Specifies the position on the curve from which the YDistance to an anchored object is measured.

See Also | Example

Signature

object.YPositionFrom

object

<u>AecAnchorEntToCurve</u> The object this property applies to.

YPositionFrom

AecCurveWidthPosition enum; read-write

The position on the curve from which $\underline{\mathsf{YDistance}}$ is

measured.

YPositionTo Property

Specifies the position on the anchored object to measure YDistance to.

See Also | Example

Signature

object.YPositionTo

object

AecAnchorEntToCurve The object this property

applies to.

YPositionTo

AecPosition enum; read-write

The position that **YDistance** measures to.

YSpacing Property

Specifies the spacing between nodes along the Y direction of the layout grid.

See Also | Example

Signature

object.YSpacing

object

AecLayoutGrid2D, AecLayoutGrid3D

The object this property applies to.

YSpacing

ACAD_DISTANCE; read-write

The spacing between nodes along the Y direction of

the layout grid.

ZCount Property

Specifies the number of nodes along the Z direction.

See Also | Example

Signature

object.ZCount

object

AecLayoutGrid2D, AecLayoutGrid3D

The object this property applies to.

ZCount

Long; read-write

The number of nodes along the Z direction.

ZDistance Property

Specifies the distance above the curve to the anchored object.

See Also | Example

Signature

object.ZDistance

object

AecAnchorEntToCurve The object this property

applies to.

ZDistance

Double; read-write

Distance above the curve to the anchored object.

ZOffset Property

Specifies the offset in the Z direction of the view block within the multiview block.

See Also | Example

Signature

object.ZOffset

object

AecViewBlock The object this property applies to.

ZOffset

Double; read-write

The offset in the Z direction of the view block within

the multi-view block.

Zoom Property

Specifies the focal length of the camera, in millimeters.

See Also | Example

Signature

object.Zoom

object

AecCamera The object this property applies to.

Zoom

Double; read-write

Length of the lens in millimeters.

Remarks

The LENSLENGTH system variable will be reset to the Zoom value when the camera generates a view.

ZPositionFrom Property

Specifies the position on the curve from which the ZDistance to an anchored object is measured.

See Also | Example

Signature

object.ZPositionFrom

object

<u>AecAnchorEntToCurve</u> The object this property

applies to.

ZPositionFrom

AecCurveHeightPosition enum; read-write

The position on the curve from which **ZDistance** is

measured.

ZPositionTo Property

Specifies the position on the anchored object to measure ZDistance to.

See Also | Example

Signature

object. ZPosition To

object

AecAnchorEntToCurve The object this property

applies to.

ZPositionTo

AecHeight enum; read-write

The position that **ZDistance** measures to.

ZRotation Property

Specifies the rotation of the anchored object around the Z axis.

See Also | Example

Signature

object.ZRotation

object

AecAnchorEntToLayoutCell, AecAnchorEntToLayoutNode, AecAnchorEntToLayoutVolume,

AecAnchorEntToCurve, AecAnchorLeadEntToNode

The object this property applies to.

ZRotation

Double; read-write

Rotation of the anchored object around the Z axis.

ZSpacing Property

Specifies the spacing between nodes along the Z direction of the layout grid.

See Also | Example

Signature

object.ZSpacing

object

<u>AecLayoutGrid2D</u>, <u>AecLayoutGrid3D</u>

The object this property applies to.

ZSpacing

ACAD_DISTANCE; read-write

The spacing between nodes along the Z direction of

the layout grid.

Modified Event

Triggered when an object or collection in the drawing has been modified.

See Also | Example

Signature

object.Modified(Entity)

Aec2dSection, AecAnchor, AecAnchorEntToCurve,

AecAnchorEntToGridAssembly,

AecAnchorEntToLayoutCell,

AecAnchorEntToLayoutNode,

<u>AecAnchorEntToLayoutVolume</u>,

AecAnchorLeadEntToNode, AecAnchorToRef,

AecBlockRef, AecCamera, AecCellLayoutTool,

AecClipVol, AecClipVolRes, AecDictionary,

AecDictRecord, AecEditInPlaceProfile, AecEntity,

AecEntRef, AecGeo, AecGridAssembly,

AecLayerKeyStyle, AecLayerKeyStyles,

AecLayoutCurve, AecLayoutGrid2D,

AecLayoutGrid3D, AecLayoutTool,

AecMaskBlockRef, AecMaskBlockStyle,

<u>AecMaskBlockStyles</u>, <u>AecMassElement</u>,

<u>AecMassElementStyle</u>, <u>AecMassElementStyles</u>,

AecMassGroup, AecMVBlockRef,

AecMVBlockStyle, AecMVBlockStyles, AecObject,

AecPolygon, AecPolygonStyle, AecPolygonStyles,

AecProfileStyle, AecProfileStyles, AecSlice,

<u>AecVolumeLayoutTool</u>

An object expression that evaluates to a valid container object.

Entity

A Drawing Object

The object in the drawing that is modified can be any one of the drawing objects.

Remarks

This event will be triggered whenever the object is modified. Modification includes whenever the value of a property is set, even if the new value is equal to the current value.

When coding in VBA, you must provide an event handler for all objects enabled for the Modified event. If you do not provide a handler, VBA may terminate unexpectedly.

No events will be fired while a modal dialog is being displayed.

AutoCAD object and layer colors.

ACAD_COLOR

Name	Value	Description
acByBlock	0	The color defined for the object's block.
acRed	1	• (red)
acYellow	2	• (yellow)
acGreen	3	• (green)
acCyan	4	• (cyan)
acBlue	5	• (blue)
acMagenta	6	• (magenta)
acWhite	7	♦ (white)
acByLayer (not valid for Layer object)	256	The color of the layer the object is on.

colors.

	Value	Description
	0	The color defined for the object's block.
	1	• (red)
	2	• (yellow)
	3	• (green)
	4	• (cyan)
	5	• (blue)
	6	• (magenta)
	7	♦ (white)
)	256	The color of the layer the object is on.

Lineweights.

AcLineWeight		
Name	Value	
acLnWtByBlock	-2	
acLnWtByLayer	-1	
acLnWtByLwDefault	-3	
acLnWt000	0	
acLnWt005	5	
acLnWt009	9	
acLnWt013	13	
acLnWt015	15	
acLnWt018	18	
acLnWt020	20	
acLnWt025	25	
acLnWt030	30	
acLnWt035	35	
acLnWt040	40	
acLnWt050	50	
acLnWt053	53	
acLnWt060	60	
acLnWt070	70	
acLnWt080	80	
acLnWt090	90	
acLnWt100	100	
acLnWt106	106	

acLnWt140 140 acLnWt158 158 acLnWt200 200	acLnWt120	120
acLnWt200 200	acLnWt140	140
	acLnWt158	158
2cI nWt211 211	acLnWt200	200
active (211	acLnWt211	211

Drawing units.

AcMeasurementUnit		
Name Value		
acEnglish English (Imperial) units		
acMetric Metric units		

The state of the application window.

AcWindowState			
Name Value Description			
acNorm	1	Normal (neither minimized or maximized)	
acMin	2	Minimized	
acMax	3	Maximized	

Unit of measurement.

AecBuiltInUnit

Name	Value	Description
aecUnitAcre	57	Acre
aecUnitAmpere	5	Ampere
aecUnitAngstrom	17	Angstrom
aecUnitAre	58	Are (100 square meters)
aecUnitAstronomicalUnit	18	Astronomical Unit
aecUnitBarn	59	Barn
aecUnitBarrel	98	Barrel
aecUnitBoardFoot	99	Board Foot
aecUnitBolt	36	Bolt (40 yards)
aecUnitBushel	100	Bushel
aecUnitCable	35	Cable (720 feet)
aecUnitCaliber	32	Caliber (.01 inches)
aecUnitCandela	7	Candela
aecUnitCentare	60	.01 are (1 square meter)
aecUnitCentiliter	101	Centiliter
aecUnitCentimeter	24	Centimeter
aecUnitCentury	175	Century
aecUnitChain	37	Chain
aecUnitChainEngineers	38	100 feet (30.48 meters)
aecUnitCord	102	Cord
aecUnitCubicAngstrom	122	Cubic Angstrom
aecUnitCubicAstronomicalUnit	123	Cubic Astronomical Unit

aecUnitCubicBolt	124	Cubic Bolt
aecUnitCubicCable	125	Cubic Cable
aecUnitCubicCaliber	126	Cubic Caliber
aecUnitCubicCentimeter	103	Cubic Centimeter
aecUnitCubicChain	127	Cubic Chain
aecUnitCubicCubit	128	Cubic Cubit
aecUnitCubicDecimeter	177	Cubic Decimeter
aecUnitCubicFathom	129	Cubic Fathom
aecUnitCubicFoot	130	Cubic Foot
aecUnitCubicFurlong	131	Cubic Furlong
aecUnitCubicGigameter	132	Cubic Gigameter
aecUnitCubicHand	133	Cubic Hand
aecUnitCubicHectometer	134	Cubic Hectometer
aecUnitCubicInch	135	Cubic Inch
aecUnitCubicLeagueNautical	136	Cubic League Nautical
aecUnitCubicLeagueStatute	137	Cubic League Statute
aecUnitCubicLightYear	138	Cubic Light Year
aecUnitCubicLink	139	Cubic Link
aecUnitCubicMeter	96	Cubic Meter
aecUnitCubicMil	140	Cubic Mil
aecUnitCubicMileNautical	141	Cubic Mile Nautical
aecUnitCubicMileStatute	142	Cubic Mile Statute
aecUnitCubicMillimeter	143	Cubic Millimeter
aecUnitCubicMillimicron	144	Cubic Millimicron
aecUnitCubicLeagueNautical	136	Cubic Nautical League
aecUnitCubicMileNautical	141	Cubic Nautical Mile
aecUnitCubicPace	145	Cubic Pace
aecUnitCubicPalm	146	Cubic Palm
aecUnitCubicParsec	147	Cubic Parsec

aecUnitCubicPerch	148	Cubic Perch
aecUnitCubicRope	149	Cubic Rope
aecUnitCubicSkein	150	Cubic Skein
aecUnitCubicSpan	151	Cubic Span
aecUnitCubicLeagueStatute	137	Cubic Statute League
aecUnitCubicMileStatute	142	Cubic Statute Mile
aecUnitCubicSurveyFoot	152	Cubic Survey Foot
aecUnitCubicYard	153	Cubic Yard
aecUnitCubit	40	Cubit
aecUnitDalton	154	Dalton (atomic mass unit)
aecUnitDay	169	Day
aecUnitDecade	174	Decade
aecUnitDecimeter	23	Decimeter
aecUnitDecistere	104	Decistere
aecUnitDekaliter	105	Dekaliter
aecUnitDekameter	22	Dekameter
aecUnitDekastere	106	Dekastere
aecUnitDimensionless	0	Dimensionless
aecUnitDram	107	Dram
aecUnitDryPint	108	Dry Pint
aecUnitDryQuart	109	Dry Quart
aecUnitFathoms	34	Fathom
aecUnitFirkin	110	Firkin
aecUnitFluidOunce	116	Fluid Ounce
aecUnitFluidPint	118	Fluid Pint
aecUnitFluidQuart	119	Fluid Quart
aecUnitFoot	30	Foot
aecUnitFortnight	171	Fortnight
aecUnitFurlong	41	Furlong

aecUnitGallon	111	Gallon
aecUnitGigameter	19	Gigameter
aecUnitGill	112	Gill (.5 cups)
aecUnitGrain	155	Grain
aecUnitGram	156	Gram
aecUnitHand	42	Hand (4 inches)
aecUnitHectare	61	Hectare
aecUnitHectometer	21	Hectometer
aecUnitHogshead	113	Hogshead
aecUnitHour	168	Hour
aecUnitHundredweight	157	Hundredweight
aecUnitInch	31	Inch
aecUnitJoule	12	Joule
aecUnitKelvin	4	Kelvin
aecUnitKilogram	1	Kilogram
aecUnitKiloliter	114	Kiloliter
aecUnitKilometer	20	Kilometer
aecUnitLeagueNautical	45	League Nautical
aecUnitLeagueStatute	46	League Statute
aecUnitLightYear	28	Light Year
aecUnitLink	39	Link (7.92 inches)
aecUnitLiter	97	Liter
aecUnitLongTon	158	2240 pounds (1.016 metric tons)
aecUnitLumen	11	Lumen
aecUnitLux	16	Lux
aecUnitMeter	2	Meter
aecUnitMicron	26	Micron
aecUnitMil	47	Mil

aecUnitMile	44	Mile
aecUnitMileNautical	43	Mile Nautical
aecUnitMillenium	176	Millenium
aecUnitMillimeter	25	Millimeter
aecUnitMillimicron	27	Millimicron
aecUnitMinim	115	Minim
aecUnitMinute	167	Minute
aecUnitMole	6	Mole
aecUnitLeagueNautical	45	Nautical League
aecUnitTonne	166	Metric Ton (tonne)
aecUnitMileNautical	43	Nautical Mile
aecUnitNewton	10	Newton
aecUnitOunce	159	Ounce
aecUnitOunceTroy	160	Ounce Troy
aecUnitPace	48	Pace
aecUnitPalm	49	Palm (3 inches)
aecUnitParsec	29	Parsec
aecUnitPascals	13	Pascal
aecUnitPeck	117	Peck
aecUnitPennyweight	161	Pennyweight
aecUnitPound	162	Pound
aecUnitRadian	8	Radian
aecUnitRegisterTon	120	Register Ton
aecUnitRod	50	Rod
aecUnitRope	51	rope
aecUnitSecond	3	Second
aecUnitSiderealYear	172	Sidereal Year
aecUnitSkein	52	Skein

aecUnitSlug	163	Slug
aecUnitSpan	53	Span
aecUnitSquareAngstrom	62	Square Angstrom
aecUnitSquareAstronomicalUnit	63	Square Angstrom
aecUnitSquareBolt	64	Square Bolt
aecUnitSquareCable	65	Square Cable
aecUnitSquareCaliber	66	Square Caliber
aecUnitSquareCentimeter	67	Square Centimeter
aecUnitSquareChain	94	Square Chain
aecUnitSquareCubit	68	Square Cubit
aecUnitSquareDecimeter	69	Square Decimeter
aecUnitSquareFathom	70	Square Fathom
aecUnitSquareFoot	56	Square Foot
aecUnitSquareFurlong	71	Square Furlong
aecUnitSquareGigameter	72	Square Gigameter
aecUnitSquareHand	73	Square Hand
aecUnitSquareInch	74	Square Inch
aecUnitSquareKilometer	75	Square Kilometer
aecUnitSquareLeagueNautical	76	Square League Nautical
aecUnitSquareLeagueStatute	77	Square League Statute
aecUnitSquareLightYear	78	Square Light Year
aecUnitSquareLink	79	Square Link
aecUnitSquareMeters	55	Square Meter
aecUnitSquareMicron	80	Square Micron
aecUnitSquareMil	81	Square Mil
aecUnitSquareMileNautical	82	Square Mile Nautical
aecUnitSquareMileStatute	83	Square Mile Statute
aecUnitSquareMillimicron	84	Square Millimicron

aecUnitSquareLeagueNautical	76	Square Nautical League
aecUnitSquareMileNautical	82	Square Nautical Mile
aecUnitSquarePace	85	Square Pace
aecUnitSquarePalm	86	Square Palm
aecUnitSquareParsec	87	Square Parsec
aecUnitSquarePerch	88	Square Perch
aecUnitSquareRope	89	Square Rope
aecUnitSquareSkein	90	Square Skein
aecUnitSquareSpan	91	Square Span
aecUnitSquareLeagueStatute	77	Square Statute League
aecUnitSquareMileStatute	83	Square Statute Mile
aecUnitSquareSurveyFoot	92	Square Survey Foot
aecUnitSquareYard	93	Square Yard
aecUnitLeagueStatute	46	Statute League
aecUnitSteradian	9	Steradian
aecUnitStone	164	Stone
aecUnitSurveyFoot	54	Survey Foot
aecUnitTon	165	Ton
aecUnitTonne	166	Tonne (metric ton)
aecUnitTownship	95	Township (36 square miles)
aecUnitTropicalYear	173	365 days, 5 hours, 48 minutes, 45.51 seconds (astronomical year)
aecUnitTun	121	Tun
aecUnitVolt	15	Volt
aecUnitWatt	14	Watt
aecUnitWeek	170	Week
aecUnitYards	33	Yard

Position on the curve.

AecCurveHeightPosition		
Name	Value	Description
aecCurveHeightPositionBottom	0	Bottom
aecCurveHeightPositionCenter	1	Center
aecCurveHeightPositionTop	2	Тор

Position on the curve.

AecCurvePosition			
Name	Value	Description	
aecCurvePositionStart	0	Start	
aecCurvePositionMiddle	1	Middle	
aecCurvePositionEnd	2	End	

Position on the curve.

AecCurveWidthPosition		
Name	Value	Description
aecCurveWidthPositionLeft	0	Left
aecCurveWidthPositionCenter	1	Center
aecCurveWidthPositionRight	2	Right

Position measured to.

AecEdgePosition			
Name	Value	Description	
aecEdgePositionStart	0	Start	
aecEdgePositionCenter	1	Center	
aecEdgePositionEnd	2	End	

Position on the object.

AecHeight			
Name	Value	Description	
aecHeightBottom	0	Bottom	
aecHeightCenter	1	Center	
aecHeightTop	2	Тор	

Alignment in relation to baseline.

AecInfillAlignment		
Name	Value	Description
aecInfillAlignCentered	0	Centered
aecInfillAlignFrontFlush	1	In front of the baseline
aecInfillAlignBackFlush	2	In back of the baseline

The shape of a layout grid.

AecLayoutShape			
Name	Value	Description	
aecLayoutShapeRectangular	0	Rectangular layout	
aecLayoutShapeRadial	1	Radial layout	
aecLayoutShapeCustom	2	Custom layout	

Type of node spacing along the curve.

Name Value Description aecLayoutTypeManualSpacing 0 Manual spacing aecLayoutTypeAutoSpacingEven 1 Spaced evenly along the curve aecLayoutTypeAutoSpacingBay 2 Automatic spacing, determined by the specified distance

between nodes

Subtype of the mass element.

AecMassElementSubType		
Name	Value	Description
aecMassElementSubTypeUnspecified	-1	Unspecified
aecMassElementSubTypeProfileInternal	0	ProfileInternal
aecMassElementSubTypeProfileExternal	1	ProfileExternal

Shape of the mass element.

AecMassElementType

Name	Value	Description
aecMassElementTypeArch	1	Arch
aecMassElementTypeBarrelVault	2	Barrel vault
aecMassElementTypeBox	0	Box
aecMassElementTypeDoric	3	Simple column
aecMassElementTypeCone	4	Cone
aecMassElementTypeCylinder	5	Cylinder
aecMassElementTypeDome	6	Dome
aecMassElementTypeExtrusion	12	Extrusion of a profile
aecMassElementTypeFreeForm	14	Free form
aecMassElementTypeGable	7	Gable
aecMassElementTypeIsoscelesTriangle	8	Isosceles triangle
aecMassElementTypePyramid	10	Pyramid
aecMassElementTypeRevolution	13	Revolved profile
aecMassElementTypeRightTriangle	9	Right triangle
aecMassElementTypeSphere	11	Sphere

Type of operation combining objects.

AecMassOperation		
Name	Value	Description
aecMassOperationAdd	0	Addition
aecMassOperationSubtract	1	Subtraction
aecMassOperationIntersect	2	Intersection

Placement of edges on AEC Polygon vertices.

AecPolygonStyleEdgeWidthJustification		
Name	Value	Description
aecPolygonEdgeWidthJustificationIn	0	Inside edge
aecPolygonEdgeWidthJustificationCenter 1 Center		
aecPolygonEdgeWidthJustificationOut	2	Outside edge

Position on the object.

AecPosition				
Name	Value	Description		
aecPositionFront	0	Front		
aecPositionCenter	1	Center		
aecPositionBack	2	Back		

Add Example

```
Sub Example_Add_AecProfile()
```

```
'This example defines a custom profile consisting of
' of two triangles, one within the other.
Dim doc As AecBaseDocument
Dim app As New AecBaseApplication
Dim profileStyle As AecProfileStyle
Dim profile As New AecProfile
Dim profileName As String
app.Init ThisDrawing.Application
Set doc = app.ActiveDocument
'Get existing or add a new profile style.
On Error Resume Next
profileName = "BOOBYPRIZE"
Set profileStyle = doc.ProfileStyles.Add(profileName)
' If error occurred, the profile probably already exists.
If Err.Number 0 Then
  Err.Clear
  On Error GoTo 0
  Set profileStyle = doc.ProfileStyles.Item(profileName)
End If
' Define definition points for rings.
Dim OuterPts(0 To 7) As Double
Dim InnerPts(0 To 7) As Double
OuterPts(0) = 0: OuterPts(1) = 0
OuterPts(2) = 648: OuterPts(3) = 0
OuterPts(4) = 324: OuterPts(5) = 324
OuterPts(6) = 0: OuterPts(7) = 0
```

InnerPts(0) = 120: InnerPts(1) = 60

```
InnerPts(2) = 528: InnerPts(3) = 60
InnerPts(4) = 324: InnerPts(5) = 264
InnerPts(6) = 120: InnerPts(7) = 60
```

- ' Create and set the outer ring. Dim ring1 As AecRing Set ring1 = profile.Rings.Add ring1.FromPoints (OuterPts) ring1.Void = False
- ' Create and set the inner ring. Dim ring2 As AecRing Set ring2 = profile.Rings.Add ring2.FromPoints (InnerPts) ring2.Void = True
- ' Set the profile definition.
 Set profileStyle.profile = profile

MsgBox ("Profile """ & profileName & """ created.")

AdjustSizing Example

Sub Example_AdjustSizing()

- 'This example modifies the top offset of the selected object
- ' in relation to its anchor point on the grid assembly. It
- ' prompts the user to select the object to be modified,
- ' and then it prompts the user to specify the number of inches
- ' to adjust the offset by. The AdjustSizing property is checked,
- ' and if it is not currently set to True, it is reset and the
- ' the user is advised of the change.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- 'assembly.

Dim ent As AcadEntity
Dim geo As AecGeo
Dim anchor As AecAnchor

Dim offset As String
Dim offset_adjust As Double

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "AdjustSizing Example" ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

```
If anchor Is Nothing Then
```

MsgBox "Selected object is not anchored.", vbExclamation, "AdjustSizin; ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

' AdjustSizing must be set to True in order for offset change to take effect.

If anchor.AdjustSizing = False Then

anchor.AdjustSizing = True

MsgBox "AdjustSizing reset from False to True"

End If

MsgBox "Top offset of object was: " & anchor.TopOffset, vbInformation,

' Prompt user to specify amount to adjust offset by. offset_adjust = ThisDrawing.Utility.GetReal("Enter the number of inches

'Change offset by specified amount. anchor.TopOffset = anchor.TopOffset + offset_adjust

ThisDrawing.Regen (acActiveViewport)

MsgBox "New top offset is: " & anchor.TopOffset, vbInformation, "TopO End If

Else

MsgBox "Object selected is not an AEC entity.", vbInformation, "AdjustSizi End If

AllowVariation Example

```
Sub Example_AllowVariation()
```

'This example will add anchor a new mass element to cell in a grid assembly.

```
Dim grid As AecGridAssembly
Dim poly As AecPolygon
Dim pt As Variant
Dim obj As AcadObject
ThisDrawing.Utility.GetEntity obj, pt, "Select grid to attach to"
If TypeOf obj Is AecGridAssembly Then
  Set grid = obj
  Set poly = ThisDrawing.ModelSpace.AddCustomObject("AecPolygon")
  Dim anchor As New AecAnchorEntToGridAssembly
  anchor.Reference = grid
  ' anchor the mass element to the first cell in the grid
  anchor.Cell = 2
  anchor. Allow Variation = True
  anchor.BottomOffset = 6
  anchor.LeftOffset = 6
  anchor.RightOffset = 6
  anchor.TopOffset = 6
  anchor.YAlignment = aecInfillAlignFrontFlush
  anchor.AdjustSizing = True
  poly.AttachAnchor anchor
Else
  MsgBox "No Layout Grid selected", vbInformation, "Node Example"
End If
```

AlternateName Example

Sub Example_AlternateName_Add_Aec()

- 'This example list the multi-view block styles in the current
- ' drawing, and allows the user to add an alternate name to each
- ' style (if an alternate name does not already exist).
- ' Use this example with a drawing that contains one or more
- ' multi-view block styles.

Dim app As New AecBaseApplication Dim doc As AecBaseDocument Dim mvblockStyles As AecMVBlockStyles Dim mvblockStyle As AecMVBlockStyle

Dim msg As String Dim altname As String Dim reply As Integer

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

- ' Get the collection of multi-view block styles in the drawing. Set mvblockStyles = doc.mvblockStyles
- ' If there are no multi-view block styles defined in the
- ' drawing, alert user and then exit.

If mvblockStyles.count = 0 Then

msg = "There are no multi-view block styles in the drawing."
MsgBox msg, vbExclamation, "AlternateName Example"
Exit Sub

End If

- 'List the name and alternate name of each multi-view block
- ' style in the current drawing. If AlternateName is blank,
- 'allow user to set it.

For Each mvblockStyle In mvblockStyles

```
msg = "MV block style name: " & mvblockStyle.Name & vbCrLf
 altname = mvblockStyle.AlternateName
 If altname = "" Then
   msg = msg & " There is no alternate name for " & mvblockStyle.Name &
   reply = MsgBox(msg, vbYesNo, "Set alternate name at Command prompt?
   ' Prompt user to enter an alternate name, if they chose
   ' to do so.
   If reply = vbYes Then
     doc.Utility.Prompt list
     altname = doc.Utility.GetString(True, "Alternate name --> ")
     mvblockStyle.AlternateName = altname
   End If
 Else
   msg = msg & " Alternate name: " & mvblockStyle.AlternateName & vbC
   MsgBox msg, vbInformation, "AlternameName Example"
 End If
Next
```

Angle1 Example

```
Sub Example_Angle1()
```

'This example shows the deviation of cut plane at starting edge where zero is 'perpendicular to the building elevation line

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Angle1 is: " & clip.Angle1, vbInformatic
End If

Next

If count = 0 Then
MsgBox "No ClipVol Present in Drawing", vbInformation, "Angle1 Exampl
End If
```

Angle2 Example

```
Sub Example_Angle2()
```

'This example shows the deviation of cut plane at starting edge where zero is 'perpendicular to the building elevation line

AngleFromNode Example

Sub Example_AngleFromNode() 'This example will find the angle from the node for a bubble attached to a ' column grid Dim obj As AcadObject Dim bubble As AecMVBlockRef Dim anchor As AecAnchor Dim leaderAnchor As AecAnchorLeadEntToNode Dim pt As Variant ThisDrawing.Utility.GetEntity obj, pt, "Select bubble" If TypeOf obj Is AecMVBlockRef Then Set bubble = obj **Set** anchor = obj.GetAnchor If TypeOf anchor Is AecAnchorLeadEntToNode Then **Set** leaderAnchor = anchor MsgBox "Angle from node = " & ThisDrawing. Utility. Angle To String (lea-Else MsgBox "Not anchored to column grid", vbInformation, "Example Angle End If Else MsgBox "Not a bubble", vbInformation, "Example AngleFromNode"

End Sub

End If

Angular Azimuth Example

Sub Example_AngularAzimuth()

Style = "south azimuths."

```
'This example returns the AngularAzimuth setting for the current drawing.
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences
'Convert the constant to a string.
Dim Style As String
Dim AngAzm As Long
' Determine Angular Azimuth setting
AngAzm = dbPref.AngularAzimuth
Select Case AngAzm
Case 0
  Style = "bearings."
Case 1
  Style = "north azimuths."
Case 2
```

End Select

MsgBox "The current value for AngularAzimuth is " & Style, vbInformation, '

Angular Display Format Example

Sub Example_AngularDisplayFormat()

'This example returns the AngularDisplayFormat setting for the current drawir

```
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences
```

```
'Convert the constant to a string.

Dim strUnits As String

Select Case dbPref.AngularDisplayFormat

Case 0

strUnits = "Degrees."

Case 1

strUnits = "Degrees/Minutes/Seconds"

Case 2

strUnits = "Grads."

