

New Objects

Visit the Office Developer Center on the Microsoft Developer Network Web site for the latest information about programming with Office PowerPoint 2003, including product news, technical articles, downloads, and samples.

No new objects are added to the Office PowerPoint 2003 object model.



New Properties (Alphabetical List)

Visit the Office Developer Center on the Microsoft Developer Network Web site for the latest information about programming with Office PowerPoint 2003, including product news, technical articles, downloads, and samples.

The following table lists properties added to the Office PowerPoint 2003 object model (sorted alphabetically).

New Property	Object(s)
<u>DocumentLibraryVersions</u>	Presentation
<u>Permission</u>	Presentation
<u>SharedWorkspace</u>	Presentation
<u>Sync</u>	Presentation

New Properties (by Object)

Visit the Office Developer Center on the Microsoft Developer Network Web site for the latest information about programming with Office PowerPoint 2003, including product news, technical articles, downloads, and samples.

The following table lists properties added to the Office PowerPoint 2003 object model (sorted by object name).

Object	New Properties
Presentation	DocumentLibraryVersions , Permission , SharedWorkspace , Sync
<hr/>	

New Methods

Visit the Office Developer Center on the Microsoft Developer Network Web site for the latest information about programming with Office PowerPoint 2003, including product news, technical articles, downloads, and samples.

The following table lists methods added to the Office PowerPoint 2003 object model.

New Method	Object
Presentation	<u>SendFaxOverInternet</u>

--	--

New Events

Visit the Office Developer Center on the Microsoft Developer Network Web site for the latest information about programming with Office PowerPoint 2003, including product news, technical articles, downloads, and samples.

The following table lists events added to the Office PowerPoint 2003 object model.

New Event	Object
PresentationSync	Application

Using Events with the Application Object

To create an event handler for an event of the [Application](#) object, you need to complete the following three steps:

1. Declare an object variable in a class module to respond to the events.
2. Write the specific event procedures.
3. Initialize the declared object from another module.

Declare the Object Variable

Before you can write procedures for the events of the **Application** object, you must create a new class module and declare an object of type **Application** with events. For example, assume that a new class module is created and called EventClassModule. The new class module contains the following code.

```
Public WithEvents App As Application
```

Write the Event Procedures

After the new object has been declared with events, it appears in the **Object** list in the class module, and you can write event procedures for the new object. (When you select the new object in the **Object** list, the valid events for that object are listed in the **Procedure** list.) Select an event from the **Procedure** list; an empty procedure is added to the class module.

```
Private Sub App_NewPresentation()
```

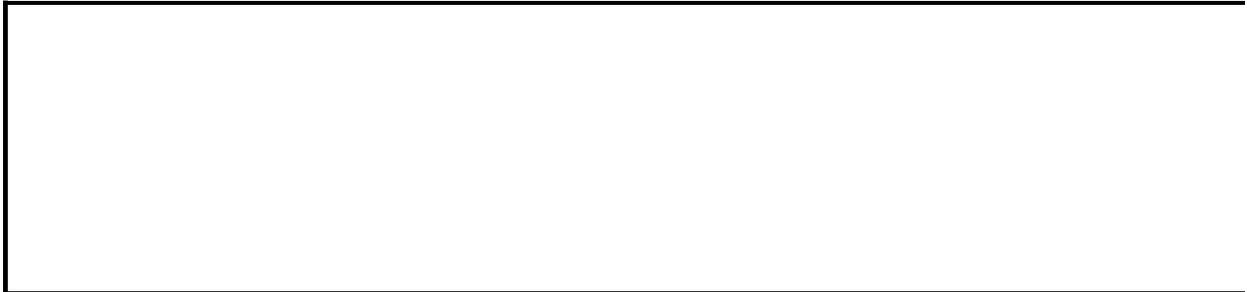
```
End Sub
```

Initializing the Declared Object

Before the procedure will run, you must connect the declared object in the class module (App in this example) with the **Application** object. You can do this with the following code from any module.

```
Dim X As New EventClassModule
Sub InitializeApp()
    Set X.App = Application
End Sub
```

Run the InitializeApp procedure. After the procedure is run, the App object in the class module points to the Microsoft PowerPoint **Application** object, and the event procedures in the class module will run when the events occur.



Working with shapes (drawing objects)

Shapes, or drawing objects, are represented by three different objects: the [Shapes](#) collection, the [ShapeRange](#) collection, and the [Shape](#) object. In general, you use the **Shapes** collection to create shapes and when you want to iterate through all the shapes on a slide; you use the **Shape** object when you want to modify a single shape; and you use the **ShapeRange** collection when you want to modify multiple shapes the same way you can work with multiple selected shapes in the user interface.

Setting properties for a shape

Many formatting properties of shapes aren't set by properties that apply directly to the **Shape** or **ShapeRange** object. Instead, related shape attributes are grouped under secondary objects, such as the **FillFormat** object, which contains all the properties that relate to the shape's fill, or the **LinkFormat** object, which contains all the properties that are unique to linked OLE objects. To set properties for a shape, you must first return the object that represents the set of related shape attributes and then set properties of that returned object. For example, you use the **Fill** property to return the **FillFormat** object, and then you set the **ForeColor** property of the **FillFormat** object to set the fill foreground color for the specified shape, as shown in the following example.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(1).Fill.ForeColor.RGB = RGB(255, 0, 0)
```

Applying a property or method to several shapes at the same time

In the user interface, there are some operations you can perform with several shapes selected; for example, you can select several shapes and set all their individual fills at once. There are other operations you can only perform with a single shape selected; for example, you can only edit the text in a shape if a single shape is selected.

In Visual Basic, there are two ways to apply properties and methods to a set of shapes. These two ways allow you to perform any operation that you can perform on a single shape on a range of shapes, whether or not you can perform the same operation in the user interface.

- If the operation works on multiple selected shapes in the user interface, you can perform the same operation in Visual Basic by constructing a **ShapeRange** collection that contains the shapes you want to work with, and applying the appropriate properties and methods directly to the **ShapeRange** collection.
- If the operation doesn't work on multiple selected shapes in the user interface, you can still perform the operation in Visual Basic by looping through the **Shapes** collection or through a **ShapeRange** collection that contains the shapes you want to work with, and applying the appropriate properties and methods to the individual **Shape** objects in the collection.

Many properties and methods that apply to the **Shape** object and **ShapeRange** collection fail if applied to certain kinds of shapes. For example, the [TextFrame](#) property fails if applied to a shape that cannot contain text. If you are not positive that each shape in a **ShapeRange** collection can have a certain property or method applied to it, don't apply the property or method to the **ShapeRange** collection. If you want to apply one of these properties or methods to a collection of shapes, you must loop through the collection and test each individual shape to make sure it is an appropriate type of shape before applying the property or method to it.

Applying a property or method to a ShapeRange collection

If you can perform an operation on multiple selected shapes in the user interface at the same time, you can do the programmatic equivalent by constructing a **ShapeRange** collection and then applying the appropriate properties or methods to it. The following example constructs a shape range that contains the AutoShapes named "Big Star" and "Little Star" on myDocument and applies a gradient fill to them.

```
Set myDocument = ActivePresentation.Slides(1)
Set myRange = myDocument.Shapes _
    .Range(Array("Big Star", "Little Star"))
myRange.Fill.PresetGradient msoGradientHorizontal, _
    1, msoGradientBrass
```

The following are general guidelines for how properties and methods behave when they're applied to a **ShapeRange** collection.

- Applying a method to a the collection is equivalent to applying the method to each individual **Shape** object in that collection.
- Setting the value of a property of the collection is equivalent to setting the value of the property of each individual shape in that range.
- A property of the collection that returns a constant returns the value of the property for an individual shape in the collection if all shapes in the collection have the same value for that property. If not all shapes in the collection have the same value for the property, it returns the "mixed" constant.
- A property of the collection that returns a simple data type (such as **Long**, **Single**, or **String**) returns the value of the property for an individual shape if all shapes in the collection have the same value for that property.
- The value of some properties can be returned or set only if there's exactly one shape in the collection. If there's more than one shape in the collection, a run-time error occurs. This is generally the case for returning or setting properties when the equivalent action in the user interface is possible only with a single shape (actions such as editing text in a shape or editing the points of a freeform).

The preceding guidelines also apply when you are setting properties of shapes that are grouped under secondary objects of the **ShapeRange** collection, such as the **FillFormat** object. If the secondary object represents operations that can be performed on multiple selected objects in the user interface, you will be able to return the object from a **ShapeRange** collection and set its properties. For example, you can use the **Fill** property to return the **FillFormat** object that represents the fills of all the shapes in the **ShapeRange** collection. Setting the properties of this **FillFormat** object will set the same properties for all the individual shapes in the **ShapeRange** collection.

Looping through a Shapes or ShapeRange collection

Even if you cannot perform an operation on several shapes in the user interface at the same time by selecting them and then using a command, you can perform the equivalent action programmatically by looping through the **Shapes** collection or through a **ShapeRange** collection that contains the shapes you want to work with, and applying the appropriate properties and methods to the individual **Shape** objects in the collection. The following example loops through all the shapes on myDocument and adds text to each shape that is an AutoShape.

```
Set myDocument = ActivePresentation.Slides(1)
For Each sh In myDocument.Shapes
    If sh.Type = msoAutoShape Then
        sh.TextFrame.TextRange.InsertAfter " (version 1)"
    End If
Next
```

The following example constructs **ShapeRange** collection that contains all the currently selected shapes in the active window and sets the text in each shape in the collection that can contain text.

```
For Each sh in ActiveWindow.Selection.ShapeRange
    If sh.HasTextFrame Then
        sh.TextFrame.TextRange = "Initially selected"
    End If
Next
```

Aligning, distributing, and grouping shapes in a shape range

Use the [Align](#) and [Distribute](#) methods to position a set of shapes relative to one another or relative to the document that contains them. Use the [Group](#) method or the [Regroup](#) method to form a single grouped shape from a set of shapes.



Working with Panes and Views

Changing the Active View

You can return or set the current view in the active document window with the [ViewType](#) property. This example changes the view in the active document window to slide view.

```
ActiveWindow.ViewType = ppViewSlide
```

Changing Panes in Normal View

In normal view, you can use the **ViewType** property with the active [Pane](#) object to return the active pane. The **ViewType** property returns one of the following **PpViewType** constants, identifying the active pane: **ppViewNotesPage**, **ppViewOutline**, or **ppViewSlide**. All other views have only one pane and the **ViewType** property returns the same **PpViewType** constant value as the active document window.

You can activate a pane by setting the **ViewType** property or by using the [Activate](#) method. This example returns the value of the **ViewType** property to identify the active view and active pane. If the active view is normal view and the active pane is the notes pane, then the slide pane is activated with the **Activate** method.

```
With ActiveWindow
    If .ViewType = ppViewNormal and _
        .ActivePane.ViewType = ppViewNotesPage Then
        .Panes(2).Activate
    End If
End With
```

Resizing panes

You can use the [SplitHorizontal](#) and the [SplitVertical](#) properties to reposition the pane dividers in normal view to the specified percentage of the available document window. This resizes the panes on either side of the divider. The maximum value of these properties is always less than 100% because the slide pane has a minimum size that depends on a 10% zoom level. This example sets the percentage of the available document window height that the slide pane occupies to 65 percent, leaving the notes pane at 35 percent.

```
ActiveWindow.SplitVertical = 65
```



Publishing a Web Presentation

In Microsoft PowerPoint, you can publish a presentation directly to a Web server and you can edit HTML documents directly in PowerPoint.

Saving a Presentation as a Web Page

Saving a presentation as a Web page is the process of creating and saving an HTML version of a presentation. To do this, use the [SaveAs](#) method, as shown in the following example which saves the current presentation as c:\myfile.htm.

```
ActivePresentation.SaveAs "c:\myfile.htm", ppSaveAsHTMLv3, msoTrue
```

Publishing a Web Presentation

Publishing a Web presentation is the process of creating an HTML version of a presentation and saving it to a Web server or a file server using the [Publish](#) method. This differs from saving a presentation as a Web page using the [SaveAs](#) method in that when you publish a Web presentation, you can customize the presentation by setting various attributes, and you can publish the presentation directly to a Web server. After setting various properties of the [WebOptions](#) object, this example publishes the active presentation to a Web server with the URL address <http://www.someones.homepage/mallard.htm>.

```
With ActivePresentation
  With .WebOptions
    .FrameColors = ppFrameColorsWhiteTextOnBlack
    .RelyonVML = True
    .OrganizeInFolder = True
  End With
  With .PublishObjects(1)
    .FileName = "http://www.someones.homepage/mallard.htm"
    .SourceType = ppPublishAll
    .SpeakerNotes = True
    .Publish
  End With
End With
```

Web Options and Default Web Options

When using the **Publish** method, you can customize the appearance, content, browser support, editing support, graphics formats, screen resolution, file organization, and encoding of the HTML document by setting properties of the [DefaultWebOptions](#) object and the [WebOptions](#) object. The **DefaultWebOptions** object contains application-level properties. These settings are overridden by any presentation-level property settings that have the same name, contained in the **WebOptions** object.

This example sets various application-level properties for Web publishing. They will be the default settings for any current or future loaded presentation until the settings are changed again. The code then resets the [ResizeGraphics](#) property for the active presentation, which overrides the application-level default. It publishes the active presentation as "c:\mallard.htm."

```
With Application.DefaultWebOptions
    .FrameColors = ppFrameColorsWhiteTextOnBlack
    .IncludeNavigation = False
    .ResizeGraphics = True
End With
With ActivePresentation
    .WebOptions.ResizeGraphics = False
    With .PublishObjects(1)
        .FileName = "c:\mallard.htm"
        .SourceType = ppPublishAll
        .SpeakerNotes = True
        .Publish
    End With
End With
```

Opening an HTML document in PowerPoint

To edit an HTML document in PowerPoint, open the HTML document using the [Open](#) method. This example opens the file "myfile.htm" for editing.

```
Presentations.Open Filename:="c:\Windows\myfile.htm"
```



[Show All](#)

Using ActiveX controls on slides

You can add controls to your slides to provide a sophisticated way to exchange information with the user while a slide show is running. For example, you could use controls on slides to allow viewers of a show designed to be run in a kiosk a way to choose options and then run a custom show based on the viewer's choices.

For general information on adding and working with controls, see [Using ActiveX Controls on a Document](#) and [Creating a Custom Dialog Box](#).

Keep the following points in mind when you are working with controls on slides.

- A control on a slide is in [design mode](#) except when the slide show is running.
- If you want a control to appear on all slides in a presentation, add it to the slide master.
- The **Me** keyword in an event procedure for a control on a slide refers to the slide, not the control.

Writing event code for controls on slides is very similar to writing event code for controls on forms. The following procedure sets the background for the slide the button named "cmdChangeColor" is on when the button is clicked.

```
Private Sub cmdChangeColor_Click()  
    With Me  
        .FollowMasterBackground = Not .FollowMasterBackground  
        .Background.Fill.PresetGradient _  
            msoGradientHorizontal, 1, msoGradientBrass  
    End With  
End Sub
```

You may want to use controls to provide your slide show with navigation tools that are more complex than those built into Microsoft PowerPoint. For example, if you add two buttons named "cmdBack" and "cmdForward" to the slide master and write the following code behind them, all slides based on the master (and set to show master background graphics) will have these professional-looking navigation buttons that will be active during a slide show.

```
Private Sub cmdBack_Click()  
    Me.Parent.SlideShowWindow.View.Previous  
End Sub
```

```
Private Sub cmdForward_Click()  
    Me.Parent.SlideShowWindow.View.Next  
End Sub
```

To work with all the ActiveX controls on a slide without affecting the other shapes on the slide, you can construct a [ShapeRange](#) collection that contains only controls. You can then apply properties and methods to the entire collection or iterate through the collection to work with each control individually. The following example aligns all the controls on slide one in the active presentation and distributes them vertically.

```
With ActivePresentation.Slides(1).Shapes  
    numShapes = .Count  
    If numShapes > 1 Then  
        numControls = 0  
        ReDim ctrlArray(1 To numShapes)  
        For i = 1 To numShapes  
            If .Item(i).Type = msoOLEControlObject Then  
                numControls = numControls + 1  
                ctrlArray(numControls) = .Item(i).Name  
            End If  
        Next  
        If numControls > 1 Then  
            ReDim Preserve ctrlArray(1 To numControls)  
            Set ctrlRange = .Range(ctrlArray)  
            ctrlRange.Distribute msoDistributeVertically, True  
            ctrlRange.Align msoAlignLefts, True  
        End If  
    End If  
End With
```



[Show All](#)

Using ActiveX controls on a document

Just as you can add ActiveX [controls](#) to [custom dialog boxes](#), you can add controls directly to a document when you want to provide a sophisticated way for the user to interact directly with your macro without the distraction of dialog boxes. Use the following procedure to add ActiveX controls to your document. For more specific information about using ActiveX controls in Microsoft PowerPoint, see [Using ActiveX controls on slides](#).

1. [Add controls to the document](#)

Display the **Control Toolbox**, click the control you want to add, and then click the document.

2. [Set control properties](#)

Right-click a control in design mode and click **Properties** to display the Properties window.

3. [Initialize the controls](#)

You can initialize controls in a procedure.

4. [Write event procedures](#)

All controls have a predefined set of events. For example, a command button has a Click event that occurs when the user clicks the command button. You can write event procedures that run when the events occur.

5. [Use control values while code is running](#)

Some properties can be set at run time.



Creating a custom dialog box

Use the following procedure to create a custom dialog box:

1. [Create a UserForm](#)

On the **Insert** menu in the Visual Basic Editor, click **UserForm**.

2. [Add controls to the UserForm](#)

Find the control you want to add in the **Toolbox** and drag the control onto the form.

3. [Set control properties](#)

Right-click a control in design mode and click **Properties** to display the Properties window.

4. [Initialize the controls](#)

You can initialize controls in a procedure before you show a form, or you can add code to the Initialize event of the form.

5. [Write event procedures](#)

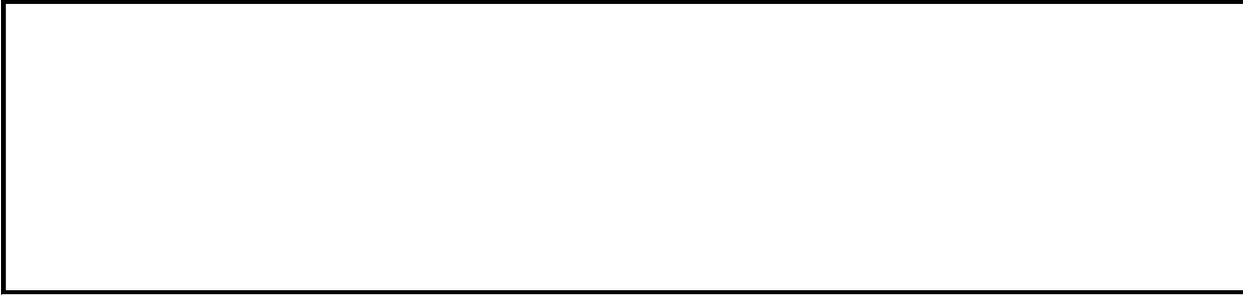
All controls have a predefined set of events. For example, a command button has a Click event that occurs when the user clicks the command button. You can write event procedures that run when the events occur.

6. [Show the dialog box](#)

Use the **Show** method to display a UserForm.

7. [Use control values while code is running](#)

Some properties can be set at run time. Changes made to the dialog box by the user are lost when the dialog box is closed.



[Show All](#)

Controlling one Microsoft Office application from another

If you want to run code in one Microsoft Office application that works with the objects in another application, follow these steps.

1. Set a reference to the other application's type library in the **References** dialog box (**Tools** menu). After you have done this, the objects, properties, and methods will show up in the Object Browser and the syntax will be checked at compile time. You can also get context-sensitive Help on them.
2. Declare object variables that will refer to the objects in the other application as specific types. Make sure you qualify each type with the name of the application that is supplying the object. For example, the following statement declares a variable that will point to a Microsoft Word document, and another that refers to a Microsoft Excel application.

```
Dim appWD As Word.Application, wbXL As Excel.Application
```

Note You must follow the steps above if you want your code to be [early bound](#).

3. Use the **New** keyword with the [OLE Programmatic Identifier](#) of the object you want to work with in the other application, as shown in the following example. If you want to see the session of the other application, set the **Visible** property to **True**.

```
Dim appWD As Word.Application
```

```
Set appWD = New Word.Application  
appWD.Visible = True
```

4. Apply properties and methods to the object contained in the variable. For example, the following instruction creates a new Word document.

```
Dim appWD As Word.Application
```

```
Set appWD = New Word.Application  
appWD.Documents.Add
```

5. When you are done working with the other application, use the **Quit** method to close it, as shown in the following example.