Case 3

strUnits = "Radians"

Case 4

strUnits = "Surveyor"

End Select
```

MsgBox "The current value for AngularDisplayFormat is " & strUnits, _ vbInformation, "AngularDisplayFormat Example"

AngularPrecision Example

Sub Example_AngularPrecision()

'This example returns the AngularPrecision setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for AngularPrecision is " & dbPref.AngularPrecisi vbInformation, "AngularPrecision Example"

Application Example

Sub Example_Application()

- 'This example creates a line and then uses the
- ' Application property of the line to return the
- 'application name.

Dim lineObj As AcadLine

Dim startPoint(0 To 2) As Double

Dim endPoint(0 To 2) As Double

Dim myApp As AcadApplication

```
'Create a new line reference

startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0

endPoint(0) = 2: endPoint(1) = 2: endPoint(2) = 0

Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)

lineObj.Update
```

MsgBox "The application name is: " & myApp.name, vbInformation, "Applica End Sub

^{&#}x27;Return the application for the object Set myApp = lineObj.Application

^{&#}x27; Display the name of the application

ApplyResize Example

Sub Example_ApplyResize()

```
'This example will take a mass element anchored to a 2D Layout Grid, and apply a Resize Offset to it.
```

```
Dim obj As AcadObject
Dim anchor As AecAnchorEntToLayoutCell
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select anchored element"
If TypeOf obj Is AecMassElement Then
Dim mass As AecMassElement
Set mass = obj
If TypeOf mass.GetAnchor Is AecAnchorEntToLayoutCell Then
Set anchor = mass.GetAnchor
anchor.ResizeOffset = -22
anchor.ApplyResize = False
End If
End If
```

Area Example

Examples:

1 AecProfile

1 AecSlice

Sub Example_Area_AecProfile()

'This example shows the area of the profile of an AecPolygon

Dim obj As Object
Dim pt As Variant
Dim poly As AecPolygon

ThisDrawing. Utility. GetEntity obj, pt, "Select an AEC Polygon"

If TypeOf obj Is AecPolygon Then

Set poly = obj

MsgBox "Profile Area: " & poly.Profile.Area, vbInformation, "Area Exampl Else

MsgBox "Not a Polygon or no Profile Found", vbInformation, "Area Examp End If

End Sub

Sub Example_Area_AecSlice()

'This example will display the area of one slice in the current drawing

Dim slice As AecSlice

Dim obj As Object

Dim counter As Integer

```
counter = 0
For Each obj In ThisDrawing.ModelSpace
If TypeOf obj Is AecSlice And counter < 1 Then
    counter = counter + 1
    Set slice = obj
    MsgBox "Slice Area: " & slice.Area, vbInformation, "Area Example"
End If
Next

If counter = 0 Then
    MsgBox "No Slice in Drawing", vbInformation, "Area Example"
End If</pre>
```

AreaDisplayUnit Example

Sub Example_AreaDisplayUnit()

```
'This example returns the AreaDisplayUnit setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
```

'Convert the area display unit to a string.

Dim unit As String

Select Case dbPref.AreaDisplayUnit

Case aecUnitSquareInch

unit = "square inch."

Case aecUnitSquareFoot

unit = "square foot."

Case aecUnitSquareYard

unit = "square yard."

Case aecUnitSquareMil

unit = "square millimeters."

Case aecUnitSquareCentimeter

```
unit = "square centimeters."

Case aecUnitSquareDecimeter
  unit = "square decimeters."

Case aecUnitSquareMeters
  unit = "square meters."

End Select

MsgBox "The current value for AreaDisplayUnit is " & unit, __
```

vbInformation, "AreaDisplayUnit Example"

AreaPrecision Example

Sub Example_AreaPrecision()

'This example displays the AreaPrecision setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The current value for AreaPrecision is: " & dbPref.AreaPrecision, _ vbInformation, "AreaPrecision Example"

AreaSuffix Example

Sub Example_AreaSuffix()

'This example displays the AreaSuffix setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The current value for AreaSuffix is: " & dbPref.AreaSuffix, _ vbInformation, "AreaSuffix Example"

ArrayPolar Example

```
Sub Example_ArrayPolar()
  'This example creates a circle and then performs a polar array
  ' on that circle.
  ' Create the circle
  Dim circleObj As AcadCircle
  Dim center(0 To 2) As Double
  Dim radius As Double
  center(0) = 2\#: center(1) = 2\#: center(2) = 0\#
  radius = 1
  Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
  ZoomAll
  MsgBox "Perform the polar array on the circle.", , "ArrayPolar Example"
  ' Define the polar array
  Dim noOfObjects As Integer
  Dim angleToFill As Double
  Dim basePnt(0 To 2) As Double
  noOfObjects = 4
  angleToFill = 3.14
                          ' 180 degrees
  basePnt(0) = 4\#: basePnt(1) = 4\#: basePnt(2) = 0\#
  'The following example will create 4 copies of an object
  by rotating and copying it about the point (3,3,0).
  Dim retObj As Variant
  retObj = circleObj.ArrayPolar(noOfObjects, angleToFill, basePnt)
  ZoomAll
  MsgBox "Polar array completed.", , "ArrayPolar Example"
End Sub
```

ArrayRectangular Example

```
Sub Example_ArrayRectangular()
```

'This example creates a circle and then performs

' a rectangular array on that circle.

```
' Create the circle
Dim circleObj As AcadCircle
Dim center(0 To 2) As Double
```

Dim radius As Double

center(0) = 2#: center(1) = 2#: center(2) = 0#

radius = 0.5

Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)

ThisDrawing.Application.ZoomAll

MsgBox "Perform the rectangular array on the circle.", , "ArrayRectangular Ex

' Define the rectangular array

Dim numberOfRows As Long

Dim numberOfColumns As Long

Dim numberOfLevels As Long

Dim distanceBwtnRows As Double

Dim distanceBwtnColumns As Double

Dim distanceBwtnLevels As Double

numberOfRows = 5

numberOfColumns = 5

numberOfLevels = 2

distanceBwtnRows = 1

distanceBwtnColumns = 1

distanceBwtnLevels = 1

' Create the array of objects

Dim retObj As Variant

retObj = circleObj.ArrayRectangular(numberOfRows, numberOfColumns, nur

ZoomAll

MsgBox "Rectangular array completed.", , "ArrayRectangular Example"

AttachAnchor Example

Sub Example_AttachAnchor()

'This example will make a mass element and and anchor it to a line.

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to mass.AttachAnchor anchor ' attach the anchor to the mass element
```

AttachEntity Example

Sub Example_AttachEntity()

'This example will make a mass element and and anchor it to a line.

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double

pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)
```

anchor.Reference = line ' set the line as the object (curve) to anchor to
anchor.AttachEntity mass ' attach the mass element to the anchor

BasePoint Example

```
Sub Example_BasePoint()
```

```
'This example displays the BasePoint setting for the current drawing.
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
Dim varPoint As Variant
Dim point(0 To 2) As Double
Dim basePoint As String
' Get the base point
varPoint = dbPref.BasePoint
point(0) = varPoint(0)
point(1) = varPoint(1)
point(2) = varPoint(2)
```

' Format base point to a string

basePoint = Format(point(0)) + ", " + Format(point(1)) + ", " + Format(point(2))
MsgBox "The current value for BasePoint is: " & basePoint, vbInformation, "E

BasePointNE Example

```
Sub Example_BasePointNE()
```

```
'This example displays the BasePointNE setting for the current drawing.
```

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

Dim varPoint As Variant

Dim point(0 To 2) As Double

Dim basePointNE As String

```
' Get the base point
```

```
varPoint = dbPref.basePointNE
```

point(0) = varpoint(0)

point(1) = varpoint(1)

point(2) = varpoint(2)

basePointNE = Format(point(0)) + ", " + Format(point(1)) + ", " + Format(point(1)))

^{&#}x27; Format base point to a string

MsgBox "The current value for BasePointNE is: " & basePointNE, _ vbInformation, "BasePointNE Example"

Blocks Example

Sub Example_Blocks()

- 'This example first finds the blocks collection
- ' using the Blocks property. It then adds a new
- ' block to the block collection.

Dim blkColl As AcadBlocks
Dim newBlock As AcadBlock
Dim insertionPnt(0 To 2) As Double

' Get the Blocks collection Set blkColl = ThisDrawing.Blocks

- ' Define the insertion point for the new block insertionPnt(0) = 0#: insertionPnt(1) = 0#: insertionPnt(2) = 0#
- ' Add the new block into the blocks collection
 Set newBlock = blkColl.Add(insertionPnt, "TEST")
 MsgBox "A block called " & newBlock.name & " has been added to the blocks
 End Sub

BottomOffset Example

Sub Example_BottomOffset()

- 'This example modifies the bottom offset of the selected object
- ' in relation to its anchor point on the grid assembly. It
- ' prompts the user to select the object to be modified,
- ' and then it prompts the user to specify the number of inches
- ' to adjust the offset by. The number can be either positive
- ' or negative.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- 'assembly.

Dim ent As AcadEntity
Dim geo As AecGeo
Dim anchor As AecAnchor

Dim offset As String Dim offset adjust As Double

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "BottomOffset Example" ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

If anchor Is Nothing Then

MsgBox "Selected object is not anchored.", vbExclamation, "BottomOffs: ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

' AdjustSizing must be set to True in order for offset change to take effect. anchor.AdjustSizing = True

MsgBox "Bottom offset of object was: " & anchor.BottomOffset, vbInfori

' Prompt user to specify amount to adjust offset by. offset_adjust = ThisDrawing.Utility.GetReal("Enter the number of inches

'Change offset by specified amount.
anchor.BottomOffset = anchor.BottomOffset + offset_adjust
ThisDrawing.Regen (acActiveViewport)
MsgBox "New bottom offset is: " & anchor.BottomOffset, vbInformation,
End If

Else

MsgBox "Object selected is not an AEC entity.", vbInformation, "BottomOf End If

Bounds Example

```
Sub Example_Bounds()
```

'This example will display the bounds of the first profile in the drawing.

```
Dim doc As AecArchBaseDocument
Dim cProfileStyles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Dim profile As AecProfile
Set doc = AecArchBaseApplication.ActiveDocument
Set cProfileStyles = doc.ProfileStyles
If cProfileStyles.Count > 0 Then
  Set profileStyle = cProfileStyles.Item(0)
  Set profile = profileStyle.profile
  MsgBox "Profile Bounds: " & vbCR & _
      Top Left: " & Format(profile.bounds(0), "0.000") & ", " & Format(prof.
   "Bottom Right: " & Format(profile.bounds(2), "0.000") & ", " & Format(pr
    vbInformation, "Bounds Example"
Else
  MsgBox "No AEC Profiles in the drawing", vbInformation, "Bounds Examp
End If
```

Cell Example

```
Sub Example_Cell()
```

'This example will add anchor a new mass element to cell in a 2D layout grid.

```
Dim grid As AecGridAssembly
Dim mass As AecMassElement
Dim pt As Variant
Dim obj As AcadObject
ThisDrawing.Utility.GetEntity obj, pt, "Select grid to attach to"
If TypeOf obj Is AecGridAssembly Then
  Set grid = obj
  Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement
  Dim anchor As New AecAnchorEntToGridAssembly
  anchor.Reference = grid
  ' anchor the mass element to the first cell in the grid
  anchor.Cell = 1
  mass.AttachAnchor anchor
Else
  MsgBox "No Layout Grid selected", vbInformation, "Node Example"
End If
```

CenterOnNode Example

Sub Example_CenterOnNode()

```
'This example attaches a Mass Element to a 2D Layout Grid, and centers the
```

```
Dim obj As AcadObject
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select Layout Grid"
If TypeOf obj Is AecLayoutGrid2D Then
  Dim grid As AecLayoutGrid2D
  Set grid = obj
  Dim mass As AecMassElement
  Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement
  Dim anchor As New AecAnchorEntToLayoutNode
  anchor.Reference = grid
  anchor.Node = 1
  anchor.CenterOnNode = True
  mass.AttachAnchor anchor
Else
  MsgBox "No Layout Grid selected", vbInformation, "CenterOnNode Examp
End If
```

^{&#}x27; the node.

Centroid Example

```
Sub Example_Centroid()
```

'This example will display the centroid of the first profile in the drawing.

```
Dim doc As AecArchBaseDocument
Dim cProfileStyles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Dim profile As AecProfile

Set doc = AecArchBaseApplication.ActiveDocument

Set cProfileStyles = doc.ProfileStyles
If cProfileStyles.Count > 0 Then
    Set profileStyle = cProfileStyles.Item(0)
    Set profile = profileStyle.profile
    MsgBox "Profile Centroid: " & profile.Centroid(0) & ", " & profile.Centroid vbInformation, "Centroid Example"

Else
    MsgBox "No AEC Profiles in the drawing", vbInformation, "Centroid Exam End If
```

Clear Example

Sub Example_Clear_AecProfile()

- 'This example copies a profile and uses the Clear method to
- ' remove all the rings from the new copy.
- 'The example looks for a profile style named
- ' "Hinged Double Full Lite." If you do not have this style
- ' in the current drawing, change the name of the style to one
- ' that exists in your drawing.

Dim doc As AecBaseDocument
Dim app As New AecBaseApplication
Dim profileStyle As AecProfileStyle
Dim profile As New AecProfile
Dim copied_profile As New AecProfile

Dim profileName As String Dim msg As String

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

' Get the specified profile style.

On Error Resume Next profileName = "Hinged - Double - Full Lite" Set profileStyle = doc.ProfileStyles.Item(profileName)

' If an error occurred, the profile probably doesn't exist.

If Err.Number 0 Then

MsgBox "Profile " & profileName & " does not exist.", vbExclamation, "Cle Exit Sub

End If

' Set an AecProfile object to the style you retrieved. Set profile = profileStyle.profile

'Copy the profile.
copied_profile.CopyFrom profile
msg = "Copied profile had " & copied_profile.Rings.count & "rings." & vbCrL

'Remove the rings in the copied profile using the Clear method. copied_profile.Clear msg = msg & "After Clear, the copied profile has " & copied_profile.Rings.cou

MsgBox msg, vbInformation, "Clear Example"

End Sub

Contains Example

```
Sub Example_Contains ()
```

'This example will check if an AecRing contains another AecRing.

```
On Error Resume Next
Dim pointList1(0 To 9) As Double
Dim pointList2(0 To 9) As Double
pointList1(0) = 0: pointList1(1) = 0
pointList1(2) = 1: pointList1(3) = 0
pointList1(4) = 1: pointList1(5) = 1
pointList1(6) = 0: pointList1(7) = 1
pointList1(8) = 0: pointList1(9) = 0
pointList1(0) = 0.25: pointList1(1) = 0.25
pointList1(2) = 0.75: pointList1(3) = 0.25
pointList1(4) = 0.75: pointList1(5) = 0.75
pointList1(6) = 0.25: pointList1(7) = 0.75
pointList1(8) = 0.25: pointList1(9) = 0.25
Dim ring1 As AecRing
Dim ring2 As AecRing
Dim profile 1 As New AecProfile
Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Dim profile As AecProfile
Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item("ContainsRing")
If profileStyle Is Nothing Then
  Set profileStyle = cprofiles.Add("ContainsRing")
End If
Set ring1 = profile1.Rings.Add
Set ring2 = profile1.Rings.Add
```

```
ring1.FromPoints (pointList1)
ring2.FromPoints (pointList2)
MsgBox "Ring1 contains Ring2 is: " & ring1.Contains(ring2)
```

ContourLinesPerSurface Example

Sub Example_ContourLinesPerSurface()

- 'This example returns the current setting of
- 'ContourLinesPerSurface. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currContourLinesPerSurface As Integer Dim newContourLinesPerSurface As Integer

- 'Retrieve the current ContourLinesPerSurface value currContourLinesPerSurface = ThisDrawing.preferences.ContourLinesPerSurf MsgBox "The current value for ContourLinesPerSurface is " & currContourLinesPerSurface)."
- 'Change the value for ContourLinesPerSurface newContourLinesPerSurface = 2001 ThisDrawing.preferences.ContourLinesPerSurface = newContourLinesPerSurf MsgBox "The new value for ContourLinesPerSurface is " & newContourLines
- 'Reset ContourLinesPerSurface to its original value
 ThisDrawing.preferences.ContourLinesPerSurface = currContourLinesPerSurf
 MsgBox "The ContourLinesPerSurface value is reset to " & currContourLinesl
 End Sub

ConvertToCurrentAreaDisplay Example

Sub Example_ConvertToCurrentAreaDisplay()

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

' Set the area display units to square yards

dbPref.AreaDisplayUnit = aecUnitSquareYard

Dim SourceArea As Double

Dim TargetArea As Double

SourceArea = 36.0 * 36.0 'Square units, assumes drawing set to inch units

TargetArea = dbPref.ConvertToCurrentAreaDisplay(SourceArea)

MsgBox "The area to convert is " & sourcearea & " square inches." & vbCrLf

"The AreaDisplayUnit is set to square yard." & vbCrLf & _

"The value for ConvertToCurrentAreaDisplay is " & TargetArea & " square

^{&#}x27;This example returns the value of ConvertToCurrentAreaDisplay

^{&#}x27; for the current drawing.

$vbInformation, "ConvertToCurrentAreaDisplay\ Example"$

ConvertToCurrentVolumeDisplay Example

Sub Example_ConvertToCurrentVolumeDisplay()

```
'This example returns the value of ConvertToCurrentVolumeDisplay
```

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

' Set the volume display units to cubic yards

dbPref.VolumeDisplayUnit = aecUnitCubicYard

Dim SourceArea As Double

Dim TargetArea As Double

SourceArea = 36# * 36# * 36# 'Cubic inches

TargetArea = dbPref.ConvertToCurrentVolumeDisplay(SourceArea)

MsgBox "The area to convert is " & SourceArea & " cubic inches." & vbCrLf

"The VolumeDisplayUnit is set to cubic yard." & vbCrLf & _

"The value for ConvertToCurrentVolumeDisplay is " & TargetArea & " " &

^{&#}x27; for the current drawing.

$vbInformation, "ConvertToCurrentVolumeDisplay\ Example"$

CoordinatePrecision Example

Sub Example_CoordinatePrecision()

'This example returns the CoordinatePrecision setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for CoordinatePrecision is: " & dbPref. Coordinate vbInformation, "CoordinatePrecision Example"

Copy Example

```
Sub Example_Copy()
  'This example creates a circle and then copies
  ' that circle. The new circle is then moved
  ' and colored red.
  ' Create the circle
  Dim circleObj As AcadCircle
  Dim center(0 To 2) As Double
  Dim radius As Double
  center(0) = 2\#: center(1) = 2\#: center(2) = 0\#
  radius = 0.5
  Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
  ZoomAll
  MsgBox "Copy the circle.", , "Copy Example"
  'Copy the circle
  Dim copyCircleObj As AcadCircle
  Set copyCircleObj = circleObj.Copy()
  ' Define the points that make up the move vector
  Dim point1(0 To 2) As Double
  Dim point2(0 To 2) As Double
  point1(0) = 0: point1(1) = 0: point1(2) = 0
  point2(0) = 2: point2(1) = 0: point2(2) = 0
  MsgBox "Move the copied circle 2 units in the X direction.", , "Copy Example
  ' Move the circle and color it
  copyCircleObj.Move point1, point2
  copyCircleObj.Color = acRed
  ZoomAll
  MsgBox "Move completed.", , "Copy Example"
```

CopyFrom Example

Sub Example_CopyFrom_AecProfile()

- 'This example copies an existing profile using the CopyFrom method.
- 'The example looks for a profile style named
- ' "Hinged Double Full Lite." If you do not have this style
- ' in the current drawing, change the name of the style to one
- ' that exists in your drawing.

Dim doc As AecBaseDocument
Dim app As New AecBaseApplication
Dim profileStyle As AecProfileStyle
Dim profile As New AecProfile
Dim copied_profile As New AecProfile

Dim profileName As String Dim msg As String

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

' Get the specified profile style.
On Error Resume Next
profileName = "Hinged - Double - Full Lite"
Set profileStyle = doc.ProfileStyles.Item(profileName)

' If an error occurred, the profile probably doesn't exist.

If Err.Number 0 Then

MsgBox "Profile " & profileName & " does not exist.", vbExclamation, "Co Exit Sub

End If

' Set an AecProfile object to the style you retrieved. Set profile = profileStyle.profile

^{&#}x27;Copy the profile.

copied_profile.CopyFrom profile

msg = "Copied profile had " & copied_profile.Rings.count & " rings." & vbCrl MsgBox msg, vbInformation, "CopyFrom Example"

CopyObjects Example

Sub Example_CopyObjects()

'This example creates a Circle object and uses the CopyObjects

Dim DOC1 As AcadDocument
Dim circleObj1 As AcadCircle, circleObj2 As AcadCircle
Dim circleObj1Copy As AcadCircle, circleObj2Copy As AcadCircle
Dim centerPoint(0 To 2) As Double
Dim radius1 As Double, radius2 As Double
Dim radius1Copy As Double, radius2Copy As Double
Dim objCollection(0 To 1) As Object
Dim retObjects As Variant

```
'Define the Circle object
centerPoint(0) = 0: centerPoint(1) = 0: centerPoint(2) = 0
radius1 = 5#: radius2 = 7#
radius1Copy = 1#: radius2Copy = 2#
```

' Create a new drawing Set DOC1 = Documents.Add

' Add two circles to the drawing
Set circleObj1 = DOC1.ModelSpace.AddCircle(centerPoint, radius1)
Set circleObj2 = DOC1.ModelSpace.AddCircle(centerPoint, radius2)
ThisDrawing.Application.ZoomAll

'Copy objects

'First put the objects to be copied into a form compatible with CopyObjects Set objCollection(0) = circleObj1
Set objCollection(1) = circleObj2

'Copy object and get back a collection of the new objects (copies) retObjects = DOC1.CopyObjects(objCollection)

'Get newly created object and apply new properties to the copies

^{&#}x27; method to make a copy of the new Circle.

Set circleObj1Copy = retObjects(0)
Set circleObj2Copy = retObjects(1)

circleObj1Copy.radius = radius1Copy
circleObj1Copy.Color = acRed
circleObj2Copy.radius = radius2Copy
circleObj2Copy.Color = acRed

ThisDrawing.Application.ZoomAll

MsgBox "Circles copied." End Sub

Count Example

Sub Example_Count()

'This example will display the number of MultiView Block Styles in the currer

Dim doc As AecArchBaseDocument
Dim cMvBlockStyles As AecMVBlockStyles

Set doc = AecArchBaseApplication.ActiveDocument

Set cMvBlockStyles = doc.MVBlockStyles
MsgBox "Number of MVBlock Styles in this drawing is: " & cMvBlockStyles

Database Example

Sub Example_Database()

- 'This example references the Database object obtained from the model space.
- ' We then display information from the Database object.

Dim Database As AcadDatabase

' Attach to Database object Set Database = ThisDrawing.ModelSpace.Database MsgBox "We now have access to the properties and methods of the Database o

'Retrieve the number of Blocks in this database MsgBox "The number of Blocks in this database is: " & Database.Blocks.coun

' Release object Set Database = Nothing MsgBox "The Database object has been released!" End Sub

DatabaseScale Example

Sub Example_DatabaseScale()

'This example returns the Database Scale setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for DatabaseScale is: " & dbPref.DatabaseScale, _ vbInformation, "DatabaseScale Example"

Delete Example

Sub Example_Delete()

```
'This example prompts users to select a drawing object, then
' deletes that object. It then issues the Undo command to
' reverse the delete.
Dim obj As AcadObject
Dim ent As AecEntity
Dim pickPt As Variant
On Error Resume Next
ThisDrawing.Utility.GetEntity obj, pickPt, "Please select an AEC object:" & vb
On Error GoTo 0
' Verify that an AEC object was selected.
If obj Is Nothing Then
 MsgBox "Nothing selected.", vbExclamation, "Delete Example"
 Exit Sub
ElseIf Not (TypeOf obj Is AecEntity) Then
 MsgBox "The object you selected is not an AEC object.", vbExclamation, "D
 Exit Sub
Else
 Set ent = obi
  Set obj = Nothing
End If
' Delete the selected object.
ent.Delete
ThisDrawing.Regen (acActiveViewport)
MsgBox "The selected object was temporarily deleted.", vbExclamation, "Dele
' Undo the delete.
```

ThisDrawing.SendCommand "_undo" & vbCr & vbCr

Depth Example

Examples:

- l AecLayoutGrid2D
- l AecLayoutGrid3D
- 1 AecMassElement

Sub Example_Depth_AecLayoutGrid2D()

'This example displays the depth of a 2D layout grid

Dim obj As Object Dim pt As Variant Dim grid As AecLayoutGrid2D Dim msg As String Dim angAsString As String

' Ask user to select a grid.
ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
 Set grid = obj
 Set obj = Nothing

- ' If the grid is radial, then Depth is an angle, in radians.
- 'There is no way to check the grid type, because there is
- ' no Shape property exposed. So this example displays both
- ' the raw value (to three decimal places) and the value in
- ' degrees (converted from radians).

msg = msg & "Depth is " & Format(grid.Depth, "0.000")

'Convert the radian value to degrees with a precision of 2. angAsString = ThisDrawing.Utility.AngleToString(grid.Depth, acDegrees, 2 msg = msg & " (if radial, this is " & angAsString & " degrees)" & vbCrLf

```
MsgBox msg, vbInformation, "Depth Example"
Else
MsgBox "Not a 2D Layout Grid", vbExclamation, "Depth Example"
End If
```

End Sub

```
Sub Example_Depth_AecLayoutGrid3D()
```

This example displays the depth of a 3D layout grid Dim obj As Object Dim pt As Variant Dim grid As AecLayoutGrid3D
ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid" If TypeOf obj Is AecLayoutGrid3D Then Set grid = obj MsgBox "Grid Depth is: " & grid.Depth, vbInformation, "Depth Example" Else MsgBox "Not a 3D Layout Grid", vbExclamation, "Depth Example" End If

End Sub

```
Sub Example_Depth_AecMassElement()
```

'This example shows the size of the mass element in its relative Y direction

```
Dim object As Object
Dim mass As AecMassElement
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecMassElement Then
count = count + 1
Set mass = object
MsgBox "Mass Element " & count & " Depth is: " & mass.Depth, vbInfor
End If
```

Next

If count = 0 Then

MsgBox "No Mass Elements Present in Drawing", vbInformation, "Depth E End If

Description Example

Sub Example_Description()

'This example will display the description of an AEC Entity.

Dim obj As AcadObject
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select AEC Object"
If TypeOf obj Is AecGeo Then
 Dim AecGeo As AecGeo
 ' AecGeo is the base class for all the drawn AEC Entities
 Set AecGeo = obj
 MsgBox "AEC Description:" & AecGeo.Description, vbInformation, "Desci
Else
 MsgBox "No AEC Entity selected", vbInformation, "Description Example"
End If

Dictionaries Example

Sub Example_Dictionaries()

- 'This example finds the current dictionary collection and
- ' adds a new dictionary to that collection.

Dim dict As AcadDictionary

Set dict = ThisDrawing.Dictionaries.Add("TEST")

MsgBox "A new dictionary called " & dict.name & " has been added to the dic End Sub

DimStyles Example

Sub Example_DimStyles()

'This example adds a new dimension style.

Dim dimStyle As AcadDimStyle

Set dimStyle = ThisDrawing.DimStyles.Add("TEST")

MsgBox "A new dimension style called " & dimStyle.name & " has been added End Sub

DisplaySilhouette Example

Sub Example_DisplaySilhouette()

- 'This example returns the current setting of
- ' DisplaySilhouette. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currDisplaySilhouette As Boolean

- 'Retrieve the current DisplaySilhouette value currDisplaySilhouette = ThisDrawing.preferences.DisplaySilhouette MsgBox "The current value for DisplaySilhouette is " & preferences.DisplaySi
- 'Change the value for DisplaySilhouette
 ThisDrawing.preferences.DisplaySilhouette = Not (currDisplaySilhouette)
 MsgBox "The new value for DisplaySilhouette is " & preferences.DisplaySilhouette)
- 'Reset DisplaySilhouette to its original value
 ThisDrawing.preferences.DisplaySilhouette = currDisplaySilhouette
 MsgBox "The DisplaySilhouette value is reset to " & preferences.DisplaySilho
 End Sub

Document Example

Sub Example_Document()

- 'This example creates a circle in model space and then
- ' finds the name of the document that the circle resides in.

Dim circleObj As AcadCircle Dim centerPoint(0 To 2) As Double Dim radius As Double