```
appWd.Quit
```



OLE Programmatic Identifiers

You can use an OLE programmatic identifier (sometimes called a ProgID) to create an Automation object. The following tables list OLE programmatic identifiers for ActiveX controls, Microsoft Office applications, and Microsoft Office Web Components.

[ActiveX Controls](#)

[Microsoft Access](#)

[Microsoft Excel](#)

[Microsoft Graph](#)

[Microsoft Office Web Components](#)

[Microsoft Outlook](#)

[Microsoft PowerPoint](#)

[Microsoft Word](#)

ActiveX Controls

To create the ActiveX controls listed in the following table, use the corresponding OLE programmatic identifier.

To create this control	Use this identifier
CheckBox	Forms.CheckBox.1
ComboBox	Forms.ComboBox.1
CommandButton	Forms.CommandButton.1
Frame	Forms.Frame.1
Image	Forms.Image.1
Label	Forms.Label.1
ListBox	Forms.ListBox.1
MultiPage	Forms.MultiPage.1
OptionButton	Forms.OptionButton.1
ScrollBar	Forms.ScrollBar.1
SpinButton	Forms.SpinButton.1
TabStrip	Forms.TabStrip.1
TextBox	Forms.TextBox.1
ToggleButton	Forms.ToggleButton.1

Microsoft Access

To create the Microsoft Access objects listed in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Access available on the machine where the macro is running.

To create this object	Use one of these identifiers
Application	Access.Application
CurrentData	Access.CodeData, Access.CurrentData
CurrentProject	Access.CodeProject, Access.CurrentProject
DefaultWebOptions	Access.DefaultWebOptions

Microsoft Excel

To create the Microsoft Excel objects listed in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Excel available on the machine where the macro is running.

To create this object	Use one of these identifiers	Comments
Application	Excel.Application	
Workbook	Excel.AddIn	
Workbook	Excel.Chart	Returns a workbook containing two worksheets; one for the chart and one for its data. The chart worksheet is the active worksheet.
Workbook	Excel.Sheet	Returns a workbook with one worksheet.

Microsoft Graph

To create the Microsoft Graph objects listed in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Graph available on the machine where the macro is running.

To create this object Use one of these identifiers

Application MSGraph.Application

Chart MSGraph.Chart

Microsoft Office Web Components

To create the Microsoft Office Web Components objects listed in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Microsoft Office Web Components available on the machine where the macro is running.

To create this object	Use one of these identifiers
ChartSpace	OWC10.Chart
DataSourceControl	OWC10.DataSourceControl
ExpandControl	OWC.ExpandControl
PivotTable	OWC10.PivotTable
RecordNavigationControl	OWC10.RecordNavigationControl
Spreadsheet	OWC10.Spreadsheet

Microsoft Outlook

To create the Microsoft Outlook object given in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Outlook available on the machine where the macro is running.

To create this object Use one of these identifiers

Application	Outlook.Application
--------------------	---------------------

Microsoft PowerPoint

To create the Microsoft PowerPoint object given in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of PowerPoint available on the machine where the macro is running.

To create this object Use one of these identifiers

Application	PowerPoint.Application
--------------------	------------------------

Microsoft Word

To create the Microsoft Word objects listed in the following table, use one of the corresponding OLE programmatic identifiers. If you use an identifier without a version number suffix, you create an object in the most recent version of Word available on the machine where the macro is running.

To create this object Use one of these identifiers

Application	Word.Application
Document	Word.Document, Word.Template
Global	Word.Global



ActionSettings Collection Object

Multiple objects [└ ActionSettings](#)
└ [ActionSetting](#)
└ Multiple objects

A collection that contains the two [ActionSetting](#) objects for a shape or text range. One **ActionSetting** object represents how the specified object reacts when the user clicks it during a slide show, and the other **ActionSetting** object represents how the specified object reacts when the user moves the mouse pointer over it during a slide show.

Using the ActionSettings Collection

Use the [ActionSettings](#) property to return the **ActionSettings** collection. Use **ActionSettings(index)**, where *index* is either **ppMouseClick** or **ppMouseOver**, to return a single **ActionSetting** object. The following example specifies that the CalculateTotal macro be run whenever the mouse pointer passes over the shape during a slide show.

```
With ActivePresentation.Slides(1).Shapes(3) _  
    .ActionSettings(ppMouseOver)  
    .Action = ppActionRunMacro  
    .Run = "CalculateTotal"  
    .AnimateAction = True  
End With
```



[Show All](#)

AddIns Collection Object

[Application](#) └ [AddIns](#)
└ [AddIn](#)

A collection of [AddIn](#) objects that represent all the Microsoft PowerPoint-specific add-ins available to PowerPoint, regardless of whether or not they're loaded. This does not include [Component Object Model \(COM\) add-ins](#).

Using the AddIns Collection

Use the **AddIns** method to return the **AddIns** collection. The following example displays the names of all the add-ins that are currently loaded in PowerPoint.

```
For Each ad In AddIns
    If ad.Loaded Then MsgBox ad.Name
Next
```

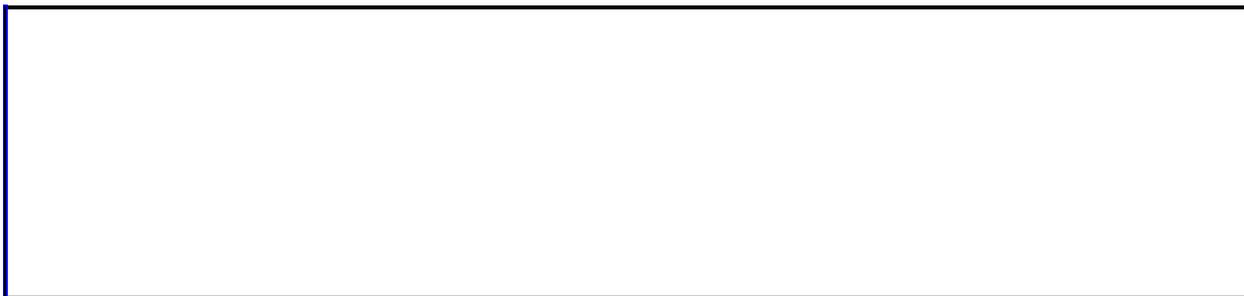
Use the **Add** method to add a PowerPoint-specific add-in to the list of those available. The **Add** method adds an add-in to the list but doesn't load the add-in. To load the add-in, set the **Loaded** property of the add-in to **True** after you use the **Add** method. You can perform these two actions in a single step, as shown in the following example (note that you use the name of the add-in, not its title, with the **Add** method).

```
AddIns.Add("graphdrs.ppa").Loaded = True
```

Use **AddIns(index)**, where *index* is the add-in's title or index number, to return a single **AddIn** object. The following example loads the hypothetical add-in titled "my ppt tools."

```
AddIns("my ppt tools").Loaded = True
```

Don't confuse the add-in title with the add-in name, which is the file name of the add-in. You must spell the add-in title exactly as it's spelled in the **Add-Ins** dialog box, but the capitalization doesn't have to match.



Adjustments Object

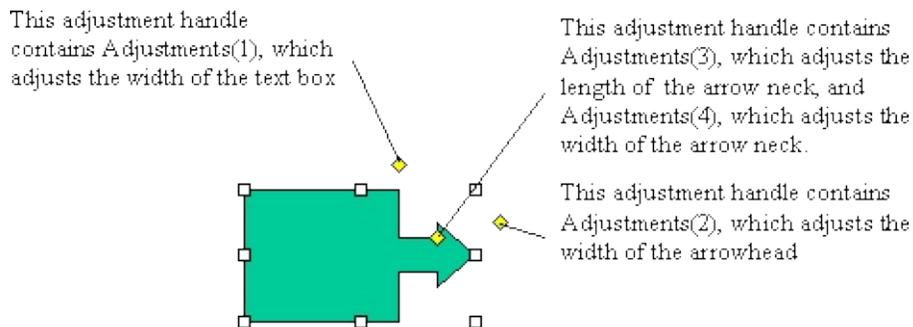
Multiple objects [Adjustments](#)

Contains a collection of adjustment values for the specified AutoShape, WordArt object, or connector. Each adjustment value represents one way an adjustment handle can be adjusted. Because some adjustment handles can be adjusted in two ways— for instance, some handles can be adjusted both horizontally and vertically— a shape can have more adjustment values than it has adjustment handles. A shape can have up to eight adjustments.

Using the Adjustments Object

Use the **Adjustments** property to return an **Adjustments** object. Use **Adjustments(index)**, where *index* is the adjustment value's index number, to return a single adjustment value.

Different shapes have different numbers of adjustment values, different kinds of adjustments change the geometry of a shape in different ways, and different kinds of adjustments have different ranges of valid values. For example, the following illustration shows what each of the four adjustment values for a right-arrow callout contributes to the definition of the callout's geometry.



Note Because each adjustable shape has a different set of adjustments, the best way to verify the adjustment behavior for a specific shape is to manually create an instance of the shape, make adjustments with the macro recorder turned on, and then examine the recorded code.

The following table summarizes the ranges of valid adjustment values for different types of adjustments. In most cases, if you specify a value that's beyond the range of valid values, the closest valid value will be assigned to the adjustment.

Type of adjustment	Valid values
Linear (horizontal	Generally the value 0.0 represents the left or top edge of the shape and the value 1.0 represents the right or bottom edge of the shape. Valid values correspond to valid adjustments you can make to the shape manually. For example, if you can only pull an adjustment handle half way across the shape manually, the maximum value for

or vertical) the corresponding adjustment will be 0.5. For shapes such as connectors and callouts, where the values 0.0 and 1.0 represent the limits of the rectangle defined by the starting and ending points of the connector or callout line, negative numbers and numbers greater than 1.0 are valid values.

Radial An adjustment value of 1.0 corresponds to the width of the shape. The maximum value is 0.5, or half way across the shape.

Angle Values are expressed in degrees. If you specify a value outside the range – 180 to 180, it will be normalized to be within that range.

The following example adds a right-arrow callout to myDocument and sets adjustment values for the callout. Note that although the shape has only three adjustment handles, it has four adjustments. Adjustments three and four both correspond to the handle between the head and neck of the arrow.

```
Set myDocument = ActivePresentation.Slides(1)
Set rac = myDocument.Shapes _
    .AddShape(msoShapeRightArrowCallout, 10, 10, 250, 190)
With rac.Adjustments
    .Item(1) = 0.5      'adjusts width of text box
    .Item(2) = 0.15    'adjusts width of arrow head
    .Item(3) = 0.8     'adjusts length of arrow head
    .Item(4) = 0.4     'adjusts width of arrow neck
End With
```



AnimationBehaviors Collection

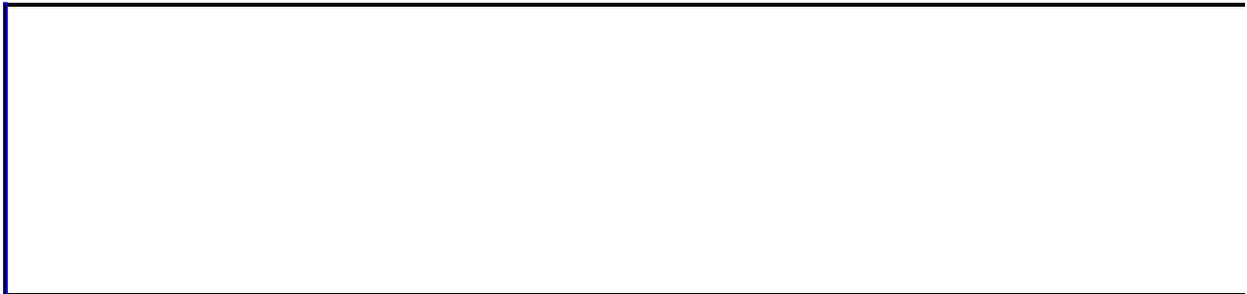
[Effect](#) └ [AnimationBehaviors](#)
└ [AnimationBehavior](#)
└ Multiple objects

Represents a collection of [AnimationBehavior](#) objects.

Using the AnimationBehaviors collection

Use the [Add](#) method to add an animation behavior. The following example adds a five-second animated rotation behavior to the main animation sequence on the first slide.

```
Sub AnimationObject()  
    Dim timeMain As TimeLine  
  
    'Reference the main animation timeline  
    Set timeMain = ActivePresentation.Slides(1).TimeLine  
  
    'Add a five-second animated rotation behavior  
    'as the first animation in the main animation sequence  
    timeMain.MainSequence(1).Behaviors.Add Type:=msoAnimTypeRotation  
End Sub
```



AnimationPoints Collection

[PropertyEffect](#) └ [AnimationPoints](#)
└ [AnimationPoint](#)

Represents a collection of animation points for a [PropertyEffect](#) object.

Using the AnimationPoints collection

Use the [Points](#) property of the [PropertyEffect](#) object to return an **AnimationPoints** collection object. The following example adds an animation point to the first behavior in the active presentation's main animation sequence.

```
Sub AddPoint()  
    ActivePresentation.Slides(1).TimeLine.MainSequence(1) _  
        .Behaviors(1).PropertyEffect.Points.Add  
End Sub
```

Transitions from one animation point to another can sometimes be abrupt or choppy. Use the [Smooth](#) property to make transitions smoother. This example smoothes the transitions between animation points.

```
Sub SmoothTransition()  
    ActivePresentation.Slides(1).TimeLine.MainSequence(1) _  
        .Behaviors(1).PropertyEffect.Points.Smooth = msoTrue  
End Sub
```



[Show All](#)

Borders Collection Object

Multiple objects [└ Borders](#)
└ [LineFormat](#)
└ [ColorFormat](#)

A collection of **[LineFormat](#)** objects that represent the borders and diagonal lines of a cell or range of cells in a table.

Using the Borders Collection

Each **Cell** object or **CellRange** collection has six elements in the **Borders** collection. You cannot add objects to the **Borders** collection.

Use **Borders**(*index*), where *index* identifies the cell border or diagonal line, to return a single **Border** object. *Index* can be any [PPBorderStyle](#) constant.

PPBorderStyle can be one of these PPBorderStyle constants.

ppBorderBottom

ppBorderLeft

ppBorderRight

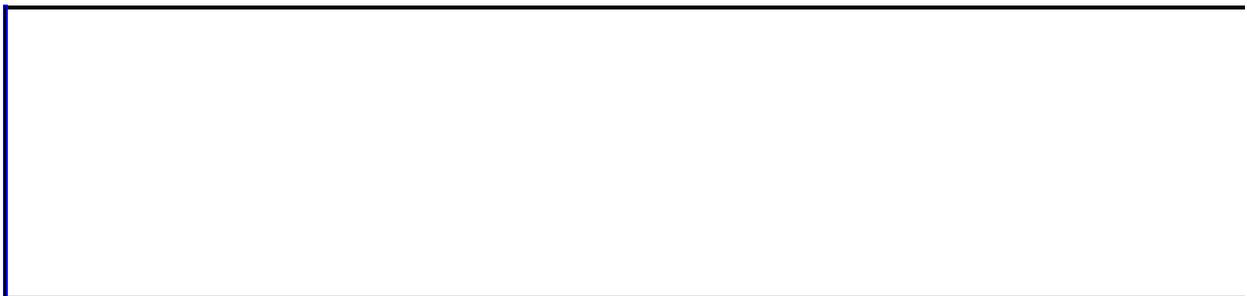
ppBorderTop

ppBorderDiagonalDown

ppBorderDiagonalUp

Use the [DashStyle](#) property to apply a dashed line style to a **Border** object. This example selects the second row from the table and applies a dashed line style to the bottom border.

```
ActiveWindow.Selection.ShapeRange.Table.Rows(2) _  
    .Cells.Borders(ppBorderBottom).DashStyle = msoLineDash
```



CellRange Collection Object

Multiple objects [CellRange](#)
[Borders](#)

A collection of [Cell](#) objects in a table column or row. The **CellRange** collection represents all the cells in the specified column or row. To use the **CellRange** collection, use the **Cells** keyword.

Using the CellRange Collection

Use the **Cells** property to return the **CellRange** collection. This example sets the right border for the cells in the first column of the table to a dashed line style.

```
With ActivePresentation.Slides(2).Shapes(5).Table.Columns(1).Cells
    .Borders(ppBorderRight).DashStyle = msoLineDash
End With
```

This example returns the number of cells in row one of the selected table.

```
num = Activewindow.Selection.ShapeRange.Table.Rows(1).Cells.Count
```

Use **Cell**(*row*, *column*), where *row* is the row number and *column* is the column number, or **Cells**(*index*), where *index* is the number of the cell in the specified row or column, to return a single **Cell** object. Cells are numbered from left to right in rows and from top to bottom in columns. With right-to-left language settings, this scheme is reversed. The example below merges the first two cells in row one of the table in shape five on slide two.

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Cell(1, 1).Merge MergeTo:=.Cell(1, 2)
End With
```

Remarks

Although the collection object is named **CellRange** and is shown in the Object Browser, this keyword is not used in programming the PowerPoint object model. The keyword **Cells** is used instead.

You cannot programmatically add cells to or delete cells from a PowerPoint table. Use the **AddTable** method with the **Table** object to add a new table. Use the **Add** method of the **Columns** or **Rows** collections to add a column or row to a table. Use the **Delete** method of the **Columns** or **Rows** collections to delete a column or row from a table.



ColorSchemes Collection Object

[Presentation](#) └ [ColorSchemes](#)
└ [ColorScheme](#)

A collection of all the [ColorScheme](#) objects in the specified presentation. Each **ColorScheme** object represents a color scheme, which is a set of colors that are used together on a slide.

Using the ColorSchemes Collection

Use the [ColorSchemes](#) property to return the **ColorSchemes** collection. Use **ColorSchemes(index)**, where index is the color scheme index number, to return a single **ColorScheme** object. The following example deletes color scheme two from the active presentation.

```
ActivePresentation.ColorSchemes(2).Delete
```

Use the [Add](#) method to create a new color scheme and add it to the **ColorSchemes** collection. The following example adds a color scheme to the active presentation and sets the title color and background color for the color scheme (because no argument was used with the **Add** method, the added color scheme is initially identical to the first standard color scheme in the presentation).

```
With ActivePresentation.ColorSchemes.Add  
    .Colors(ppTitle).RGB = RGB(255, 0, 0)  
    .Colors(ppBackground).RGB = RGB(128, 128, 0)  
End With
```

Set the [ColorScheme](#) property of a **Slide**, **SlideRange**, or **Master** object to return the color scheme for one slide, a set of slides, or a master, respectively. The following example sets the color scheme for all the slides in the active presentation to the third color scheme in the presentation.

```
With ActivePresentation  
    .Slides.Range.ColorScheme = .ColorSchemes(3)  
End With
```



Columns Collection Object

[Table](#) └ [Columns](#)
└ [Column](#)
└ [CellRange](#)

A collection of [Column](#) objects that represent the columns in a table.

Using the Columns Collection

Use the **Columns** property to return the **Columns** collection. This example finds the first table in the active presentation, counts the number of **Column** objects in the **Columns** collection, and displays information to the user.

```
Dim ColCount, sl, sh As Integer

With ActivePresentation
    For sl = 1 To .Slides.Count
        For sh = 1 To .Slides(sl).Shapes.Count
            If .Slides(sl).Shapes(sh).HasTable Then
                ColCount = .Slides(sl).Shapes(sh) _
                    .Table.Columns.Count
                MsgBox "Shape " & sh & " on slide " & sl & _
                    " contains the first table and has " & _
                    ColCount & " columns."
            Exit Sub
        End If
    Next
Next
End With
```

Use the [Add](#) method to add a column to a table. This example creates a column in an existing table and sets the width of the new column to 72 points (one inch).

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Columns.Add.Width = 72
End With
```

Use **Columns(index)** to return a single **Column** object. *Index* represents the position of the column in the **Columns** collection (usually counting from left to right; although the [TableDirection](#) property can reverse this). This example selects the first column of the table in shape five on the second slide.

```
ActivePresentation.Slides(2).Shapes(5).Table.Columns(1).Select
```



Comments Collection

Multiple objects [↳ Comments](#)
[↳ Comment](#)

Represents a collection of [Comment](#) objects.

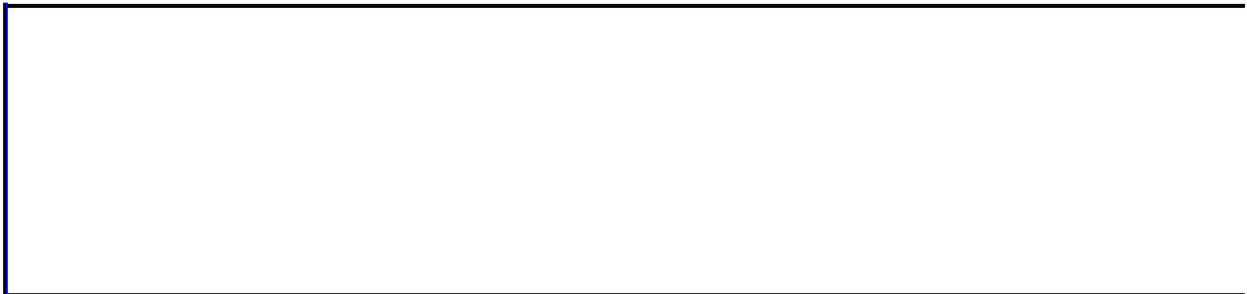
Using the Comments collection

Use the [Comments](#) property to refer to the **Comments** collection. The following example displays the number of comments on the current slide.

```
Sub CountComments()  
    MsgBox "You have " & ActiveWindow.Selection.SlideRange(1) _  
        .Comments.Count & " comments on this slide."  
End Sub
```

Use the [Add](#) method to add a comment to a slide. This example adds a new comment to the first slide of the active presentation.

```
Sub AddComment()  
    Dim sldNew As Slide  
    Dim cmtNew As Comment  
  
    Set sldNew = ActivePresentation.Slides.Add(Index:=1, _  
        Layout:=ppLayoutBlank)  
    Set cmtNew = sldNew.Comments.Add(Left:=12, Top:=12, _  
        Author:="Jeff Smith", AuthorInitials:"JS", _  
        Text:"You might consider reviewing the new specs " & _  
        "for more up-to-date information.")  
End Sub
```



Designs Collection

[Presentation](#) └ [Designs](#)
└ [Design](#)

Represents a collection of slide design templates.

Using the Designs collection

Use the [Designs](#) property of the [Presentation](#) object to reference a design template.

To add or clone an individual design template, use the **Designs** collection's [Add](#) or [Clone](#) methods, respectively. To refer to an individual design template, use the [Item](#) method.

To load a design template, use the [Load](#) method.

The following example adds a new design template to the **Designs** collection and confirms it was added correctly.

```
Sub AddDesignMaster()  
    With ActivePresentation.Designs  
        .Add designName:="MyDesignName"  
        MsgBox .Item("MyDesignName").Name  
    End With  
End Sub
```



DiagramNodes Collection

[Diagram](#) └ [DiagramNodes](#)
└ [DiagramNode](#)
└ Multiple objects

A collection of [DiagramNode](#) objects that represents all the nodes in a diagram.

Using the DiagramNodes collection

Use the [Nodes](#) property of the [Diagram](#) object to return a **DiagramNodes** collection. Use the [Item](#) method to select and work with a single diagram node in a diagram. This example assumes the first shape on the first slide in the active presentation is a diagram, selects the first node, and deletes it.

```
Sub FillDiagramNode()  
    ActivePresentation.Slides(1).Shapes(1).Diagram.Nodes.Item(1).Del  
End Sub
```

Use the [SelectAll](#) method to select and work with all nodes in a diagram. This example assumes the first shape on the first slide in the active presentation is a diagram, selects all nodes, and fills them with the specified pattern.

```
Sub FillDiagramNodes()  
    ActivePresentation.Slides(1).Shapes(1).Diagram.Nodes.SelectAll  
    ActiveWindow.Selection.ShapeRange.Fill.Patterned _  
        Pattern:=msoPatternSmallConfetti  
End Sub
```



DocumentWindows Collection Object

Multiple objects [DocumentWindows](#)
└─ [DocumentWindow](#)
 └─ Multiple objects

A collection of all the [DocumentWindow](#) objects that are currently open in PowerPoint. This collection doesn't include open slide show windows, which are included in the [SlideShowWindows](#) collection.

Using the DocumentWindows Collection

Use the [Windows](#) property to return the **DocumentWindows** collection. The following example tiles the open document windows.

```
Windows.Arrange ppArrangeTiled
```

Use the [NewWindow](#) method to create a document window and add it to the **DocumentWindows** collection. The following example creates a new window for the active presentation.

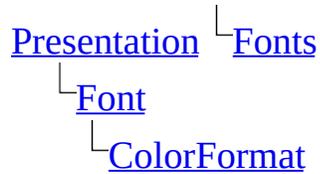
```
ActivePresentation.NewWindow
```

Use **Windows(index)**, where *index* is the window index number, to return a single **DocumentWindow** object. The following example closes document window two.

```
Windows(2).Close
```



Fonts Collection Object



A collection of all the **Font** objects in the specified presentation. Each **Font** object represents a font that's used in the presentation.

The **Fonts** collection is used by the Geni Wizard to determine whether any of the fonts in the specified presentation won't be supported when Genigraphics images the slides. If you just want to set character formatting for a particular bullet or text range, use the **Font** property to return the **Font** object for the bullet or text range.

The Genigraphics wizard enables users to transmit their presentations directly to Genigraphics for conversion into film slides, overhead transparencies, or other specialized media formats. For more information about the services Genigraphics provides, visit the Genigraphics Web site at <http://www.genigraphics.com/>. This service may not be available outside the United States.

Using the Fonts Object

Use the [Fonts](#) property to return the **Fonts** collection. The following example displays the number of fonts used in the active presentation.

```
MsgBox ActivePresentation.Fonts.Count
```

Use **Fonts(index)**, where *index* is the font's name or index number, to return a single **Font** object. The following example checks to see whether font one in the active presentation is embedded in the presentation.

```
If ActivePresentation.Fonts(1).Embedded = True Then  
    MsgBox "Font 1 is embedded"
```



HeadersFooters Object

Multiple objects [↳ HeadersFooters](#)
[↳ HeaderFooter](#)

Contains all the [HeaderFooter](#) objects on the specified slide, notes page, handout, or master. Each **HeaderFooter** object represents a header, footer, date and time, or slide number.

Note **HeaderFooter** objects aren't available for [Slide](#) objects that represent notes pages. The **HeaderFooter** object that represents a header is available only for a notes master or handout master.

Using the HeaderFooters Object

Use the [HeaderFooters](#) property to return the **HeaderFooters** object. Use the [DateAndTime](#) , [Footer](#) , [Header](#) , or [SlideNumber](#) property to return an individual **HeaderFooter** object. The following example sets the footer text for slide one in the active presentation.

```
ActivePresentation.Slides(1).HeaderFooters.Footer _  
    .Text = "Volcano Coffee"
```



NamedSlideShows Collection Object

[SlideShowSettings](#) └ [NamedSlideShows](#)
└ [NamedSlideShow](#)

A collection of all the [NamedSlideShow](#) objects in the presentation. Each **NamedSlideShow** object represents a custom slide show.

Using the NamedSlideShows Collection

Use the [NamedSlideShows](#) property to return the **NamedSlideShows** collection. Use **NamedSlideShows(index)**, where *index* is the custom slide show name or index number, to return a single **NamedSlideShow** object. The following example deletes the custom slide show named "Quick Show."

```
ActivePresentation.SlideShowSettings _  
    .NamedSlideShows("Quick Show").Delete
```

Use the [Add](#) method to create a new slide show and add it to the **NamedSlideShows** collection. The following example adds to the active presentation the named slide show "Quick Show" that contains slides 2, 7, and 9. The example then runs this custom slide show.

```
Dim qSlides(1 To 3) As Long  
With ActivePresentation  
    With .Slides  
        qSlides(1) = .Item(2).SlideID  
        qSlides(2) = .Item(7).SlideID  
        qSlides(3) = .Item(9).SlideID  
    End With  
    With .SlideShowSettings  
        .NamedSlideShows.Add "Quick Show", qSlides  
        .RangeType = ppShowNamedSlideShow  
        .SlideShowName = "Quick Show"  
        .Run  
    End With  
End With
```



Panes Collection Object

[DocumentWindow](#) └ [Panes](#)
└ [Pane](#)

A collection of **Pane** objects that represent the slide, outline, and notes panes in the document window for normal view, or the single pane of any other view in the document window.

Using the Panes Collection

Use the **Panes** property to return the **Panes** collection. The following example tests for the number of panes in the active window. If the value is one, indicating any view other than normal view, then normal view is activated and the vertical pane divider is set to divide the document window at 15% outline pane and 85% slide pane.

```
With ActiveWindow
  If .Panes.Count = 1 Then
    .ViewType = ppViewNormal
    .SplitHorizontal = 15
  End If
End With
```

Remarks

In normal view, the **Panes** collection contains three members. All other document window views have only a single pane, resulting in a **Panes** collection with one member.

[Show All](#)

Placeholders Collection Object

[Shapes](#) └ [Placeholders](#)
└ [Shape](#)
└ Multiple objects

A collection of all the **Shape** objects that represent [placeholders](#) on the specified slide. Each **Shape** object in the **Placeholders** collection represents a placeholder for text, a chart, a table, an organizational chart, or some other type of object. If the slide has a title, the title is the first placeholder in the collection.

Using the Placeholders Collection

Use the [Placeholders](#) property to return the **Placeholders** collection. Use **Placeholders(index)**, where *index* is the placeholder index number, to return a **Shape** object that represents a single placeholder. Note that for any slide that has a title, `Shapes.Title` is equivalent to `Shapes.Placeholders(1)`. The following example adds a new slide with a Bulleted List slide layout to the beginning of the presentation, sets the text for the title, and then adds two paragraphs to the text placeholder.

```
Set sObj = ActivePresentation.Slides.Add(1, ppLayoutText).Shapes
sObj.Title.TextFrame.TextRange.Text = "This is the title text"
sObj.Placeholders(2).TextFrame.TextRange.Text = _
    "Item 1" & Chr(13) & "Item 2"
```

You can delete individual placeholders by using the [Delete](#) method, and you can restore deleted placeholders by using the [AddPlaceholder](#) method, but you cannot add any more placeholders to a slide than it had when it was created. To change the number of placeholders on a given slide, set the [Layout](#) property.



Presentations Collection Object

[Application](#) └ [Presentations](#)
└ [Presentation](#)
└ Multiple objects

A collection of all the [Presentation](#) objects in PowerPoint. Each **Presentation** object represents a presentation that's currently open in PowerPoint.

Using the Presentations Collection

Use the [Presentations](#) property to return the **Presentations** collection. Use the [Add](#) method to create a new presentation and add it to the collection. The following example creates a new presentation, adds a slide to the presentation, and then saves the presentation.

```
Set newPres = Presentations.Add(True)
newPres.Slides.Add 1, 1
newPres.SaveAs "Sample"
```

Use **Presentations(index)**, where *index* is the presentation's name or index number, to return a single **Presentation** object. The following example prints presentation one.

```
Presentations(1).PrintOut
```

Use the [Open](#) method to open a presentation and add it to the **Presentations** collection. The following example opens the file Sales.ppt as a read-only presentation.

```
Presentations.Open FileName:="sales.ppt", ReadOnly:=True
```

Remarks

The **Presentations** collection doesn't include open add-ins, which are a special kind of hidden presentation. You can, however, return a single open add-in if you know its file name. For example `Presentations("oscar.ppa")` will return the open add-in named "Oscar.ppa" as a **Presentation** object. However, it is recommended that the **AddIns** collection be used to return open add-ins.



PrintRanges Collection Object

[PrintOptions](#) └ [PrintRanges](#)
└ [PrintRange](#)

A collection of all the [PrintRange](#) objects in the specified presentation. Each **PrintRange** object represents a range of consecutive slides or pages to be printed.

Using the PrintRanges Collection

Use the [Ranges](#) property to return the **PrintRanges** collection. The following example clears all previously defined print ranges from the collection for the active presentation.

```
ActivePresentation.PrintOptions.Ranges.ClearAll
```

Use the [Add](#) method to create a **PrintRange** object and add it to the **PrintRanges** collection. The following example defines three print ranges that represent slide 1, slides 3 through 5, and slides 8 and 9 in the active presentation and then prints the slides in these ranges.

```
With ActivePresentation.PrintOptions
    .RangeType = ppPrintSlideRange
    With .Ranges
        .ClearAll
        .Add 1, 1
        .Add 3, 5
        .Add 8, 9
    End With
End With
ActivePresentation.PrintOut
```

Use **Ranges(index)**, where *index* is the print range index number, to return a single **PrintRange** object. The following example displays a message that indicates the starting and ending slide numbers for print range one in the active presentation.

```
With ActivePresentation.PrintOptions.Ranges
    If .Count > 0 Then
        With .Item(1)
            MsgBox "Print range 1 starts on slide " & .Start & _
                " and ends on slide " & .End
        End With
    End If
End With
```



PublishObjects Collection Object

[Presentation](#) └ [PublishObjects](#)
└ [PublishObject](#)

A collection of [PublishObject](#) objects representing the set of complete or partial loaded presentations that are available for publishing to HTML.

Using the PublishObjects Collection

Use the **PublishObjects** property to return the **PublishObjects** collection. This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```

Use **Item(index)**, where *index* is always "1", to return the single **PublishObject** object for a loaded presentation. There can be only one **PublishObject** object for each loaded presentation.

This example defines the **PublishObject** object to be the entire active presentation by setting the **SourceType** property to **ppPublishAll**.

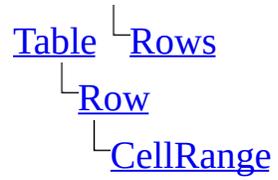
```
ActivePresentation.PublishObjects.Item(1).SourceType = ppPublishAll
```

Remarks

You can specify the content and attributes of the published presentation by setting various properties of the **PublishObject** object. For example, the [SourceType](#) property defines the portion of a loaded presentation to be published. The [RangeStart](#) property and the [RangeEnd](#) property specify the range of slides to publish, and the [SpeakerNotes](#) property designates whether or not to publish the speaker's notes.

You cannot add to the **PublishObjects** collection.

Rows Collection Object



A collection of [Row](#) objects that represent the rows in a table.

Using the Rows Collection

Use the [Rows](#) property to return the **Rows** collection. This example changes the height of all rows in the specified table to 160 points.

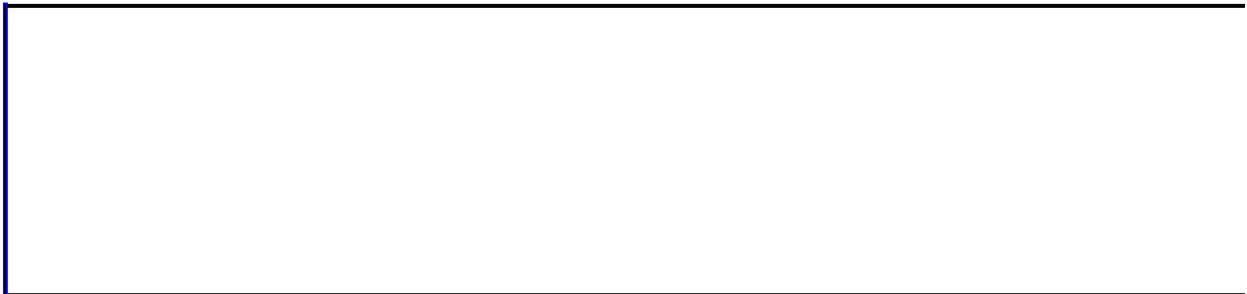
```
Dim i As Integer
With ActivePresentation.Slides(2).Shapes(4).Table
    For i = 1 To .Rows.Count
        .Rows.Height = 160
    Next i
End With
```

Use the [Add](#) method to add a row to a table. This example inserts a row before the second row in the referenced table.

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows.Add (2)
```

Use **Rows(index)**, where *index* is a number that represents the position of the row in the table, to return a single **Row** object. This example deletes the first row from the table in shape five on slide two.

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows(1).Delete
```



RulerLevels Collection Object

[Ruler](#) └ [RulerLevels](#)
└ [RulerLevel](#)

A collection of all the [RulerLevel](#) objects on the specified ruler. Each **RulerLevel** object represents the first-line and left indent for text at a particular outline level. This collection always contains five members— one for each of the available outline levels.

Using the RulerLevels Collection

Use the [Levels](#) property to return the **RulerLevels** collection. The following example sets the margins for the five outline levels in body text in the active presentation.

```
With ActivePresentation.SlideMaster.TextStyles(ppBodyStyle).Ruler
    .Levels(1).FirstMargin = 0
    .Levels(1).LeftMargin = 40
    .Levels(2).FirstMargin = 60
    .Levels(2).LeftMargin = 100
    .Levels(3).FirstMargin = 120
    .Levels(3).LeftMargin = 160
    .Levels(4).FirstMargin = 180
    .Levels(4).LeftMargin = 220
    .Levels(5).FirstMargin = 240
    .Levels(5).LeftMargin = 280
End With
```



Sequence Collection

[TimeLine](#) └ [Sequence](#)
└ [Effect](#)
└ Multiple objects

Represents a collection of [Effect](#) objects for a slide's interactive animation sequences. The **Sequence** collection is a member of the [Sequences](#) collection.

Using the Sequence collection

Use the [MainSequence](#) property of the [TimeLine](#) object to return a **Sequence** object.

Use the [AddEffect](#) method to add a new **Sequence** object. This example adds a shape and an animation sequence to the first shape on the first slide in the active presentation.

```
Sub NewEffect()  
    Dim effNew As Effect  
    Dim shpFirst As Shape  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence.  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlinds)  
End Sub
```



Sequences Collection

[TimeLine](#) └ [Sequences](#)
└ [Sequence](#)

Represents a collection of [Sequence](#) objects. Use a **Sequence** object to add, find, modify, and clone animation effects.

Using the Sequences collection

Use the [InteractiveSequences](#) property of the [TimeLine](#) object to return a **Sequences** collection. Use the [Add](#) method to add an interactive animation sequence. The following example adds two shapes on the first slide of the active presentation and sets interactive effect for the star shape so that when you click on the bevel shape, the star shape is be animated.

```
Sub AddNewSequence()  
    Dim shp1 As Shape  
    Dim shp2 As Shape  
    Dim interEffect As Effect  
  
    Set shp1 = ActivePresentation.Slides(1).Shapes.AddShape _  
        (Type:=msoShape32pointStar, Left:=100, _  
        Top:=100, Width:=200, Height:=200)  
    Set shp2 = ActivePresentation.Slides(1).Shapes.AddShape _  
        (Type:=msoShapeBevel, Left:=400, _  
        Top:=200, Width:=150, Height:=100)  
  
    With ActivePresentation.Slides(1).TimeLine.InteractiveSequences.  
        Set interEffect = .AddEffect(shp2, msoAnimEffectBlinds, _  
            trigger:=msoAnimTriggerOnShapeClick)  
        interEffect.Shape = shp1  
    End With  
End Sub
```



ShapeNodes Collection Object

Multiple objects [↳ ShapeNodes](#)
[↳ ShapeNode](#)

A collection of all the [ShapeNode](#) objects in the specified freeform. Each **ShapeNode** object represents either a node between segments in a freeform or a control point for a curved segment of a freeform. You can create a freeform manually or by using the [BuildFreeform](#) and [ConvertToShape](#) methods.

Using the ShapeNodes Collection

Use the **Nodes** property to return the **ShapeNodes** collection. The following example deletes node four in shape three on myDocument. For this example to work, shape three must be a freeform with at least four nodes.

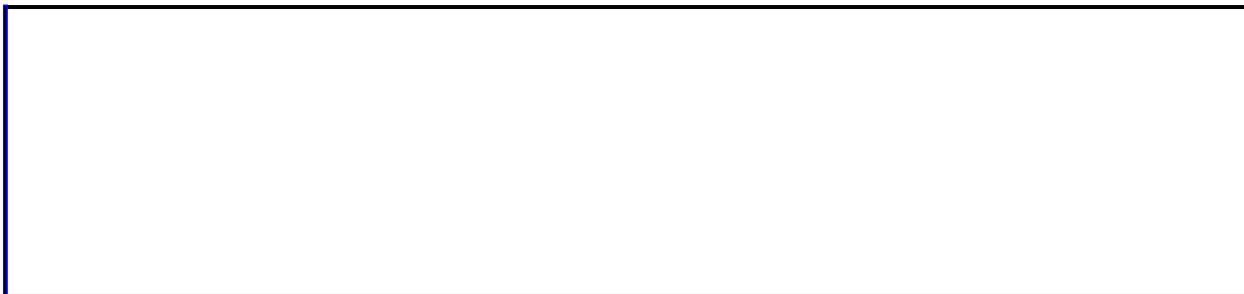
```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).Nodes.Delete 4
```

Use the [Insert](#) method to create a new node and add it to the **ShapeNodes** collection. The following example adds a smooth node with a curved segment after node four in shape three on myDocument. For this example to work, shape three must be a freeform with at least four nodes.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    .Insert 4, msoSegmentCurve, msoEditingSmooth, 210, 100
End With
```

Use **Nodes(index)**, where *index* is the node index number, to return a single **ShapeNode** object. If node one in shape three on myDocument is a corner point, the following example makes it a smooth point. For this example to work, shape three must be a freeform.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Nodes(1).EditingType = msoEditingCorner Then
        .Nodes.SetEditingType 1, msoEditingSmooth
    End If
End With
```



Shapes Collection Object

Multiple objects [└ Shapes](#)
└ Multiple objects

A collection of all the [Shape](#) objects on the specified slide. Each **Shape** object represents an object in the drawing layer, such as an AutoShape, freeform, OLE object, or picture.

Note If you want to work with a subset of the shapes on a document— for example, to do something to only the AutoShapes on the document or to only the selected shapes— you must construct a [ShapeRange](#) collection that contains the shapes you want to work with. For an overview of how to work either with a single shape or with more than one shape at a time, see [Working with Shapes \(Drawing Objects\)](#)).

Using the Shapes Collection

Use the **Shapes** property to return the **Shapes** collection. The following example selects all the shapes in the active presentation.