```
'Define the circle centerPoint(0) = 0#: centerPoint(1) = 0#: centerPoint(2) = 0# radius = 5#
```

' Create the Circle object in model space Set circleObj = ThisDrawing.ModelSpace.AddCircle(centerPoint, radius) ZoomAll

' Find the document name for the circle Dim currDoc As AcadDocument Set currDoc = circleObj.Document

MsgBox "The circle resides in: " & currDoc.name

EdgeWidth Example

Sub Example_EdgeWidth()

'This example will display the edge width of the selected AecPolygon

Dim obj As AcadObject
Dim pt As Variant

ThisDrawing.Utility.GetEntity obj, pt, "Select AecPolygon"

If TypeOf obj Is AecPolygon Then
Dim polygon As AecPolygon
Dim polygonStyle As AecPolygonStyle
Set polygon = obj
Set polygonStyle = polygon.Style
MsgBox "Polygon EdgeWidth = " & polygonStyle.EdgeWidth, vbInformatic
Else
MsgBox "No AecPolygon Selected", vbInformation, "EdgeWidth Example"

End Sub

End If

EdgeWidthJustify Example

Sub Example_EdgeWidthJustify()

'This example will display the edge width justification of the selected AecPoly

Dim obj As AcadObject Dim pt As Variant

ThisDrawing.Utility.GetEntity obj, pt, "Select AecPolygon"

If TypeOf obj Is AecPolygon Then

Dim polygon As AecPolygon

Dim polygonStyle As AecPolygonStyle

Set polygon = obj

Set polygonStyle = polygon.Style

Select Case polygonStyle.EdgeWidthJustify

 ${\color{blue}\textbf{Case} aec Polygon Edge Width Justification Center}$

MsgBox "Polygon EdgeWidthJustify CENTER", vbInformation, "Edge Case aecPolygonEdgeWidthJustificationIn

MsgBox "Polygon EdgeWidthJustify is IN", vbInformation, "EdgeWidthJustificationOut

MsgBox "Polygon EdgeWidthJustify is OUT", vbInformation, "EdgeW End Select

Else

MsgBox "No AecPolygon Selected", vbInformation, "EdgeWidthJustify Exa End If

ElevationModelSpace Example

Sub Example_ElevationModelSpace()

- 'This example changes the model space elevation of the current drawing
- ' and then resets it to the original value again.

Dim currElevation As Double

currElevation = ThisDrawing.ElevationModelSpace
MsgBox "The current model space elevation is " & ThisDrawing.ElevationMo

' Change the elevation
ThisDrawing.ElevationModelSpace = currElevation + 2
MsgBox "The new model space elevation is " & ThisDrawing.ElevationModel

'Reset the elevation to its original value
ThisDrawing.ElevationModelSpace = currElevation
MsgBox "The model space elevation is reset to " & ThisDrawing.ElevationMo

ElevationPaperSpace Example

Sub Example_ElevationPaperSpace()

- 'This example changes the paperspace elevation of the current drawing
- ' and then resets it to the original value again.

Dim currElevation As Double

currElevation = ThisDrawing.ElevationPaperSpace
MsgBox "The current paper space elevation is " & ThisDrawing.ElevationPaper

'Change the elevation
ThisDrawing.ElevationPaperSpace = currElevation + 2
MsgBox "The new paperspace elevation is " & ThisDrawing.ElevationPaperSpace Control of the contro

'Reset the elevation to its original value
ThisDrawing.ElevationPaperSpace = currElevation
MsgBox "The paperspace elevation is reset to " & ThisDrawing.ElevationPape

ElevationPrecision Example

Sub Example_ElevationPrecision()

'This example returns the ElevationPrecision setting for the current drawing.

Dim dbPref As AecDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for ElevationPrecision is: " & dbPref.ElevationPre vbInformation, "ElevationPrecision Example"

EndOffset Example

Sub Example_EndOffset()

'This example displays the end offset of a selected AEC Layout Curve Object

Dim obj As Object
Dim pt As Variant
Dim layoutCurve As AecLayoutCurve

ThisDrawing.Utility.GetEntity obj, pt, "Select a Node on an AEC Layout Curve

If TypeOf obj Is AecLayoutCurve Then
Set layoutCurve = obj
MsgBox "FndOffset is: " & layoutCurve FndOffs

MsgBox "EndOffset is: " & layoutCurve.EndOffset, vbInformation, "EndOffs Else

MsgBox "Not a AEC Layout Curve", vbExclamation, "EndOffset Example" End If

ExpandKey Example

Sub Example_ExpandKey()

'This example shows how to expand a layer key for the current layer standard

Dim doc As AecArchBaseDocument

Dim dbPref As AecArchBaseDatabasePreferences

Dim cLayerKeyStyles As AecLayerKeyStyles

Dim layerKeyStyle As AecLayerKeyStyle

Dim cLayerKeys As AecLayerKeys

Dim layerKey As AecLayerKey

Set doc = AecArchBaseApplication.ActiveDocument

Set cLayerKeyStyles = doc.LayerKeyStyles

Set dbPref = doc.Preferences

' Sets the layer key style to the current layer standard

Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)

MsgBox "Layer Key WALL expands to: " & layerKeyStyle.ExpandKey("WAL vbInformation, "Expand Example"

ExportFreeForm Example

Sub Example_ExportFreeForm()

```
'This example creates a mass element, after prompting users
```

- ' for its insertion point, and then exports the mass element
- 'boundaries to a file.
- 'You can run this code in conjunction with Example_ImportFreeForm,
- ' which imports the saved boundaries.

Dim massElement As AecMassElement Dim obj As AcadObject Dim center_at_origin As Boolean

Dim object_selected As Boolean object_selected = False

'Create a new mass element with a pyramid shape.

Set massElement = ThisDrawing.ModelSpace.AddCustomObject("AecMassEl massElement.Type = aecMassElementTypePyramid

'Specify a location for the mass element.

pt = ThisDrawing.Utility.GetPoint(, "Select the insertion point:")

If Err.Number 0 Then

MsgBox ("error when getting a point." & vbCrLf)

Exit Sub

End If

massElement.Location = pt massElement.Rotation = 0

' Set the pyramid's dimensions to 12' by 12' by 12' massElement.Width = 144 massElement.Depth = 144 massElement.Height = 144

- ' Indicate that original coordinates of the boundary
- ' representation are to be saved.

center_at_origin = True

'Export the mass element boundaries.
massElement.ExportFreeForm "c:\temp\freeform-massElement", center_at_ori

FacetDeviation Example

Sub Example_FacetDeviation()

'This example displays the FacetDeviation setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The setting for FacetDeviation is: " & dbPref.FacetDeviation, _ vbInformation, "FacetDeviation Example"

FlipX Example

```
Sub Example_FlipX()
```

```
'This example will flip an object that uses an AecAnchorEntToCurve about the
Dim obj As AcadObject
Dim pnt As Variant
ThisDrawing.Utility.GetEntity obj, pnt, vbCrLf & "Select a Geo anchored to a
If TypeOf obj Is AecGeo Then
Dim geo As AecGeo
Set geo = obj
Dim anchor As AecAnchor
Set anchor = geo.GetAnchor
If TypeOf anchor Is AecAnchorEntToCurve Then
    Dim curveAnchor As AecAnchorEntToCurve
    Set curveAnchor = anchor
    If curveAnchor.FlipX Then
       MsgBox "FlipX is True", vbInformation, "FlipX Example"
    Else
       MsgBox "FlipX is False", vbInformation, "FlipX Example"
    End If
    ' The following line will flip the current anchor
    curveAnchor.FlipX = Not curveAnchor.FlipX
    MsgBox "Anchor not of type AecAnchorEntToCurve", vbExclamation, "I
  End If
Else
  MsgBox "Not an AecGeo Object", vbExclamation, "FlipX Example"
End If
```

FlipY Example

```
Sub Example_FlipY()
```

```
'This example will flip an object that uses an AecAnchorEntToCurve about the
Dim obj As AcadObject
Dim pnt As Variant
ThisDrawing.Utility.GetEntity obj, pnt, vbCrLf & "Select a Geo anchored to a
If TypeOf obj Is AecGeo Then
Dim geo As AecGeo
Set geo = obi
Dim anchor As AecAnchor
Set anchor = geo.GetAnchor
If TypeOf anchor Is AecAnchorEntToCurve Then
    Dim curveAnchor As AecAnchorEntToCurve
    Set curveAnchor = anchor
    If curveAnchor.FlipY Then
       MsgBox "FlipY is True", vbInformation, "FlipY Example"
    Else
       MsgBox "FlipY is False", vbInformation, "FlipY Example"
    End If
    ' The following line will flip the current anchor
    curveAnchor.FlipY = Not curveAnchor.FlipY
    MsgBox "Anchor not of type AecAnchorEntToCurve", vbExclamation, "I
  End If
Else
  MsgBox "Not an AecGeo Object", vbExclamation, "FlipY Example"
End If
```

FlipZ Example

```
Sub Example_FlipZ()
```

```
'This example will flip an object that uses an AecAnchorEntToCurve about the
Dim obj As AcadObject
Dim pnt As Variant
ThisDrawing.Utility.GetEntity obj, pnt, vbCrLf & "Select a Geo anchored to a
If TypeOf obj Is AecGeo Then
Dim geo As AecGeo
Set geo = obi
Dim anchor As AecAnchor
Set anchor = geo.GetAnchor
If TypeOf anchor Is AecAnchorEntToCurve Then
    Dim curveAnchor As AecAnchorEntToCurve
    Set curveAnchor = anchor
    If curveAnchor.FlipZ Then
       MsgBox "FlipZ is True", vbInformation, "FlipZ Example"
    Else
       MsgBox "FlipZ is False", vbInformation, "FlipZ Example"
    End If
    ' The following line will flip the current anchor
    curveAnchor.FlipZ = Not curveAnchor.FlipZ
    MsgBox "Anchor not of type AecAnchorEntToCurve", vbExclamation, "I
  End If
Else
  MsgBox "Not an AecGeo Object", vbExclamation, "FlipZ Example"
End If
```

FromPoints Example

Sub Example_FromPoints()

'This example will make a new profile from a set of points.

```
On Error Resume Next
Dim pointList(0 To 7) As Double
pointList(0) = 0.5: pointList(1) = 0.5
pointList(2) = 0.067: pointList(3) = -0.25
pointList(4) = 0.933: pointList(5) = -0.25
pointList(6) = 0.5: pointList(7) = 0.5
Dim ring As AecRing
Dim profile As New AecProfile
Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item("FromPoints")
If profileStyle Is Nothing Then
  Set profileStyle = cprofiles.Add("FromPoints")
End If
Set ring = profile.Rings.Add
ring.FromPoints pointList
Set profileStyle.profile = profile
```

FromPolyline Example

```
Sub Example_FromPolyline()
```

'This example creates AEC Profile from a 2D Polyline.

```
Dim plineObj As AcadPolyline
```

Dim points(0 To 14) As Double

$$points(0) = 1: points(1) = 1: points(2) = 0:$$

$$points(3) = 1: points(4) = 2: points(5) = 0:$$

$$points(6) = 2: points(7) = 2: points(8) = 0:$$

$$points(9) = 3: points(10) = 2: points(11) = 0:$$

$$points(12) = 4$$
: $points(13) = 4$: $points(14) = 0$:

Set plineObj = ThisDrawing.ModelSpace.AddPolyline(points)

On Error Resume Next

Dim ring As AecRing
Dim profile As New AecProfile

Dim doc As AecArchBaseDocument

^{&#}x27; Define the 2D polyline points

^{&#}x27;The 3rd element is ignored

^{&#}x27; Create a 2D Polyline object in model space

```
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle

Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item("FromPolyline")

If profileStyle Is Nothing Then
Set profileStyle = cprofiles.Add("FromPolyline")

End If
Set ring = profile.Rings.Add

ring.FromPolyline plineObj

plineObj.delete
Set profileStyle.profile = profile
```

GenerateLayerExample

Sub Example_GenerateLayer()

'This example shows how to generate a layer with the current layer standard

Dim doc As AecArchBaseDocument

Dim dbPref As AecArchBaseDatabasePreferences

Dim cLayerKeyStyles As AecLayerKeyStyles

Dim layerKeyStyle As AecLayerKeyStyle

Dim cLayerKeys As AecLayerKeys

Dim layerKey As AecLayerKey

Set doc = AecArchBaseApplication.ActiveDocument

Set cLayerKeyStyles = doc.LayerKeyStyles

Set dbPref = doc.Preferences

' Sets the layer key style to the current layer standard

Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)

Dim layer As AcadLayer

Set layer = layerKeyStyle.GenerateLayer("WALL") MsgBox "Layer Key WAI vbInformation, "Expand Example"

GetAnchor Example

```
Sub Example_GetAnchor()
```

```
'This example will get the anchor attached to an AEC Entity.
Dim ent As AcadEntity
Dim geo As AecGeo
Dim pt As Variant
Dim anchor As AecAnchor
ThisDrawing.Utility.GetEntity ent, pt, "Selected anchored object"
If TypeOf ent Is AecGeo Then
  Set geo = ent
  Set anchor = geo.GetAnchor
  If anchor Is Nothing Then
    MsgBox "AEC Entity is not anchored", vbInformation, "GetAnchor Exan
  Else
    MsgBox "AEC Entity is anchored with an " & anchor.ObjectName & " ot
  End If
Else
  MsgBox "AEC Entity not selected", vbInformation, "GetAnchor Example"
End If
```

GetBoundingBox Example

Sub Example_GetBoundingBox()

- 'This example creates a line in model space. It then finds the
- ' bounding box for the line and displays the corners of the box.

Dim startPoint(0 To 2) As Double Dim endPoint(0 To 2) As Double Dim lineObj As AcadLine

'Create the Line object in model space startPoint(0) = 2#: startPoint(1) = 2#: startPoint(2) = 0# endPoint(0) = 4#: endPoint(1) = 4#: endPoint(2) = 0# Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint) ZoomAll

Dim minExt As Variant Dim maxExt As Variant

- ' Return the bounding box for the line and return the minimum
- ' and maximum extents of the box in the minExt and maxExt variables. lineObj.GetBoundingBox minExt, maxExt
- ' Print the min and max extents

```
MsgBox "The extents of the bounding box for the line are:" & vbCrLf _ & "Min Extent: " & minExt(0) & "," & minExt(1) & "," & minExt(2) _ & vbCrLf & "Max Extent: " & maxExt(0) & "," & maxExt(1) & "," & maxI
```

GetExtensionDictionary Example

Sub Example_GetExtensionDictionary()

- 'This example creates a Circle object in model space and
- ' adds a new Extension Dictionary to the Circle object

Dim EDictionary As AcadDictionary Dim circleObj As AcadCircle Dim centerPoint(0 To 2) As Double Dim radius As Double

' Define the Circle object centerPoint(0) = 0: centerPoint(1) = 0: centerPoint(2) = 0 radius = 5#

'Create the Circle object in model space Set circleObj = ThisDrawing.ModelSpace.AddCircle(centerPoint, radius)

ThisDrawing.Application.ZoomAll

MsgBox "Circle object Extension Dictionary state before: " & circleObj.HasEx

'Create an Extension Dictionary for the new Circle Set EDictionary = circleObj.GetExtensionDictionary

MsgBox "Circle object Extension Dictionary state after: " & circleObj.HasExte End Sub

GetXData Example

lineObj.SetXData DataType, Data

```
Sub Example_GetXData()
```

'This example creates a line and attaches extended data to that line.

```
' Create the line
Dim lineObj As AcadLine
Dim startPt(0 To 2) As Double, endPt(0 To 2) As Double
startPt(0) = 1#: startPt(1) = 1#: startPt(2) = 0#
endPt(0) = 5\#: endPt(1) = 5\#: endPt(2) = 0\#
Set lineObj = ThisDrawing.ModelSpace.AddLine(startPt, endPt)
ZoomAll
'Initialize all the xdata values. Note that first data in the list should be
'application name and first datatype code should be 1001
Dim DataType(0 To 9) As Integer
Dim Data(0 To 9) As Variant
Dim reals3(0 To 2) As Double
Dim worldPos(0 To 2) As Double
DataType(0) = 1001: Data(0) = "Test\_Application"
DataType(1) = 1000: Data(1) = "This is a test for xdata"
DataType(2) = 1003: Data(2) = "0"
                                              ' laver
DataType(3) = 1040: Data(3) = 1.23479137438413E+40 'real
DataType(4) = 1041: Data(4) = 1237324938
                                                  ' distance
DataType(5) = 1070: Data(5) = 32767
                                               ' 16 bit Integer
DataType(6) = 1071: Data(6) = 32767
                                               ' 32 bit Integer
DataType(7) = 1042: Data(7) = 10
                                             'scaleFactor
reals3(0) = -2.95: reals3(1) = 100: reals3(2) = -20
DataType(8) = 1010: Data(8) = reals3
                                              ' real
worldPos(0) = 4: worldPos(1) = 400.999999999: worldPos(2) = 2.798989
                                                ' world space position
DataType(9) = 1011: Data(9) = worldPos
' Attach the xdata to the line
```

'Return the xdata for the line Dim xdataOut As Variant Dim xtypeOut As Variant lineObj.GetXData "", xtypeOut, xdataOut

Groups Example

Sub Example_Groups()

- 'This example finds the current Groups collection and
- ' adds a new group to that collection.

Dim groupColl As AcadGroups Set groupColl = ThisDrawing.Groups

' Create a dimension style named "TEST" in current drawing Dim testGroup As AcadGroup Set testGroup = groupColl.Add("TEST")

MsgBox "A new group called " & testGroup.name & " has been added to the C End Sub

Handle Example

```
Sub Example_Handle()
  'This example creates several objects in model space.
  ' It then iterates through model space and displays the
  ' Handle for each object found.
  'Create a Ray object in model space
  Dim rayObj As AcadRay
  Dim basePoint(0 To 2) As Double
  Dim SecondPoint(0 To 2) As Double
  basePoint(0) = 3\#: basePoint(1) = 3\#: basePoint(2) = 0\#
  SecondPoint(0) = 1#: SecondPoint(1) = 3#: SecondPoint(2) = 0#
  Set rayObj = ThisDrawing.ModelSpace.AddRay(basePoint, SecondPoint)
  'Create a polyline object in model space
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 5) As Double
  points(0) = 3: points(1) = 7
  points(2) = 9: points(3) = 2
  points(4) = 3: points(5) = 5
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  'Create a line object in model space
  Dim lineObj As AcadLine
  Dim startPoint(0 To 2) As Double
  Dim endPoint(0 To 2) As Double
  startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
  endPoint(0) = 2: endPoint(1) = 2: endPoint(2) = 0
  Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)
  'Create a circle object in model space
  Dim circObj As AcadCircle
  Dim centerPt(0 To 2) As Double
  Dim radius As Double
  centerPt(0) = 5: centerPt(1) = 3: centerPt(2) = 0
  radius = 3
```

Set circObj = ThisDrawing.ModelSpace.AddCircle(centerPt, radius)

```
' Create an ellipse object in model space
Dim ellObj As AcadEllipse
Dim majAxis(0 To 2) As Double
Dim center(0 To 2) As Double
Dim radRatio As Double
center(0) = 5\#: center(1) = 5\#: center(2) = 0\#
majAxis(0) = 10: majAxis(1) = 20#: majAxis(2) = 0#
radRatio = 0.3
Set ellObj = ThisDrawing.ModelSpace.AddEllipse(center, majAxis, radRatio)
ZoomAll
' Iterate through the model space collection and display
' the handle of each entity found.
Dim entHandle As String
Dim entry As AcadEntity
For Each entry In ThisDrawing.ModelSpace
  entHandle = entry.handle
  entry.Highlight (True)
```

MsgBox "The handle of this object is " & entHandle, vbInformation, "Handl

Next

entry. Highlight (False)

HandleToObject Example

Sub Example_HandleToObject()

- 'This example creates a spline in model space. It then returns the
- ' handle for the spline. The spline is returned from the handle, and
- ' then colored.

```
' Create the spline
```

Dim splineObj As AcadSpline

Dim startTan(0 To 2) As Double

Dim endTan(0 To 2) As Double

Dim fitPoints(0 To 8) As Double

ZoomAll

' Find the handle of the spline

Dim handle As Integer

handle = splineObj.handle

MsgBox "The handle of the Spline is: " & splineObj.handle, , "HandleToObjec

' Find an object from a given handle

Dim tempObj As AcadObject

Set tempObj = ThisDrawing.HandleToObject(handle)

' Now use the newly initialized object variable to color the object red tempObj.Color = acRed

ThisDrawing.Regen True
MsgBox "The Spline is now red.", , "HandleToObject Example"

Has Example

Sub Example_Has()

'This example will check the wall styles collection for a given wall style.

```
Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cWallStyles As AecWallStyles
Set cWallStyles = doc.WallStyles
Dim styleName As String
styleName = "Brick"
If cWallStyles.Has(styleName) Then
MsgBox "Wall style: ' " & styleName & " ' in drawing", vbInformation, "Has E
Else
MsgBox "Wall style: ' " & styleName & " ' not in drawing", vbInformation, "H
End If
```

HasExtensionDictionary Example

Sub Example_HasExtensionDictionary()

- 'This example will iterate through each object in the current drawing and
- ' determine if that object has an associated Extension Dictionary

Dim DrawingObject As AcadObject Dim ExtensionDictionaryResults As String

' Make sure this drawing contains objects before we continue
If ThisDrawing.ModelSpace.count = 0 Then
 MsgBox "There are no objects in the current drawing."
 Exit Sub
End If

For Each DrawingObject In ThisDrawing.ModelSpace

' Determine if object contains Extension Dictionary Select Case DrawingObject.HasExtensionDictionary

Case True

ExtensionDictionaryResults = ExtensionDictionaryResults & Drawing(Case False

ExtensionDictionaryResults = ExtensionDictionaryResults & Drawing(End Select

Next

MsgBox ExtensionDictionaryResults End Sub

Height Example

Examples:

- 1 AecClipVol
- 1 AecLayoutGrid3D
- 1 AecMassElement

```
Sub Example_Height_AecClipVol()
```

'This example shows the height of the upper cut plane above the 'building elevation line position

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Height is: " & clip.Height, vbInformation
End If

Next
```

If count = 0 Then

MsgBox "No ClipVol Present in Drawing", vbInformation, "Height Example

End If

```
Sub Example_Height_AecLayoutGrid3D()
```

Dim object As Object Dim grid As AecLayoutGrid3D Dim count As Integer 'initalize count = 0 For Each object In ThisDrawing.ModelSpace If TypeOf object Is AecLayoutGrid3D Then count = count + 1 Set grid = object MsgBox "3D Grid" & count & " Height is: " & grid.Height, vbInformation, "Height Example" End If Next If count = 0 Then MsgBox "No 3D Layout Grids Present in Drawing", vbInformation, "Height Example" End If

End Sub

```
Sub Example_Height_AecMassElement()
```

'This example shows the size of the mass element in its relative Z direction

```
Dim object As Object
Dim mass As AecMassElement
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecMassElement Then
count = count + 1
Set mass = object
MsgBox "Mass Element " & count & " Height is: " & mass.Height, vbInfo End If
```

Next

```
If count = 0 Then

MsgBox "No Mass Elements Present in Drawing", vbInformation, "Height I

End If
```

Highlight Example

Sub Example_Highlight()

```
'This example creates several objects in model space.
' It then iterates through model space and highlights each
' object and displays it's Entity Name.
'Create a Ray object in model space
Dim rayObj As AcadRay
Dim basePoint(0 To 2) As Double
Dim SecondPoint(0 To 2) As Double
basePoint(0) = 3\#: basePoint(1) = 3\#: basePoint(2) = 0\#
SecondPoint(0) = 1#: SecondPoint(1) = 3#: SecondPoint(2) = 0#
Set rayObj = ThisDrawing.ModelSpace.AddRay(basePoint, SecondPoint)
'Create a polyline object in model space
Dim plineObj As AcadLWPolyline
Dim points(0 To 5) As Double
points(0) = 3: points(1) = 7
points(2) = 9: points(3) = 2
points(4) = 3: points(5) = 5
Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
plineObj.Closed = True
'Create a line object in model space
Dim lineObj As AcadLine
Dim startPoint(0 To 2) As Double
Dim endPoint(0 To 2) As Double
startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
endPoint(0) = 2: endPoint(1) = 2: endPoint(2) = 0
Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)
'Create a circle object in model space
Dim circObj As AcadCircle
Dim centerPt(0 To 2) As Double
Dim radius As Double
centerPt(0) = 20: centerPt(1) = 30: centerPt(2) = 0
radius = 3
```

```
Set circObj = ThisDrawing.ModelSpace.AddCircle(centerPt, radius)
```

```
' Create an ellipse object in model space
Dim ellObj As AcadEllipse
Dim majAxis(0 To 2) As Double
Dim center(0 To 2) As Double
Dim radRatio As Double
center(0) = 5\#: center(1) = 5\#: center(2) = 0\#
majAxis(0) = 10: majAxis(1) = 20#: majAxis(2) = 0#
radRatio = 0.3
Set ellObj = ThisDrawing.ModelSpace.AddEllipse(center, majAxis, radRatio)
ZoomAll
' Iterate through the model space collection and
' highlight each entity.
Dim entName As String
Dim entry As AcadEntity
For Each entry In ThisDrawing.ModelSpace
  entName = entry.ObjectName
  entry.Highlight (True)
  MsgBox "The name of this object is " & entName, vbInformation, "Highligh
  entry. Highlight (False)
Next
```

Hyperlinks Example

```
Sub Example_HyperLinks()
  'This example creates a Circle object in model space and
  ' adds a new Hyperlink to its Hyperlink collection
  Dim Hyperlinks As AcadHyperlinks
  Dim Hyperlink As AcadHyperlink
  Dim circleObj As AcadCircle
  Dim centerPoint(0 To 2) As Double
  Dim radius As Double
  Dim HLList As String
  ' Define the Circle object
  centerPoint(0) = 0: centerPoint(1) = 0: centerPoint(2) = 0
  radius = 5#
  'Create the Circle object in model space
  Set circleObj = ThisDrawing.ModelSpace.AddCircle(centerPoint, radius)
  ThisDrawing.Application.ZoomAll
  ' Get reference to the Circle's Hyperlinks collection
  Set Hyperlinks = circleObj.Hyperlinks
  ' Add a new Hyperlink complete with all properties
  Set Hyperlink = Hyperlinks.Add("AutoDesk")
  Hyperlink.URL = "www.autodesk.com"
  Hyperlink.URLDescription = "Autodesk Main Site"
  Hyperlink.URLNamedLocation = "MY LOCATION"
  ' Read and display a list of existing Hyperlinks and
  ' their properties for this object
  For Each Hyperlink In Hyperlinks
                                                                   " & vbCrI
    HLList = HLList & "
    HLList = HLList & "URL: " & Hyperlink.URL & vbCrLf
    HLList = HLList & "URL Description: " & Hyperlink.URLDescription & vl
    HLList = HLList & "URL Named Location: " & Hyperlink.URLNamedLocation: "
```

Next

MsgBox "The circle has " & Hyperlinks.count & " Hyperlink: " & vbCrLf & H End Sub

ImportFreeForm Example

Sub Example_ImportFreeForm()

- 'This example imports a previously exported boundary representation
- ' of a mass element and applies it to a free-form mass element.
- ' It prompts users to specify the location of the new mass element.
- 'You can run this code in conjunction with Example_ExportFreeForm
- '(run Example_ExportFreeForm first).

Dim massElement As AecMassElement Dim pt As Variant Dim center_at_origin As Boolean

'Create a new mass element Set massElement = ThisDrawing.ModelSpace.AddCustomObject("AecMassEle

'Set the mass element type to "free-form" massElement.Type = aecMassElementTypeFreeForm

'Select a location for the mass element
pt = ThisDrawing.Utility.GetPoint(, "Select the insertion point:")
If Err.Number 0 Then
 MsgBox ("error when getting a point." & vbCrLf)
 Exit Sub
End If

massElement.Location = pt

- 'Use mass element Location as insertion point. center_at_origin = True
- 'Import the boundary representation.
 massElement.ImportFreeForm "c:\temp\freeform-massElement"

Index Example

Examples:

- l AecLayerKey
- 1 AecLayerOverrideSetting
- 1 AecLayoutCurveNode
- 1 AecLayoutGrid2DNode
- 1 AecLayoutGrid2DXNode
- 1 AecLayoutGrid2DYNode
- 1 AecLayoutGrid3DNode
- 1 AecLayoutGrid3DXNode
- l <u>AecLayoutGrid3DYNode</u>
- l <u>AecLayoutGrid3DZNode</u>
- 1 AecRing
- l AecViewBlock

Sub Example_Index_AecLayerKey()

'This example shows the index of the layer generated by the key

Dim db As New AecBaseDatabase Dim layerKey As AecLayerKey

db.Init ThisDrawing.Database

Set layerKey = db.LayerKeyStyles.Item(0).Keys.Item(0)
MsgBox "Layer Index: " & layerKey.Index, vbInformation, "Index Example"

End If

```
Sub Example_Index_AecLayerOverrideSetting()
  'This example shows the index of the overridesetting
  Dim db As New AecBaseDatabase
  Dim setting As AecLayerOverrideSetting
  db.Init ThisDrawing.Database
  Set setting = db.LayerKeyStyles.Item(0).overrideSettings.Item(0)
  MsgBox "Setting Index: " & setting.Index, vbInformation, "Index Example"
End Sub
Sub Example_Index_AecLayoutCurveNode()
 'This example displays the color of the nodes owner layout curve
 Dim obj As Object
 Dim pt As Variant
 Dim layoutCurve As AecLayoutCurve
 Dim node As AecLayoutCurveNode
 ThisDrawing.Utility.GetEntity obj, pt, "Select a Layout Curve"
 If TypeOf obj Is AecLayoutCurve Then
   Set layoutCurve = obj
   Set node = layoutCurve.nodes.Item(0)
   MsgBox "Color is: " & node.layoutCurve.Color, vbInformation, "LayoutCurv
```

MsgBox "Not a Layout Curve", vbExclamation, "LayoutCurve Example"

```
Sub Example_Index_AecLayoutGrid2DNode()

"This example displays the index of a layout node

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
Set grid = obj
Set node = grid.XNodes.Item(0)
MsgBox "Node Index is: " & node.Index, vbInformation, "Index Example"

Else
MsgBox "Not a 2D Layout Grid", vbExclamation, "Index Example"
End If
```

Sub Example_Index_AecLayoutGrid2DXNode()

'This example displays the index of a layout Xnode

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
Set grid = obj
Set node = grid.XNodes.Item(0)
MsgBox "XNode Index is: " & node.Index, vbInformation, "Index Example'
Else
MsgBox "Not a 2D Layout Grid", vbExclamation, "Index Example"
End If
```

Sub Example_Index_AecLayoutGrid2DYNode()

'This example displays the index of a layout Ynode

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
   Set node = grid.YNodes.Item(0)
   MsgBox "YNode Index is: " & node.Index, vbInformation, "Index Example"
Else
   MsgBox "Not a 2D Layout Grid", vbExclamation, "Index Example"
End If
```

End Sub

Sub Example_Index_AecLayoutGrid3DNode()

'This example displays the index of a layout node

```
Dim obj As Object
  Dim pt As Variant
  Dim grid As AecLayoutGrid3D
  Dim node As AecLayoutGrid3DNode
  ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
  If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.XNodes.Item(0)
    MsgBox "Node Index is: " & node.Index, vbInformation, "Index Example"
  Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "Index Example"
  End If
End Sub
Sub Example_Index_AecLayoutGrid3DXNode()
  This example displays the index of a layout Xnode
  Dim obj As Object
  Dim pt As Variant
  Dim grid As AecLayoutGrid3D
  Dim node As AecLayoutGrid3DNode
  ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
  If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.XNodes.Item(0)
    MsgBox "XNode Index is: " & node.Index, vbInformation, "Index Example'
  Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "Index Example"
  End If
```

```
Sub Example_Index_AecLayoutGrid3DYNode()
  'This example displays the index of a layout Ynode
  Dim obj As Object
  Dim pt As Variant
  Dim grid As AecLayoutGrid3D
  Dim node As AecLayoutGrid3DNode
  ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
  If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.YNodes.Item(0)
    MsgBox "YNode Index is: " & node.Index, vbInformation, "Index Example"
  Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "Index Example"
  End If
End Sub
Sub Example_Index_AecLayoutGrid3DZNode()
  'This example displays the index of a layout Znode
  Dim obj As Object
  Dim pt As Variant
  Dim grid As AecLayoutGrid3D
  Dim node As AecLayoutGrid3DNode
  ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
  If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
```

```
Set node = grid.ZNodes.Item(0)
   MsgBox "ZNode Index is: " & node.Index, vbInformation, "Index Example'
Else
   MsgBox "Not a 3D Layout Grid", vbExclamation, "Index Example"
End If
```

```
Sub Example_Index_AecRing()
```

'This example will display the index of the ring

Dim db As New AecBaseDatabase
Dim ring As AecRing

db.Init ThisDrawing.Database

Set ring = db.ProfileStyles.Item(0).Profile.rings.Item(0)
MsgBox "Ring Index: " & ring.Index, vbInformation, "Index Example"

End Sub

Sub Example_Index_AecViewBlock()

'This example shows the index of the first viewblock of a multiviewblock

Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef
Dim viewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then

Set blockRef = obj

Set viewBlocks = blockRef.viewBlocks
MsgBox "Index of View Block 1: " & viewBlocks.Item(0).Index, vbInforma
Else
MsgBox "Not a Multiview Block", vbInformation, "Index Example"
End If

Init Example

Sub Example_Init()

- 'This example initializes the application object, using
- ' the current drawing, then lists the number of objects in
- ' the drawing's model space.

Dim app As New AecBaseApplication Dim doc As AecBaseDocument

Dim nbr_of_objects As Integer Dim msg As String

- ' Initialize the application object. app.Init ThisDrawing.Application
- ' Access the current drawing database. Set doc = app.ActiveDocument
- ' Get the number of objects in the drawing's model space. nbr_of_objects = doc.ModelSpace.Count

msg = "There are " & nbr_of_objects & " objects in this drawing's model space MsgBox msg, vbInformation, "Init Example"

Intersect Example

```
Sub Example_Intersect ()
```

'This example will add make two profiles from rings, and intersect the first with

```
On Error Resume Next
Dim pointList1(0 To 9) As Double
Dim pointList2(0 To 7) As Double
pointList1(0) = 0: pointList1(1) = 0
pointList1(2) = 1: pointList1(3) = 0
pointList1(4) = 1: pointList1(5) = 1
pointList1(6) = 0: pointList1(7) = 1
pointList1(8) = 0: pointList1(9) = 0
pointList2(0) = 0.5: pointList2(1) = 0.5
pointList2(2) = 0.067: pointList2(3) = -0.25
pointList2(4) = 0.933: pointList2(5) = -0.25
pointList2(6) = 0.5: pointList2(7) = 0.5
Dim ring1 As AecRing
Dim ring2 As AecRing
Dim profile1 As New AecProfile
Dim profile As New AecProfile
Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item("IntersectRing")
If profileStyle Is Nothing Then
  Set profileStyle = cprofiles.Add("IntersectRing")
End If
Set ring1 = profile1.Rings.Add
Set ring2 = profile2.Rings.Add
```

```
ring1.FromPoints (pointList1)
ring2.FromPoints (pointList2)
profile1.Intersect profile2
Set profileStyle.profile = profile1
```

IntersectWith Example

```
Sub Example_IntersectWith()
  'This example creates a line and circle and finds the points at
  ' which they intersect.
  ' Create the line
  Dim lineObj As AcadLine
  Dim startPt(0 To 2) As Double
  Dim endPt(0 To 2) As Double
  startPt(0) = 1: startPt(1) = 1: startPt(2) = 0
  endPt(0) = 5: endPt(1) = 5: endPt(2) = 0
  Set lineObj = ThisDrawing.ModelSpace.AddLine(startPt, endPt)
  ' Create the circle
  Dim circleObj As AcadCircle
  Dim centerPt(0 To 2) As Double
  Dim radius As Double
  centerPt(0) = 3: centerPt(1) = 3: centerPt(2) = 0
  radius = 1
  Set circleObj = ThisDrawing.ModelSpace.AddCircle(centerPt, radius)
  ZoomAll
  ' Find the intersection points between the line and the circle
  Dim intPoints As Variant
  intPoints = lineObj.IntersectWith(circleObj, acExtendNone)
  ' Print all the intersection points
  Dim I As Integer, j As Integer, k As Integer
  Dim str As String
  If VarType(intPoints) <> vbEmpty Then
    For I = LBound(intPoints) To UBound(intPoints)
       str = "Intersection Point[" & k & "] is: " & intPoints(j) & "," & intPoints(j
       MsgBox str, , "IntersectWith Example"
       str = ""
       I = I + 2
       j = j + 3
       k = k + 1
```

Next End If End Sub

Item Example

Sub Example_Item()

- 'This example lists the number of keys in the Standard
- ' layer key styles collection, and then lists the name of
- ' each layer key styles collection in the drawing. The example
- ' uses both the string and integer methods of specifying the
- ' index of an Item.

Dim app As New AecBaseApplication Dim doc As AecBaseDocument Dim layerkeystyles As AecLayerKeyStyles Dim keystyle As AecLayerKeyStyle Dim msg As String

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

' Get the drawing's collection of layer key styles.

Set layerkeystyles = doc layerkeystyles

Set layerkeystyles = doc.layerkeystyles

msg = "Number of Layer Key Styles in this drawing is: " & layerkeystyles.Cou

'Get the standard layer key style, if it exists (it should!)

On Error Resume Next

Set keystyle = layerkeystyles.Item("Standard")

'List the number of keys in the Standard layer key style.

If Err Then

msg = msg & "Standard layer key style does not exist in this drawing" Err.Clear

Else

msg = "Standard layer key style contains " & keystyle.Keys.Count & " keys.'

End If

On Error GoTo 0

^{&#}x27;Loop through the layer key styles collection, listing the name

^{&#}x27; of each layer key style in the collection.

MsgBox msg, vbInformation, "Item Example"

Keys Example

```
Sub Example_Keys()
```

End Sub

'This example shows the how to get the layer key from the layer key style.

```
Dim doc As AecArchBaseDocument
  Dim dbPref As AecArchBaseDatabasePreferences
  Dim cLayerKeyStyles As AecLayerKeyStyles
  Dim layerKeyStyle As AecLayerKeyStyle
  Dim cLayerKeys As AecLayerKeys
  Dim layerKey As AecLayerKey
  Set doc = AecArchBaseApplication.ActiveDocument
  Set cLayerKeyStyles = doc.LayerKeyStyles
  Set dbPref = doc.Preferences
  ' Sets the layer key style to the current layer standard
  Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
  Set cLayerKeys = layerKeyStyle.Keys
  For Each layerKey In cLayerKeys
    Debug.Print layerKey.Name
    Debug.Print " Color - " & layerKey.Color
    Debug.Print " Layer
                        - " & layerKey.Layer
    Debug.Print "LineType - " & layerKey.Linetype
    Debug.Print " Lineweight - " & layerKey.Lineweight
    Debug.Print " Plotstyle - " & layerKey.PlotStyleName
    Debug.Print " Plottable - " & layerKey.Plottable
    Debug.Print "Removable - " & layerKey.Removeable
  Next
< PRE class= Code >
```

Layer Example

Sub Example_Layer()

'This example lists the layer keys in the layer key style

Dim app As New AecBaseApplication
Dim doc As AecBaseDocument
Dim dbPref As AecBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys

Dim layor Koy As As a layor Koy

Dim layerKey As AecLayerKey

Dim msg As String

'Initialize the application object and access the current drawing. app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

' Get the drawing's collection of layer key styles. Set cLayerKeyStyles = doc.layerkeystyles

' Get the preferences object. Set dbPref = doc.Preferences

' Set the layer key style to the current layer standard.
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)

' Get the collection of layer keys in the style. Set cLayerKeys = layerKeyStyle.Keys

' Loop through the collection and list some properties of each key.

^{&#}x27; of the document's standard layer.

```
For Each layerKey In cLayerKeys

msg = msg & " " & layerKey.Name & ":" & vbCrLf

msg = msg & " Color - " & layerKey.Color & vbCrLf

msg = msg & " Layer - " & layerKey.Layer & vbCrLf

msg = msg & " LineType - " & layerKey.Linetype & vbCrLf

msg = msg & " Lineweight - " & layerKey.Lineweight & vbCrLf

msg = msg & " Plotstyle - " & layerKey.PlotStyleName & vbCrLf

Next
```

MsgBox msg, vbInformation, "Layer Example"

LayerFile Example

```
Sub Example_LayerFile()
```

```
'This example displays the LayerFile setting for the current drawing.
```

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

```
Dim layerfile As String
```

layerfile = dbPref.LayerFile

'Test if layer file is set

If layerfile = "" Then

layerfile = "not set"

End If

MsgBox "The current value for LayerFile is: " & layerfile, vbInformation, "LayerFile is: " & layerFile is: " &

LayerKeyStyle Example

Examples:

- l AecLayerKey
- l AecLayerKeys
- 1 AecLayerOverrideSetting
- 1 <u>AecLayerOverrideSettings</u>

```
Sub Example_LayerKeyStyle_AecLayerKey()
```

'This example shows the layer style generated by the key

```
Dim db As New AecBaseDatabase
Dim layerKey As AecLayerKey
```

db.Init ThisDrawing.Database

```
Set layerKey = db.LayerKeyStyles.Item(0).Keys.Item(0)
MsgBox "Layer Style: " & layerKey.layerKeyStyle.Name, vbInformation, "LayerKeyStyle.Name, vbInformation," LayerKeyStyle.Name, vbInformation, "LayerKeyStyle.Name, vbInformation, "LayerKeyStyle.Name, vbInformation, vbInformation,
```

End Sub

```
Sub Example_LayerKeyStyle_AecLayerKeys()
```

'This example shows the name of the owner layer key style

```
Dim db As New AecBaseDatabase
Dim layerKeys As AecLayerKeys
```

db.Init ThisDrawing.Database

Set layerKeys = db.LayerKeyStyles.Item(0).Keys

Sub Example_LayerKeyStyle_AecLayerOverrideSetting()

'This example shows the name of the owner of the overridesetting

Dim db As New AecBaseDatabase
Dim setting As AecLayerOverrideSetting

db.Init ThisDrawing.Database

Set setting = db.LayerKeyStyles.Item(0).overrideSettings.Item(0)
MsgBox "Setting Owners Name: " & setting.layerKeyStyle.Name, vbInformation of the setting o

End Sub

Sub Example_LayerKeyStyle_AecLayerOverrideSettings()

Dim db As New AecBaseDatabase
Dim overrideSettings As AecLayerOverrideSettings

db.Init ThisDrawing.Database

Set overrideSettings = db.LayerKeyStyles.Item(0).overrideSettings
MsgBox "Layer Override Settings owner name: " & overrideSettings.layerKey

LayerKeyStyles Example

Sub Example_LayerKeyStyles()

'This example will display the number of Layer Key Styles in the current draw

Dim doc As AecArchBaseDocument
Dim cLayerKeyStyles As AecLayerKeyStyles

Set doc = AecArchBaseApplication.ActiveDocument

Set cLayerKeyStyles = doc.LayerKeyStyles
MsgBox "Number of Layer Key Styles in this drawing is: " & cLayerKeyStyle

Layers Example

Sub Example_Layers()

- 'This example finds the current Layers collection and
- ' adds a new layer to that collection.

Dim layerColl As AcadLayers Set layerColl = ThisDrawing.Layers

'Create a dimension style named "TEST" in current drawing Dim testlayer As AcadLayer Set testlayer = layerColl.Add("TEST")

MsgBox "A new layer called " & testlayer.name & " has been added to the Lay End Sub

LayerStandard Example

```
Sub Example_LayerStandard()
```

```
'This example displays the LayerStandard setting for the current drawing.
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
Dim laystandard As String
laystandard = dbPref.layerstandard
'Test if layer standard is set
If laystandard = "" Then
  laystandard = "not set"
End If
MsgBox "The current value for LayerStandard is: " & laystandard, _
```

vbInformation, "LayerStandard Example"

LayoutCurve Example

Examples:

l <u>AecLayoutCurveNode</u>

1 AecLayoutCurveNodes

Sub Example_LayoutCurve_AecLayoutCurveNode()

'This example displays the color of the nodes owner layout curve

Dim obj As Object

Dim pt As Variant

Dim layoutCurve As AecLayoutCurve

Dim node As AecLayoutCurveNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a Layout Curve"

If TypeOf obj Is AecLayoutCurve Then

Set layoutCurve = obj

Set node = layoutCurve.nodes.Item(0)

MsgBox "Color is: " & node.layoutCurve.Color, vbInformation, "LayoutCurv Else

MsgBox "Not a Layout Curve", vbExclamation, "LayoutCurve Example" End If

End Sub

Sub Example_LayoutCurve_AecLayoutCurveNodes()

'This example displays the color of the node collections owner curve

Dim obj As Object Dim pt As Variant Dim layoutCurve As AecLayoutCurve Dim nodes As AecLayoutCurveNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a Layout Curve"

If TypeOf obj Is AecLayoutCurve Then
 Set layoutCurve = obj
 Set nodes = layoutCurve.nodes
 MsgBox "Color is: " & nodes.layoutCurve.Color, vbInformation, "LayoutCur
Else
 MsgBox "Not a Layout Curve", vbExclamation, "LayoutCurve Example"
End If

LayoutGrid2D Example

Examples:

- l AecLayoutGrid2DNode
- 1 AecLayoutGrid2DNodes
- 1 AecLayoutGrid2DXNode
- 1 AecLayoutGrid2DXNodes
- 1 AecLayoutGrid2DYNode
- l <u>AecLayoutGrid2DYNodes</u>

```
Sub Example_LayoutGrid2D_AecLayoutGrid2DNode()
```

'This example displays the index of a layout node

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
   Set node = grid.XNodes.Item(0)
   MsgBox "Node LayoutGrid's Depth is: " & node.LayoutGrid2D.Depth, vbIr.
```

Else

Mag Day "Not a 2D Layout Grid" whEvelemation "Layout Grid 2D Evernle

MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example End If

```
Sub Example_LayoutGrid2D_AecLayoutGrid2DNodes()
```

'This example displays the index of a layout node

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim nodes As AecLayoutGrid2DNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
 Set grid = obj
 Set nodes = grid.XNodes
 MsgBox "Node collection owner depth is: " & nodes.LayoutGrid2D.Depth,
 Else
 MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example

End Sub

End If

Sub Example_LayoutGrid2D_AecLayoutGrid2DXNode()

'This example displays the depth of the owner grid of a layout xnode

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
   Set node = grid.XNodes.Item(0)
   MsgBox "Owner Grid Depth is: " & node.LayoutGrid2D.Depth, vbInformat
Else
```

MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example End If

End Sub

Sub Example_LayoutGrid2D_AecLayoutGrid2DXNodes()

'This example displays the depth of the owner grid of a collection of layout Xn

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim nodes As AecLayoutGrid2DNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
 Set grid = obj
 Set nodes = grid.XNodes
 MsgBox "Owner Grid Depth is: " & nodes.LayoutGrid2D.Depth, vbInforma
Else
 MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example

End Sub

End If

Sub Example_LayoutGrid2D_AecLayoutGrid2DYNode()

'This example displays the depth of the owner grid of a layout xnode

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim node As AecLayoutGrid2DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
   Set node = grid.YNodes.Item(0)
   MsgBox "Owner Grid Depth is: " & node.LayoutGrid2D.Depth, vbInformat
Else
   MsgBox "Not a 2D Layout Grid", vbEyslamation, "LayoutGrid2D Eyample
```

MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example End If

End Sub

Sub Example_LayoutGrid2D_AecLayoutGrid2DYNodes()

'This example displays the depth of the owner grid of a collection of layout Yn

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D
Dim nodes As AecLayoutGrid2DNodes
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
Set grid = obj
Set nodes = grid.YNodes
MsgBox "Owner Grid Depth is: " & nodes.LayoutGrid2D.Depth, vbInforma
Else
```

MsgBox "Not a 2D Layout Grid", vbExclamation, "LayoutGrid2D Example End If

LayoutGrid3D Example

Examples:

- 1 AecLayoutGrid3DNode
- l <u>AecLayoutGrid3DNodes</u>
- 1 AecLayoutGrid3DXNode
- 1 AecLayoutGrid3DXNodes
- 1 AecLayoutGrid3DYNode
- l <u>AecLayoutGrid3DYNodes</u>
- l AecLayoutGrid3DZNode
- 1 AecLayoutGrid3DZNodes

Sub Example_LayoutGrid3D_AecLayoutGrid3DNode()

'This example displays the index of a layout node

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim node As AecLayoutGrid3DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.XNodes.Item(0)
    MsgBox "Node LayoutGrid's Depth is: " & node.LayoutGrid3D.Depth, vbIr
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example
End If
```

Sub Example_LayoutGrid3D_AecLayoutGrid3DNodes()

'This example displays the index of a layout node

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

Dim nodes As AecLayoutGrid3DNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

If TypeOf obj Is AecLayoutGrid3D Then

Set grid = obj

Set nodes = grid.XNodes

MsgBox "Node collection owner depth is: " & nodes.LayoutGrid3D.Depth, Else

MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example End If

End Sub

Sub Example_LayoutGrid3D_AecLayoutGrid3DXNode()

'This example displays the depth of the owner grid of a layout xnode

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

Dim node As AecLayoutGrid3DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.XNodes.Item(0)
    MsgBox "Owner Grid Depth is: " & node.LayoutGrid3D.Depth, vbInformat
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example
End If
```

Sub Example_LayoutGrid3D_AecLayoutGrid3DXNodes()

'This example displays the depth of the owner grid of a collection of layout Xn

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim nodes As AecLayoutGrid3DNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
Set nodes = grid.XNodes
MsgBox "Owner Grid Depth is: " & no
```

MsgBox "Owner Grid Depth is: " & nodes.LayoutGrid3D.Depth, vbInforma Else

MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example End If

End Sub

Sub Example_LayoutGrid3D_AecLayoutGrid3DYNode()

'This example displays the depth of the owner grid of a layout Ynode

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim node As AecLayoutGrid3DNode
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set node = grid.YNodes.Item(0)
    MsgBox "Owner Grid Depth is: " & node.LayoutGrid3D.Depth, vbInformat
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example
```

End Sub

End If

Sub Example_LayoutGrid3D_AecLayoutGrid3DYNodes()

'This example displays the depth of the owner grid of a collection of layout Yn

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim nodes As AecLayoutGrid3DNodes
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Set nodes = grid.YNodes
    MsgBox "Owner Grid Depth is: " & nodes.LayoutGrid3D.Depth, vbInforma
Else
```

MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example End If

```
Sub Example_LayoutGrid3D_AecLayoutGrid3DZNode()
```

'This example displays the depth of the owner grid of a layout Znode

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim node As AecLayoutGrid3DNode

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
   Set grid = obj
   Set node = grid.ZNodes.Item(0)
   MsgBox "Owner Grid Depth is: " & node.LayoutGrid3D.Depth, vbInformat
Else
```

MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example End If

End Sub

Sub Example_LayoutGrid3D_AecLayoutGrid3DZNodes()

'This example displays the depth of the owner grid of a collection of layout Zno

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
Dim nodes As AecLayoutGrid3DNodes

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
```

```
If TypeOf obj Is AecLayoutGrid3D Then
   Set grid = obj
```

Set nodes = grid.ZNodes

MsgBox "Owner Grid Depth is: " & nodes.LayoutGrid3D.Depth, vbInforma Else

MsgBox "Not a 3D Layout Grid", vbExclamation, "LayoutGrid3D Example End If

Layouts Example

Sub Example_Layouts()

- 'This example will access the Layouts collection for the current drawing
- ' and list basic information about the Layouts in the drawing.

Dim Layouts As AcadLayouts, Layout As ACADLayout Dim msg As String

' Get layouts collection from document object Set Layouts = ThisDrawing.Layouts

msg = vbCrLf & vbCrLf 'Start with a space

' Get the names of every layout in this drawing For Each Layout In Layouts msg = msg & Layout.name & vbCrLf Next

' Display a list of available layouts
MsgBox "There are " & Layouts.count & " layouts in " & _
ThisDrawing.WindowTitle & ":" & msg
End Sub

LeaderExtension1 Example

Sub Example_LeaderExtension1() 'This example will find the 1st Leader Extension for a bubble attached to a ' column grid Dim obj As AcadObject Dim bubble As AecMVBlockRef Dim anchor As AecAnchor Dim leaderAnchor As AecAnchorLeadEntToNode Dim pt As Variant ThisDrawing.Utility.GetEntity obj, pt, "Select bubble" If TypeOf obj Is AecMVBlockRef Then Set bubble = obj **Set** anchor = obj.GetAnchor If TypeOf anchor Is AecAnchorLeadEntToNode Then **Set** leaderAnchor = anchor MsgBox "Leader Extension1 = " & leaderAnchor.LeaderExtension1 Else MsgBox "Not anchored to column grid", vbInformation, "Example Leade End If Else MsgBox "Not a bubble", vbInformation, "Example LeaderExtension1"

End Sub

End If

LeaderExtension2 Example

Sub Example_LeaderExtension2()

```
' column grid
Dim obj As AcadObject
Dim bubble As AecMVBlockRef
Dim anchor As AecAnchor
Dim leaderAnchor As AecAnchorLeadEntToNode
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select bubble"
If TypeOf obj Is AecMVBlockRef Then
  Set bubble = obj
  Set anchor = obj.GetAnchor
  If TypeOf anchor Is AecAnchorLeadEntToNode Then
    Set leaderAnchor = anchor
    MsgBox "Leader Extension2 = " & leaderAnchor.LeaderExtension2
  Else
    MsgBox "Not anchored to column grid", vbInformation, "Example Leade
  End If
Else
  MsgBox "Not a bubble", vbInformation, "Example LeaderExtension2"
End If
```

'This example will find the 2nd Leader Extension for a bubble attached to a

LeftOffset Example

Sub Example_LeftOffset()

- 'This example returns the left offset of the selected object
- ' to the grid assembly.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- ' assembly.

Dim ent As AcadEntity
Dim geo As AecGeo
Dim anchor As AecAnchor

Dim offset As String

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "LeftOffset Example" ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

If anchor Is Nothing Then

MsgBox "Selected object is not anchored.", vbExclamation, "LeftOffset E ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

MsgBox "Left offset of object: " & anchor.LeftOffset, vbInformation, "Le

End If
Else
MsgBox "Object selected is not an AEC entity.", vbInformation, "LeftOffset
End If

Limits Example

```
Sub Example_Limits()
```

- 'This example finds the current limits for the drawing.
- ' It then changes the limits for the drawing. The grid
- ' is turned on to show the limits.

'Turn on the grid for the active viewport
ThisDrawing.ActiveViewport.GridOn = True

ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport

' Find the current limits

Dim currLimits As Variant

currLimits = ThisDrawing.Limits

MsgBox "The current drawing limits are " & vbCrLf _

& "Lower-left corner " & ThisDrawing.Limits(0) & ", " & ThisDrawing.L

& "Upper-right corner " & ThisDrawing.Limits(2) & ", " & ThisDrawing.l

' Change the limits

Dim newLimits(0 To 3) As Double

newLimits(0) = 2#: newLimits(1) = 2#: newLimits(2) = 4#: newLimits(3) = 4#

ThisDrawing.Limits = newLimits

ThisDrawing.Regen acActiveViewport

MsgBox "The new drawing limits are " & vbCrLf _

& "Lower-left corner " & ThisDrawing.Limits(0) & ", " & ThisDrawing.L

& "Upper-right corner " & ThisDrawing.Limits(2) & ", " & ThisDrawing.l

' Reset the drawing limits

ThisDrawing.Limits = currLimits

ThisDrawing.Regen acActiveViewport

MsgBox "The drawing limits have been reset to " & vbCrLf _

& "Lower-left corner " & ThisDrawing.Limits(0) & ", " & ThisDrawing.L

& "Upper-right corner " & ThisDrawing.Limits(2) & ", " & ThisDrawing.l End Sub

LinearDisplayFormat Example

Sub Example_LinearDisplayFormat()

linearformat = "Fractional"

```
'This example returns the LinearDispalyFormat setting for the current drawing
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
'Convert the linear display format to a string.
Dim linearformat As String
Select Case dbPref.LinearDisplayFormat
Case 1
  linearformat = "Scientific"
Case 2
  linearformat = "Decimal"
Case 3
  linearformat = "Engineering"
Case 4
  linearformat = "Architectural"
Case 5
```

End Select

MsgBox "The current value for LinearDisplayFormat is: " & linearformat, vbIı

LinearPrecision Example

Sub Example_LinearPrecision()

'This example returns the LinearPrecision setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for LinearPrecision is: " & dbPref.LinearPrecisio vbInformation, "LinearPrecision Example"

LinearUnit Example

```
Sub Example_LinearUnit()
```

```
'This example returns the LinearUnit setting for the current drawing.
```

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

Dim unit As String

Select Case dbPref.LinearUnit

Case aecUnitInch

unit = "inch"

Case aecUnitFoot

unit = "foot"

Case aecUnitYards

unit = "yard"

Case aecUnitMillimeter

unit = "millimeters"

Case aecUnitCentimeter

unit = "centimeters"

^{&#}x27;Convert the linear display unit to a string.