```
ActivePresentation.Slides(1).Shapes.SelectAll
```

Note If you want to do something (like delete or set a property) to all the shapes on a document at the same time, use the [Range](#) method with no argument to create a **ShapeRange** object that contains all the shapes in the **Shapes** collection, and then apply the appropriate property or method to the **ShapeRange** object.

Use the [AddCallout](#), [AddComment](#), [AddConnector](#), [AddCurve](#), [AddLabel](#), [AddLine](#), [AddMediaObject](#), [AddOLEObject](#), [AddPicture](#), [AddPlaceholder](#), [AddPolyline](#), [AddShape](#), [AddTable](#), [AddTextbox](#), [AddTextEffect](#), or [AddTitle](#) method to create a new shape and add it to the **Shapes** collection. Use the [BuildFreeform](#) method in conjunction with the [ConvertToShape](#) method to create a new freeform and add it to the collection. The following example adds a rectangle to the active presentation.

```
ActivePresentation.Slides(1).Shapes.AddShape Type:=msoShapeRectangle  
Left:=50, Top:=50, Width:=100, Height:=200
```

Use **Shapes(index)**, where *index* is the shape's name or index number, to return a single **Shape** object. The following example sets the fill to a preset shade for shape one in the active presentation.

```
ActivePresentation.Slides(1).Shapes(1).Fill _  
.PresetGradient Style:=msoGradientHorizontal, Variant:=1, _  
PresetGradientType:=msoGradientBrass
```

Use **Shapes.Range(index)**, where *index* is the shape's name or index number or an array of shape names or index numbers, to return a [ShapeRange](#) collection that represents a subset of the **Shapes** collection. The following example sets the fill pattern for shapes one and three in the active presentation.

```
ActivePresentation.Slides(1).Shapes.Range(Array(1, 3)).Fill _  
    .Patterned Pattern:=msoPatternHorizontalBrick
```

Use **Shapes.Placeholders(index)**, where *index* is the placeholder number, to return a **Shape** object that represents a placeholder. If the specified slide has a title, use **Shapes.Placeholders(1)** or **Shapes.Title** to return the title placeholder. The following example adds a slide to the active presentation and then adds text to both the title and the subtitle (the subtitle is the second placeholder on a slide with this layout).

```
With ActivePresentation.Slides.Add(Index:=1, Layout:=ppLayoutTitle).  
    .Title.TextFrame.TextRange = "This is the title text"  
    .Placeholders(2).TextFrame.TextRange = "This is subtitle text"  
End With
```



SlideRange Collection Object

Multiple objects [SlideRange](#)
└─ Multiple objects

A collection that represents a notes page or a slide range, which is a set of slides that can contain as little as a single slide or as much as all the slides in a presentation. You can include whichever slides you want— chosen from all the slides in the presentation or from all the slides in the selection— to construct a slide range. For example, you could construct a **SlideRange** collection that contains the first three slides in a presentation, all the selected slides in the presentation, or all the title slides in the presentation.

Using the SlideRange Collection

This section describes how to:

- Return a set of slides that you specify by name or index number
- Return all or some of the selected slides in a presentation
- Return a notes page
- Apply properties and methods to a slide range

Returning a set of slides that you specify by name or index number

Use **Slides.Range**(*index*), where *index* is the name or index number of the slide or an array that contains either names or index numbers of slides, to return a **SlideRange** collection that represents a set of slides in a presentation. You can use the **Array** function to construct an array of names or index numbers. The following example sets the background fill for slides one and three in the active presentation.

```
With ActivePresentation.Slides.Range(Array(1, 3))
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient msoGradientHorizontal, _
        1, msoGradientLateSunset
End With
```

The following example sets the background fill for the slides named "Intro" and "Big Chart" in the active presentation. Note that slides are assigned automatically generated names of the form *Slide**n* (where *n* is an integer) when they're created. To assign a more meaningful name to a slide, use the [Name](#) property.

```
With ActivePresentation.Slides.Range(Array("Intro", "Big Chart"))
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient msoGradientHorizontal, _
        1, msoGradientLateSunset
End With
```

Although you can use the [Range](#) method to return any number of slides, it's simpler to use the [Item](#) method if you only want to return a single member of the **SlideRange** collection. For example, `Slides(1)` is simpler than `Slides.Range(1)`.

Returning all or some of the selected slides in a presentation

Use the [SlideRange](#) property of the [Selection](#) object to return all the slides in the selection. The following example sets the background fill for all the selected slides in window one, assuming that there's at least one slide selected.

```
With Windows(1).Selection.SlideRange
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient msoGradientHorizontal, _
        1, msoGradientLateSunset
End With
```

Use **Selection.SlideRange(*index*)**, where *index* is the slide name or index number, to return a single slide from the selection. The following example sets the background fill for slide two in the collection of selected slides in window one, assuming that there are at least two slides selected.

```
With Windows(1).Selection.SlideRange(2)
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient msoGradientHorizontal, _
        1, msoGradientLateSunset
End With
```

Returning a notes page

Use the **NotesPage** property to return a **SlideRange** collection that represents the specified notes page. The following example inserts text into placeholder two (the notes area) on the notes page for slide one in the active presentation.

```
ActivePresentation.Slides(1).NotesPage.Shapes _  
    .Placeholders(2).TextFrame.TextRange.InsertAfter "Added Text"
```

Applying a property or method to a slide range

Just as you can work with several slides at the same time in the user interface by selecting them and applying a command, you can work with several slides at the same time programmatically by constructing a **SlideRange** collection and applying properties or methods to it. And just as some commands in the user interface that work on single slides aren't valid when multiple slides are selected, some properties and methods that work on a **Slide** object or on a **SlideRange** collection that contains only one slide will fail if they're applied to a **SlideRange** collection that contains more than one slide. In general, if you can't do something manually when more than one slide is selected (such as return the individual shapes on one of the slides), you can't do it programmatically by using a **SlideRange** collection that contains more than one slide.

For those operations that work in the user interface whether you have a single slide or multiple slides selected (such as copying the selection to the Clipboard or setting the slide background fill), the associated properties and methods will work on a **SlideRange** collection that contains more than one slide. Here are some general guidelines for how these properties and methods behave when they're applied to multiple slides.

- Applying a method to a **SlideRange** collection is equivalent to applying the method to all the **Slide** objects in that range as a group.
- Setting the value of a property of the **SlideRange** collection is equivalent to setting the value of the property in each slide in that range individually (for a property that takes an enumerated type, setting the value to the "Mixed" value has no effect).
- A property of the **SlideRange** collection that returns an enumerated type returns the value of the property for an individual slide in the collection if all slides in the collection have the same value for that property. If the slides in the collection don't all have the same value for the property, the property returns the "Mixed" value.
- A property of the **SlideRange** collection that returns a simple data type (such as **Long**, **Single**, or **String**) returns the value of the property for an individual slide in the collection if all slides in the collection have the same value for that property. If the slides in the collection don't all have the same value for the property, the property will return - 2 or generate an error. For

example, using the **Name** property on a **SlideRange** object that contains multiple slides will generate an error because each slide has a different value for its **Name** property.

- Some formatting properties of slides aren't set by properties and methods that apply directly to the **SlideRange** collection, but by properties and methods that apply to an object contained in the **SlideRange** collection, such as the **ColorScheme** object. If the contained object represents operations that can be performed on multiple objects in the user interface, you'll be able to return the object from a **SlideRange** collection that contains more than one slide, and its properties and methods will follow the preceding rules. For example, you can use the **ColorScheme** property to return the **ColorScheme** object that represents the color schemes used on all the slides in the specified **SlideRange** collection. Setting properties for this **ColorScheme** object will also set these properties for the **ColorScheme** objects on all the individual slides in the **SlideRange** collection.



Slides Collection Object

[Presentation](#) └ [Slides](#)
└ [Slide](#)
└ Multiple objects

A collection of all the [Slide](#) objects in the specified presentation.

Using the Slides Collection

This section describes how to:

- Create a slide and add it to the collection
- Return a single slide that you specify by name, index number, or slide ID number
- Return a subset of the slides in the presentation
- Apply a property or method to all the slides in the presentation at the same time

Creating a slide and adding it to the collection

Use the [Slides](#) property to return a **Slides** collection. Use the [Add](#) method to create a new slide and add it to the collection. The following example adds a new slide to the active presentation.

```
ActivePresentation.Slides.Add 2, ppLayoutBlank
```

Returning a single slide that you specify by name, index number, or slide ID number

Use `Slides(index)`, where *index* is the slide name or index number, or use the `Slides.FindBySlideID(index)`, where *index* is the slide ID number, to return a single **Slide** object. The following example sets the layout for slide one in the active presentation.

```
ActivePresentation.Slides(1).Layout = ppLayoutTitle
```

The following example sets the layout for the slide named "Big Chart" in the active presentation. Note that slides are assigned automatically generated names of the form *Slide n* (where *n* is an integer) when they're created. To assign a more meaningful name to a slide, use the [Name](#) property.

```
ActivePresentation.Slides("Big Chart").Layout = ppLayoutTitle
```

Returning a subset of the slides in the presentation

Use **Slides.Range**(*index*), where *index* is the slide index number or name or an array of slide index numbers or an array of slide names, to return a [SlideRange](#) object that represents a subset of the **Slides** collection. The following example sets the background fill for slides one and three in the active presentation.

```
With ActivePresentation.Slides.Range(Array(1, 3))  
    .FollowMasterBackground = False  
    .Background.Fill.PresetGradient msoGradientHorizontal, _  
        1, msoGradientLateSunset  
End With
```

Applying a property or method to all the slides in the presentation at the same time

If you want to do something to all the slides in your presentation at the same time (such as delete all of them or set a property for all of them), use **Slides.Range** with no argument to construct a **SlideRange** collection that contains all the slides in the **Slides** collection, and then apply the appropriate property or method to the **SlideRange** collection. The following example sets the background fill for all the slides in the active presentation

```
With ActivePresentation.Slides.Range
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient msoGradientHorizontal, _
        1, msoGradientLateSunset
End With
```



SlideShowWindows Collection Object

[Application](#) └ [SlideShowWindows](#)
└ [SlideShowWindow](#)
└ Multiple objects

A collection of all the [SlideShowWindow](#) objects that represent the open slide shows in PowerPoint.

Using the SlideShowWindows Collection

Use the [SlideShowWindows](#) property to return the **SlideShowWindows** collection. Use **SlideShowWindows(index)**, where *index* is the window index number, to return a single **SlideShowWindow** object. The following example reduces the height of slide show window one if it's a full-screen window.

```
With SlideShowWindows(1)
    If .IsFullScreen Then
        .Height = .Height - 20
    End If
End With
```

Use the [Run](#) method to create a new slide show window and add it to the **SlideShowWindows** collection. The following example runs a slide show of the active presentation.

```
ActivePresentation.SlideShowSettings.Run
```



TabStops Collection Object

[Ruler](#) └ [TabStops](#)
└ [TabStop](#)

A collection of all the [TabStop](#) objects on one ruler.

Using the TabStops Collection

Use the [TabStops](#) property to return the **TabStops** collection. The following example clears all the tab stops for the text in shape two on slide one in the active presentation.

```
With ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.Ruler.TabStops  
    For t = .Count To 1 Step -1  
        .Item(t).Clear  
    Next  
End With
```

Use the [Add](#) method to create a tab stop and add it to the **TabStops** collection. The following example adds a tab stop to the body-text style on the slide master for the active presentation. The new tab stop will be positioned 2 inches (144 points) from the left edge of the ruler and will be left aligned.

```
ActivePresentation.SlideMaster _  
    .TextStyles(ppBodyStyle).Ruler.TabStops.Add ppTabStopLeft, 144
```



TextStyleLevels Collection Object

[TextStyle](#) └ [TextStyleLevels](#)
└ [TextStyleLevel](#)
└ Multiple objects

A collection of all the outline text levels. This collection always contains five members, each of which is represented by a [TextStyleLevel](#) object.

Using the TextStyleLevels Collection

Use **Levels(index)**, where *index* is a number from 1 through 5 that corresponds to the outline level, to return a single **TextStyleLevel** object. The following example sets the font name and font size for level-one body text on all the slides in the active presentation.

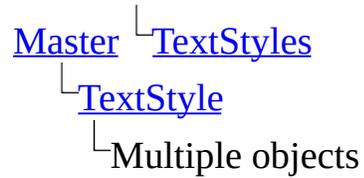
```
With ActivePresentation.SlideMaster _  
    .TextStyles(ppBodyStyle).Levels(1)  
    With .Font  
        .Name = "Arial"  
        .Size = 36  
    End With  
End With
```

The following example sets the font size for text at each outline level for the notes body area on all the notes pages in the active presentation.

```
With ActivePresentation.NotesMaster.TextStyles(ppBodyStyle).Levels  
    .Item(1).Font.Size = 34  
    .Item(2).Font.Size = 30  
    .Item(3).Font.Size = 25  
    .Item(4).Font.Size = 20  
    .Item(5).Font.Size = 15  
End With
```



TextStyles Collection Object

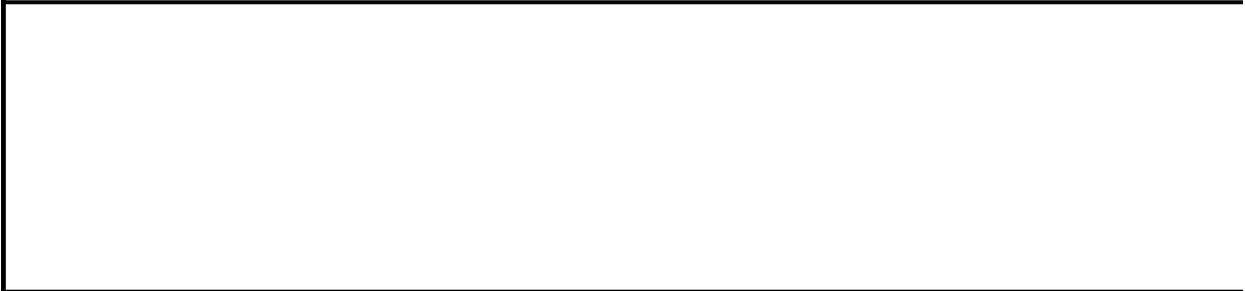


A collection of three text styles — title text, body text, and default text — each of which is represented by a [TextStyle](#) object. Each text style contains a [TextFrame](#) object that describes how text is placed within the text bounding box, a [Ruler](#) object that contains tab stops and outline indent formatting information, and a [TextStyleLevels](#) collection that contains outline text formatting information.

Using the TextStyles Collection

Use **TextStyles**(*index*), where *index* is either **ppBodyStyle**, **ppDefaultStyle**, or **ppTitleStyle**, to return a single **TextStyle** object. This example sets the margins for the notes body area on all the notes pages in the active presentation.

```
With ActivePresentation.NotesMaster _  
    .TextStyles(ppBodyStyle).TextFrame  
    .MarginBottom = 50  
    .MarginLeft = 50  
    .MarginRight = 50  
    .MarginTop = 50  
End With
```



ActionSetting Object

[ActionSettings](#) └ [ActionSetting](#)
└ Multiple objects

Contains information about how the specified shape or text range reacts to mouse actions during a slide show. The **ActionSetting** object is a member of the [ActionSettings](#) collection. The **ActionSettings** collection contains one **ActionSetting** object that represents how the specified object reacts when the user clicks it during a slide show and one **ActionSetting** object that represents how the specified object reacts when the user moves the mouse pointer over it during a slide show.

Using the ActionSetting Object

Use **ActionSettings**(*index*), where *index* is the either **ppMouseClick** or **ppMouseOver**, to return a single **ActionSetting** object. The following example sets the mouse-click action for the text in the third shape on slide one in the active presentation to an Internet link.

```
With ActivePresentation.Slides(1).Shapes(3) _  
    .TextFrame.TextRange.ActionSettings(ppMouseClick)  
    .Action = ppActionHyperlink  
    .Hyperlink.Address = "http://www.microsoft.com"  
End With
```

Remarks

If you've set properties of the **ActionSetting** object that don't seem to be taking effect, make sure that you've set the [Action](#) property to the appropriate value.

[Show All](#)

AddIn Object

[AddIns](#) \perp [AddIn](#)

Represents a single add-in, either loaded or not loaded. The **AddIn** object is a member of the [AddIns](#) collection. The **AddIns** collection contains all of the PowerPoint-specific add-ins available, regardless of whether or not they're loaded. The collection does not include [Component Object Model \(COM\) add-ins](#).

Using the AddIn Object

Use **AddIns**(*index*), where *index* is the add-in's title or index number, to return a single **AddIn** object. The following example loads the My Ppt Tools add-in.

```
AddIns("my ppt tools").Loaded = True
```

The add-in title, shown above, should not be confused with the add-in name, which is the file name of the add-in. You must spell the add-in title exactly as it's spelled in the **Add-Ins** dialog box, but the capitalization doesn't have to match.

The index number represents the position of the add-in in the **Available Add-Ins** list in the **Add-Ins** dialog box. The following example displays the names of all the add-ins that are currently loaded in PowerPoint.

```
For i = 1 To AddIns.Count  
    If AddIns(i).Loaded Then MsgBox AddIns(i).Name  
Next
```

Remarks

Use the [Add](#) method to add a PowerPoint-specific add-in to the list of those available. Note, however, that using this method doesn't load the add-in. To load the add-in, set the [Loaded](#) property of the add-in to **True** after you use the **Add** method. You can perform both of these actions in a single step, as shown in the following example (note that you use the name of the add-in, not its title, with the **Add** method).

```
AddIns.Add("generic.ppa").Loaded = True
```

Use **AddIns(index)**, where *index* is the add-in's title, to return a reference to the loaded add-in. The following example sets the `presAddin` variable to the add-in titled "my ppt tools" and sets the `myName` variable to the name of the add-in.

```
Set presAddin = AddIns("my ppt tools")  
With presAddin  
    myName = .Name  
End With
```



AnimationBehavior Object

[AnimationBehaviors](#) \perp [AnimationBehavior](#)

\perp Multiple objects

Represents the behavior of an animation effect, the main animation sequence, or an interactive animation sequence. The **AnimationBehavior** object is a member of the [AnimationBehaviors](#) collection.

Using the AnimationBehavior object

Use [Behaviors](#) (*index*), where *index* is the number of the behavior in the sequence of behaviors, to return a single **AnimationBehavior** object. The following example sets the positions of the a rotation's starting and ending points. This example assumes that the first behavior for the main animation sequence is a [RotationEffect](#) object.

```
Sub Change()  
    With ActivePresentation.Slides(1).TimeLine.MainSequence(1) _  
        .Behaviors(1).RotationEffect  
        .From = 1  
        .To = 180  
    End With  
End Sub
```



AnimationPoint Object

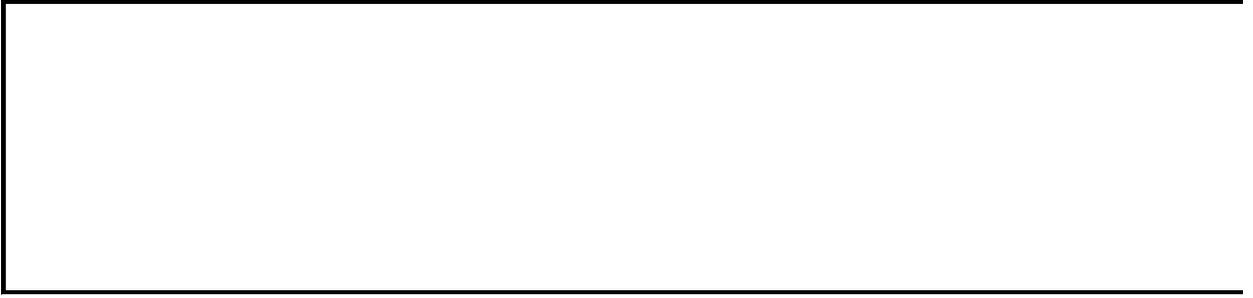
[AnimationPoints](#) ⊥ [AnimationPoint](#)

Represents an individual animation point for an animation behavior. The **AnimationPoint** object is a member of the [AnimationPoints](#) collection. The **AnimationPoints** collection contains all the animation points for an animation behavior.

Using the AnimationPoint object

To add or reference an **AnimationPoint** object, use the [Add](#) or [Item](#) method, respectively. Use the [Time](#) property of an **AnimationPoint** object to set timing between animation points. Use the [Value](#) property to set other animation point properties, such as color. The following example adds three animation points to the first behavior in the active presentation's main animation sequence, and then it changes colors at each animation point.