```
Case aecUnitDecimeter
```

```
unit = "decimeters"
```

Case aecUnitMeter

unit = "meters"

End Select

MsgBox "The current value for LinearUnit is: " & unit, vbInformation, "Linear

LineType Example

```
Sub Example_LineType ()
```

'This example shows the linetype of the layer generated by the key

```
Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKey
Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
Set cLayerKeys = layerKeyStyle.Keys
For Each layerKey In cLayerKeys
  Debug.Print layerKey.Name
  Debug.Print " Color
                       - " & layerKey.Color
  Debug.Print " Layer - " & layerKey.Layer
  Debug.Print "LineType - " & layerKey.Linetype
  Debug.Print " Lineweight - " & layerKey.Lineweight
  Debug.Print " Plotstyle - " & layerKey.PlotStyleName
  Debug.Print " Plottable - " & layerKey.Plottable
  Debug.Print " Removable - " & layerKey.Removeable
Next
```

Linetypes Example

Sub Example_Linetypes()

- 'This example finds the linetypes collection and
- ' lists all the available linetypes in the collection.

Dim linetypeColl As AcadLineTypes Dim entry As AcadLineType Dim msg As String

'Return the linetype collection object of the active document Set linetypeColl = ThisDrawing.Linetypes ' List all available linetypes For Each entry In linetypeColl msg = msg & entry.name & vbCrLf

Next

MsgBox "The linetypes available in this drawing are:" & vbCrLf & msg, vbInf **End Sub**

LinetypeScale Example

Sub Example_LinetypeScale()

- 'This example creates a line and finds the linetype scale
- ' for the line. It then changes the linetype scale, and finally
- ' resets the linetype scale back to the original value.

Dim startPoint(0 To 2) As Double

Dim endPoint(0 To 2) As Double

Dim lineObj As AcadLine

Dim currLTScale As Double

'Create a Line object in model space

startPoint(0) = 2#: startPoint(1) = 2#: startPoint(2) = 0#

endPoint(0) = 4#: endPoint(1) = 4#: endPoint(2) = 0#

Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)

lineObj.Update

currLTScale = lineObj.LinetypeScale

MsgBox "The linetype scale for the line is:" & lineObj.LinetypeScale, vbInfor

' Set the linetype scale of a Line to .5

lineObj.LinetypeScale = 0.5

lineObj.Update

MsgBox "The new linetype scale for the line is:" & lineObj.LinetypeScale, vbI

'Reset the linetype scale of a Line to what is was before

lineObj.LinetypeScale = currLTScale

lineObj.Update

MsgBox "The linetype scale for the line is reset to:" & lineObj.LinetypeScale, End Sub

LineWeight Example

```
Sub Example_LineWeight ()
```

'This example shows the lineweight of the layer generated by the key

```
Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKey
Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
Set cLayerKeys = layerKeyStyle.Keys
For Each layerKey In cLayerKeys
  Debug.Print layerKey.Name
  Debug.Print " Color
                       - " & layerKey.Color
  Debug.Print " Layer - " & layerKey.Layer
  Debug.Print "LineType - " & layerKey.Linetype
  Debug.Print " Lineweight - " & layerKey.Lineweight
  Debug.Print " Plotstyle - " & layerKey.PlotStyleName
  Debug.Print " Plottable - " & layerKey.Plottable
  Debug.Print " Removable - " & layerKey.Removeable
Next
```

LineweightDisplay Example

Sub Example_LineweightDisplay()

- 'This example reads and modifies the preference value which controls
- ' whether symbol names may include extended character sets, or more
- ' than 31 characters.

' When finished, this example resets the preference value back to

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.LineWeightDisplay MsgBox "The LineweightDisplay preference is set to: " & originalValue
- ' Modify the LineweightDisplay preference by toggling the value ACADPref.LineWeightDisplay = Not (ACADPref.LineWeightDisplay) newValue = ACADPref.LineWeightDisplay
 MsgBox "The LineweightDisplay preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- ' * Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref. Line Weight Display = original Value

MsgBox "The LineweightDisplay preference was reset back to: " & originalVa End Sub

^{&#}x27; it's original value.

Location Example

```
Sub Example_Location ()

'This example displays the location of an AEC Entity

Dim obj As Object
Dim pt As Variant
Dim geo As AecGeo

ThisDrawing.Utility.GetEntity obj, pt, "Select an AEC Entity"

If TypeOf obj Is AecGeo Then
    Set geo = obj
    MsgBox "AEC Entity Location: " & geo.Location(0) & ", " & geo.Location(Else
    MsgBox "Not an AEC Entity", vbExclamation, "Location Example"
End If
```

LowerExtension Example

```
Sub Example_LowerExtension()
```

'This example shows the distance of the lower cut plane below building elevation line position

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Lower Extension is: " & clip.LowerExte
End If

Next

If count = 0 Then
MsgBox "No ClipVol Present in Drawing", vbInformation, "LowerExtenstio
End If
```

MaskBlockStyles Example

Sub Example_MaskBlockStyles()

Dim doc As AecBaseDocument

'This example will display the number of Mask Block Styles in the current dra

```
Dim app As New AecBaseApplication
Dim cMaskBlocks As AecMaskBlockStyles

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument
Set cMaskBlocks = doc.MaskBlockStyles

MsgBox "Number of MaskBlock Styles in this drawing is: "__
```

& cMaskBlocks.Count, vbInformation, "MaskBlockStyles Example"

MassGroup Example

Examples:

1 AecMassElement

l AecMassGroup

Sub Example_MassGroup_AecMassElement()

'This example shows the name of the mass' group, if applicable

```
Dim object As Object
Dim mass As AecMassElement
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecMassElement Then
count = count + 1
Set mass = object
MsgBox "Mass Element " & count & " Mass Group Name is: " & mass.M
End If

Next

If count = 0 Then
MsgBox "No Mass Elements Present in Drawing", vbInformation, "MassGreend If
```

Sub Example_MassGroup_AecMassGroup()

'This example shows the Mass Group the Mass Element is attached to.

Dim ent As AcadEntity
Dim pt As Variant
Dim mass As AecMassElement
Dim massGroup As AecMassGroup

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Mass Element"

```
If TypeOf ent Is AecMassElement Then
Set mass = ent
Set massGroup = mass.massGroup
If Not massGroup Is Nothing Then
MsgBox "Mass Group is: " & massGroup.Name, vbInformation, "Mass G
Else
MsgBox "Mass Element is not part of a Mass Group", vbInformation, "M
End If
```

Else

MsgBox "Not an AecMassElement", vbExclamation, "Mass Group Example End If

MassGroups Example

Sub Example_MassGroups()

'This example will display the number of Mass Groups in the current drawing.

```
Dim app As New AecBaseApplication
Dim Aecdoc As AecBaseDocument
Dim massgroups As AecMassGroups
```

app.Init ThisDrawing.Application

```
Set Aecdoc = app.ActiveDocument
Set Aecdoc = AecArchBaseApplication.ActiveDocument
Set massgroups = Aecdoc.massgroups
MsgBox "There are " & massgroups.count & " mass groups in this drawing"
```

MaxActiveViewports Example

Sub Example_MaxActiveViewports()

- 'This example returns the current setting of
- ' MaxActiveViewports. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currMaxActiveViewports As Integer Dim newMaxActiveViewports As Integer

- 'Retrieve the current MaxActiveViewports value currMaxActiveViewports = ThisDrawing.preferences.MaxActiveViewports MsgBox "The current value for MaxActiveViewports is " & currMaxActiveViewports"
- 'Change the value for MaxActiveViewports
 newMaxActiveViewports = 20
 ThisDrawing.preferences.MaxActiveViewports = newMaxActiveViewports
 MsgBox "The new value for MaxActiveViewports is " & newMaxActiveView]
- 'Reset MaxActiveViewports to its original value
 ThisDrawing.preferences.MaxActiveViewports = currMaxActiveViewports
 MsgBox "The MaxActiveViewports value is reset to " & currMaxActiveViewp
 End Sub

MeasurementUnit Example

Sub Example_MeasurementUnit()

```
'This example returns the MeasurementUnit setting for the current drawing.
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
'Convert the measurement unit to a string.
Dim unit As String
If dbPref.MeasurementUnit = acEnglish Then
  unit = "English"
Else
  unit = "Metric"
End If
MsgBox "The current value for MeasurementUnit is " & unit, _
  vbInformation, "MeasurementUnit Example"
```

Mirror Example

```
Sub Example_Mirror()
  'This example creates a light weight polyline
  ' and then mirrors that polyline.
  ' Create the polyline
  Dim plineObj As AcadLWPolyline
  Dim points (0 To 11) As Double
  points(0) = 1: points(1) = 1
  points(2) = 1: points(3) = 2
  points(4) = 2: points(5) = 2
  points(6) = 3: points(7) = 2
  points(8) = 4: points(9) = 4
  points(10) = 4: points(11) = 1
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  ZoomAll
  ' Define the mirror axis
  Dim point1(0 To 2) As Double
  Dim point2(0 To 2) As Double
  point1(0) = 0: point1(1) = 4.25: point1(2) = 0
  point2(0) = 4: point2(1) = 4.25: point2(2) = 0
  MsgBox "Mirror the polyline.", , "Mirror Example"
  ' Mirror the polyline
  Dim mirrorObj As AcadLWPolyline
  Set mirrorObj = plineObj.Mirror(point1, point2)
  mirrorObj.Color = acRed
  ZoomAll
  MsgBox "Mirror completed.", , "Mirror Example"
End Sub
```

Mirror3D Example

```
Sub Example_Mirror3D()
  'This example creates a box in model space.
  ' It then mirrors the box about a plane and colors
  ' the new box red.
  Dim boxObj As Acad3DSolid
  Dim length As Double, width As Double, height As Double
  Dim center(0 To 2) As Double
  ' Define the box
  center(0) = 5\#: center(1) = 5\#: center(2) = 0
  length = 5#: width = 7: height = 10#
  'Create the box (3DSolid) object in model space
  Set boxObj = ThisDrawing.ModelSpace.AddBox(center, length, width, height)
  ' Define the mirroring plane with three points
  Dim mirrorPt1(0 To 2) As Double
  Dim mirrorPt2(0 To 2) As Double
  Dim mirrorPt3(0 To 2) As Double
  mirrorPt1(0) = 1.25: mirrorPt1(1) = 0: mirrorPt1(2) = 0
  mirrorPt2(0) = 1.25: mirrorPt2(1) = 2: mirrorPt2(2) = 0
  mirrorPt3(0) = 1.25: mirrorPt3(1) = 2: mirrorPt3(2) = 2
  ' Mirror the box
  Dim mirrorBoxObj As Acad3DSolid
  Set mirrorBoxObj = boxObj.Mirror3D(mirrorPt1, mirrorPt2, mirrorPt3)
  mirrorBoxObj.Color = acRed
  'Change the viewing direction of the viewport
  Dim NewDirection(0 To 2) As Double
  NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
```

ThisDrawing.ActiveViewport.direction = NewDirection

ZoomAll

ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport

ModelSpace Example

```
Sub Example_ModelSpace()
```

- 'This example adds a line and a circle to model space.
- 'The line is added using a user-defined variable representing
- ' the model space. The circle is added without using the
- ' user-defined variable. Either use of the ModelSpace
- ' property is valid.

```
' Define the line
```

Dim lineObj As AcadLine

Dim startPoint(0 To 2) As Double

Dim endPoint(0 To 2) As Double

startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0

endPoint(0) = 4: endPoint(1) = 4: endPoint(2) = 0

' Add the line to model space using the mspace variable

Dim mspace As AcadModelSpace

Set mspace = ThisDrawing.ModelSpace

Set lineObj = mspace.AddLine(startPoint, endPoint)

' Define a circle

Dim circleObj As AcadCircle

Dim center(0 To 2) As Double

Dim radius As Double

center(0) = 4: center(1) = 4: center(2) = 0

radius = 1

' Add the circle to modelspace without using the mspace variable Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)

ZoomAll

Modified Example

'This code is placed in a module and is run to initialized the event handler.

```
Public eh As New EventHandler
Sub Notify()
  Dim obj As AcadObject
  For Each obj In ThisDrawing.ModelSpace
    If TypeOf obj Is AecMassElement Then
         'This sets an event handler (defined below) on the Mass Element
       Set eh.obj = obj
       Exit For
    End If
  Next
End Sub
'This is the Event Handler code
' It is placed in a "Class Module"
Public WithEvents obj As AcadObject
Private Sub obj_Modified(ByVal pObject As IAcadObject)
  Dim ent As AecMassElement
  Set ent = pObject
  Dim loc As Variant
  loc = ent.Location
  Dim str As String
  str = "AecMassElement at (" & loc(0) & ", " & _
                    loc(1) & ", " & _
```

loc(2) & ")"

MsgBox str, , "Object Event"

Move Example

```
Sub Example_Move()
  'This example creates a circle and then performs
  ' a move on that circle.
  ' Create the circle
  Dim circleObj As AcadCircle
  Dim center(0 To 2) As Double
  Dim radius As Double
  center(0) = 2#: center(1) = 2#: center(2) = 0#
  radius = 0.5
  Set circleObj = ThisDrawing.ModelSpace.AddCircle(center, radius)
  ZoomAll
  ' Define the points that make up the move vector
  Dim point1(0 To 2) As Double
  Dim point2(0 To 2) As Double
  point1(0) = 0: point1(1) = 0: point1(2) = 0
  point2(0) = 2: point2(1) = 0: point2(2) = 0
  MsgBox "Move the circle 2 units in the X direction.", , "Move Example"
  ' Move the circle
  circleObj.Move point1, point2
  ZoomAll
  MsgBox "Move completed.", , "Move Example"
```

MVBlockRef Example

Examples:

1 AecViewBlock

1 AecViewBlocks

Sub Example_Name_AecViewBlock()

'This example shows the MVBlockRef's number of colors

Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then

Set blockRef = obj

MsgBox "Colors of View Blocks MVBlockRef: " & blockRef.viewBlocks.It Else

MsgBox "Not a Multiview Block", vbInformation, "MVBlockRef Example" End If

End Sub

Sub Example_Name_AecViewBlocks()

'This example shows the number of colors of the owner of the Viewblock collecti

Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef

Dim viewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then

Set blockRef = obj

Set viewBlocks = blockRef.viewBlocks

MsgBox "Colors of MVBlockRef Owner: " & viewBlocks.MVBlockRef.Co

Else

MsgBox "Not a Multiview Block", vbInformation, "MVBlockRef Example"

End If

MVBlockStyles Example

Sub Example_MVBlockStyles()

'This example will display the number of MultiView Block Styles in the currer

Dim doc As AecArchBaseDocument
Dim cMvBlockStyles As AecMVBlockStyles

Set doc = AecArchBaseApplication.ActiveDocument

Set cMvBlockStyles = doc.MVBlockStyles
MsgBox "Number of MVBlock Styles in this drawing is: " & cMvBlockStyles

Name Example

Examples:

- l AecMassGroup
- l AecCamera
- l AecLayerKey
- 1 AecViewBlock

```
Sub Example_Name_AecMassGroup()
```

'This example shows the name of the mass group

```
Dim object As Object
Dim group As AecMassGroup
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace
```

```
If TypeOf object Is AecMassGroup Then
count = count + 1
Set group = object
MsgBox "Mass Group " & count & " Name is: " & group.Name, vbInforn
```

End If

Next

```
If count = 0 Then
```

MsgBox "No Mass Element Groups Present in Drawing", vbInformation, "NEnd If

```
Sub Example_Name_AecCamera()
```

'This example displays the name of a selected camera object

Dim obj As Object
Dim pt As Variant
Dim camera As AecCamera

ThisDrawing.Utility.GetEntity obj, pt, "Select a Camera"

```
If TypeOf obj Is AecCamera Then
Set camera = obj
MsgBox "Name is: " & camera.Name, vbInformation, "Name Example"
Else
MsgBox "Not a Camera", vbExclamation, "Name Example"
End If
```

End Sub

Sub Example_Name_AecLayerKey()

'This example shows the name of the layer generated by the key

Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKeys

Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard

```
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
      Set cLayerKeys = layerKeyStyle.Keys
       For Each layerKey In cLayerKeys
             Debug.Print layerKey.Name
             Debug.Print " Color - " & layerKey.Color
             Debug.Print " Layer - " & layerKey.Layer
             Debug.Print "LineType - " & layerKey.Linetype
             Debug.Print " Lineweight - " & layerKey.Lineweight
             Debug.Print " Plotstyle - " & layerKey.PlotStyleName
             Debug.Print " Plottable - " & layerKey.Plottable
             Debug.Print "Removable - " & layerKey.Removeable
       Next
End Sub
Sub Example_Name_AecViewBlock()
'This example shows the name of the first viewblock of a multiviewblock
       Dim obj As Object
       Dim pt As Variant
       Dim blockRef As AecMVBlockRef
       Dim viewBlocks As AecViewBlocks
      ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"
       If TypeOf obj Is AecMVBlockRef Then
              Set blockRef = obi
             Set viewBlocks = blockRef.viewBlocks
             MsgBox "Name of View Block 1: " & viewBlocks.Item(0).Name, vbInformation of View Block 1: " & viewBlock 1: "
       Else
             MsgBox "Not a Multiview Block", vbInformation, "Name Example"
       End If
```

Node Example

```
Sub Example_Node()
  'This example will add anchor a new mass element to a 2D layout grid in the
  ' drawing.
  Dim grid As AecLayoutGrid2D
  Dim mass As AecMassElement
  Dim pt As Variant
  Dim obj As AcadObject
  ThisDrawing.Utility.GetEntity obj, pt, "Select grid to attach to"
  If TypeOf obj Is AecLayoutGrid2D Then
    Set grid = obj
    Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement
    Dim anchor As New AecAnchorEntToLayoutNode
    anchor.Reference = grid
    ' anchor the mass element to the last node on the grid
    Dim lastNode As Long
    lastNode = grid.XNodes.Count * grid.YNodes.Count
    anchor.Node = lastNode
    mass.AttachAnchor anchor
  Else
    MsgBox "No Layout Grid selected", vbInformation, "Node Example"
```

End Sub

End If

Nodes Example

Sub Example_Nodes()

'This example displays the number of nodes on the layout curve

Dim obj As Object
Dim pt As Variant
Dim layoutCurve As AecLayoutCurve

ThisDrawing.Utility.GetEntity obj, pt, "Select a Node on an AEC Layout Curve

If TypeOf obj Is AecLayoutCurve Then
Set layoutCurve = obj
MsgBox "Number of nodes is: " & layoutCurve.nodes.Count, vbInformation,
Else
MsgBox "Not a AEC Layout Curve", vbEyclamation, "Nodes Example"

MsgBox "Not a AEC Layout Curve", vbExclamation, "Nodes Example" End If

Normal Example

```
Sub Example_Normal ()

'This example displays the normal vector of an AEC Entity

Dim obj As Object
Dim pt As Variant
Dim geo As AecGeo

ThisDrawing.Utility.GetEntity obj, pt, "Select an AEC Entity"

If TypeOf obj Is AecGeo Then
    Set geo = obj
    MsgBox "AEC Entity Normal: " & geo.Normal(0) & ", " & geo.Normal(1) & Else
    MsgBox "Not an AEC Entity", vbExclamation, "Normal Example"
End If
```

NorthRotation Example

Sub Example_NorthRotation()

'This example returns the NorthRotation setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for NorthRotation is: " & dbPref.NorthRotation, _ vbInformation, "NorthRotation Example"

ObjectID Example

```
Sub Example ObjectID()
  'This example creates several objects in model space.
  ' It then iterates through model space and displays the
  'ObjectID for each object found.
  'Create a Ray object in model space
  Dim rayObj As AcadRay
  Dim basePoint(0 To 2) As Double
  Dim SecondPoint(0 To 2) As Double
  basePoint(0) = 3\#: basePoint(1) = 3\#: basePoint(2) = 0\#
  SecondPoint(0) = 1#: SecondPoint(1) = 3#: SecondPoint(2) = 0#
  Set rayObj = ThisDrawing.ModelSpace.AddRay(basePoint, SecondPoint)
  'Create a polyline object in model space
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 5) As Double
  points(0) = 3: points(1) = 7
  points(2) = 9: points(3) = 2
  points(4) = 3: points(5) = 5
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  'Create a line object in model space
  Dim lineObj As AcadLine
  Dim startPoint(0 To 2) As Double
  Dim endPoint(0 To 2) As Double
  startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
  endPoint(0) = 2: endPoint(1) = 2: endPoint(2) = 0
  Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)
  'Create a circle object in model space
  Dim circObj As AcadCircle
  Dim centerPt(0 To 2) As Double
  Dim radius As Double
  centerPt(0) = 5: centerPt(1) = 3: centerPt(2) = 0
  radius = 3
```

```
Set circObj = ThisDrawing.ModelSpace.AddCircle(centerPt, radius)
```

```
' Create an ellipse object in model space
Dim ellObj As AcadEllipse
Dim majAxis(0 To 2) As Double
Dim center(0 To 2) As Double
Dim radRatio As Double
center(0) = 5\#: center(1) = 5\#: center(2) = 0\#
majAxis(0) = 10: majAxis(1) = 20#: majAxis(2) = 0#
radRatio = 0.3
Set ellObj = ThisDrawing.ModelSpace.AddEllipse(center, majAxis, radRatio)
ZoomAll
' Iterate through the model space collection and display
' the ObjectID of each entity found.
Dim entObjectID As Long
Dim entry As AcadEntity
For Each entry In ThisDrawing.ModelSpace
  entObjectID = entry.objectID
  entry.Highlight (True)
  MsgBox "The ObjectID of this object is " & entObjectID, vbInformation, "C
  entry. Highlight (False)
Next
```

ObjectIDToObject Example

```
Sub Example_ObjectIDToObject()
```

- 'This example creates a spline in model space. It then returns the
- ' handle for the spline. The spline is returned from the handle, and
- ' then colored.

```
' Create the spline
```

Dim splineObj As AcadSpline

Dim startTan(0 To 2) As Double

Dim endTan(0 To 2) As Double

Dim fitPoints(0 To 8) As Double

ZoomAll

' Find the objectID of the spline Dim objectID As Long

objectID = splineObj.objectID

MsgBox "The objectID of the Spline is: " & splineObj.objectID, , "ObjectIDTc

' Find an object from a given objectID

Dim tempObj As AcadObject

Set tempObj = ThisDrawing.ObjectIdToObject(objectID)

' Now use the newly initialized object variable to color the object red tempObj.Color = acRed

ThisDrawing.Regen True
MsgBox "The Spline is now red.", , "ObjectIDToObject Example"

ObjectName Example

radius = 3

```
Sub Example_ObjectName()
  'This example creates several objects in model space.
  ' It then iterates through model space and displays the
  ' Object Name for each object found.
  'Create a Ray object in model space
  Dim rayObj As AcadRay
  Dim basePoint(0 To 2) As Double
  Dim SecondPoint(0 To 2) As Double
  basePoint(0) = 3\#: basePoint(1) = 3\#: basePoint(2) = 0\#
  SecondPoint(0) = 1#: SecondPoint(1) = 3#: SecondPoint(2) = 0#
  Set rayObj = ThisDrawing.ModelSpace.AddRay(basePoint, SecondPoint)
  'Create a polyline object in model space
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 5) As Double
  points(0) = 3: points(1) = 7
  points(2) = 9: points(3) = 2
  points(4) = 3: points(5) = 5
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  'Create a line object in model space
  Dim lineObj As AcadLine
  Dim startPoint(0 To 2) As Double
  Dim endPoint(0 To 2) As Double
  startPoint(0) = 0: startPoint(1) = 0: startPoint(2) = 0
  endPoint(0) = 2: endPoint(1) = 2: endPoint(2) = 0
  Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint)
  'Create a circle object in model space
  Dim circObj As AcadCircle
  Dim centerPt(0 To 2) As Double
  Dim radius As Double
  centerPt(0) = 20: centerPt(1) = 30: centerPt(2) = 0
```

```
Set circObj = ThisDrawing.ModelSpace.AddCircle(centerPt, radius)
```

```
' Create an ellipse object in model space
Dim ellObj As AcadEllipse
Dim majAxis(0 To 2) As Double
Dim center(0 To 2) As Double
Dim radRatio As Double
center(0) = 5\#: center(1) = 5\#: center(2) = 0\#
majAxis(0) = 10: majAxis(1) = 20#: majAxis(2) = 0#
radRatio = 0.3
Set ellObj = ThisDrawing.ModelSpace.AddEllipse(center, majAxis, radRatio)
ZoomAll
' Iterate through the model space collection and display
' the object name of each entity found.
Dim objName As String
Dim entry As AcadEntity
For Each entry In ThisDrawing.ModelSpace
  objName = entry.ObjectName
  entry.Highlight (True)
  MsgBox "The name of this object is " & objName, vbInformation, "ObjectN
  entry. Highlight (False)
Next
```

ObjectSortByPlotting Example

Sub Example_ObjectSortByPlotting()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by plotting order.
- ' When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortByPlotting MsgBox "The ObjectSortByPlotting preference is set to: " & originalValue
- ' Modify the ObjectSortByPlotting preference by toggling the value ACADPref.ObjectSortByPlotting = Not (ACADPref.ObjectSortByPlotting) newValue = ACADPref.ObjectSortByPlotting
 MsgBox "The ObjectSortByPlotting preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref. Object Sort By Plotting = original Value

MsgBox "The ObjectSortByPlotting preference was reset back to: " & original End Sub

ObjectSortByPSOutput Example

Sub Example_ObjectSortByPSOutput()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by PostScript output order.
- 'When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortByPSOutput MsgBox "The ObjectSortByPSOutput preference is set to: " & originalValue
- ' Modify the ObjectSortByPSOutput preference by toggling the value ACADPref.ObjectSortByPSOutput = Not (ACADPref.ObjectSortByPSOutput) newValue = ACADPref.ObjectSortByPSOutput
 MsgBox "The ObjectSortByPSOutput preference has been set to: " & newValu
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.ObjectSortByPSOutput = originalValue

MsgBox "The ObjectSortByPSOutput preference was reset back to: " & origin End Sub

ObjectSortByRedraws Example

Sub Example_ObjectSortByRedraws()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by redraw order.
- ' When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortByRedraws MsgBox "The ObjectSortByRedraws preference is set to: " & originalValue
- ' Modify the ObjectSortByRedraws preference by toggling the value ACADPref.ObjectSortByRedraws = Not (ACADPref.ObjectSortByRedraws) newValue = ACADPref.ObjectSortByRedraws

 MsgBox "The ObjectSortByRedraws preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.ObjectSortByRedraws = originalValue

MsgBox "The ObjectSortByRedraws preference was reset back to: " & origina End Sub

ObjectSortByRegens Example

Sub Example_ObjectSortByRegens()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by regeneration order.
- ' When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortByRegens MsgBox "The ObjectSortByRegens preference is set to: " & originalValue
- ' Modify the ObjectSortByRegens preference by toggling the value ACADPref.ObjectSortByRegens = Not (ACADPref.ObjectSortByRegens) newValue = ACADPref.ObjectSortByRegens
 MsgBox "The ObjectSortByRegens preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- this preference in effect

ACADPref.ObjectSortByRegens = originalValue

MsgBox "The ObjectSortByRegens preference was reset back to: " & original End Sub

ObjectSortBySelection Example

Sub Example_ObjectSortBySelection()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by object selection.
- ' When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortBySelection MsgBox "The ObjectSortBySelection preference is set to: " & originalValue
- ' Modify the ObjectSortBySelection preference by toggling the value ACADPref.ObjectSortBySelection = Not (ACADPref.ObjectSortBySelection) newValue = ACADPref.ObjectSortBySelection

 MsgBox "The ObjectSortBySelection preference has been set to: " & newValue"
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.ObjectSortBySelection = originalValue

MsgBox "The ObjectSortBySelection preference was reset back to: " & origina End Sub

ObjectSortBySnap Example

Sub Example_ObjectSortBySnap()

- 'This example reads and modifies the preference value which controls
- ' the sorting of drawing objects by object snap.
- 'When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.ObjectSortBySnap MsgBox "The ObjectSortBySnap preference is set to: " & originalValue
- ' Modify the ObjectSortBySnap preference by toggling the value ACADPref.ObjectSortBySnap = Not (ACADPref.ObjectSortBySnap) newValue = ACADPref.ObjectSortBySnap MsgBox "The ObjectSortBySnap preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- '* Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.ObjectSortBySnap = originalValue

MsgBox "The ObjectSortBySnap preference was reset back to: " & originalVal End Sub

Offset Example

```
Sub Example_Offset()
```

'This example will find the offset of the MVBlock tag to the end of the leader.

```
Dim obj As AcadObject
Dim bubble As AecMVBlockRef
Dim anchor As AecAnchor
Dim leaderAnchor As AecAnchorLeadEntToNode
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select bubble"
If TypeOf obj Is AecMVBlockRef Then
  Set bubble = obj
  Set anchor = obj.GetAnchor
  If TypeOf anchor Is AecAnchorLeadEntToNode Then
    Set leaderAnchor = anchor
    Dim offset As Variant
    Dim offsetString As String
    offset = leaderAnchor.offset
    offsetString = offset(0) & ", " & offset(1) & ", " & offset(2)
    MsgBox "Offset from node = " & offsetString, vbInformation, "Offset Ex-
  Else
    MsgBox "Not anchored to column grid", vbInformation, "Offset Example
  End If
Else
  MsgBox "Not a bubble", vbInformation, "Offset Example"
End If
```

OLELaunch Example

Sub Example_OLELaunch()

- 'This example reads and modifies the preference value which controls
- ' whether to launch the parent application when plotting OLE objects.
- ' When finished, this example resets the preference value back to
- ' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

'Read and display the original value originalValue = ACADPref.OLELaunch MsgBox "The OLELaunch preference is set to: " & originalValue

' Modify the OLELaunch preference by toggling the value ACADPref.OLELaunch = Not (ACADPref.OLELaunch) newValue = ACADPref.OLELaunch MsgBox "The OLELaunch preference has been set to: " & newValue

- 'Reset the preference back to it's original value
- ' * Note: Comment out this last section to leave the change to

' this preference in effect

ACADPref. OLELaunch = original Value

MsgBox "The OLELaunch preference was reset back to: " & originalValue End Sub

Operation Example

Examples:

1 AecMassElement

l AecMassGroup

```
Sub Example_Operation_AecMassElement()
```

Case aecMassOperationSubtract

'This example shows the operation of a mass element?

```
Dim object As Object
Dim mass As AecMassElement
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecMassElement Then
count = count + 1
Set mass = object

Select Case mass.Operation
Case aecMassOperationAdd
MsgBox "Mass Element " & count & " Operation: Add", vbInformation
Case aecMassOperationIntersect
MsgBox "Mass Element " & count & " Operation: Intersect", vbInform
```

MsgBox "Mass Element " & count & " Operation: Subtract", vbInform

End If

End Select

Next

```
If count = 0 Then
                                               MsgBox "No Mass Elements Present in Drawing", vbInformation, "Operational Control of the Control
End If
```

End Sub

```
Sub Example_Operation_AecMassGroup()
```

'This example shows the operation of a mass group?

```
Dim object As Object
Dim group As AecMassGroup
Dim count As Integer
'initalize
count = 0
```

For Each object In ThisDrawing.ModelSpace

```
If TypeOf object Is AecMassGroup Then
  count = count + 1
  Set group = object
```

Select Case group. Operation

Case aecMassOperationAdd

MsgBox "Mass Group " & count & " Operation: Add", vbInformation, Case aecMassOperationIntersect

MsgBox "Mass Group " & count & " Operation: Intersect", vbInformat Case aecMassOperationSubtract

MsgBox "Mass Group " & count & " Operation: Subtract", vbInformati **End Select**

End If

Next

If count = 0 Then
 MsgBox "No Mass Element Groups Present in Drawing", vbInformation, "C
End If

OverridesEnabled Example

Sub Example_OverridesEnabled()

Dim db As New AecBaseDatabase

Dim cLayerKeyStyles As AecLayerKeyStyles

Dim layerKeyStyle As aeclayerkeystyle

MsgBox "Layer Overrides Enabled: " & layerKeyStyle.OverridesEnabled, vbI1

OverrideSettings Example

Sub Example_OverrideSettings()

```
Dim db As New AecBaseDatabase
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As aeclayerkeystyle
Dim overrideSettings As AecLayerOverrideSettings
db.Init ThisDrawing.Database
```

```
Set cLayerKeyStyles = db.LayerKeyStyles
Set layerKeyStyle = cLayerKeyStyles.Item(0)
Set overrideSettings = layerKeyStyle.overrideSettings
```

OwnerID Example

Sub Example_OwnerID()

- 'This example creates an MText object in model space
- ' and then finds the OwnerID for the object.

Dim MTextObj As AcadMText
Dim corner(0 To 2) As Double
Dim width As Double
Dim text As String
corner(0) = 0#: corner(1) = 10#: corner(2) = 0#
width = 10
text = "This is the text String for the mtext Object"

'Creates the mtext Object
Set MTextObj = ThisDrawing.ModelSpace.AddMText(corner, width, text)
ZoomAll

MsgBox "The OwnerID for the MText object is: " & MTextObj.OwnerID

PaperSpace Example

Sub Example_PaperSpace()

'This example sets the variable pspace to the paperspace entity collection.

Dim pspace As AcadPaperSpace Set pspace = ThisDrawing.PaperSpace

PlotConfigurations Example

Sub Example_PlotConfigurations()

- 'This example will access the PlotConfigurations collection for the current dra
- ' add a plot configuration, and list basic information about the
- ' plot configurations in the drawing.

Dim PlotConfigurations As AcadPlotConfigurations Dim PlotConfiguration As AcadPlotConfiguration Dim msg As String

'Get PlotConfigurations collection from document object Set PlotConfigurations = ThisDrawing.PlotConfigurations

' If there aren't any plot configurations, then we'll add one
If PlotConfigurations.count = 0 Then

'*** Customize the new configuration to your satisfaction ***
PlotConfigurations.Add "NEW_CONFIGURATION"
End If

msg = vbCrLf & vbCrLf 'Start with a space

' Get the names of the plot configurations in this drawing For Each PlotConfiguration In PlotConfigurations msg = msg & PlotConfiguration.name & vbCrLf Next

'Display a list of available plot configurations
MsgBox "There are " & PlotConfigurations.count & " plot configuration(s) in '
ThisDrawing.WindowTitle & ":" & msg
End Sub

PlotStyleName Example

```
Sub Example_PlotStyleName()
```

'This example shows the plotstyle name of the layer generated by the key

```
Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKey
Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
Set cLayerKeys = layerKeyStyle.Keys
For Each layerKey In cLayerKeys
  Debug.Print layerKey.Name
  Debug.Print " Color
                       - " & layerKey.Color
  Debug.Print " Layer - " & layerKey.Layer
  Debug.Print "LineType - " & layerKey.Linetype
  Debug.Print " Lineweight - " & layerKey.Lineweight
  Debug.Print "Plotstyle - " & layerKey.PlotStyleName
  Debug.Print " Plottable - " & layerKey.Plottable
  Debug.Print " Removable - " & layerKey.Removeable
Next
```

Plottable Example

```
Sub Example_Plottable()
```

'This example shows whether the layer is plottable

```
Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKey
Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
Set cLayerKeys = layerKeyStyle.Keys
For Each layerKey In cLayerKeys
  Debug.Print layerKey.Name
  Debug.Print " Color
                       - " & layerKey.Color
  Debug.Print " Layer - " & layerKey.Layer
  Debug.Print "LineType - " & layerKey.Linetype
  Debug.Print " Lineweight - " & layerKey.Lineweight
  Debug.Print "Plotstyle - " & layerKey.PlotStyleName
  Debug.Print " Plottable - " & layerKey.Plottable
  Debug.Print "Removable - " & layerKey.Removeable
Next
```

PolygonStyles Example

Sub Example_PolygonStyles()

'This example will display the number of Polygon Styles in the current drawin

Dim doc As AecArchBaseDocument
Dim cPolygonStyles As AecPolygonStyles

Set doc = AecArchBaseApplication.ActiveDocument

Set cPolygonStyles = doc.PolygonStyles

MsgBox "Number of Polygon Styles in this drawing is: " & cPolygonStyles.Cc $\,$

Preferences Example

Sub Example_Preferences()

'This example returns the ElevationPrecision setting for the current drawing.

Dim dbPref As AecDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.Preferences

MsgBox "The current value for ElevationPrecision is: " & dbPref.ElevationPre vbInformation, "ElevationPrecision Example"

Profile Example

Examples:

- 1 AecProfileStyle
- 1 AecRing
- 1 AecRings

```
Sub Example_Profile_AecProfileStyle()
```

'This example will display the area of the first profile style in the collection

Dim db As New AecBaseDatabase Dim pStyle As AecProfileStyle

db.Init ThisDrawing.Database

Set pStyle = db.ProfileStyles.Item(0)
MsgBox "Profile Area: " & pStyle.Profile.Area, vbInformation, "Profile Example."

End Sub

Sub Example_Profile_AecRing()

'This example will display the area of the ring's owner profile

Dim db As New AecBaseDatabase
Dim ring As AecRing

db.Init ThisDrawing.Database

Set ring = db.ProfileStyles.Item(0).Profile.rings.Item(0)
MsgBox "Ring Profile Area: " & ring.Profile.Area, vbInformation, "Profile Ex

End Sub

Sub Example_Profile_AecRings()

'This example will display the area of the profile that owns the ring collection

Dim db As New AecBaseDatabase Dim pStyle As AecProfileStyle

db.Init ThisDrawing.Database

Set pStyle = db.ProfileStyles.Item(0)
MsgBox "Rings Profile Area: " & pStyle.Profile.rings.Profile.Area, vbInforma

ProfileStyle Example

Sub Example_ProfileStyle()

Dim ent As Object Dim pt As Variant Dim mass As AecMassElement

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Mass Element"

If TypeOf ent Is AecMassElement Then Set mass = ent

MsgBox "ProfileStyle is: " & mass.ProfileStyle.Name, vbInformation, "Prof Else

MsgBox "Not an AecMassElement", vbExclamation, "ProfileStyle Example End If

ProfileStyleName Example

```
Sub Example_ProfileStyleName ()
```

'This example shows the profile style name for the mass element

```
Dim object As Object
Dim mass As AecMassElement
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecMassElement Then
count = count + 1
Set mass = object
MsgBox "Mass Element " & count & " Name is: " & mass.ProfileStyleNa
End If

Next

If count = 0 Then
MsgBox "No Mass Elements Present in Drawing", vbInformation, "Profile S
End If
```

ProfileStyles Example

Sub Example_ProfileStyles()

'This example will display the number of Profile Styles in the current drawing.

Dim doc As AecArchBaseDocument Dim cProfileStyles As AecProfileStyles

Set doc = AecArchBaseApplication.ActiveDocument

Set cProfileStyles = doc.ProfileStyles
MsgBox "Number of Profile Styles in this drawing is: " & cProfileStyles.Coun

ProjectName Example

Sub Example_ProjectName()

'This example displays the ProjectName setting for the current drawing

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The setting for ProjectName is: " & dbPref.ProjectName, _ vbInformation, "ProjectName Example"

Radius Example

```
Sub Example_Radius()
```

'This example shows the radius of a mass element

```
Dim mass As AecMassElement
Dim pt1(0 To 2) As Double
pt1(0) = 100: pt1(1) = 100: pt1(2) = 0

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
mass.Type = aecMassElementTypeCone
mass.Height = 100
mass.Radius = 50
mass.Location = pt1
```

MsgBox "Mass Element Radius = " & mass.Radius, vbInformation, "Radius E:

ReferenceExample

```
Sub Example_Reference()
```

'This example will make a mass element and anchor it to a line.

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to mass.AttachAnchor anchor ' attach the anchor to the mass element
```

RegisteredApplications Example

Sub Example_RegisteredApplications()

- 'This example finds the current RegisteredApplications collection and
- ' adds a new RegisteredApplication to that collection.

Dim RegAppColl As AcadRegisteredApplications Set RegAppColl = ThisDrawing.RegisteredApplications

' Create a RegisteredApp named "TEST" in the current drawing
Dim RegAppObj As AcadRegisteredApplication
Set RegAppObj = RegAppColl.Add("TEST")
MsgBox "A new registered application called " & RegAppObj.name & " has both End Sub

ReleaseAnchor Example

Sub Example_ReleaseAnchor()

'This example will release the anchor attached to an AEC Entity.

```
Dim ent As AcadEntity
Dim geo As AecGeo
Dim pt As Variant
Dim anchor As AecAnchor
ThisDrawing.Utility.GetEntity ent, pt, "Selected anchored object"
If TypeOf ent Is AecGeo Then
  Set geo = ent
  Set anchor = geo.GetAnchor
  If anchor Is Nothing Then
    MsgBox "AEC Entitiy is not anchored", vbInformation, "ReleaseAnchor I
  Else
    geo.ReleaseAnchor ' releases the anchor from the entity
    MsgBox "AEC Entity is released", vbInformation, "ReleaseAnchor Exam
  End If
Else
  MsgBox "AEC Entity not selected", vbInformation, "ReleaseAnchor Examp
End If
```

Remove Example

Sub Example_Remove_LayerKeyStyle()

'This example adds a new layer key style to the layer key

Dim app As New AecBaseApplication Dim doc As AecBaseDocument Dim layerkeystyles As AecLayerKeyStyles Dim layerkeystyle As AecLayerKeyStyle

Dim msg As String

'Initialize.

app.Init ThisDrawing.Application
Set doc = app.ActiveDocument

'Get the layer key styles collection and note the number

' of objects in the collection.

Set layerkeystyles = doc.layerkeystyles

msg = "There were " & layerkeystyles.Count _

& " layer key styles in this drawing." & vbCrLf

' Add a new layer key style to the collection.

Set layerkeystyle = layerkeystyles.Add("ExampleStyle") layerkeystyle.Description = "This is a test"

msg = msg & "After adding a layer key style, there are " _ & layerkeystyles.Count & " of them."

MsgBox msg, vbInformation, "Remove Example"

'Note the number of objects in the layer key styles collection now.
msg = "The new layer key style had " & layerkeystyle.Keys.Count & " keys--"

'Remove the layer key style that was just added. layerkeystyles.Remove ("ExampleStyle")

^{&#}x27; styles collection, then removes the new style.

msg = msg & " but I've since removed the style..."
MsgBox msg, vbInformation, "Remove Example"

Removeable Example

```
Sub Example_Removeable()
```

'This example shows whether the layer is removeable

```
Dim doc As AecArchBaseDocument
Dim dbPref As AecArchBaseDatabasePreferences
Dim cLayerKeyStyles As AecLayerKeyStyles
Dim layerKeyStyle As AecLayerKeyStyle
Dim cLayerKeys As AecLayerKeys
Dim layerKey As AecLayerKey
Set doc = AecArchBaseApplication.ActiveDocument
Set cLayerKeyStyles = doc.LayerKeyStyles
Set dbPref = doc.Preferences
' Sets the layer key style to the current layer standard
Set layerKeyStyle = cLayerKeyStyles.Item(dbPref.LayerStandard)
Set cLayerKeys = layerKeyStyle.Keys
For Each layerKey In cLayerKeys
  Debug.Print layerKey.Name
  Debug.Print " Color
                       - " & layerKey.Color
  Debug.Print " Layer - " & layerKey.Layer
  Debug.Print "LineType - " & layerKey.Linetype
  Debug.Print " Lineweight - " & layerKey.Lineweight
  Debug.Print "Plotstyle - " & layerKey.PlotStyleName
  Debug.Print " Plottable - " & layerKey.Plottable
  Debug.Print " Removable - " & layerKey.Removeable
Next
```

RenderSmoothness Example

Sub Example_RenderSmoothness()

- 'This example returns the current setting of
- ' RenderSmoothness. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currRenderSmoothness As Double Dim newRenderSmoothness As Double

- 'Retrieve the current RenderSmoothness value currRenderSmoothness = ThisDrawing.preferences.RenderSmoothness MsgBox "The current value for RenderSmoothness is " & currRenderSmoothn
- 'Change the value for RenderSmoothness newRenderSmoothness = 2.5 ThisDrawing.preferences.RenderSmoothness = newRenderSmoothness MsgBox "The new value for RenderSmoothness is " & newRenderSmoothness
- 'Reset RenderSmoothness to its original value
 ThisDrawing.preferences.RenderSmoothness = currRenderSmoothness
 MsgBox "The RenderSmoothness value is reset to " & currRenderSmoothness,
 End Sub

ResizeOffset Example

Sub Example_ResizeOffset()

```
'This example will take a mass element anchored to a 2D Layout Grid, and 'apply a Resize Offset to it.
```

```
Dim obj As AcadObject
Dim anchor As AecAnchorEntToLayoutCell
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select anchored element"
If TypeOf obj Is AecMassElement Then
Dim mass As AecMassElement
Set mass = obj
If TypeOf mass.GetAnchor Is AecAnchorEntToLayoutCell Then
Set anchor = mass.GetAnchor
anchor.ResizeOffset = -22
anchor.ApplyResize = False
End If
End If
```

RightOffset Example

Sub Example_RightOffset()

- 'This example returns the right offset of the selected object
- ' to the grid assembly.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- 'assembly.

Dim ent As AcadEntity Dim geo As AecGeo Dim anchor As AecAnchor

Dim offset As String

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "RightOffset Example" ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

If anchor Is Nothing Then

MsgBox "Selected object is not anchored.", vbExclamation, "RightOffset ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

MsgBox "Right offset of object: " & anchor.RightOffset, vbInformation, "

End If
Else
MsgBox "Object selected is not an AEC entity.", vbInformation, "RightOffse End If

Rings Example

```
Sub Example_Rings()
```

'This example shows the number of rings of the profile of an AecPolygon

```
Dim obj As Object
Dim pt As Variant
Dim poly As AecPolygon

ThisDrawing.Utility.GetEntity obj, pt, "Select an AECPolygon"

If TypeOf obj Is AecPolygon Then
Set poly = obj
MsgBox "Profile Rings: " & poly.Profile.rings.Count, vbInformation, "Rings Else
MsgBox "Not a Polygon or no Profile Found", vbInformation, "Rings Exam End If
```

Rise Example

```
Sub Example_Rise()
```

'This example shows the rise of the mass element if gable shape

Rotate Example

```
Sub Example Rotate()
  'This example creates a light weight polyline
  ' and then rotates that polyline.
  ' Create the polyline
  Dim plineObj As AcadLWPolyline
  Dim points (0 To 11) As Double
  points(0) = 1: points(1) = 2
  points(2) = 1: points(3) = 3
  points(4) = 2: points(5) = 3
  points(6) = 3: points(7) = 3
  points(8) = 4: points(9) = 4
  points(10) = 4: points(11) = 2
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  ZoomAll
  MsgBox "Rotate the polyline by 45 degrees.", , "Rotate Example"
  ' Define the rotation
  Dim basePoint(0 To 2) As Double
  Dim rotationAngle As Double
  basePoint(0) = 4: basePoint(1) = 4.25: basePoint(2) = 0
  rotationAngle = 0.7853981 ' 45 degrees
  ' Rotate the polyline
  plineObj.Rotate basePoint, rotationAngle
  ZoomAll
  MsgBox "Rotation completed.", , "Rotate Example"
End Sub
```

Rotate3D Example

```
Sub Example_Rotate3D()
```

- 'This example creates a box in model space.
- ' It then rotates the box about an axis.

Dim boxObj As Acad3DSolid Dim length As Double, width As Double, height As Double Dim center(0 To 2) As Double

```
' Define the box
center(0) = 5#: center(1) = 5#: center(2) = 0
length = 5#: width = 7: height = 10#
```

- ' Create the box (3DSolid) object in model space Set boxObj = ThisDrawing.ModelSpace.AddBox(center, length, width, height)
- 'Change the viewing direction of the viewport
 Dim NewDirection(0 To 2) As Double
 NewDirection(0) = -1: NewDirection(1) = -1: NewDirection(2) = 1
 ThisDrawing.ActiveViewport.direction = NewDirection
 ThisDrawing.ActiveViewport = ThisDrawing.ActiveViewport
 ThisDrawing.Regen True
- ' Define the rotation axis with two points Dim rotatePt1(0 To 2) As Double Dim rotatePt2(0 To 2) As Double Dim rotateAngle As Double

```
rotatePt1(0) = -3: rotatePt1(1) = 4: rotatePt1(2) = 0
rotatePt2(0) = -3: rotatePt2(1) = -4: rotatePt2(2) = 0
rotateAngle = 30
rotateAngle = rotateAngle * 3.141592 / 180#
```

- 'Draw a line between the two axis points so that it is visible.
- 'This is optional. It is not required for the rotation.

Dim axisLine As AcadLine

Set axisLine = ThisDrawing.ModelSpace.AddLine(rotatePt1, rotatePt2)

axisLine.Update
MsgBox "Rotate the box 30 degrees about the axis shown.", , "Rotate3D Exam

'Rotate the box boxObj.Rotate3D rotatePt1, rotatePt2, rotateAngle ThisDrawing.Regen True MsgBox "The box is rotated 30 degrees.", , "Rotate3D Example"

Rotation Example

```
Sub Example_Rotation()
```

'This example shows the rotation referance angle for an AEC Entity

```
Dim object As Object
Dim pt As Variant
Dim geo As AecGeo

ThisDrawing.Utility.GetEntity object, pt, "Select AEC Entity:"

If TypeOf object Is AecGeo Then
Set geo = object
Dim rotString As String
rotString = ThisDrawing.Utility.AngleToString(geo.Rotation, acDegrees, 4)
MsgBox "AEC Entity Rotation is: " & rotString, vbInformation, "Rotation E
Else
MsgBox "No AEC Entity selected.", vbInformation, "Rotation Example"
End If
```

SaveAsDefault Example

Sub Example_SaveAsDefault()

```
'This example sets the TextHeight setting to 0.25
```

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

dbPref.TextHeight = 0.25

dbPref.SaveAsDefault

MsgBox "The new default value for TextHeight is: " & dbPref.TextHeight, _ vbInformation, "SaveAsDefault Example"

^{&#}x27; and saves it as the default.

ScaleEntity Example

Sub Example ScaleEntity()

```
'This example creates a light weight polyline
  ' and then scales that polyline.
  ' Create the polyline
  Dim plineObj As AcadLWPolyline
  Dim points(0 To 11) As Double
  points(0) = 1: points(1) = 2
  points(2) = 1: points(3) = 3
  points(4) = 2: points(5) = 3
  points(6) = 3: points(7) = 3
  points(8) = 4: points(9) = 4
  points(10) = 4: points(11) = 2
  Set plineObj = ThisDrawing.ModelSpace.AddLightWeightPolyline(points)
  plineObj.Closed = True
  ZoomAll
  MsgBox "Scale the polyline by 0.5", , "ScaleEntity Example"
  ' Define the scale
  Dim basePoint(0 To 2) As Double
  Dim scalefactor As Double
  basePoint(0) = 4: basePoint(1) = 4.25: basePoint(2) = 0
  scalefactor = 0.5
  ' Scale the polyline
  plineObj.ScaleEntity basePoint, scalefactor
  ZoomAll
  MsgBox "Scale completed.", , "ScaleEntity Example"
End Sub
```

ScaleOnInsert Example

```
Sub Example_ScaleOnInsert()
```

```
'This example returns the ScaleOnInsert setting for the current drawing.
Dim dbPref As AecArchBaseDatabasePreferences
Set dbPref = AecArchBaseApplication.ActiveDocument.preferences
'Convert the scale on insert value to a string.
Dim scaleins As String
If dbPref.ScaleOnInsert = True Then
  scaleins = "scale on insert."
Else
  scaleins = "do not scale on insert."
End If
MsgBox "The current value for ScaleOnInsert is " & scaleins, _
  vbInformation, "ScaleOnInsert Example"
```

ScaleX Example

Examples:

1 AecMaskBlockRef

1 AecMVBlockRef

```
Sub Example_ScaleX_AecMaskBlockRef()
```

'This example shows the X scale of the mask block

Dim obj As Object

Dim pt As Variant

Dim mask As AecMaskBlockRef

ThisDrawing.Utility.GetEntity obj, pt, "Select Mask Block"

If TypeOf obj Is AecMaskBlockRef Then

Set mask = obj

 $MsgBox \ "Scale X \ is: " \& \ mask.Scale X, \ vbInformation, "Scale X \ Example"$

Else

MsgBox "Not a Mask Block", vbExclamation, "ScaleX Example"

End If

End Sub

Sub Example_ScaleX_AecMVBlockRef()

Dim ent As AcadEntity

Dim pt As Variant

Dim mvBlock As AecMVBlockRef

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Multi-View Block"

```
If TypeOf ent Is AecMVBlockRef Then
    Set mvBlock = ent
    MsgBox "Scale X is: " & mvBlock.ScaleX, vbInformation, "ScaleX Exampl
Else
    MsgBox "Not an AecMVBlockRef", vbExclamation, "ScaleX Example"
End If
```

ScaleXY Example

```
Sub Example_ScaleXY ()

Dim ring As AecRing
Dim profile As AecProfile

Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle

Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item(0)

Set profileStyle.profile = profile.ScaleXY(2, 1)
```

ScaleY Example

Examples:

1 AecMaskBlockRef

1 AecMVBlockRef

```
Sub Example_ScaleY_AecMaskBlockRef()
```

'This example shows the Y scale of the mask block

Dim obj As Object

Dim pt As Variant

Dim mask As AecMaskBlockRef

ThisDrawing.Utility.GetEntity obj, pt, "Select Mask Block"

If TypeOf obj Is AecMaskBlockRef Then

Set mask = obj

MsgBox "Scale Y is: " & mask.ScaleY, vbInformation, "ScaleY Example"

Else

MsgBox "Not a Mask Block", vbExclamation, "ScaleY Example" End If

End Sub

Sub Example_ScaleY_AecMVBlockRef()

Dim ent As AcadEntity

Dim pt As Variant

Dim mvBlock As AecMVBlockRef

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Multi-View Block"

```
If TypeOf ent Is AecMVBlockRef Then
Set mvBlock = ent
MsgBox "Scale Y is: " & mvBlock.ScaleY, vbInformation, "ScaleY Example
Else
MsgBox "Not an AecMVBlockRef", vbExclamation, "ScaleY Example"
End If
```

ScaleZ Example

Examples:

1 AecMaskBlockRef

1 AecMVBlockRef

```
Sub Example_ScaleZ_AecMaskBlockRef()
```

'This example shows the Z scale of the mask block

Dim obj As Object

Dim pt As Variant

Dim mask As AecMaskBlockRef

ThisDrawing.Utility.GetEntity obj, pt, "Select Mask Block"

```
If TypeOf obj Is AecMaskBlockRef Then
```

Set mask = obj

MsgBox "Scale Z is: " & mask.ScaleZ, vbInformation, "ScaleZ Example"

Else

MsgBox "Not a Mask Block", vbExclamation, "ScaleZ Example" End If

End Sub

Sub Example_ScaleZ_AecMVBlockRef()

Dim ent As AcadEntity

Dim pt As Variant

Dim mvBlock As AecMVBlockRef

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Multi-View Block"

```
If TypeOf ent Is AecMVBlockRef Then
Set mvBlock = ent
MsgBox "Scale Z is: " & mvBlock.ScaleZ, vbInformation, "ScaleZ Example
Else
MsgBox "Not an AecMVBlockRef", vbExclamation, "ScaleZ Example"
End If
```

SegmentPerPolyline Example

Sub Example_SegmentPerPolyline()

- 'This example returns the current setting of
- ' SegmentPerPolyline. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currSegmentPerPolyline As Integer Dim newSegmentPerPolyline As Integer

- 'Retrieve the current SegmentPerPolyline value currSegmentPerPolyline = ThisDrawing.preferences.SegmentPerPolyline MsgBox "The current value for SegmentPerPolyline is " & currSegmentPerPol
- 'Change the value for SegmentPerPolyline newSegmentPerPolyline = 2001 ThisDrawing.preferences.SegmentPerPolyline = newSegmentPerPolyline MsgBox "The new value for SegmentPerPolyline is " & newSegmentPerPolyli
- 'Reset SegmentPerPolyline to its original value
 ThisDrawing.preferences.SegmentPerPolyline = currSegmentPerPolyline
 MsgBox "The SegmentPerPolyline value is reset to " & currSegmentPerPolyline
 End Sub

SelfIntersects Example

Sub Example_SelfIntersects()

'This example shows if an AecPolygon self intersects

```
Dim obj As Object
Dim pt As Variant
Dim poly As AecPolygon

ThisDrawing.Utility.GetEntity obj, pt, "Select an AECPolygon"

If TypeOf obj Is AecPolygon Then
Set poly = obj
MsgBox "Profile SelfIntersects?: " & poly.Profile.SelfIntersects, vbInformat

Else
MsgBox "Not a Polygon or no Profile Found", vbInformation, "SelfIntersect

End If
```

SetXData Example

lineObj.SetXData DataType, Data

```
Sub Example_SetXdata()
```

'This example creates a line and attaches extended data to that line.