```
Sub AniPoint()  
  
    Dim sldNewSlide As Slide  
    Dim shpHeart As Shape  
    Dim effCustom As Effect  
    Dim aniBehavior As AnimationBehavior  
    Dim aptNewPoint As AnimationPoint  
  
    Set sldNewSlide = ActivePresentation.Slides.Add _  
        (Index:=1, Layout:=ppLayoutBlank)  
    Set shpHeart = sldNewSlide.Shapes.AddShape _  
        (Type:=msoShapeHeart, Left:=100, Top:=100, _  
        Width:=200, Height:=200)  
    Set effCustom = sldNewSlide.TimeLine.MainSequence _  
        .AddEffect(shpHeart, msoAnimEffectCustom)  
    Set aniBehavior = effCustom.Behaviors.Add(msoAnimTypeProperty)  
  
    With aniBehavior.PropertyEffect  
        .Property = msoAnimShapeFillColor  
        Set aptNewPoint = .Points.Add  
        aptNewPoint.Time = 0.2  
        aptNewPoint.Value = RGB(0, 0, 0)  
        Set aptNewPoint = .Points.Add  
        aptNewPoint.Time = 0.5  
        aptNewPoint.Value = RGB(0, 255, 0)  
        Set aptNewPoint = .Points.Add  
        aptNewPoint.Time = 1  
        aptNewPoint.Value = RGB(0, 255, 255)  
    End With  
  
End Sub
```



AnimationSettings Object

Multiple objects [AnimationSettings](#)
└─ Multiple objects

Represents the special effects applied to the animation for the specified shape during a slide show.

Using the AnimationSettings Object

Use the [AnimationSettings](#) property of the **Shape** object to return the **AnimationSettings** object. The following example adds a slide that contains both a title and a three-item list to the active presentation, and then it sets the list to be animated by first-level paragraphs, to fly in from the left when animated, to dim to the specified color after being animated, and to animate its items in reverse order.

```
Set sObjs = ActivePresentation.Slides.Add(2, ppLayoutText).Shapes
sObjs.Title.TextFrame.TextRange.Text = "Top Three Reasons"
With sObjs.Placeholders(2)
    .TextFrame.TextRange.Text = _
        "Reason 1" & VBNewLine & "Reason 2" & VBNewLine & "Reason 3"
    With .AnimationSettings
        .TextLevelEffect = ppAnimateByFirstLevel
        .EntryEffect = ppEffectFlyFromLeft
        .AfterEffect = ppAfterEffectDim
        .DimColor.RGB = RGB(100, 120, 100)
        .AnimateTextInReverse = True
    End With
End With
```



Application Object

[Application](#) └ Multiple objects

Represents the entire Microsoft PowerPoint application. The **Application** object contains:

- Application-wide settings and options (the name of the active printer, for example).
- Properties that return top-level objects, such as **ActivePresentation**, **Windows**, and so on.

Using the Application Object

Use the **Application** property to return the **Application** object. The following example returns the path to the application file.

```
Dim MyPath As String  
MyPath = Application.Path
```

The following example creates a PowerPoint **Application** object in another application, starts PowerPoint (if it's not already running), and opens an existing presentation named "Ex_a2a.ppt."

```
Set ppt = New Powerpoint.Application  
ppt.Visible = True  
ppt.Presentations.Open "c:\My Documents\ex_a2a.ppt"
```

Remarks

When you are writing code that will run from PowerPoint, the following properties of the **Application** object can be used without the object qualifier: **ActivePresentation, ActiveWindow, AddIns, Assistant, CommandBars, Presentations, SlideShowWindows, Windows**. For example, instead of writing `Application.ActiveWindow.Height = 200`, you can write `ActiveWindow.Height = 200`.

AutoCorrect Object

[Application](#) └ [AutoCorrect](#)

Represents the AutoCorrect functionality in Microsoft PowerPoint.

Using the AutoCorrect object

Use the [AutoCorrect](#) property to return an **AutoCorrect** object. The following example disables displaying the AutoCorrect options buttons.

```
Sub HideAutoCorrectOpButton()  
    With Application.AutoCorrect  
        .DisplayAutoCorrectOptions = msoFalse  
        .DisplayAutoLayoutOptions = msoFalse  
    End With  
End Sub
```



BulletFormat Object

[ParagraphFormat](#) └ [BulletFormat](#)
└ [Font](#)

Represents bullet formatting.

Using the **BulletFormat** Object

Use the [Bullet](#) property to return the **BulletFormat** object. The following example sets the bullet size and color for the paragraphs in shape two on slide one in the active presentation.

```
With ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat.Bullet
    .Visible = True
    .RelativeSize = 1.25
    .Character = 169
  With .Font
    .Color.RGB = RGB(255, 255, 0)
    .Name = "Symbol"
  End With
End With
End With
```



CalloutFormat Object

Multiple objects [↳ CalloutFormat](#)

Contains properties and methods that apply to line callouts.

Using the CalloutFormat Object

Use the **Callout** property to return a **CalloutFormat** object. The following example specifies the following attributes of shape three (a line callout) on myDocument:

- The callout will have a vertical accent bar that separates the text from the callout line.
- The angle between the callout line and the side of the callout text box will be 30 degrees.
- There will be no border around the callout text.
- The callout line will be attached to the top of the callout text box.
- The callout line will contain two segments.

For this example to work, shape three must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Callout
    .Accent = True
    .Angle = msoCalloutAngle30
    .Border = False
    .PresetDrop msoCalloutDropTop
    .Type = msoCalloutThree
End With
```



Cell Object

[CellRange](#) └ [Cell](#)
└ Multiple objects

Represents a table cell. The **Cell** object is a member of the [CellRange](#) collection. The **CellRange** collection represents all the cells in the specified column or row. To use the **CellRange** collection, use the **Cells** keyword.

Using the Cell Object

Use **Cell**(*row*, *column*), where *row* is the row number and *column* is the column number, or **Cells**(*index*), where *index* is the number of the cell in the specified row or column, to return a single **Cell** object. Cells are numbered from left to right in rows and from top to bottom in columns. With right-to-left language settings, this scheme is reversed. The following example merges the first two cells in row one of the table in shape five on slide two.

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Cell(1, 1).Merge MergeTo:=.Cell(1, 2)
End With
```

This example sets the bottom border for cell one in the first column of the table to a dashed line style.

```
With ActivePresentation.Slides(2).Shapes(5).Table.Columns(1) _
    .Cells(1)
    .Borders(ppBorderBottom).DashStyle = msoLineDash
End With
```

Use the [Shape](#) property to access the **Shape** object and to manipulate the contents of each cell. This example deletes the text in the first cell (row 1, column 1), inserts new text, and then sets the width of the entire column to 110 points.

```
With ActivePresentation.Slides(2).Shapes(5).Table.Cell(1, 1)
    .Shape.TextFrame.TextRange.Delete
    .Shape.TextFrame.TextRange.Text = "Rooster"
    .Parent.Columns(1).Width = 110
End With
```

Remarks

You cannot programmatically add cells to or delete cells from a PowerPoint table. Use the **Add** method of the **Columns** or **Rows** collections to add a column or row to a table. Use the **Delete** method of the **Columns** or **Rows** collections to delete a column or row from a table.

--

ColorEffect Object

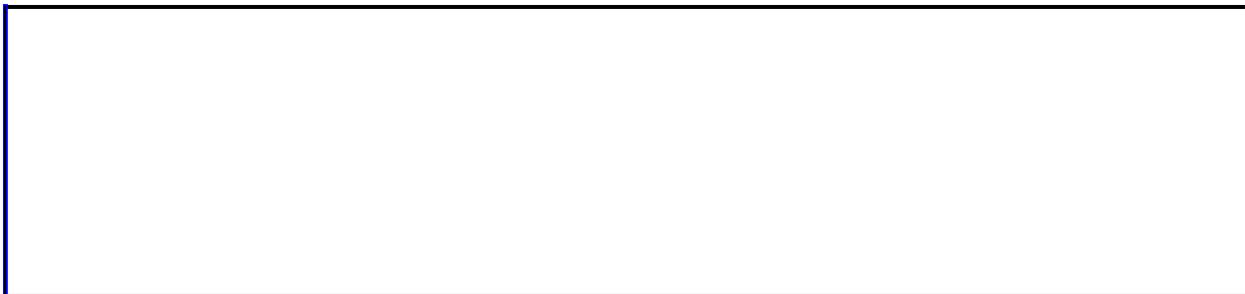
[AnimationBehavior](#) └ [ColorEffect](#)
└ [ColorFormat](#)

Represents a color effect for an animation behavior.

Using the ColorEffect object

Use the [ColorEffect](#) property of the [AnimationBehavior](#) object to return a **ColorEffect** object. Color effects can be changed using the **ColorEffect** object's [From](#) and [To](#) properties, as shown below. Color effects are initially set using the [To](#) property, and then can be changed by a specific number using the [By](#) property. The following example adds a shape to the first slide of the active presentation and sets a color effect animation behavior to change the fill color of the new shape.

```
Sub ChangeColorEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(Type:=msoAnimTypeColor)  
  
    With bhvEffect.ColorEffect  
        .From.RGB = RGB(Red:=255, Green:=0, Blue:=0)  
        .To.RGB = RGB(Red:=0, Green:=0, Blue:=255)  
    End With  
End Sub
```



ColorFormat Object

Multiple objects [↳ ColorFormat](#)

Represents the color of a one-color object, the foreground or background color of an object with a gradient or patterned fill, or the pointer color. You can set colors to an explicit red-green-blue value (by using the [RGB](#) property) or to a color in the color scheme (by using the [SchemeColor](#) property).

Using the ColorFormat Object

Use one of the properties listed in the following table to return a **ColorFormat** object.

Use this property	With this object	To return a ColorFormat object that represents this
<u>DimColor</u>	AnimationSettings	Color used for dimmed objects
<u>BackColor</u>	FillFormat	Background fill color (used in a shaded or patterned fill)
<u>ForeColor</u>	FillFormat	Foreground fill color (or simply the fill color for a solid fill)
<u>Color</u>	Font	Bullet or character color
<u>BackColor</u>	LineFormat	Background line color (used in a patterned line)
<u>ForeColor</u>	LineFormat	Foreground line color (or just the line color for a solid line)
<u>ForeColor</u>	ShadowFormat	Shadow color
<u>PointerColor</u>	SlideShowSettings	Default pointer color for a presentation
<u>PointerColor</u>	SlideShowView	Temporary pointer color for a view of a slide show
<u>ExtrusionColor</u>	ThreeDFormat	Color of the sides of an extruded object

Use the [SchemeColor](#) property to set the color of a slide element to one of the colors in the standard color scheme. The following example sets the text color for shape one on slide two in the active presentation to the standard color-scheme title color.

```
ActivePresentation.Slides(2).Shapes(1).TextFrame _  
    .TextRange.Font.Color.SchemeColor = ppTitle
```

Use the [RGB](#) property to set a color to an explicit red-green-blue value. The following example adds a rectangle to myDocument and then sets the foreground color, background color, and gradient for the rectangle's fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, 90, 90, 90, 50).F
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(170, 170, 170)
    .TwoColorGradient msoGradientHorizontal, 1
End With
```



ColorScheme Object

Multiple objects [ColorScheme](#)

Represents a color scheme, which is a set of eight colors used for the different elements of a slide, notes page, or handout, such as the title or background. (Note that the color schemes for slides, notes pages, and handouts in a presentation can be set independently.) Each color is represented by an [RGBColor](#) object. The **ColorScheme** object is a member of the [ColorSchemes](#) collection. The **ColorSchemes** collection contains all the color schemes in a presentation.

Using the ColorScheme Object

This section describes how to do the following:

- Return a **ColorScheme** object from the collection of all the color schemes in the presentation
- Return the **ColorScheme** object attached to a specific slide or master
- Return the color of a single slide element from a **ColorScheme** object

Returning a **ColorScheme** object from the collection of all the color schemes in the presentation

Use **ColorSchemes**(*index*), where *index* is the color scheme index number, to return a single **ColorScheme** object. The following example deletes color scheme two from the active presentation.

```
ActivePresentation.ColorSchemes(2).Delete
```

Returning the ColorScheme object attached to a specific slide or master

Set the [ColorScheme](#) property of a **Slide**, **SlideRange**, or **Master** object to return the color scheme for one slide, a set of slides, or a master, respectively. The following example creates a color scheme based on the current slide, adds the new color scheme to the collection of standard color schemes for the presentation, and sets the color scheme for the slide master to the new color scheme. All new slides based on the master will have this color scheme.

```
Set newScheme = ActiveWindow.View.Slide.ColorScheme
newScheme.Colors(ppTitle).RGB = RGB(0, 150, 250)
Set newStandardScheme = _
    ActivePresentation.ColorSchemes.Add(newScheme)
ActivePresentation.SlideMaster.ColorScheme = newStandardScheme
```

Returning the color of a single slide element from a ColorScheme object

Use the [Colors](#) method to return an **RGBColor** object that represents the color of a single slide-element type. You can set an **RGBColor** object to another **RGBColor** object, or you can use the [RGB](#) property to set or return the explicit red-green-blue (RGB) value for an **RGBColor** object. The following example sets the background color in color scheme one to red and sets the title color to the title color that's defined for color scheme two.

```
With ActivePresentation.ColorSchemes
    .Item(1).Colors(ppBackground).RGB = RGB(255, 0, 0)
    .Item(1).Colors(ppTitle) = .Item(2).Colors(ppTitle)
End With
```



Column Object

[Columns](#) | [Column](#)
| [CellRange](#)

Represents a table column. The **Column** object is a member of the [Columns](#) collection. The **Columns** collection includes all the columns in a table.

Using the Column Object

Use **Columns**(*index*) to return a single **Column** object. *Index* represents the position of the column in the **Columns** collection (usually counting from left to right; although the [TableDirection](#) property can reverse this). This example selects the first column of the table in shape five on the second slide.

```
ActivePresentation.Slides(2).Shapes(5).Table.Columns(1).Select
```

Use the **Cell** object to indirectly reference the **Column** object. This example deletes the text in the first cell (row 1, column 1), inserts new text, and then sets the width of the entire column to 110 points.

```
With ActivePresentation.Slides(2).Shapes(5).Table.Cell(1, 1)
    .Shape.TextFrame.TextRange.Delete
    .Shape.TextFrame.TextRange.Text = "Rooster"
    .Parent.Columns(1).Width = 110
End With
```

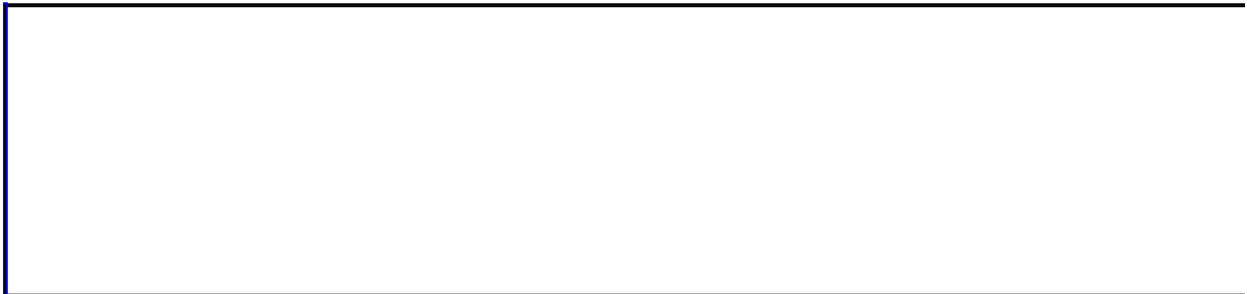
Use the [Add](#) method to add a column to a table. This example creates a column in an existing table and sets the column width to 72 points (one inch).

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Columns.Add.Width = 72
End With
```

Remarks

Use the [Cells](#) property to modify the individual cells in a **Column** object. This example selects the first column in the table and applies a dashed line style to the bottom border.

```
ActiveWindow.Selection.ShapeRange.Table.Columns(1) _  
    .Cells.Borders(ppBorderBottom).DashStyle = msoLineDash
```



CommandEffect Object

[AnimationBehavior](#) \perp [CommandEffect](#)

Represents a command effect for an animation behavior. You can send events, call functions, and send OLE verbs to embedded objects using this object.

Using the CommandEffect Object

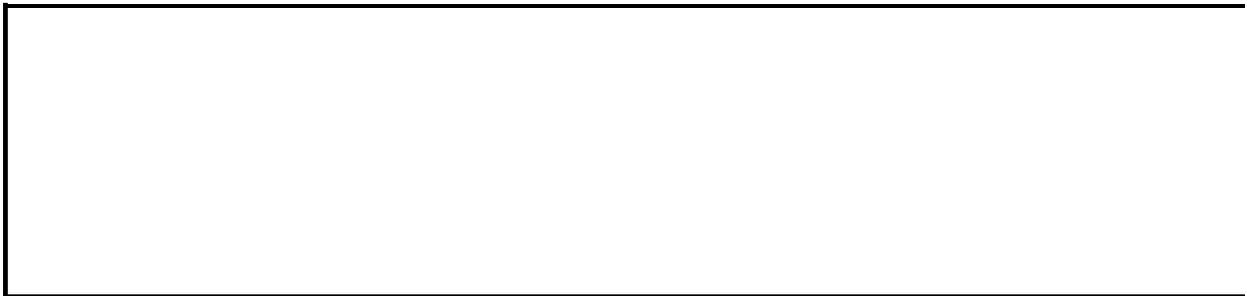
Use the **CommandEffect** property of the [AnimationBehavior](#) object to return a **CommandEffect** object. Command effects can be changed using the **CommandEffect** object's **Command** and **Type** properties.

Example

The following example shows how to set a command effect animation behavior.

```
Set bhvEffect = effectNew.Behaviors.Add(msoAnimTypeCommand)
```

```
With bhvEffect.CommandEffect  
    .Type = msoAnimCommandTypeVerb  
    .Command = Play  
End With
```



Comment Object

[Comments](#) | [Comment](#)

Represents a comment on a given slide or slide range. The **Comment** object is a member of the [Comments](#) collection object.

Using the Comment object

Use [Comments](#) (*index*), where *index* is the number of the comment, or the [Item](#) method to access a single comment on a slide. This example displays the author of the first comment on the first slide. If there are no comments, it displays a message stating such.

```
Sub ShowComment()  
    With ActivePresentation.Slides(1).Comments  
        If .Count > 0 Then  
            MsgBox "The first comment on this slide is by " & _  
                .Item(1).Author  
        Else  
            MsgBox "There are no comments on this slide."  
        End If  
    End With  
End Sub
```

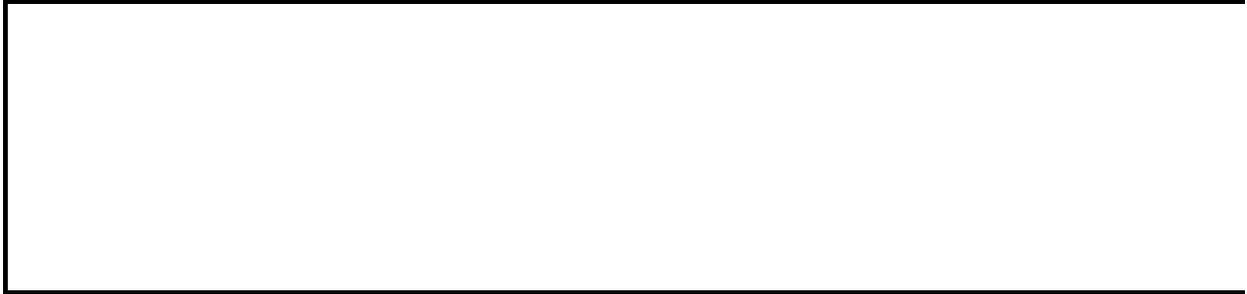
Use the following properties to access comment data:

Author	The author's full name
AuthorIndex	The author's index in the list of comments
AuthorInitials	The author's initials
DateTime	The date and time the comment was created
Text	The text of the comment
Left, Top	The comment's screen coordinates

This example displays a message containing the author, date and time, and contents of all the messages on the first slide.