```
' Create the line
Dim lineObj As AcadLine
Dim startPt(0 To 2) As Double, endPt(0 To 2) As Double
startPt(0) = 1#: startPt(1) = 1#: startPt(2) = 0#
endPt(0) = 5#: endPt(1) = 5#: endPt(2) = 0#
Set lineObj = ThisDrawing.ModelSpace.AddLine(startPt, endPt)
ZoomAll
'Initialize all the xdata values. Note that first data in the list should be
'application name and first datatype code should be 1001
Dim DataType(0 To 9) As Integer
Dim Data(0 To 9) As Variant
Dim reals3(0 To 2) As Double
Dim worldPos(0 To 2) As Double
DataType(0) = 1001: Data(0) = "Test\_Application"
DataType(1) = 1000: Data(1) = "This is a test for xdata"
DataType(2) = 1003: Data(2) = "0"
                                              ' laver
DataType(3) = 1040: Data(3) = 1.23479137438413E+40 'real
DataType(4) = 1041: Data(4) = 1237324938
                                                  ' distance
DataType(5) = 1070: Data(5) = 32767
                                               ' 16 bit Integer
DataType(6) = 1071: Data(6) = 32767
                                               ' 32 bit Integer
DataType(7) = 1042: Data(7) = 10
                                             'scaleFactor
reals3(0) = -2.95: reals3(1) = 100: reals3(2) = -20
DataType(8) = 1010: Data(8) = reals3
                                              ' real
worldPos(0) = 4: worldPos(1) = 400.999999999: worldPos(2) = 2.798989
DataTvpe(9) = 1011: Data(9) = worldPos
                                                ' world space position
' Attach the xdata to the line
```

'Return the xdata for the line Dim xdataOut As Variant Dim xtypeOut As Variant lineObj.GetXData "", xtypeOut, xdataOut

Side1 Example

```
Sub Example_Side1()
```

'This example shows the length of cut plane along the starting edge

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Side 1 is: " & clip.Side1, vbInformation,
End If

Next

If count = 0 Then
MsgBox "No ClipVol Present in Drawing", vbInformation, "Side1 Example'
End If
```

Side2 Example

```
Sub Example_Side2()
```

'This example shows the length of cut plane along the ending edge

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Side 2 is: " & clip.Side2, vbInformation,
End If

Next

If count = 0 Then
MsgBox "No ClipVol Present in Drawing", vbInformation, "Side2 Example'
End If
```

SolidFill Example

Sub Example_SolidFill()

- 'This example reads and modifies the preference value which controls
- ' whether multilines, traces, solids, all hatches (including solid-fill)
- ' and wide polylines are filled in.

'When finished, this example resets the preference value back to

' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

'Read and display the original value originalValue = ACADPref.SolidFill MsgBox "The SolidFill preference is set to: " & originalValue

' Modify the SolidFill preference by toggling the value ACADPref.SolidFill = Not (ACADPref.SolidFill) newValue = ACADPref.SolidFill MsgBox "The SolidFill preference has been set to: " & newValue

- 'Reset the preference back to it's original value
- ' * Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.SolidFill = originalValue

MsgBox "The SolidFill preference was reset back to: " & originalValue End Sub

StartOffset Example

Sub Example_StartOffset()

'This example displays the start offset of a selected AEC Layout Curve object

Dim obj As Object
Dim pt As Variant
Dim layoutCurve As AecLayoutCurve

ThisDrawing.Utility.GetEntity obj, pt, "Select a Node on an AEC Layout Curve"

If TypeOf obj Is AecLayoutCurve Then
Set layoutCurve = obj
MsgBox "StartOffset is: " & layoutCurve.StartOffset, vbInformation, "StartOffset
Else
MsgBox "Nata AEC I accept Council of Englanation, "StartOffset England"

MsgBox "Not a AEC Layout Curve", vbExclamation, "StartOffset Example" End If

Style Example

```
Sub Example_Style()
  'This example shows the style of an AEC Entity
  Dim obj As Object
  Dim pt As Variant
  ThisDrawing.Utility.GetEntity obj, pt, "Select an AEC Object"
  If TypeOf obj Is AecGeo Then
    On Error Resume Next
    Dim name As String
    name = obj.Style.name
    ' If the entity doesn't use a style, this will fail
    If (name "") Then
       MsgBox "Style is: " & name, vbInformation, "Style Example"
    Else
       MsgBox "AEC Entity does not have a style", vbInformation, "Style Exam
    End If
  Else
    MsgBox "Not a AEC Entity", vbExclamation, "Style Example"
  End If
End Sub
```

StyleName Example

```
Sub Example_StyleName()
  'This example shows the style name of an AEC Entity
  Dim obj As Object
  Dim pt As Variant
  ThisDrawing.Utility.GetEntity obj, pt, "Select an AEC Object"
  If TypeOf obj Is AecGeo Then
    On Error Resume Next
    Dim name As String
    name = obj.StyleName
    ' If the entity doesn't use a style, this will fail
    If (name "") Then
       MsgBox "Style is: " & name, vbInformation, "Style Example"
    Else
       MsgBox "AEC Entity does not have a style", vbInformation, "Style Exam
    End If
  Else
    MsgBox "Not a AEC Entity", vbExclamation, "Style Example"
  End If
End Sub
```

Subtract Example

```
Sub Example_Subtract ()
```

'This example adds two profiles from rings, and subtracts the second from the fir

```
On Error Resume Next
Dim pointList1(0 To 9) As Double
Dim pointList2(0 To 7) As Double
pointList1(0) = 0: pointList1(1) = 0
pointList1(2) = 1: pointList1(3) = 0
pointList1(4) = 1: pointList1(5) = 1
pointList1(6) = 0: pointList1(7) = 1
pointList1(8) = 0: pointList1(9) = 0
pointList2(0) = 0.5: pointList2(1) = 0.5
pointList2(2) = 0.067: pointList2(3) = -0.25
pointList2(4) = 0.933: pointList2(5) = -0.25
pointList2(6) = 0.5: pointList2(7) = 0.5
Dim ring1 As AecRing
Dim ring2 As AecRing
Dim profile 1 As New AecProfile
Dim profile2 As New AecProfile
Dim doc As AecArchBaseDocument
Set doc = AecArchBaseApplication.ActiveDocument
Dim cprofiles As AecProfileStyles
Dim profileStyle As AecProfileStyle
Set cprofiles = doc.ProfileStyles
Set profileStyle = cprofiles.Item("SubtractRing")
If profileStyle Is Nothing Then
  Set profileStyle = cprofiles.Add("SubtractRing")
End If
Set ring1 = profile1.Rings.Add
Set ring2 = profile2.Rings.Add
```

```
ring1.FromPoints (pointList1)
ring2.FromPoints (pointList2)
profile1.Subtract profile2
Set profileStyle.profile = profile1
```

TextFrameDisplay Example

Sub Example_TextFrameDisplay()

- 'This example returns the current setting of
- 'TextFrameDisplay. It then changes the value, and finally
- ' it resets the value back to the original setting.

Dim currTextFrameDisplay As Boolean

- 'Retrieve the current TextFrameDisplay value currTextFrameDisplay = ThisDrawing.preferences.TextFrameDisplay MsgBox "The current value for TextFrameDisplay is " & preferences.TextFram
- 'Change the value for TextFrameDisplay
 ThisDrawing.preferences.TextFrameDisplay = Not (currTextFrameDisplay)
 MsgBox "The new value for TextFrameDisplay is " & preferences.TextFrameI
- 'Reset TextFrameDisplay to its original value
 ThisDrawing.preferences.TextFrameDisplay = currTextFrameDisplay
 MsgBox "The TextFrameDisplay value is reset to " & preferences.TextFrameDisplay
 End Sub

TextHeight Example

Sub Example_TextHeight()

'This example displays the TextHeight setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The current value for TextHeight is: " & dbPref.TextHeight, _ vbInformation, "TextHeight Example"

TextStyles Example

Sub Example_TextStyles()

- 'This example finds the current Textstyles collection and
- ' adds a new Text style to that collection.

Dim TextColl As AcadTextStyles Set TextColl = ThisDrawing.TextStyles

'Create a Text style named "TEST" in the current drawing Dim textStyle As AcadTextStyle Set textStyle = TextColl.Add("TEST")

MsgBox "A new Text style called " & textStyle.name & " has been added to th End Sub

TopOffset Example

Sub Example_TopOffset()

- 'This example modifies the top offset of the selected object
- ' in relation to its anchor point on the grid assembly. It
- ' prompts the user to select the object to be modified,
- ' and then it prompts the user to specify the number of inches
- ' to adjust the offset by. The number can be either positive
- ' or negative.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- 'assembly.

Dim ent As AcadEntity
Dim geo As AecGeo
Dim anchor As AecAnchor

Dim offset As String Dim offset adjust As Double

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "TopOffset Example"

ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

If anchor Is Nothing Then

MsgBox "Selected object is not anchored.", vbExclamation, "TopOffset E ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

' AdjustSizing must be set to True in order for offset change to take effect. anchor.AdjustSizing = True

MsgBox "Top offset of object was: " & anchor.TopOffset, vbInformation,

' Prompt user to specify amount to adjust offset by.

offset_adjust = ThisDrawing.Utility.GetReal("Enter the number of inches

'Change offset by specified amount.
anchor.TopOffset = anchor.TopOffset + offset_adjust
ThisDrawing.Regen (acActiveViewport)
MsgBox "New top offset is: " & anchor.TopOffset, vbInformation, "TopO End If

Else

MsgBox "Object selected is not an AEC entity.", vbInformation, "TopOffset End If

TransformBy Example

```
Sub Example_TransformBy()
                 'This example creates a line and rotates it 90 degrees
                 'using a transformation matrix.
                  ' Create a line
                Dim lineObj As AcadLine
                 Dim startPt(0 To 2) As Double
                Dim endPt(0 To 2) As Double
                startPt(0) = 2: startPt(1) = 1: startPt(2) = 0
                endPt(0) = 5: endPt(1) = 1: endPt(2) = 0
                Set lineObj = ThisDrawing.ModelSpace.AddLine(startPt, endPt)
                lineObj.Update
                  'Initialize the transMat variable with a transformation matrix
                 ' that will rotate an object by 90 degrees about the point(0,0,0)
                 '(More examples of transformation matrices are listed below)
                  Dim transMat(0 To 3, 0 To 3) As Double
                 transMat(0, 0) = 0\#: transMat(0, 1) = -1\#: transMat(0, 2) = 0\#: transMat(0, 3) = 0\#: transM
                transMat(1, 0) = 1#: transMat(1, 1) = 0#: transMat(1, 2) = 0#: transMat(1, 3) = 0
                transMat(2, 0) = 0\#: transMat(2, 1) = 0\#: transMat(2, 2) = 1\#: transMat(2, 3) = 1\#: transMat(3, 3) = 1\#: transMa
                transMat(3, 0) = 0\#: transMat(3, 1) = 0\#: transMat(3, 2) = 0\#: transMat(3, 3) = 0\#: transMa
                'Transform the line using the defined transformation matrix
                 MsgBox "Transform the line.", , "TransformBy Example"
                lineObj.TransformBy (transMat)
                 ZoomAll
```

' More examples of transformation matrices:

MsgBox "The line is transformed.", , "TransformBy Example"

'Rotation Matrix: 90 Degrees about point 0,0,0
'0.000000 -1.000000 0.000000 0.000000
'1.000000 0.000000 0.000000 0.000000
'0.000000 1.000000 0.000000

'0.000000 0.000000 0.000000 1.000000

- 'Rotation Matrix: 45 Degrees about point 5,5,0
 - '0.707107 -0.707107 0.000000 5.000000
 - '0.707107 0.707107 0.000000 -2.071068
 - $^{\circ}0.000000 \ 0.000000 \ 1.000000 \ 0.000000$
 - '0.000000 0.000000 0.000000 1.000000
- 'Translation Matrix: move an object by 10,10,0
 - '1.000000 0.000000 0.000000 10.000000
 - '0.000000 1.000000 0.000000 10.000000
 - '0.000000 0.000000 1.000000 0.000000
 - '0.000000 0.000000 0.000000 1.000000
- 'Scaling Matrix: scale by 10,10 at point 0,0,0
 - 10.000000 0.000000 0.000000 0.000000
 - '0.000000 10.000000 0.000000 0.000000
 - $^{\prime}\,0.000000\,\,0.000000\,\,10.000000\,\,0.000000$
 - $^{\prime}\,0.000000\,\,0.000000\,\,0.000000\,\,1.000000$
- ' Scaling Matrix: scale by 10 at point 2,2
 - ' 10.000000 0.000000 0.000000 -18.000000
 - 0.000000 10.000000 0.000000 -18.000000
 - $\hbox{'0.000000} \ 0.000000 \ 10.000000 \ 0.000000$
 - '0.000000 0.000000 0.000000 1.000000

Type Example

Examples:

l AecLayoutCurve

1 AecMassElement

```
Sub Example_Type_AecLayoutCurve()
```

'This example displays the spacing rule used on a selected aec layout curve obje

```
Dim obj As Object
Dim pt As Variant
Dim layoutCurve As AecLayoutCurve
Dim layoutType As AecLayoutType
Dim str As String
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a Node on an AEC Layout Curve"

```
If TypeOf obj Is AecLayoutCurve Then
Set layoutCurve = obj
layoutType = layoutCurve.Type
Select Case layoutType
Case aecLayoutTypeAutoSpacingBay
str = "Bay Spacing"
Case aecLayoutTypeAutoSpacingEven
str = "Even Spacing"
Case aecLayoutTypeManualSpacing
str = "Manual Spacing"
End Select

MsgBox "Layout rule is: " & str, vbInformation, "Type Example"
Else
MsgBox "Not a AEC Layout Curve", vbExclamation, "Type Example"
End If
```

```
Sub Example_Type_AecMassElement()
'This example shows the type of mass element
  Dim object As Object
  Dim mass As AecMassElement
  Dim count As Integer
  'initalize
  count = 0
  For Each object In ThisDrawing.ModelSpace
    If TypeOf object Is AecMassElement Then
       count = count + 1
       Set mass = object
       Select Case mass. Type
      Case aecMassElementTypeArch
         MsgBox "Mass Element " & count & " Type: Arch", vbInformation, "T
       Case aecMassElementTypeBarrelVault
         MsgBox "Mass Element " & count & " Type: Barrel Vault ", vbInforma
       Case aecMassElementTypeBox
         MsgBox "Mass Element " & count & " Type: Box", vbInformation, "Ty
       Case aecMassElementTypeCone
         MsgBox "Mass Element " & count & " Type: Cone", vbInformation, "I
      Case aecMassElementTypeCylinder
         MsgBox "Mass Element " & count & " Type: Cylinder", vbInformation
       Case aecMassElementTypeDome
         MsgBox "Mass Element " & count & " Type: Dome", vbInformation, "
       Case aecMassElementTypeDoric
         MsgBox "Mass Element " & count & " Type: Doric", vbInformation, "...
       Case aecMassElementTypeExtrusion
         MsgBox "Mass Element " & count & " Type: Extrusion", vbInformation
       Case aecMassElementTypeGable
```

MsgBox "Mass Element " & count & " Type: Gable", vbInformation, "

Case aecMassElementTypeIsoscelesTriangle

MsgBox "Mass Element " & count & " Type: Isosceles Triangle", vbInf Case aecMassElementTypePyramid

MsgBox "Mass Element " & count & " Type: Pyramid", vbInformation Case aecMassElementTypeRevolution

MsgBox "Mass Element " & count & " Type: Revolution", vbInformati Case aecMassElementTypeRightTriangle

MsgBox "Mass Element " & count & " Type: Right Triangle", vbInforn Case aecMassElementTypeSphere

MsgBox "Mass Element " & count & " Type: Sphere", vbInformation, ' End Select

End If

Next

If count = 0 Then

MsgBox "No Mass Elements Present in Drawing", vbInformation, "Type Ex End If

UserCoordinateSystems Example

Sub Example_UserCoordinateSystems()

- 'This example finds the current UserCoordinateSystems collection and
- ' adds a new UCS to that collection.

Dim UCSColl As AcadUCSs Set UCSColl = ThisDrawing.UserCoordinateSystems

'Create a UCS named "TEST" in the current drawing Dim ucsObj As AcadUCS Dim origin(0 To 2) As Double Dim xAxisPnt(0 To 2) As Double Dim yAxisPnt(0 To 2) As Double

```
' Define the UCS
```

```
origin(0) = 4#: origin(1) = 5#: origin(2) = 3#
xAxisPnt(0) = 5#: xAxisPnt(1) = 5#: xAxisPnt(2) = 3#
yAxisPnt(0) = 4#: yAxisPnt(1) = 6#: yAxisPnt(2) = 3#
```

' Add the UCS to the UserCoordinatesSystems collection Set ucsObj = UCSColl.Add(origin, xAxisPnt, yAxisPnt, "TEST")

MsgBox "A new UCS called " & ucsObj.name & " has been added to the User End Sub

Update Example

Sub Example_Update()

- 'This example creates a line in model space. It displays a
- ' Msgbox immediately before the call to update the line and
- ' again after the call.

Dim lineObj As AcadLine

Dim startPoint(0 To 2) As Double

Dim endPoint(0 To 2) As Double

'Create a Line object in model space startPoint(0) = 2#: startPoint(1) = 2#: startPoint(2) = 0# endPoint(0) = 4#: endPoint(1) = 4#: endPoint(2) = 0# Set lineObj = ThisDrawing.ModelSpace.AddLine(startPoint, endPoint) lineObj.Color = acRed

MsgBox "Before the update.", , "Update Example"
'The following code draws an object in AutoCAD window lineObj.Update
MsgBox "After the update.", , "Update Example"

' The following call updates the entire drawing ThisDrawing.Application.Update

UseModelExtents Example

```
Sub Example_UseModelExtents()
```

'This example determines if the height and lower 'extension are set automatically using model extents

```
Dim object As Object
Dim clip As AecClipVol
Dim count As Integer

'initalize
count = 0

For Each object In ThisDrawing.ModelSpace

If TypeOf object Is AecClipVol Then
count = count + 1
Set clip = object
MsgBox "ClipVol" & count & " Use Model Extents?: " & clip.UseModel
End If

Next

If count = 0 Then
MsgBox "No ClipVol Present in Drawing", vbInformation, "UseModelExter
End If
```

UseNodeCS Example

```
Sub Example_UseNodeCS()
```

'This example attaches a Mass Element to a 2D Layout Grid, and uses the No

```
Dim obj As AcadObject
Dim pt As Variant
ThisDrawing.Utility.GetEntity obj, pt, "Select Layout Grid"
If TypeOf obj Is AecLayoutGrid2D Then
  Dim grid As AecLayoutGrid2D
  Set grid = obj
  Dim mass As AecMassElement
  Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement
  Dim anchor As New AecAnchorEntToLayoutNode
  anchor.Reference = grid
  anchor.Node = 1
  anchor.UseNodeCS = True
  mass.AttachAnchor anchor
Else
  MsgBox "No Layout Grid selected", vbInformation, "UseNodeCS Example'
End If
```

^{&#}x27; coordinate system.

Valid Example

Examples:

1 AecProfile

1 AecRing

Sub Example_Valid_AecProfile()

'This example shows if an AecPolygon is valid

Dim obj As Object Dim pt As Variant Dim poly As AecPolygon

ThisDrawing.Utility.GetEntity obj, pt, "Select an AECPolygon"

If TypeOf obj Is AecPolygon Then

Set poly = obj

MsgBox "Profile Valid: " & poly.Profile.Valid, vbInformation, "Valid Exam_lElse

MsgBox "Not a Polygon or no Profile Found", vbInformation, "Valid Examped If

End Sub

Sub Example_Valid_AecRing()

'This example will display the validity of the ring

Dim db As New AecBaseDatabase Dim ring As AecRing

db.Init ThisDrawing.Database

Set ring = db.ProfileStyles.Item(0).Profile.rings.Item(0)
MsgBox "Ring Valid?: " & ring.Valid, vbInformation, "Valid Example"

Value Example

Examples:

1 AecLayerOverrideSetting

l AecLayoutCurve

Sub Example_Value_AecLayerOverrideSetting()

'This example shows the name of the owner of the override setting

Dim db As New AecBaseDatabase
Dim setting As AecLayerOverrideSetting

db.Init ThisDrawing.Database

Set setting = db.LayerKeyStyles.Item(0).overrideSettings.Item(0)
MsgBox "Setting Value: " & setting.Value, vbInformation, "Value Example"

End Sub

Sub Example_Value_AecLayoutCurve()

'This examples show either the bay spacing or the number of nodes on the layou

Dim obj As Object Dim pt As Variant Dim layoutCurve As AecLayoutCurve Dim str As String

ThisDrawing.Utility.GetEntity obj, pt, "Select a Node on an AEC Layout Curve"

```
If TypeOf obj Is AecLayoutCurve Then
   Set layoutCurve = obj
```

```
Select Case layoutCurve.Type
Case aecLayoutTypeAutoSpacingBay
str = "Bay spacing = " & layoutCurve.Value
Case aecLayoutTypeAutoSpacingEven
str = "Number of even spaces = " & layoutCurve.Value
Case aecLayoutTypeManualSpacing
str = "Manual Spacing"
End Select

MsgBox str, vbInformation, "Value Example"
Else
MsgBox "Not a AEC Layout Curve", vbExclamation, "Value Example"
End If
```

VerticalScale Example

Sub Example_VerticalScale()

'This example will display the vertical scale of the database.

Dim db As New AecArchBaseDatabase
Dim pref As AecArchBaseDatabasePreferences

db.Init ThisDrawing.Database

Set pref = db.Preferences
MsgBox "Vertical Scale of the database: " & pref.VerticalScale, vbInformation

"VerticalScale Example"

ViewBlocks Example

```
Sub Example_ViewBlocks()
```

Dim ent As AcadEntity
Dim pt As Variant
Dim mvBlock As AecMVBlockRef
Dim cViewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity ent, pt, "Select AEC Multi-View Block"

If TypeOf ent Is AecMVBlockRef Then
 Set mvBlock = ent
 Set cViewBlocks = mvBlock.viewBlocks
 MsgBox "Number of View blocks is: " & cViewBlocks.Count, vbInformatio
Else
 MsgBox "Not an AecMVBlockRef", vbExclamation, "StyleName Example"
End If

Viewports Example

Sub Viewports()

End Sub

- 'This example finds the current Viewports collection and
- ' adds a new viewport to that collection.

Dim viewportColl As AcadViewports Set viewportColl = ThisDrawing.Viewports

'Create a viewport named "TEST" in the current drawing
Dim viewportObj As AcadViewport
Set viewportObj = viewportColl.Add("TEST")
MsgBox "A new viewport called " & viewportObj.name & " has been added to

Views Example

Sub Example_Views()

- 'This example finds the current views collection and
- ' adds a new view to that collection.

Dim viewColl As AcadViews Set viewColl = ThisDrawing.Views

'Create a view named "TEST" in the current drawing Dim viewObj As AcadView Set viewObj = viewColl.Add("TEST")

MsgBox "A new view called " & viewObj.name & " has been added to the view End Sub

Visible Example

polygon. Visible = True

Sub Example_Visible_AecBaseObject()

```
'This example adds an AecPolygon object to the current drawing,
' displays the object, then turns the object's Visible property
' to False and redisplays the drawing.
Dim obj As AcadObject
Dim pt As Variant
Dim polygon As AecPolygon
' Add an AecPolygon object to the drawing.
Set polygon = ThisDrawing.ModelSpace.AddCustomObject("AecPolygon")
'Select a location for the object onscreen.
pt = ThisDrawing.Utility.GetPoint(, "Select the insertion point:")
If Err.Number 0 Then
  MsgBox ("error when getting a point." & vbCrLf)
   Exit Sub
End If
' Place the object at the specified point, then display
' a message so that users can see the object.
polygon.Location = pt
polygon.Update
MsgBox ("AecPolygon added to drawing. Now you see it...")
' Make the object invisible, then display a message so that
' the drawing is visible and users can see that the
' object is no longer visible.
polygon. Visible = False
polygon.Update
MsgBox ("...and now you don't.")
' Make the object visible again, so that users can manipulate
' it through the user interface.
```

polygon.Update

Void Example

```
Sub Example_Void()
```

'This example will display the if the ring is void or not

Dim db As New AecBaseDatabase Dim ring As AecRing

db.Init ThisDrawing.Database

Set ring = db.ProfileStyles.Item(0).Profile.rings.Item(0)
MsgBox "Ring Void?: " & ring.Void, vbInformation, "Void Example"

Volume Example

```
Sub Example_Volume ()
```

'This example shows the volume of a mass element or mass group.

```
Dim object As Object
Dim pt As Variant
Dim mass As AecMassElement
Dim massGroup As AecMassGroup
Dim count As Integer
ThisDrawing.Utility.GetEntity object, pt, "Select Mass Element or Mass Group
If TypeOf object Is AecMassGroup Then
  Set massGroup = object
  MsgBox "Mass Group volume = " & Format(massGroup.Volume, "###0.00(
Else
  If TypeOf object Is AecMassElement Then
    Set mass = object
    MsgBox "Mass Element volume = " & Format(mass.Volume, "###0.000"
  Else
    MsgBox "Mass Element or Group not selected.", vbInformation, "Volume
  End If
End If
```

VolumeDisplay Example

Sub Example_VolumeDisplayUnit()

'This example returns the VolumeDisplayUnit setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

'Convert the volume display unit to a string.

Dim unit As String

Select Case dbPref.VolumeDisplayUnit

Case aecUnitCubicInch

unit = "cubic inch"

Case aecUnitCubicFoot

unit = "cubic foot"

Case aecUnitCubicYard

unit = "cubic yard"

Case aecUnitCubicMil

unit = "cubic millimeters"

Case aecUnitCubicCentimeter

unit = "cubic centimeters"

```
Case aecUnitCubicDecimeter
unit = "cubic decimeters"
```

Case aecUnitCubicMeter

unit = "cubic meters"

End Select

MsgBox "The current value for VolumeDisplayUnit is " & unit, _ vbInformation, "VolumeDisplayUnit Example"

VolumePrecision Example

Sub Example_VolumePrecision()

'This example displays the VolumePrecision setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The current value for VolumePrecision is: " & dbPref.VolumePrecisi vbInformation, "VolumePrecision Example"

VolumeSuffix Example

Sub Example_VolumeSuffix()

'This example displays the VolumeSuffix setting for the current drawing.

Dim dbPref As AecArchBaseDatabasePreferences

Set dbPref = AecArchBaseApplication.ActiveDocument.preferences

MsgBox "The current value for VolumeSuffix is " & dbPref.VolumeSuffix, _ vbInformation, "VolumeSuffix Example"

Width Example

Examples:

- l AecLayoutGrid2D
- 1 AecLayoutGrid3D
- 1 AecMassElement

```
Sub Example_Width_AecLayoutGrid2D()
```

'This example displays the width of a 2D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
Set grid = obj
MsgBox "Grid Width is: " & grid.Width, vbInformation, "Width Example"
Else
MsgBox "Not a 2D Layout Grid", vbExclamation, "Width Example"
End If

End Sub

Sub Example_Width_AecLayoutGrid3D()

'This example displays the width of a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

```
ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
```

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    MsgBox "Grid Width is: " & grid.Width, vbInformation, "Width Example"
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "Width Example"
End If
```

End Sub

Sub Example_Width_AecMassElement()

'This example shows the size of the mass element in its relative X direction

```
Dim obj As Object
Dim pt As Variant
Dim mass As AecMassElement
```

ThisDrawing.Utility.GetEntity obj, pt, "Select Mass Element"

```
If TypeOf obj Is AecMassElement Then<
    /font >
    Set mass =
        obj MsgBox "Mass Element Width is: " & mass.Width, vbInformation,
    Else
        MsgBox "No Mass Elements selected.", vbInformation, "Width Example"
End If
```

WindowState Example

Sub Example_WindowState()

'This example reads and displays the current window state of the AutoCAD ap

Dim CurrentState As String

' Use the "WindowState" variable to determine the window state of AutoCAD Select Case WindowState

Case acMin: CurrentState = "Minimized"
Case acMax: CurrentState = "Maximized"
Case acNorm: CurrentState = "Normal Size"

End Select

' Display window state

MsgBox "The application window is now: " & CurrentState End Sub

WindowTitle Example

Sub Example_WindowTitle()

- 'This example cycles through the documents collection
- ' and uses the WindowTitle property to create a list of all open documents.

Dim DOC As AecBaseDocument
Dim aecApp As New AecBaseApplication
Dim msg As String

' Initialize the Aec Base application object aecApp.Init ThisDrawing.Application

' If there are no open documents, then exit
If aecApp.Documents.count = 0 Then
 MsgBox "There are no open documents!"
 Exit Sub
End If

msg = vbCrLf & vbCrLf 'Start with a space

'Cycle through all open drawings and get the window title of each drawing
For Each DOC In aecApp.Documents
msg = msg & DOC.WindowTitle & vbCrLf
Next

' Display a list of open drawings MsgBox "The open drawing titles are: " & msg

XDistance Example

```
Sub Example_XDistance()
```

```
'This example will make a mass element and and anchor it 200 units
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.XDistance = 200 ' set the X Distance for the anchor
```

mass.AttachAnchor anchor 'attach the anchor to the mass element

^{&#}x27; from the start of a line.

XEndoffset Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_XEndoffset_AecLayoutGrid2D()

'This example displays the X end offset for a 2D layout grid

Dim obj As Object
Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid X End Offset is: " & grid.XEndOffset, vbInformation, "XEndOffset)
Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "XEndOffset Example" End If

End Sub

Sub Example_XEndoffset_AecLayoutGrid3D()

'This example displays the X end offset for a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid X End Offset is: " & grid.XEndOffset, vbInformation, "XEndOffset
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "XEndOffset Example"
End If

XNodes Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_XNodes_AecLayoutGrid2D()

'This example displays the number of X nodes of a 2D layout grid

Dim obj As Object
Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Selecta 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid X Nodes is: " & grid.XNodes.Count, vbInformation, "XNodes Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "XNodes Example" End If

End Sub

Sub Example_XNodes_AecLayoutGrid3D()

'This example displays the number of X nodes of a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    MsgBox "Grid X Nodes is: " & grid.XNodes.Count, vbInformation, "XNodes
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "XNodes Example"
End If
```

XOffset Example

```
Sub Example_XOffset ()
```

'This example shows the Xoffset of the first viewblock of a multiviewblock

```
Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef
Dim viewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then
Set blockRef = obj
Set viewBlocks = blockRef.viewBlocks
MsgBox "XOffset of View Block 1: " & viewBlocks.Item(0).XOffset, vbInfelse
MsgBox "Not a Multiview Block", vbInformation, "XOffset Example"

End If
```

XPositionFrom Example

Sub Example_XPositionFrom()

```
'This example will make a mass element and anchor it 200 units
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000
```

' this makes a new mass element 120 x 120 x 120 at 0,0,0

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.XDistance = 200 ' set the X Distance for the anchor anchor.XPositionFrom = aecCurvePositionEnd ' set the anchor to measure from mass.AttachAnchor anchor ' attach the anchor to the mass element

^{&#}x27; from the end of a line.

XPositionTo Example

Sub Example_XPositionTo()

```
'This example will make a mass element and anchor it's center 200 units
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
```

```
pt2(0) = 1000: pt2(1) = 1000
```

' this makes a new mass element 120 x 120 x 120 at 0,0,0

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.XDistance = 200 ' set the X Distance for the anchor anchor.XPositionFrom = aecCurvePositionEnd ' set the anchor to measure fron anchor.XPositionTo = aecEdgePositionCenter ' set the anchor to measure to the mass.AttachAnchor anchor ' attach the anchor to the mass element

^{&#}x27; from the end of a line.

XRefEdit Example

Sub Example_XRefEdit()

- 'This example reads and modifies the preference value which controls
- ' whether the current drawing can be edited in place when being
- ' referenced by another user.

' When finished, this example resets the preference value back to

' it's original value.

Dim ACADPref As AcadDatabasePreferences Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

'Read and display the original value originalValue = ACADPref.XRefEdit MsgBox "The XRefEdit preference is set to: " & originalValue

' Modify the XRefEdit preference by toggling the value ACADPref.XRefEdit = Not (ACADPref.XRefEdit) newValue = ACADPref.XRefEdit MsgBox "The XRefEdit preference has been set to: " & newValue

- 'Reset the preference back to it's original value
- ' * Note: Comment out this last section to leave the change to
- ' this preference in effect

ACADPref.XRefEdit = originalValue

MsgBox "The XRefEdit preference was reset back to: " & originalValue End Sub

XRefLayerVisibility Example

Sub Example_XRefLayerVisibility()

- 'This example reads and modifies the preference value which controls
- ' the visibility of xref-dependent layers and specifies if nested xref
- ' path changes are saved.

' When finished, this example resets the preference value back to

' it's original value.

Dim ACADPref As AcadDatabasePreferences
Dim originalValue As Variant, newValue As Variant

' Get the user preferences object Set ACADPref = ThisDrawing.preferences

- 'Read and display the original value originalValue = ACADPref.XRefLayerVisibility MsgBox "The XRefLayerVisibility preference is set to: " & originalValue
- ' Modify the XRefLayerVisibility preference by toggling the value ACADPref.XRefLayerVisibility = Not (ACADPref.XRefLayerVisibility) newValue = ACADPref.XRefLayerVisibility

 MsgBox "The XRefLayerVisibility preference has been set to: " & newValue
- 'Reset the preference back to it's original value
- ' * Note: Comment out this last section to leave the change to
- this preference in effect

ACADPref.XRefLayerVisibility = originalValue

MsgBox "The XRefLayerVisibility preference was reset back to: " & original V End Sub

XRotation Example

```
Sub Example_XRotation()
```

```
'This example will make a mass element and and anchor it to a line,
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the linee as the object (curve) to anchor to anchor.XDistance = 200 ' set the X Distance for the anchor anchor.XRotation = Atn(1) ' Rotate by 45 degrees (Atn of 1 is 45 degrees in race
```

mass.AttachAnchor anchor 'attach the anchor to the mass element

^{&#}x27; and rotate it around the X axis of the line 45 degrees

XStartOffset Example

Examples:

l AecLayoutGrid2D

1 AecLayoutGrid3D

Sub Example_XStartOffset_AecLayoutGrid2D()

'This example displays the X start offset for a 2D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid X Start Offset is: " & grid.XStartOffset, vbInformation, "XSt Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "XStartOffset Example" End If

End Sub

Sub Example_XStartOffset_AecLayoutGrid3D()

'This example displays the X start offset for a 3D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    MsgBox "Grid X Start Offset is: " & grid.XStartOffset, vbInformation, "XSt
Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "XStartOffset Example"
End If
```

XType Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

```
Sub Example_XType_AecLayoutGrid2D()
```

'This example displays the X type for a 2D layout grid

Dim obj As Object Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
```

Select Case grid.XType

Case 0:

MsgBox "Grid X Type is: Manual Spacing", vbInformation, "XType Exar Case 1:

MsgBox "Grid X Type is: Auto Spacing Even", vbInformation, "XType E Case 2:

MsgBox "Grid X Type is: Auto Spacing Bay", vbInformation, "XType Ex End Select

Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "XType Example" End If

```
Sub Example_XType_AecLayoutGrid3D()
```

'This example displays the X type for a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

If TypeOf obj Is AecLayoutGrid3D Then
 Set grid = obj

Select Case grid.XType

Case 0:

MsgBox "Grid X Type is: Manual Spacing", vbInformation, "XType Exar Case 1:

MsgBox "Grid X Type is: Auto Spacing Even", vbInformation, "XType E Case 2:

MsgBox "Grid X Type is: Auto Spacing Bay", vbInformation, "XType Ex End Select

Else

MsgBox "Not a 3D Layout Grid", vbExclamation, "XType Example" End If

XValue Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_XValue_AecLayoutGrid2D()

'This example displays the X value for a 2D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid X Value is: " & grid.XValue, vbInformation, "XValue Examp Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "XValue Example" End If

End Sub

Sub Example_XValue_AecLayoutGrid3D()

'This example displays the X value for a 3D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid X Value is: " & grid.XValue, vbInformation, "XValue Examp
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "XValue Example"
End If
```

YAlignment Example

Sub Example_YAlignment()

- 'This example looks at the way a selected object is anchored
- ' in relation to the baseline of a grid assembly.
- ' Use this example with a drawing that contains a window
- ' assembly and one or more AEC objects attached to the
- ' assembly.

Dim ent As AcadEntity Dim geo As AecGeo Dim anchor As AecAnchor

Dim alignment As String

On Error Resume Next 'Handle errors in code.

' Prompt user to select an object.

ThisDrawing.Utility.GetEntity ent, pt, "Select object anchored to window asser

- ' Make sure user selected an AEC object, and that the object
- ' is anchored to a grid assembly.

If ent Is Nothing Then

MsgBox "Nothing was selected.", vbExclamation, "YAlignment Example" ElseIf TypeOf ent Is AecGeo Then

Set geo = ent

' Get the anchor the object is attached to.

Set anchor = geo.GetAnchor

On Error GoTo 0

If anchor Is Nothing Then

MsgBox "Selected object is not anchored.", vbExclamation, "YAlignment ElseIf Not TypeOf anchor Is AecAnchorEntToGridAssembly Then

MsgBox "Object is anchored, but not to a grid assembly.", vbExclamation Else

```
Select Case anchor.YAlignment
Case aecInfillAlignCentered
alignment = "Centered."
Case aecInfillAlignFrontFlush
alignment = "In front of the baseline."
Case aecInfillAlignBackFlush
alignment = "In back of the baseline."
Case Else
alignment = "Unknown"
End Select

MsgBox "Y Alignment of object: " & alignment, vbInformation, "YAlign End If
Else
MsgBox "Object selected is not an AEC entity.", vbInformation, "YAlignme End If
```

YDistance Example

```
Sub Example_YDistance()
```

```
'This example will make a mass element and and anchor it 200 units 'to the left of a line.

Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

'this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
'draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line 'set the line as the object (curve) to anchor to anchor.YDistance = 200 'set the Y Distance for the anchor
```

mass.AttachAnchor anchor 'attach the anchor to the mass element

YEndOffset Example

Examples:

l AecLayoutGrid2D

1 AecLayoutGrid3D

Sub Example_YEndOffset_AecLayoutGrid2D()

'This example displays the Y end offset for a 2D layout grid

Dim obj As Object
Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid Y End Offset is: " & grid.YEndOffset, vbInformation, "YEnc Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "YEndOffset Example" End If

End Sub

Sub Example_YEndOffset_AecLayoutGrid3D()

'This example displays the Y end offset for a 3D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Y End Offset is: " & grid.YEndOffset, vbInformation, "YEncElse
MsgBox "Not a 3D Layout Grid", vbExclamation, "YEndOffset Example"
End If
```

YNodes Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_YNodes_AecLayoutGrid2D()

'This example displays the number of Y nodes for a 2D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then
Set grid = obj
MsgBox "Grid Y Nodes is: " & grid.YNodes.Count, vbInformation, "YNode Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "YNodes Example" End If

End Sub

Sub Example_YNodes_AecLayoutGrid3D()

'This example displays the number of Y nodes for a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Y Nodes is: " & grid.YNodes.Count, vbInformation, "YNode
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "YNodes Example"
End If
```

YOffset Example

```
Sub Example_YOffset ()
```

'This example shows the Yoffset of the first viewblock of a multiviewblock

```
Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef
Dim viewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then
Set blockRef = obj
Set viewBlocks = blockRef.viewBlocks
MsgBox "YOffset of View Block 1: " & viewBlocks.Item(0).YOffset, vbInfelse
MsgBox "Not a Multiview Block", vbInformation, "YOffset Example"

End If
```

YPositionFrom Example

Sub Example_YPositionFrom()

```
'This example will make a mass element and anchor it 200 units
```

```
Dim mass As AecMassElement
Dim line As AcadLine
```

Dim anchor As New AecAnchorEntToCurve

Dim pt1(0 To 2) As Double

Dim pt2(0 To 2) As Double

```
pt2(0) = 1000: pt2(1) = 1000
```

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.YDistance = 200 ' set the Y Distance for the anchor anchor.YPositionFrom = aecCurvePositionEnd ' set the anchor to measure from mass.AttachAnchor anchor ' attach the anchor to the mass element

^{&#}x27; from the left of a line.

^{&#}x27; this makes a new mass element 120 x 120 x 120 at 0,0,0

YPositionTo Example

Sub Example_YPositionTo()

```
'This example will make a mass element and anchor it's center 200 units
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
```

```
pt2(0) = 1000: pt2(1) = 1000
```

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.YDistance = 200 ' set the Y Distance for the anchor anchor.YPositionTo = aecEdgePositionCenter ' set the anchor to measure to the mass.AttachAnchor anchor ' attach the anchor to the mass element

^{&#}x27; from the left of a line.

^{&#}x27; this makes a new mass element 120 x 120 x 120 at 0,0,0

YRotation Example

```
Sub Example_YRotation()
```

```
'This example will add anchor a new mass element to a 2D layout grid in the
' drawing.
Dim grid As AecLayoutGrid2D
Dim mass As AecMassElement
Dim pt As Variant
Dim obj As AcadObject
ThisDrawing.Utility.GetEntity obj, pt, "Select grid to attach to"
If TypeOf obj Is AecLayoutGrid2D Then
  Set grid = obj
  Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement
  Dim anchor As New AecAnchorEntToLayoutNode
  anchor.Reference = grid
  ' anchor the mass element to the first node on the grid
  anchor.Node = 1
  anchor.YRotation = Atn(1) ' 45 degrees
  mass.AttachAnchor anchor
  MsgBox "No Layout Grid selected", vbInformation, "Node Example"
```

End Sub

End If

YStartOffset Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_YStartOffset_AecLayoutGrid2D()

'This example displays the Y start offset for a 2D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid Y Start Offset is: " & grid.YStartOffset, vbInformation, "YSt Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "YStartOffset Example" End If

End Sub

Sub Example_YStartOffset_AecLayoutGrid3D()

'This example displays the Y start offset for a 3D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Y Start Offset is: " & grid.YStartOffset, vbInformation, "YSt
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "YStartOffset Example"
End If
```

YType Example

Examples:

l AecLayoutGrid2D

l AecLayoutGrid3D

```
Sub Example_YType_AecLayoutGrid2D()
```

'This example displays the Y type for a 2D layout grid

Dim obj As Object
Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid2D Then
   Set grid = obj
```

Select Case grid.YType

Case 0:

MsgBox "Grid Y Type is: Manual Spacing", vbInformation, "YType Exar. Case 1:

MsgBox "Grid Y Type is: Auto Spacing Even", vbInformation, "YType E Case 2:

MsgBox "Grid Y Type is: Auto Spacing Bay", vbInformation, "YType Ex End Select

Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "YType Example" End If

```
Sub Example_YType_AecLayoutGrid3D()
```

'This example displays the Y type for a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

If TypeOf obj Is AecLayoutGrid3D Then
 Set grid = obj

Select Case grid.YType

Case 0:

MsgBox "Grid Y Type is: Manual Spacing", vbInformation, "YType Exar. Case 1:

MsgBox "Grid Y Type is: Auto Spacing Even", vbInformation, "YType E Case 2:

MsgBox "Grid Y Type is: Auto Spacing Bay", vbInformation, "YType Ex End Select

Else

MsgBox "Not a 3D Layout Grid", vbExclamation, "YType Example" End If

YValue Example

Examples:

1 AecLayoutGrid2D

l AecLayoutGrid3D

Sub Example_YValue_AecLayoutGrid2D()

'This example displays the Y value for a 2D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid2D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 2D Layout Grid"

If TypeOf obj Is AecLayoutGrid2D Then

Set grid = obj

MsgBox "Grid Y Value is: " & grid.YValue, vbInformation, "YValue Examp Else

MsgBox "Not a 2D Layout Grid", vbExclamation, "YValue Example" End If

End Sub

Sub Example_YValue_AecLayoutGrid3D()

'This example displays the Y value for a 3D layout grid

Dim obj As Object

Dim pt As Variant

Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Y Value is: " & grid.YValue, vbInformation, "YValue Examp
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "YValue Example"
End If
```

ZDistance Example

```
Sub Example_ZDistance()
```

```
'This example will make a mass element and and anchor it 200 units 'above a line.

Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

'this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
'draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line 'set the line as the object (curve) to anchor to anchor.ZDistance = 200 'set the Z Distance for the anchor
```

mass.AttachAnchor anchor 'attach the anchor to the mass element

ZEndOffset Example

```
Sub Example_ZEndOffset()
```

'This example displays the Z end offset for a 3D layout grid

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Z End Offset is: " & grid.ZEndOffset, vbInformation, "ZEnd
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "ZEndOffset Example"
End If
```

ZNodes Example

```
Sub Example_ZNodes()
```

'This example displays the number of Z nodes for a 3D layout grid

```
Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Z Nodes is: " & grid.ZNodes.Count, vbInformation, "ZNodes
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "ZNodes Example"
End If
```

ZOffset Example

```
Sub Example_ZOffset ()
```

'This example shows the Zoffset of the first viewblock of a multiviewblock

```
Dim obj As Object
Dim pt As Variant
Dim blockRef As AecMVBlockRef
Dim viewBlocks As AecViewBlocks

ThisDrawing.Utility.GetEntity obj, pt, "Select a Multiview Block"

If TypeOf obj Is AecMVBlockRef Then
Set blockRef = obj
Set viewBlocks = blockRef.viewBlocks
MsgBox "Zoffset of View Block 1: " & viewBlocks.Item(0).ZOffset, vbInfo
Else
MsgBox "Not a Multiview Block", vbInformation, "ZOffset Example"
End If
```

Zoom Example

```
Sub Example_Zoom()
```

'This example displays the zoom of a selected AecCamera object

```
Dim obj As Object
Dim pt As Variant
Dim camera As AecCamera
```

ThisDrawing.Utility.GetEntity obj, pt, "Select a Camera"

```
If TypeOf obj Is AecCamera Then
Set camera = obj
MsgBox "Zoom is: " & camera.Zoom, vbInformation, "Zoom Example"
Else
MsgBox "Not a Camera", vbExclamation, "Zoom Example"
End If
```

ZPositionFrom Example

Sub Example_ZPositionFrom()

```
'This example will make a mass element and and anchor it 200 units 'above a line.
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000
```

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.ZDistance = -200 ' set the Z Distance for the anchor anchor.ZPositionFrom = aecCurvePositionEnd ' set the anchor to measure from mass.AttachAnchor anchor ' attach the anchor to the mass element

ZPositionTo Example

Sub Example ZPositionTo()

```
'This example will make a mass element and anchor it's center 200 units 'above a line.
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000
```

' this makes a new mass element 120 x 120 x 120 at 0,0,0

Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")

' draws a line from 0,0 to 1000,1000

Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the line as the object (curve) to anchor to anchor.ZDistance = -200 ' set the Z Distance for the anchor anchor.ZPositionFrom = aecCurvePositionEnd ' set the anchor to measure from anchor.ZPositionTo = aecEdgePositionCenter ' set the anchor to measure to the mass.AttachAnchor anchor ' attach the anchor to the mass element

ZRotation Example

Sub Example_ZRotation()

```
'This example will make a mass element and and anchor it to a line,
```

```
Dim mass As AecMassElement
Dim line As AcadLine
Dim anchor As New AecAnchorEntToCurve
Dim pt1(0 To 2) As Double
Dim pt2(0 To 2) As Double
pt2(0) = 1000: pt2(1) = 1000

' this makes a new mass element 120 x 120 x 120 at 0,0,0
Set mass = ThisDrawing.ModelSpace.AddCustomObject("AecMassElement")
' draws a line from 0,0 to 1000,1000
Set line = ThisDrawing.ModelSpace.AddLine(pt1, pt2)

anchor.Reference = line ' set the linee as the object (curve) to anchor to anchor.ZDistance = 200 ' set the Z Distance for the anchor anchor.ZRotation = Atn(1) ' Rotate by 45 degrees (Atn of 1 is 45 degrees in rac mass.AttachAnchor anchor ' attach the anchor to the mass element
```

^{&#}x27; and rotate it around the Z axis of the line 45 degrees

ZStartOffset Example

Sub Example_ZStartOffset()

'This example displays the Z start offset for a 3D layout grid

Dim obj As Object
Dim pt As Variant
Dim grid As AecLayoutGrid3D

ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"

```
If TypeOf obj Is AecLayoutGrid3D Then
Set grid = obj
MsgBox "Grid Z Start Offset is: " & grid.ZStartOffset, vbInformation, "ZStartStartStartOffset"
Else
MsgBox "Not a 3D Layout Grid", vbExclamation, "ZStartOffset Example"
End If
```

ZType Example

```
Sub Example_ZType()
  'This example displays the Z type for a 3D layout grid
  Dim obj As Object
  Dim pt As Variant
  Dim grid As AecLayoutGrid3D
  ThisDrawing.Utility.GetEntity obj, pt, "Select a 3D Layout Grid"
  If TypeOf obj Is AecLayoutGrid3D Then
    Set grid = obj
    Select Case grid.ZType
    Case 0:
      MsgBox "Grid Z Type is: Manual Spacing", vbInformation, "ZType Exan
    Case 1:
      MsgBox "Grid Z Type is: Auto Spacing Even", vbInformation, "ZType Ex
    Case 2:
      MsgBox "Grid Z Type is: Auto Spacing Bay", vbInformation, "ZType Exa
    End Select
  Else
    MsgBox "Not a 3D Layout Grid", vbExclamation, "ZType Example"
  End If
```

ZValueExample

ForceHorizontal Property

Specifies whether the tag is aligned with the object or oriented horizontally.

See Also | Example

Signature

object.ForceHorizontal

object

<u>AecAnchorExtendedTagToEnt</u> The object this

property applies to.

ForceHorizontal

Boolean; read-write

TRUE: Orient the tag horizontally. FALSE: Align the tag with the object.

TrueColor Property

Returns the true color of the object.

See Also | Example

Signature

object.TrueColor

object

Aec2DSection, AecBlockRef, AecCamera, AecCellLayoutTool, AecClipVol, AecClipVolRes, AecDisplayComponentEntity, AecEditInPlaceProfile, AecEntity, AecEntRef, AecGeo, AecGridAssembly,

<u>AecLayoutCurve</u>, <u>AecLayoutGrid2D</u>, <u>AecLayoutGrid3D</u>, <u>AecLayoutTool</u>, <u>AecMaskBlockRef</u>, <u>AecMassElement</u>,

AecMassGroup, AecMvBlockRef, AecPolygon,

AecSlice, AecVolumeLayoutTool

The object or objects this property applies to.

TrueColor

<u>AcadAcCmColor</u> object; read-only AutoCAD True Color object.

ActiveConfiguration Property

Returns the active display configuration for the specified view port.

See Also | Example

Signature

RetVal = object.ActiveConfiguration(ViewPort)

object

AecDisplayConfigurations The object this property

applies to.

ViewPort

AcadObject; input-only

The AcadViewport or AcadPViewport that the

display configuration applies to.

RetVal

AecDisplayConfiguration

The active display configuration for the specified

viewport.

AddOverride Method

Adds an object or style override on the display representation.

See Also | Example

Signature

RetVal = object.AddOverride(AecObject)

Object

AecDisplayRepresentation The object or objects this

method applies to.

AecObject

AecObject; input-only

An object instance or style override.

RetVal

AecDisplayProperties

The override display properties for the display

representation.

Remarks

This method will fail if an override is already attached.

DeleteOverride Method

Deletes an existing override from the display representation.

See Also | Example

Signature

object.DeleteOverride AecObject

Object

AecDisplayRepresentation The object or objects this

method applies to.

AecObject

AecObject; input-only

An object instance or style override.

Remarks

This method will fail if there is no existing override to remove.

Duplicate Method

Creates a duplicate display representation.

See Also | Example

Signature

RetVal = object.Duplicate UniqueName

Object

AecDisplayRepresentation The object or objects this

method applies to.

UniqueName

String; input-only

The unique name for the new display

representation.

RetVal

<u>AecDisplayRepresentation</u>

The duplicated display representation.

Remarks

This method will fail if the name is not unique.

See Also

Methods and Properties:

ImportFreeForm method

See Also

Methods and Properties:

ExportFreeForm method

Collections

<u>AecLayerKeys</u>

<u>AecLayerKeyStyles</u>

<u>AecLayerOverrideSettings</u>

<u>AecLayoutCurveNodes</u>

AecLayoutGrid2DNodes

AecLayoutGrid2DXNodes

AecLayoutGrid2DYNodes

<u>AecLayoutGrid3DNodes</u>

<u>AecLayoutGrid3DXNodes</u>

<u>AecLayoutGrid3DYNodes</u>

<u>AecLayoutGrid3DZNodes</u>

<u>AecLayoutNodes</u>

<u>AecMaskBlockStyles</u>

<u>AecMassElementStyles</u>

<u>AecMassGroups</u>

<u>AecMVBlockStyles</u>

<u>AecPolygonStyles</u>

<u>AecProfileStyles</u>

<u>AecRings</u>

<u>AecViewBlocks</u>

ACAD_ANGLE data type

ACAD_ANGLE is a DOUBLE whose units are based on the AUNITS system variable, and whose precision is based on the AUPREC system variable.

See Also:

Methods and Properties:

AngleToString method

AutoCAD ActiveX and VBA Developer's Guide:

Calculating Points and Values

See Also:

Methods and Properties:

AngleToString method

AutoCAD ActiveX and VBA Developer's Guide:

Calculating Points and Values

See Also:

Methods and Properties:

AngleToString method

AutoCAD ActiveX and VBA Developer's Guide:

Calculating Points and Values

The view direction for a given display set.

AecViewDirection				
Name	Value	Description		
aecViewDirectionTop	0	Тор		
aecViewDirectionBottom	1	Bottom		
aecViewDirectionLeft	2	Left		
aecViewDirectionRight	3	Right		
aecViewDirectionFront	4	Front		
aecViewDirectionBack	5	Back		
aecViewDirectionModel	6	Model		

The fixed view direction used by a display configuration.

AecFixedViewDirection				
Name	Value	Description		
aecFixedViewDirectionTop	0	Тор		
aecFixedViewDirectionBottom	1	Bottom		
aecFixedViewDirectionLeft	2	Left		
aecFixedViewDirectionRight	3	Right		
aecFixedViewDirectionFront	4	Front		
aecFixedViewDirectionBack	5	Back		
aecFixedViewDirectionViewport	6	Viewport		

The hatch type used by a display component.

AecHatchType				
Name	Value	Description		
aecHatchTypeUserDefined	0	UserDefined		
aecHatchTypePreDefined	1	PreDefined		
aecHatchTypeCustomDefined	2	CustomDefined		
aecHatchTypeSolidFill	3	SolidFill		

The surface hatch placement.

AecSurfaceHatchPlacement				
Name	Value	Description		
aecSurfaceHatchPlacementTop	1	Тор		
aecSurfaceHatchPlacementBottom	2	Bottom		
aecSurfaceHatchPlacementLeft	3	Left		
aecSurfaceHatchPlacementRight	4	Right		
aecSurfaceHatchPlacementFront	5	Front		
aecSurfaceHatchPlacementBack	6	Back		

The surface mapping type.

AecSurfaceMappingType				
Name	Value	Description		
aecSurfaceMappingTypeDefaultMapping	0	DefaultMapping		
aecSurfaceMappingTypeFaceMapping	1	FaceMapping		
aecSurfaceMappingTypeAsSurfaceHatch	2	AsSurfaceHatch		

ACAD_DISTANCE data type

ACAD_DISTANCE is a DOUBLE whose units are based on the LUNITS system variable, and whose precision is based on the LUPREC system variable.