```
Sub SlideComments()  
    Dim cmtExisting As Comment  
    Dim cmtAll As Comments  
    Dim strComments As String  
  
    Set cmtAll = ActivePresentation.Slides(1).Comments  
  
    If cmtAll.Count > 0 Then  
        For Each cmtExisting In cmtAll  
            strComments = strComments & cmtExisting.Author & vbTab &
```

```
        cmtExisting.DateTime & vbTab & cmtExisting.Text & vb
    Next
    MsgBox "The comments in your document are as follows:" & vbCrLf
        & strComments
Else
    MsgBox "This slide doesn't have any comments."
End If
End Sub
```



ConnectorFormat Object

Multiple objects [↳ ConnectorFormat](#)
[↳ Shape](#)

Contains properties and methods that apply to connectors. A connector is a line that attaches two other shapes at points called connection sites. If you rearrange shapes that are connected, the geometry of the connector will be automatically adjusted so that the shapes remain connected.

Using the ConnectorFormat Object

Use the **ConnectorFormat** property to return a **ConnectorFormat** object. Use the [BeginConnect](#) and [EndConnect](#) methods to attach the ends of the connector to other shapes in the document. Use the [RerouteConnections](#) method to automatically find the shortest path between the two shapes connected by the connector. Use the [Connector](#) property to see whether a shape is a connector.

Note that you assign a size and a position when you add a connector to the **Shapes** collection, but the size and position are automatically adjusted when you attach the beginning and end of the connector to other shapes in the collection. Therefore, if you intend to attach a connector to other shapes, the initial size and position you specify are irrelevant. Likewise, you specify which connection sites on a shape to attach the connector to when you attach the connector, but using the **RerouteConnections** method after the connector is attached may change which connection sites the connector attaches to, making your original choice of connection sites irrelevant.

The following example adds two rectangles to myDocument and connects them with a curved connector.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 0, 0).ConnectorFormat
    .BeginConnect ConnectedShape:=firstRect, ConnectionSite:=1
    .EndConnect ConnectedShape:=secondRect, ConnectionSite:=1
    .Parent.RerouteConnections
End With
```

Remarks

Connection sites are generally numbered according to the rules presented in the following table.

Shape type	Connection site numbering scheme
AutoShapes, WordArt, pictures, and OLE objects	The connection sites are numbered starting at the top and proceeding counterclockwise.
Freeforms	The connection sites are the vertices, and they correspond to the vertex numbers.

To figure out which number corresponds to which connection site on a complex shape, you can experiment with the shape while the macro recorder is turned on and then examine the recorded code; or you can create a shape, select it, and then run the following example. This code will number each connection site and attach a connector to it.

```
Set mainshape = ActiveWindow.Selection.ShapeRange(1)
With mainshape
    bx = .Left + .Width + 50
    by = .Top + .Height + 50
End With
With ActiveWindow.View.Slide
    For j = 1 To mainshape.ConnectionSiteCount
        With .Shapes.AddConnector(msoConnectorStraight, _
            bx, by, bx + 50, by + 50)
            .ConnectorFormat.EndConnect mainshape, j
            .ConnectorFormat.Type = msoConnectorElbow
            .Line.ForeColor.RGB = RGB(255, 0, 0)
            l = .Left
            t = .Top
        End With
        With .Shapes.AddTextbox(msoTextOrientationHorizontal, _
            l, t, 36, 14)
            .Fill.Visible = False
            .Line.Visible = False
            .TextFrame.TextRange.Text = j
        End With
    Next j
End With
```



DefaultWebOptions Object

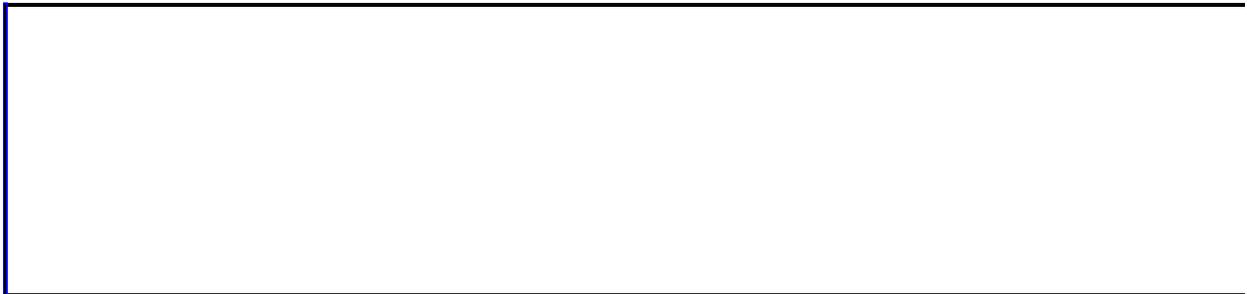
[Application](#) └ [DefaultWebOptions](#)

Contains global application-level attributes used by Microsoft PowerPoint when you publish or save a complete or partial presentation as a Web page or when you open a Web page. You can return or set attributes either at the application (global) level or at the presentation level. (Note that attribute values can be different from one presentation to another, depending on the attribute value at the time the presentation was saved.) Presentation-level attribute settings override application-level attribute settings. Presentation-level attributes are contained in the [WebOptions](#) object.

Using the DefaultWebOptions Object

Use the [DefaultWebOptions](#) property to return the **DefaultWebOptions** object. The following example checks to see whether PNG (Portable Network Graphics) are allowed as an image format, and then sets the `strImageFileType` variable accordingly.

```
Set objAppWebOptions = Application.DefaultWebOptions
With objAppWebOptions
    If .AllowPNG = True Then
        strImageFileType = "PNG"
    Else
        strImageFileType = "JPG"
    End If
End With
```



Design Object

Multiple objects [↳ Design](#)

Represents an individual slide design template. The **Design** object is a member of the [Designs](#) and [SlideRange](#) collections and the [Master](#) and [Slide](#) objects.

Using the Design object

Use the [Design](#) property of the **Master**, **Slide**, or **SlideRange** objects to access a **Design** object, for example:

- `ActivePresentation.SlideMaster.Design`
- `ActivePresentation.Slides(1).Design`
- `ActivePresentation.Slides.Range.Design`

Use the [Add](#), [Item](#), [Clone](#), or [Load](#) methods of the **Designs** collection to add, refer to, clone, or load a **Design** object, respectively. For example, to add a design template, use `ActivePresentation.Designs.Add designName:="MyDesign"`

The **Design** object's [AddTitleMaster](#) method and [HasTitleMaster](#) property can be used to add and / or query the status of a title slide master. For example:

```
Sub AddQueryTitleMaster(dsn As Design)
    dsn.AddTitleMaster
    MsgBox dsn.HasTitleMaster
End Sub
```

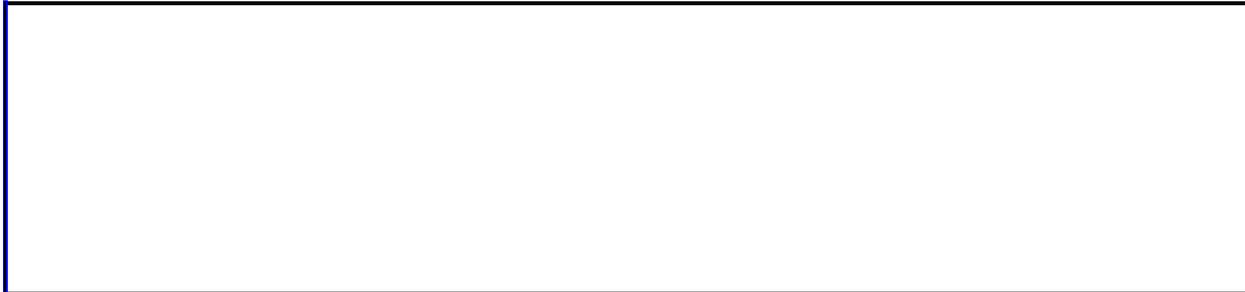


Diagram Object

Multiple objects [└Diagram](#)
[└DiagramNodes](#)

Represents a single diagram in a document. The **Diagram** object is a member of the [DiagramNode](#) and [Shape](#) objects and the [ShapeRange](#) collection.

Using the Diagram object

Use the **Diagram** property of the [Shape](#) object or [ShapeRange](#) collection to return a **Diagram** object.

Use the [Convert](#) method to change a diagram from one type to another. This example converts the first diagram on the first slide of the active presentation into a radial diagram. This example assumes that the first shape in the active presentation is a diagram and not another type of shape.

```
Sub DiagramConvert()  
    ActivePresentation.Slides(1).Shapes(1).Diagram _  
        .Convert Type:=msoDiagramRadial  
End Sub
```

Use the [Reverse](#) property to flip the order of the nodes in a diagram, so that the first node becomes the last node, and vice versa. This example reverses the order of the diagram nodes for the second shape on the first slide of the active presentation . This assumes that the second shape in the active presentation is a diagram and not another type of shape.

```
Sub DiagramReverse()  
    ActivePresentation.Slides(1).Shapes(2).Diagram.Reverse = msoTrue  
End Sub
```

A [DiagramNode](#) object can have a nested **Diagram** object. Use the [Diagram](#) property of a **DiagramNode** object to return the nested **Diagram** object.



DiagramNode Object

Multiple objects [↳ DiagramNode](#)
↳ Multiple objects

Represents a node in a diagram.

Using the DiagramNode object

To return a **DiagramNode** object, use one of the following:

- The **DiagramNode** object's [AddNode](#), [CloneNode](#), [NextNode](#) or [PrevNode](#) methods, or [Root](#) property.
- The [DiagramNodeChildren](#) collection's [AddNode](#) or [Item](#) methods, or [FirstChild](#) or [LastChild](#) properties.
- The [DiagramNodes](#) collection's [Item](#) method.
- The [Shape](#) object's or [ShapeRange](#) collection's [DiagramNode](#) property.

A diagram node can terminate, or contain other child diagrams, child diagram nodes, or child shapes:

- To refer to a child diagram, use the [Diagram](#) property.
- To refer to an individual child diagram node, use the [AddNode](#), [CloneNode](#), [NextNode](#) or [PrevNode](#) methods, or [Root](#) property.
- To refer to a collection of child diagram nodes, use the [Children](#) property.
- To refer to a shape, use the [Shape](#) or [TextShape](#) properties.

Use the [AddNode](#) method to add a node to a diagram or to a diagram node. This example assumes the third shape in the active presentation is a diagram and adds a node to it.

```
Sub AddDiagramNode()  
    ActivePresentation.Shapes(3).DiagramNode.Children.AddNode  
End Sub
```

Use the [Delete](#) method to remove a node from a diagram or diagram node. This example assumes the second shape in the presentation is a diagram and removes the first node from it.

```
Sub DeleteDiagramNode()  
    ActivePresentation.Shapes(2).DiagramNode.Children(1).Delete  
End Sub
```



DiagramNodeChildren Collection

[DiagramNode](#) └ [DiagramNodeChildren](#)
└ [DiagramNode](#)

A collection of **DiagramNode** objects that represents child nodes in a diagram.

Using the DiagramNodeChildren collection

Use the [Children](#) property of the [DiagramNode](#) object to return a **DiagramNodeChildren** collection. To add an individual child diagram node to the collection, use the [AddNode](#) method. To return individual child diagram nodes in the collection, use the [FirstChild](#) or [LastChild](#) properties or the [Item](#) method.

This example deletes the first child of the second node in the first diagram in the document. This example assumes that the first shape in the active document is a diagram with at least two nodes, one with child nodes.

```
Sub DiagramNodeChild()  
    ActiveDocument.Shapes(1).Diagram.Nodes.Item(2) _  
        .Children.FirstChild.Delete  
End Sub
```



DocumentWindow Object

Multiple objects [DocumentWindows](#)
└─ [DocumentWindow](#)
 └─ Multiple objects

Represents a document window. The **DocumentWindow** object is a member of the [DocumentWindows](#) collection. The **DocumentWindows** collection contains all the open document windows.

Using the DocumentWindow Object

Use **Windows**(*index*), where *index* is the document window index number, to return a single **DocumentWindow** object. The following example activates document window two.

```
Windows(2).Activate
```

The first member of the **DocumentWindows** collection, `Windows(1)`, always returns the active document window. Alternatively, you can use the [ActiveWindow](#) property to return the active document window. The following example maximizes the active window.

```
ActiveWindow.WindowState = ppWindowMaximized
```

Use **Panes**(*index*), where *index* is the pane index number, to manipulate panes within normal, slide, outline, or notes page views of the document window. The following example activates pane three, which is the notes pane.

```
ActiveWindow.Panes(3).Activate
```

Use the [ActivePane](#) property to return the active pane within the document window. The following example checks to see if the active pane is the outline pane. If not, it activates the outline pane.

```
mypane = ActiveWindow.ActivePane.ViewType  
If mypane <> 1 Then  
    ActiveWindow.Panes(1).Activate  
End If
```

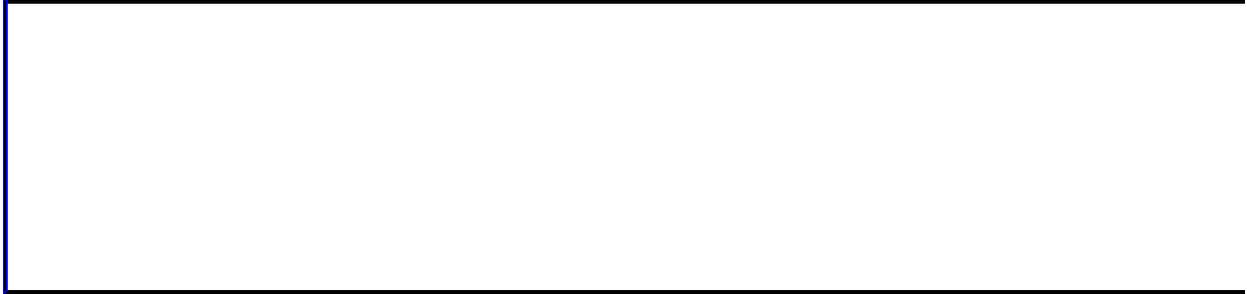
Use the [Presentation](#) property to return the presentation that's currently running in the specified document window.

Use the [Selection](#) property to return the selection.

Use the [SplitHorizontal](#) property to return the percentage of the screen width that the outline pane occupies in normal view.

Use the [SplitVertical](#) property to return the percentage of the screen height that the slide pane occupies in normal view.

Use the [View](#) property to return the view in the specified document window.



Effect Object

[Sequence](#) └ [Effect](#)
└ Multiple objects

Represents timing information about a slide animation.

Using the Effect object

Use the [AddEffect](#) method to add an effect. This example adds a shape to the first slide in the active presentation and adds an effect and a behavior to the shape.

```
Sub NewShapeAndEffect()  
    Dim shpStar As Shape  
    Dim sldOne As Slide  
    Dim effNew As Effect  
  
    Set sldOne = ActivePresentation.Slides(1)  
    Set shpStar = sldOne.Shapes.AddShape(Type:=msoShape5pointStar, _  
        Left:=150, Top:=72, Width:=400, Height:=400)  
    Set effNew = sldOne.TimeLine.MainSequence.AddEffect(Shape:=shpSt  
        EffectId:=msoAnimEffectStretchy, Trigger:=msoAnimTriggerAfte  
    With effNew  
        With .Behaviors.Add(msoAnimTypeScale).ScaleEffect  
            .FromX = 75  
            .FromY = 75  
            .ToX = 0  
            .ToY = 0  
        End With  
        .Timing.AutoReverse = msoTrue  
    End With  
End Sub
```

To refer to an existing **Effect** object, use [MainSequence](#) (*index*), where *index* is the number of the **Effect** object in the [Sequence](#) collection. This example changes the effect for the first sequence and specifies the behavior for that effect.

```
Sub ChangeEffect()  
    With ActivePresentation.Slides(1).TimeLine _  
        .MainSequence(1)  
        .EffectType = msoAnimEffectSpin  
        With .Behaviors(1).RotationEffect  
            .From = 100  
            .To = 360  
            .By = 5  
        End With  
    End With  
End Sub
```

There is always at least one **Effect** object in each slide regardless of whether the slide has animations or not.



EffectInformation Object

[Effect](#) └ [EffectInformation](#)
└ Multiple objects

Represents various animation options for an [Effect](#) object.

Using the EffectInformation object

Use the members of the **EffectInformation** object to return the current state of an **Effect** object, such as the after effect, whether the background animates along with its corresponding text, whether text animates in reverse, play settings, sound effects, text building behavior, and so on. All of the members of the **EffectInformation** object are read-only. To change any effect information properties, you must use the methods of the corresponding [Sequence](#) object.

Use the [EffectInformation](#) property of the [Effect](#) object to return an **EffectInformation** object. The following example sets the [HideWhileNotPlaying](#) property for the play settings in the main animation sequence.

```
Sub HideEffect()  
    ActiveWindow.Selection.SlideRange(1).TimeLine _  
        .MainSequence(1).EffectInformation.PlaySettings _  
        .HideWhileNotPlaying = msoTrue  
End Sub
```



EffectParameters Object

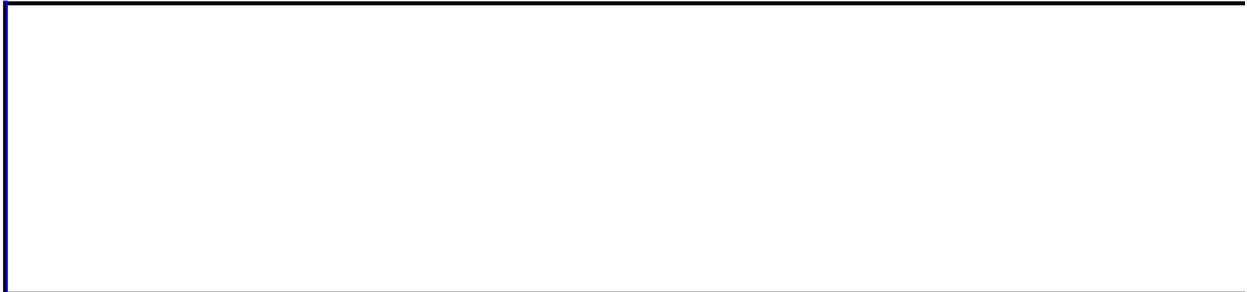
[Effect](#) └ [EffectParameters](#)
└ [ColorFormat](#)

Represents various animation parameters for an [Effect](#) object, such as colors, fonts, sizes, and directions.

Using the EffectParameters object

Use the [EffectParameters](#) property of the **Effect** object to return an **EffectParameters** object. The following example creates a shape, sets a fill effect, and changes the starting and ending fill colors.

```
Sub effParam()  
    Dim shpNew As Shape  
    Dim effNew As Effect  
  
    Set shpNew = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeHeart, Left:=100, _  
            Top:=100, Width:=150, Height:=150)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpNew, EffectID:=msoAnimEffectChangeFillC  
            Trigger:=msoAnimTriggerAfterPrevious)  
  
    With effNew.EffectParameters  
        .Color1.RGB = RGB(Red:=0, Green:=0, Blue:=255)  
        .Color2.RGB = RGB(Red:=255, Green:=0, Blue:=0)  
    End With  
  
End Sub
```



[Show All](#)

ExtraColors Object

[Presentation](#) | [ExtraColors](#)

Represents the [extra colors](#) in a presentation. The object can contain up to eight colors, each of which is represented by an [red-green-blue \(RGB\) value](#).

Using the ExtraColors Object

Use the [ExtraColors](#) property to return the **ExtraColors** object. Use **ExtraColors(index)**, where index is the extra color index number, to return the red-green-blue (RGB) value for a single extra color. The following example adds a rectangle to slide one in the active presentation and sets its fill foreground color to the first extra color. If there hasn't been at least one extra color defined for the presentation, this example will fail.

```
With ActivePresentation
  Set rect = .Slides(1).Shapes _
    .AddShape(msoShapeRectangle, 50, 50, 100, 200)
  rect.Fill.ForeColor.RGB = .ExtraColors(1)
End With
```

Use the [Add](#) method to add an extra color. The following example adds an extra color to the active presentation (if the color hasn't already been added).

```
ActivePresentation.ExtraColors.Add RGB(69, 32, 155)
```



FillFormat Object

Multiple objects [└ FillFormat](#)
[└ ColorFormat](#)

Represents fill formatting for a shape. A shape can have a solid, gradient, texture, pattern, picture, or semi-transparent fill.

Using the FillFormat Object

Use the **Fill** property to return a **FillFormat** object. The following example adds a rectangle to myDocument and then sets the gradient and color for the rectangle's fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 90, 90, 90, 80).Fill
    .ForeColor.RGB = RGB(0, 128, 128)
    .OneColorGradient msoGradientHorizontal, 1, 1
End With
```

Remarks

Many of the properties of the **FillFormat** object are read-only. To set one of these properties, you have to apply the corresponding method.



FilterEffect Object

[AnimationBehavior](#) \perp [FilterEffect](#)

Represents a filter effect for an animation behavior.

Using the FilterEffect Object

Use the **FilterEffect** property of the [AnimationBehavior](#) object to return a **FilterEffect** object. Filter effects can be changed using the **FilterEffect** object's **Reveal**, **SubType**, and **Type** properties.

Example

The following example adds a shape to the first slide of the active presentation and sets a filter effect animation behavior.

```
Sub ChangeFilterEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeFilter)  
  
    With bhvEffect.FilterEffect  
        .Type = msoAnimFilterEffectTypeWipe  
        .Subtype = msoAnimFilterEffectSubtypeUp  
        .Reveal = msoTrue  
    End With  
End Sub
```



Font Object

Multiple objects [Font](#)
[ColorFormat](#)

Represents character formatting for text or a bullet. The **Font** object is a member of the [Fonts](#) collection. The **Fonts** collection contains all the fonts used in a presentation.

Using the Font Object

This section describes how to do the following:

- Return the **Font** object that represents the font attributes of a specified bullet, a specified range of text, or all text at a specified outline level
- Return a **Font** object from the collection of all the fonts used in the presentation

Returning the Font object that represents the font attributes of a specified bullet, a specified range of text, or all text at a specified outline level

Use the [Font](#) property to return the **Font** object that represents the font attributes for a specific bullet, text range, or outline level. The following example sets the title text on slide one and sets the font properties.

```
With ActivePresentation.Slides(1).Shapes.Title _  
    .TextFrame.TextRange  
    .Text = "Volcano Coffee"  
    With .Font  
        .Italic = True  
        .Name = "Palatino"  
        .Color.RGB = RGB(0, 0, 255)  
    End With  
End With
```

Returning a Font object from the collection of all the fonts used in the presentation

Use **Fonts(index)**, where *index* is the font's name or index number, to return a single **Font** object. The following example checks to see whether font one in the active presentation is embedded in the presentation.

```
If ActivePresentation.Fonts(1).Embedded = _  
    True Then MsgBox "Font 1 is embedded"
```



FreeformBuilder Object

[FreeformBuilder](#)

Represents the geometry of a freeform while it's being built.

Using the FreeformBuilder Object

Use the [BuildFreeform](#) method to return a **FreeformBuilder** object. Use the [AddNodes](#) method to add nodes to the freeform. Use the [ConvertToShape](#) method to create the shape defined in the **FreeformBuilder** object and add it to the [Shapes](#) collection. The following example adds a freeform with four segments to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.BuildFreeform(msoEditingCorner, 360, 200)
    .AddNodes msoSegmentCurve, msoEditingCorner, _
        380, 230, 400, 250, 450, 300
    .AddNodes msoSegmentCurve, msoEditingAuto, 480, 200
    .AddNodes msoSegmentLine, msoEditingAuto, 480, 400
    .AddNodes msoSegmentLine, msoEditingAuto, 360, 200
    .ConvertToShape
End With
```



GroupShapes Collection Object

Multiple objects [↳ GroupShapes](#)

Represents the individual shapes within a grouped shape. Each shape is represented by a [Shape](#) object. Using the [Item](#) method with this object, you can work with single shapes within a group without having to ungroup them.

Using The Groupshapes Collection

Use the [GroupItems](#) property to return the **GroupShapes** collection. Use **GroupItems(index)**, where *index* is the number of the individual shape within the grouped shape, to return a single shape from the **GroupShapes** collection. The following example adds three triangles to myDocument, groups them, sets a color for the entire group, and then changes the color for the second triangle only.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeIsoscelesTriangle, 10, _
        10, 100, 100).Name = "shpOne"
    .AddShape(msoShapeIsoscelesTriangle, 150, _
        10, 100, 100).Name = "shpTwo"
    .AddShape(msoShapeIsoscelesTriangle, 300, _
        10, 100, 100).Name = "shpThree"
    With .Range(Array("shpOne", "shpTwo", "shpThree")).Group
        .Fill.PresetTextured msoTextureBlueTissuePaper
        .GroupItems(2).Fill.PresetTextured msoTextureGreenMarble
    End With
End With
```



HeaderFooter Object

[HeadersFooters](#)  [HeaderFooter](#)

Represents a header, footer, date and time, slide number, or page number on a slide or master. All the **HeaderFooter** objects for a slide or master are contained in a [HeadersFooters](#) object.

Using the HeaderFooter Object

Use one of the properties listed in the following table to return the **HeaderFooter** object.

Use this property	To return
<u>DateAndTime</u>	A HeaderFooter object that represents the date and time on the slide.
<u>Footer</u>	A HeaderFooter object that represents the footer for the slide.
<u>Header</u>	A HeaderFooter object that represents the header for the slide. This works only for notes pages and handouts, not for slides.
<u>SlideNumber</u>	A HeaderFooter object that represent the slide number (on a slide) or page number (on a notes page or a handout).

Note **HeaderFooter** objects aren't available for **Slide** objects that represent notes pages. The **HeaderFooter** object that represents a header is available only for a notes master or handout master.

You can set properties of **HeaderFooter** objects for single slides. The following example sets the footer text for slide one in the active presentation.

```
ActivePresentation.Slides(1).HeadersFooters.Footer _  
    .Text = "Volcano Coffee"
```

You can also set properties of **HeaderFooter** objects for the slide master, title master, notes master, or handout master to affect all slides, title slides, notes pages, or handouts and outlines at the same time. The following example sets the text for the footer in the slide master for the active presentation, sets the format for the date and time, and turns on the display of slide numbers. These settings will apply to all slides that are based on this master that display master graphics and that have not had their footer and date and time set individually.

```
Set mySlidesHF = ActivePresentation.SlideMaster.HeadersFooters  
With mySlidesHF  
    .Footer.Visible = True
```

```
.Footer.Text = "Regional Sales"  
.SlideNumber.Visible = True  
.DateAndTime.Visible = True  
.DateAndTime.UseFormat = True  
.DateAndTime.Format = ppDateTimeMdy  
End With
```

To clear header and footer information that has been set for individual slides and make sure all slides display the header and information you define for the slide master, run the following code before running the previous example.

```
For Each s In ActivePresentation.Slides  
    s.DisplayMasterShapes = True  
    s.HeadersFooters.Clear  
Next
```



LineFormat Object

Multiple objects [└LineFormat](#)
[└ColorFormat](#)

Represents line and arrowhead formatting. For a line, the **LineFormat** object contains formatting information for the line itself; for a shape with a border, this object contains formatting information for the shape's border.

Using the LineFormat Object

Use the **Line** property to return a **LineFormat** object. The following example adds a blue, dashed line to myDocument. There's a short, narrow oval at the line's starting point and a long, wide triangle at its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(50, 0, 128)
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



LinkFormat Object

Multiple objects [LinkFormat](#)

Contains properties and methods that apply to linked OLE objects. The [OLEFormat](#) object contains properties and methods that apply to OLE objects whether or not they're linked. The [PictureFormat](#) object contains properties and methods that apply to pictures and OLE objects.

Using the LinkFormat Object

Use the **LinkFormat** property to return a **LinkFormat** object. The following example loops through all the shapes on all the slides in the active presentation and sets all linked Microsoft Excel worksheets to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Excel.Sheet" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
```



Master Object

[Master](#) └ Multiple objects

Represents a slide master, title master, handout master, notes master, or design master.

Using the Master Object

To return a **Master** object, use the [Master](#) property of the [Slide](#) object or [SlideRange](#) collection, or use the [HandoutMaster](#), [NotesMaster](#), [SlideMaster](#), or [TitleMaster](#) property of the [Presentation](#) object. Note that some of these properties are also available from the [Design](#) object as well. The following example sets the background fill for the slide master for the active presentation.

```
ActivePresentation.SlideMaster.Background.Fill _  
    .PresetGradient msoGradientHorizontal, 1, msoGradientBrass
```

To add a title master or design to a presentation and return a **Master** object that represents the new title master or design, use the [AddTitleMaster](#) method. The following example adds a title master to the active presentation and places the title placeholder 10 points from the top of the master.

```
ActivePresentation.AddTitleMaster.Shapes.Title.Top = 10
```



MotionEvent Object

[AnimationBehavior](#) ↳ [MotionEffect](#)

Represents a motion effect for an **AnimationBehavior** object.

Using the MotionEffect object

Use the [MotionEffect](#) property of the **AnimationBehavior** object to return a **MotionEffect** object. The following example refers to the motion effect for a given animation behavior.

```
ActivePresentation.Slides(1).TimeLine.MainSequence.Item.Behaviors(1)
```

Use the [ByX](#), [ByY](#), [FromX](#), [FromY](#), [ToX](#), and [ToY](#) properties of the MotionEffect object to construct a motion path. The following example adds a shape to the first slide and creates a motion path.

```
Sub AddMotionPath()  
  
    Dim shpNew As Shape  
    Dim effNew As Effect  
    Dim aniMotion As AnimationBehavior  
  
    Set shpNew = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShape5pointStar, Left:=0, _  
        Top:=0, Width:=100, Height:=100)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpNew, effectId:=msoAnimEffectCustom, _  
        Trigger:=msoAnimTriggerWithPrevious)  
    Set aniMotion = effNew.Behaviors.Add(msoAnimTypeMotion)  
  
    With aniMotion.MotionEffect  
        .FromX = 0  
        .FromY = 0  
        .ToX = 500  
        .ToY = 500  
    End With  
  
End Sub
```



NamedSlideShow Object

[NamedSlideShows](#)  [NamedSlideShow](#)

Represents a custom slide show, which is a named subset of slides in a presentation. The **NamedSlideShow** object is a member of the [NamedSlideShows](#) collection. The **NamedSlideShows** collection contains all the named slide shows in the presentation.

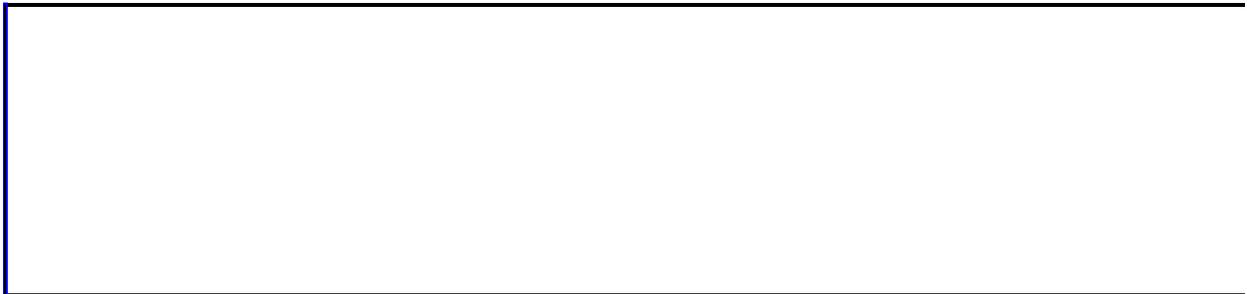
Using the NamedSlideShow Object

Use **NamedSlideShows**(*index*), where *index* is the custom slide show name or index number, to return a single **NamedSlideShow** object. The following example deletes the custom slide show named "Quick Show."

```
ActivePresentation.SlideShowSettings _  
    .NamedSlideShows("Quick Show").Delete
```

Use the [SlideIDs](#) property to return an array that contains the unique slide IDs for all the slides in the specified custom show. The following example displays the slide IDs for the slides in the custom slide show named "Quick Show."

```
idArray = ActivePresentation.SlideShowSettings _  
    .NamedSlideShows("Quick Show").SlideIDs  
For i = 1 To UBound(idArray)  
    MsgBox idArray(i)  
Next
```



ObjectVerbs Object

[OLEFormat](#) [ObjectVerbs](#)

Represents the collection of OLE verbs for the specified OLE object. OLE verbs are the operations supported by an OLE object. Commonly used OLE verbs are "play" and "edit."

Using the ObjectVerbs Object

Use the **ObjectVerbs** property to return an **ObjectVerbs** object. The following example displays all the available verbs for the OLE object contained in shape one on slide two in the active presentation. For this example to work, shape one must contain an OLE object.

```
With ActivePresentation.Slides(2).Shapes(1).OLEFormat
  For Each v In .ObjectVerbs
    MsgBox v
  Next
End With
```



OLEFormat Object

Multiple objects [OLEFormat](#)
[ObjectVerbs](#)

Contains properties and methods that apply to OLE objects. The [LinkFormat](#) object contains properties and methods that apply to linked OLE objects only. The [PictureFormat](#) object contains properties and methods that apply to pictures and OLE objects.

Using the OLEFormat Object

Use the **OLEFormat** property to return an **OLEFormat** object. The following example loops through all the shapes on all the slides in the active presentation and sets all linked Microsoft Excel worksheets to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Excel.Sheet" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
```



Options Object

[Application](#) `Options`

Represents application options in Microsoft PowerPoint.

Using the Options object

Use the [Options](#) property to return an **Options** object. The following example sets three application options for PowerPoint.

```
Sub TogglePasteOptionsButton()  
  With Application.Options  
    If .DisplayPasteOptions = False Then  
      .DisplayPasteOptions = True  
    End If  
  End With  
End Sub
```



PageSetup Object

[Presentation](#) └ [PageSetup](#)

Contains information about the page setup for slides, notes pages, handouts, and outlines in a presentation.

Using the PageSetup Object

Use the [PageSetup](#) property to return the **PageSetup** object. The following example sets all slides in the active presentation to be 11 inches wide and 8.5 inches high and sets the slide numbering for the presentation to start at 17.

```
With ActivePresentation.PageSetup
    .SlideWidth = 11 * 72
    .SlideHeight = 8.5 * 72
    .FirstSlideNumber = 17
End With
```



[Show All](#)

Pane Object

[DocumentWindow](#) └ [Panels](#)
└ [Pane](#)

An object representing one of the three [panes](#) in normal view or the single pane of any other view in the document window.

Using the Pane Object

Use **Panes**(*index*), where *index* is the index number for a pane, to return a single **Pane** object. The following table lists the names of the panes in normal view with their corresponding index numbers.

Pane	Index number
Outline	1
Slide	2
Notes	3

When using a document window view other than normal view, use **Panes**(1) to reference the single **Pane** object.

Use the [Activate](#) method to make the specified pane active.

Use the [ViewType](#) property to determine which pane is active. The following example uses the **ViewType** property to determine whether the slide pane is the active pane. If it is, then the **Activate** method makes the notes pane the active pane.

```
With ActiveWindow
    If .ActivePane.ViewType = ppViewSlide Then
        .Panes(3).Activate
    End If
End With
```

Remarks

Normal view is the only view with multiple panes. All other document window views have only a single pane, which is the document window.



ParagraphFormat Object

Multiple objects [└ ParagraphFormat](#)
[└ BulletFormat](#)

Represents the paragraph formatting of a text range.

Using the ParagraphFormat Object

Use the [ParagraphFormat](#) property to return the **ParagraphFormat** object. The following example left aligns the paragraphs in shape two on slide one in the active presentation.

```
ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange _  
    .ParagraphFormat.Alignment = ppAlignLeft
```



PictureFormat Object

Multiple objects [PictureFormat](#)

Contains properties and methods that apply to pictures and OLE objects. The [LinkFormat](#) object contains properties and methods that apply to linked OLE objects only. The [OLEFormat](#) object contains properties and methods that apply to OLE objects whether or not they're linked.

Using the PictureFormat Object

Use the **PictureFormat** property to return a **PictureFormat** object. The following example sets the brightness, contrast, and color transformation for shape one on myDocument and crops 18 points off the bottom of the shape. For this example to work, shape one must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).PictureFormat
    .Brightness = 0.3
    .Contrast = 0.7
    .ColorType = msoPictureGrayScale
    .CropBottom = 18
End With
```



PlaceholderFormat Object

Multiple objects [PlaceholderFormat](#)

Contains properties that apply specifically to placeholders, such as placeholder type.

Using the PlaceholderFormat Object

Use the [PlaceholderFormat](#) property to return a **PlaceholderFormat** object. The following example adds text to placeholder one on slide one in the active presentation if that placeholder exists and is a horizontal title placeholder.

```
With ActivePresentation.Slides(1).Shapes.Placeholders
  If .Count > 0 Then
    With .Item(1)
      Select Case .PlaceholderFormat.Type
        Case ppPlaceholderTitle
          .TextFrame.TextRange = "Title Text"
        Case ppPlaceholderCenterTitle
          .TextFrame.TextRange = "Centered Title Text"
        Case Else
          MsgBox "There's no horizontal " _
            "title on this slide"
      End Select
    End With
  End If
End With
```



PlaySettings Object

Multiple objects [PlaySettings](#)

Contains information about how the specified media clip will be played during a slide show.

Using the PlaySettings Object

Use the [PlaySettings](#) property to return the **PlaySettings** object. The following example inserts a movie named "Clock.avi" into slide one in the active presentation. It then sets it to be played automatically after the previous animation or slide transition, specifies that the slide show continue while the movie plays, and specifies that the movie object be hidden during a slide show except when it's playing.

```
Set clockMovie = ActivePresentation.Slides(1).Shapes _  
    .AddMediaObject(FileName:="C:\WINNT\clock.avi", _  
    Left:=20, Top:=20)  
With clockMovie.AnimationSettings.PlaySettings  
    .PlayOnEntry = True  
    .PauseAnimation = False  
    .HideWhileNotPlaying = True  
End With
```



Presentation Object

Multiple objects [Presentation](#)
└─Multiple objects

Represents a PowerPoint presentation. The **Presentation** object is a member of the [Presentations](#) collection. The **Presentations** collection contains all the **Presentation** objects that represent open presentations in PowerPoint.

Using the Presentation Object

This section describes how to:

- Return a presentation that you specify by name or index number
- Return the presentation in the active window
- Return the presentation in any document window or slide show window you specify

Returning a presentation that you specify by name or index number

Use **Presentations**(*index*), where *index* is the presentation's name or index number, to return a single **Presentation** object. The name of the presentation is the file name, with or without the file name extension, and without the path. The following example adds a slide to the beginning of Sample Presentation.

```
Presentations("Sample Presentation").Slides.Add 1, 1
```

Note that if multiple presentations with the same name are open, the first presentation in the collection with the specified name is returned.

Returning the presentation in the active window

Use the [ActivePresentation](#) property to return the presentation in the active window. The following example saves the active presentation.

```
ActivePresentation.Save
```

Note that if an embedded presentation is in-place active, the **ActivePresentation** property returns the embedded presentation.

Returning the presentation in any document window or slide show window you specify

Use the [Presentation](#) property to return the presentation that's in the specified document window or slide show window. The following example displays the name of the slide show running in slide show window one.

```
MsgBox SlideShowWindows(1).Presentation.Name
```



PrintOptions Object

Multiple objects [PrintOptions](#)
[PrintRanges](#)

Contains print options for a presentation.

Note Specifying the optional arguments *From*, *To*, *Copies*, and *Collate* for the [PrintOut](#) method will set the corresponding properties of the **PrintOptions** object.

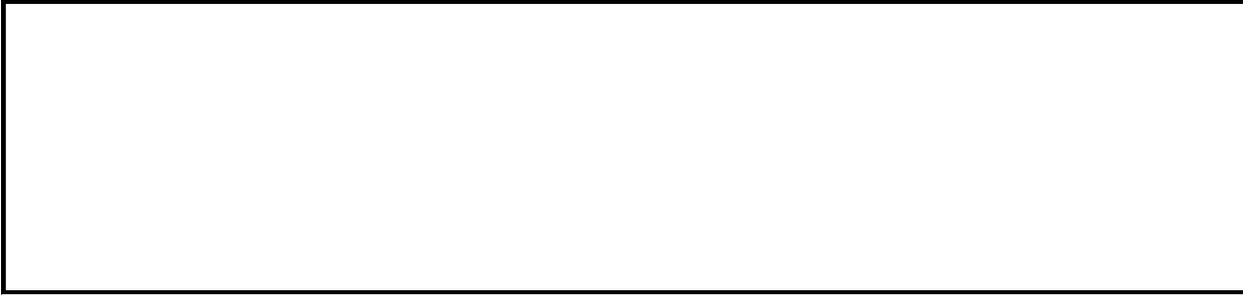
Using the PrintOptions Object

Use the [PrintOptions](#) property to return the **PrintOptions** object. The following example prints two uncollated color copies of all the slides (whether visible or hidden) in the active presentation. The example also scales each slide to fit the printed page and frames each slide with a thin border.

```
With ActivePresentation
  With .PrintOptions
    .NumberOfCopies = 2
    .Collate = False
    .PrintColorType = ppPrintColor
    .PrintHiddenSlides = True
    .FitToPage = True
    .FrameSlides = True
    .OutputType = ppPrintOutputSlides
  End With
  .PrintOut
End With
```

Use the [RangeType](#) property to specify whether to print the entire presentation or only a specified part of it. If you want to print only certain slides, set the **RangeType** property to **ppPrintSlideRange**, and use the [Ranges](#) property to specify which pages to print. The following example prints slides 1, 4, 5, and 6 in the active presentation

```
With ActivePresentation
  With .PrintOptions
    .RangeType = ppPrintSlideRange
    With .Ranges
      .Add 1, 1
      .Add 4, 6
    End With
  End With
  .PrintOut
End With
```



PrintRange Object

[PrintRanges](#) | [PrintRange](#)

Represents a single range of consecutive slides or pages to be printed. The **PrintRange** object is a member of the [PrintRanges](#) collection. The **PrintRanges** collection contains all the print ranges that have been defined for the specified presentation.

Using the PrintRange Object

Use **Ranges**(*index*), where *index* is the print range index number, to return a single **PrintRange** object. The following example displays a message that indicates the starting and ending slide numbers for print range one in the active presentation.

```
With ActivePresentation.PrintOptions.Ranges
  If .Count > 0 Then
    With .Item(1)
      MsgBox "Print range 1 starts on slide " & .Start & _
            " and ends on slide " & .End
    End With
  End If
End With
```

Use the [Add](#) method to create a **PrintRange** object and add it to the **PrintRanges** collection. The following example defines three print ranges that represent slide 1, slides 3 through 5, and slides 8 and 9 in the active presentation and then prints the slides in these ranges.

```
With ActivePresentation.PrintOptions
  .RangeType = ppPrintSlideRange
  With .Ranges
    .ClearAll
    .Add 1, 1
    .Add 3, 5
    .Add 8, 9
  End With
End With
ActivePresentation.PrintOut
```

Remarks

You can set print ranges in the **PrintRanges** collection independent of the **RangeType** setting; these ranges are retained as long as the presentation they're contained in is loaded. The ranges in the **PrintRanges** collection are applied when the **RangeType** property is set to **ppPrintSlideRange**.

--

PropertyEffect Object

[AnimationBehavior](#) \perp [PropertyEffect](#)
 \perp [AnimationPoints](#)

Represents a property effect for an [AnimationBehavior](#) object.

Using the PropertyEffect object

Use the [PropertyEffect](#) property of the **AnimationBehavior** object to return a **PropertyEffect** object. The following example refers to the property effect for a specified animation behavior.

```
ActivePresentation.Slides(1).TimeLine.MainSequence.Item(1) _  
    .Behaviors(1).PropertyEffect
```

Use the [Points](#) property to access the animation points of a particular animation behavior. If you want to change only two states of an animation behavior, use the [From](#) and [To](#) properties. This example adds a new shape to the and sets the property effect to animate the fill color from blue to red.

```
Sub AddShapeSetAnimFill()  
  
    Dim effBlinds As Effect  
    Dim shpRectangle As Shape  
    Dim animProperty As AnimationBehavior  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effBlinds = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectBlind  
  
    effBlinds.Timing.Duration = 3  
  
    Set animProperty = effBlinds.Behaviors.Add(msoAnimTypeProperty)  
  
    With animProperty.PropertyEffect  
        .Property = msoAnimColor  
        .From = RGB(Red:=0, Green:=0, Blue:=255)  
        .To = RGB(Red:=255, Green:=0, Blue:=0)  
    End With  
  
End Sub
```



PublishObject Object

[PublishObjects](#) | [PublishObject](#)

Represents a complete or partial loaded presentation that is available for publishing to HTML. The **PublishObject** object is a member of the [PublishObjects](#) collection.

Using the PublishObject Object

Use **PublishObjects**(*index*), where *index* is always "1", to return the single object for a loaded presentation. There can be only one **PublishObject** object for each loaded presentation. This example publishes slides three through five of presentation two to HTML. It names the published presentation Mallard.htm.

```
With Presentations(2).PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```

Remarks

You can specify the content and attributes of the published presentation by setting various properties of the **PublishObject** object. For example, the [SourceType](#) property defines the portion of a loaded presentation to be published. The [RangeStart](#) property and the [RangeEnd](#) property specify the range of slides to publish, and the [SpeakerNotes](#) property designates whether or not to publish the speaker's notes.

RGBColor Object

[RGBColor](#)

Represents a single color in a color scheme.

Using the RGBColor Object

Use the [Colors](#) method to return an [RGBColor](#) object. You can set an **RGBColor** object to another **RGBColor** object. You can use the [RGB](#) property to set or return the explicit red-green-blue value for an **RGBColor** object, with the exception of the **RGBColor** objects defined by the **ppNotSchemeColor** and **ppSchemeColorMixed** constants. The **RGB** property can be returned, but not set, for these two objects. The following example sets the background color in color scheme one in the active presentation to red and sets the title color to the title color that's defined for color scheme two.

```
With ActivePresentation.ColorSchemes
    .Item(1).Colors(ppBackground).RGB = RGB(255, 0, 0)
    .Item(1).Colors(ppTitle) = .Item(2).Colors(ppTitle)
End With
```



RotationEffect Object

[AnimationBehavior](#) ⊥ [RotationEffect](#)

Represents a rotation effect for an [AnimationBehavior](#) object.

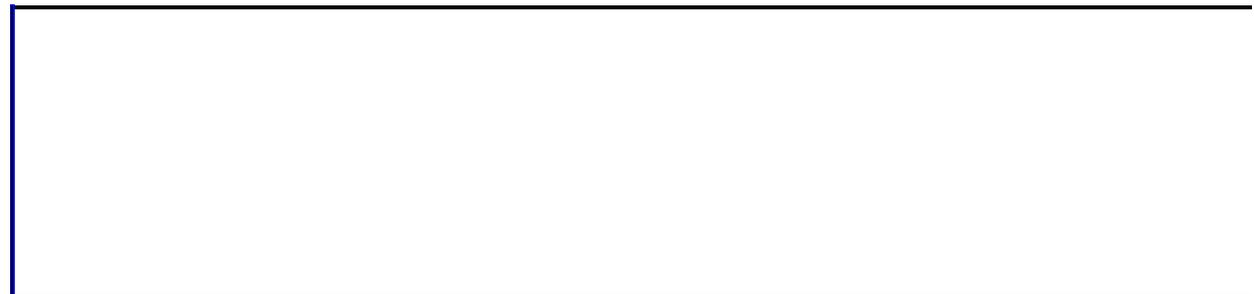
Using the RotationEffect object

Use the [RotationEffect](#) property of the **AnimationBehavior** object to return a **RotationEffect** object. The following example refers to the rotation effect for a given animation behavior.

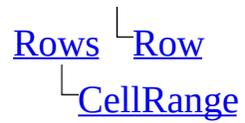
```
ActivePresentation.Slides(1).TimeLine.MainSequence.Item.Behaviors(1)
```

Use the [By](#), [From](#), and [To](#) properties of the **RotationEffect** object to affect an object's animation rotation. The following example adds a new shape to the first slide and sets the rotation animation behavior.

```
Sub AddRotation()  
  
    Dim shpNew As Shape  
    Dim effNew As Effect  
    Dim aniNew As AnimationBehavior  
  
    Set shpNew = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShape5pointStar, Left:=0, _  
        Top:=0, Width:=100, Height:=100)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpNew, effectId:=msoAnimEffectCustom)  
    Set aniNew = effNew.Behaviors.Add(msoAnimTypeRotation)  
  
    With aniNew.RotationEffect  
        'Rotate 270 degrees from current position  
        .By = 270  
    End With  
  
End Sub
```



Row Object



Represents a row in a table. The **Row** object is a member of the **Rows** collection. The **Rows** collection includes all the rows in the specified table.

Using the Row Object

Use **Rows**(*index*), where *index* is a number that represents the position of the row in the table, to return a single **Row** object. This example deletes the first row from the table in shape five on slide two of the active presentation.

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows(1).Delete
```

Use the **Select** method to select a row in a table. This example selects row one of the specified table.

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows(1).Select
```

Remarks

Use the [Cells](#) property to modify the individual cells in a **Row** object. This example selects the second row in the table and applies a dashed line style to the bottom border.

```
ActiveWindow.Selection.ShapeRange.Table.Rows(2) _  
    .Cells.Borders(ppBorderBottom).DashStyle = msoLineDash
```



Ruler Object

Multiple objects [Ruler](#)
└─ Multiple objects

Represents the ruler for the text in the specified shape or for all text in the specified text style. Contains tab stops and the indentation settings for text outline levels.

Using the Ruler Object

Use the [Ruler](#) property of the **TextFrame** object to return the **Ruler** object that represents the ruler for the text in the specified shape. Use the [TabStops](#) property to return the **TabStops** object that contains the tab stops on the ruler. Use the [Levels](#) property to return the **RulerLevels** object that contains the indentation settings for text outline levels. The following example sets a left-aligned tab stop at 2 inches (144 Points) and sets a hanging indent for the text in object two on slide one in the active presentation.

```
With ActivePresentation.Slides(1).Shapes(2).TextFrame.Ruler
    .TabStops.Add ppTabStopLeft, 144
    .Levels(1).FirstMargin = 0
    .Levels(1).LeftMargin = 36
End With
```

Use the [Ruler](#) property of the **TextStyle** object to return the **Ruler** object that represents the ruler for one of the four defined text styles (title text, body text, notes text, or default text). The following example sets the first-line indent and hanging indent for outline level one in body text on the slide master for the active presentation.

```
With ActivePresentation.SlideMaster _
    .TextStyles(ppBodyStyle).Ruler.Levels(1)
    .FirstMargin = 9
    .LeftMargin = 54
End With
```



RulerLevel Object

[RulerLevels](#) | [RulerLevel](#)

Contains first-line indent and hanging indent information for an outline level. The **RulerLevel** object is a member of the [RulerLevels](#) collection. The **RulerLevels** collection contains a **RulerLevel** object for each of the five available outline levels.

Using the RulerLevel Object

Use **RulerLevels**(*index*), where *index* is the outline level, to return a single **RulerLevel** object. The following example sets the first-line indent and hanging indent for outline level one in body text on the slide master for the active presentation.

```
With ActivePresentation.SlideMaster _  
    .TextStyles(ppBodyStyle).Ruler.Levels(1)  
    .FirstMargin = 9  
    .LeftMargin = 54  
End With
```

The following example sets the first-line indent and hanging indent for outline level one in shape two on slide one in the active presentation.

```
With ActivePresentation.SlideMaster.Shapes(2) _  
    .TextFrame.Ruler.Levels(1)  
    .FirstMargin = 9  
    .LeftMargin = 54  
End With
```



Selection Object

[DocumentWindow](#) └ [Selection](#)
└ Multiple objects

Represents the selection in the specified document window.

Using the Selection Object

Use the [Selection](#) property to return the **Selection** object. The following example places a copy of the selection in the active window on the Clipboard.

```
ActiveWindow.Selection.Copy
```

Use the [ShapeRange](#), [SlideRange](#), or [TextRange](#) property to return a range of shapes, slides, or text from the selection.

The following example sets the fill foreground color for the selected shapes in window two, assuming that there's at least one shape selected, and assuming that all selected shapes have a fill whose forecolor can be set.

```
With Windows(2).Selection.ShapeRange.Fill
    .Visible = True
    .ForeColor.RGB = RGB(255, 0, 255)
End With
```

The following example sets the text in the first selected shape in window two if that shape contains a text frame.

```
With Windows(2).Selection.ShapeRange(1)
    If .HasTextFrame Then
        .TextFrame.TextRange = "Current Choice"
    End If
End With
```

The following example cuts the selected text in the active window and places it on the Clipboard.

```
ActiveWindow.Selection.TextRange.Cut
```

The following example duplicates all the slides in the selection (if you're in slide view, this duplicates the current slide).

```
ActiveWindow.Selection.SlideRange.Duplicate
```

If you don't have an object of the appropriate type selected when you use one of these properties (for instance, if you use the **ShapeRange** property when there are no shapes selected), an error occurs. Use the [Type](#) property to determine what kind of object or objects are selected. The following example checks to see whether the selection contains slides. If the selection does contain slides, the example sets the background for the first slide in the selection.

```
With Windows(2).Selection
  If .Type = ppSelectionSlides Then
    With .SlideRange(1)
      .FollowMasterBackground = False
      .Background.Fill.PresetGradient _
        msoGradientHorizontal, 1, msoGradientLateSunset
    End With
  End If
End With
```

Remarks

The **Selection** object is deleted whenever you change slides in an active slide view (the **Type** property will return **ppSelectionNone**).

SetEffect Object

[AnimationBehavior](#) \perp [SetEffect](#)

Represents a set effect for an animation behavior. You can use the **SetEffect** object to set the value of a property.

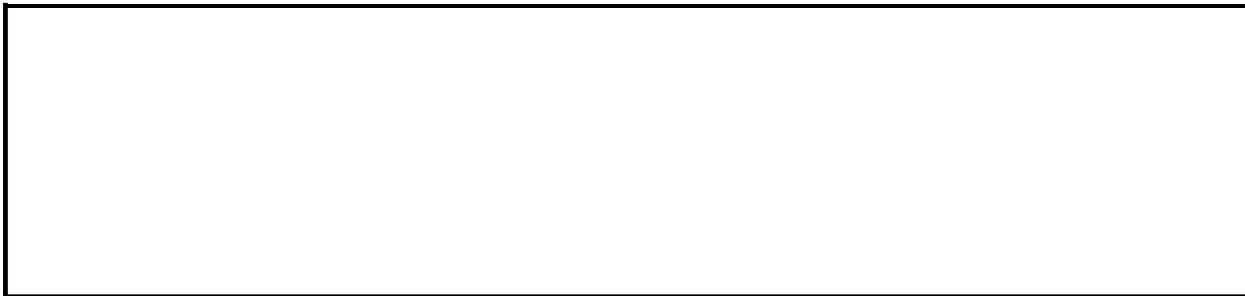
Using the SetEffect Object

Use the **SetEffect** property of the [AnimationBehavior](#) object to return a **SetEffect** object. Set effects can be changed using the **SetEffect** object's **Property** and **To** properties.

Example

The following example adds a shape to the first slide of the active presentation and sets a set effect animation behavior.

```
Sub ChangeSetEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeSet)  
  
    With bhvEffect.SetEffect  
        .Property = msoAnimShapeFillColor  
        .To = RGB(Red:=0, Green:=255, Blue:=255)  
    End With  
End Sub
```



ShadowFormat Object

Multiple objects [↳ ShadowFormat](#)
[↳ ColorFormat](#)

Represents shadow formatting for a shape.

Using the ShadowFormat Object

Use the **Shadow** property to return a **ShadowFormat** object. The following example adds a shadowed rectangle to myDocument. The semitransparent, blue shadow is offset 5 points to the right of the rectangle and 3 points above it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    50, 50, 100, 200).Shadow
    .ForeColor.RGB = RGB(0, 0, 128)
    .OffsetX = 5
    .OffsetY = -3
    .Transparency = 0.5
    .Visible = True
End With
```



[Show All](#)

Shape Object

Multiple objects  [Shape](#)
 Multiple objects

Represents an object in the drawing layer, such as an AutoShape, freeform, OLE object, or picture.

Note There are three objects that represent shapes: the **Shapes** collection, which represents all the shapes on a document; the [ShapeRange](#) collection, which represents a specified subset of the shapes on a document (for example, a **ShapeRange** object could represent shapes one and four on the document, or it could represent all the selected shapes on the document); the **Shape** object, which represents a single shape on a document. If you want to work with several shape at the same time or with shapes within the selection, use a **ShapeRange** collection. For an overview of how to work with either a single shape or with more than one shape at a time, see [Working with Shapes \(Drawing Objects\)](#) .

Using the Shape Object

This section describes how to:

- Return an existing shape on a slide, indexed by name or number.
- Return a newly created shape on a slide.
- Return a shape within the selection.
- Return the slide title and other placeholders on a slide.
- Return the shapes attached to the ends of a connector.
- Return the default shape for a presentation.
- Return a newly created freeform.
- Return a single shape from within a group.
- Return a newly formed group of shapes.

Returning an Existing Shape on a Slide

Use **Shapes**(*index*), where *index* is the shape name or the index number, to return a **Shape** object that represents a shape on a slide. The following example horizontally flips shape one and the shape named Rectangle 1 on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(1).Flip msoFlipHorizontal
myDocument.Shapes("Rectangle 1").Flip msoFlipHorizontal
```

Each shape is assigned a default name when you add it to the **Shapes** collection. To give the shape a more meaningful name, use the **Name** property. The following example adds a rectangle to myDocument, gives it the name Red Square, and then sets its foreground color and line style.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(Type:=msoShapeRectangle, _
    Top:=144, Left:=144, Width:=72, Height:=72)
    .Name = "Red Square"
    .Fill.ForeColor.RGB = RGB(255, 0, 0)
    .Line.DashStyle = msoLineDashDot
End With
```

Returning a Newly Created Shape on a Slide

To add a shape to a slide and return a **Shape** object that represents the newly created shape, use one of the following methods of the **Shapes** collection:

[AddCallout](#), [AddComment](#), [AddConnector](#), [AddCurve](#), [AddLabel](#),
[AddLine](#), [AddMediaObject](#), [AddOLEObject](#), [AddPicture](#), [AddPlaceholder](#),
[AddPolyline](#), [AddShape](#), [AddTable](#), [AddTextbox](#), [AddTextEffect](#), [AddTitle](#).

Returning a Shape Within the Selection

Use **Selection.ShapeRange(*index*)**, where *index* is the shape name or the index number, to return a **Shape** object that represents a shape within the selection. The following example sets the fill for the first shape in the selection in the active window, assuming that there's at least one shape in the selection.

```
ActiveWindow.Selection.ShapeRange(1).Fill _  
    .ForeColor.RGB = RGB(255, 0, 0)
```

Returning the Slide Title and Other Placeholders on a Slide

Use **Shapes.Title** to return a **Shape** object that represents an existing slide title. Use **Shapes.AddTitle** to add a title to a slide that doesn't already have one and return a **Shape** object that represents the newly created title. Use **Shapes.Placeholders(index)**, where *index* is the placeholder's index number, to return a **Shape** object that represents a placeholder. If you have not changed the layering order of the shapes on a slide, the following three statements are equivalent, assuming that slide one has a title.

```
ActivePresentation.Slides(1).Shapes.Title _  
    .TextFrame.TextRange.Font.Italic = True  
ActivePresentation.Slides(1).Shapes.Placeholders(1) _  
    .TextFrame.TextRange.Font.Italic = True  
ActivePresentation.Slides(1).Shapes(1).TextFrame _  
    .TextRange.Font.Italic = True
```

Returning the Shapes Attached to the Ends of a Connector

To return a **Shape** object that represents one of the shapes attached by a connector, use the [BeginConnectedShape](#) or [EndConnectedShape](#) property.

Returning the Default Shape for a Presentation

To return a **Shape** object that represents the [default shape](#) for a presentation, use the [DefaultShape](#) property.

Returning a newly created freeform

Use the [BuildFreeform](#) and [AddNodes](#) methods to define the geometry of a new freeform, and use the [ConvertToShape](#) method to create the freeform and return the **Shape** object that represents it.

Returning a Single Shape from Within a Group

Use **GroupItems**(*index*), where *index* is the shape name or the index number within the group, to return a **Shape** object that represents a single shape in a grouped shape.

Returning a Newly Formed Group of Shapes

Use the [Group](#) or [Regroup](#) method to group a range of shapes and return a single **Shape** object that represents the newly formed group. After a group has been formed, you can work with the group the same way you work with any other shape.



ShapeNode Object

[ShapeNodes](#) └ [ShapeNode](#)

Represents the geometry and the geometry-editing properties of the nodes in a user-defined freeform. Nodes include the vertices between the segments of the freeform and the control points for curved segments. The **ShapeNode** object is a member of the [ShapeNodes](#) collection. The **ShapeNodes** collection contains all the nodes in a freeform.

Using the ShapeNode Object

Use **Nodes**(*index*), where *index* is the node index number, to return a single **ShapeNode** object. If node one in shape three on myDocument is a corner point, the following example makes it a smooth point. For this example to work, shape three must be a freeform.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Nodes(1).EditingType = msoEditingCorner Then
        .Nodes.SetEditingType 1, msoEditingSmooth
    End If
End With
```



ShapeRange Collection Object

Multiple objects [ShapeRange](#)
└─ Multiple objects

Represents a shape range, which is a set of shapes on a document. A shape range can contain as few as a single shape or as many as all the shapes on the document. You can include whichever shapes you want— chosen from among all the shapes on the document or all the shapes in the selection— to construct a shape range. For example, you could construct a **ShapeRange** collection that contains the first three shapes on a document, all the selected shapes on a document, or all the freeforms on a document.

For an overview of how to work with either a single shape or with more than one shape at a time, see [Working with Shapes \(Drawing Objects\)](#) .

Using the ShapeRange Collection

This section describes how to:

- Return a set of shapes you specify by name or index number.
- Return all or some of the selected shapes on a document.

Returning a Set of Shapes You Specify by Name or Index Number

Use **Shapes.Range**(*index*), where *index* is the name or index number of the shape or an array that contains either names or index numbers of shapes, to return a **ShapeRange** collection that represents a set of shapes on a document. You can use the **Array** function to construct an array of names or index numbers. The following example sets the fill pattern for shapes one and three on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.Range(Array(1, 3)).Fill _
    .Patterned msoPatternHorizontalBrick
```

The following example sets the fill pattern for the shapes named "Oval 4" and "Rectangle 5" on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
Set myRange = myDocument.Shapes _
    .Range(Array("Oval 4", "Rectangle 5"))
myRange.Fill.Patterned msoPatternHorizontalBrick
```

Although you can use the [Range](#) method to return any number of shapes or slides, it's simpler to use the [Item](#) method if you want to return only a single member of the collection. For example, `Shapes(1)` is simpler than `Shapes.Range(1)`.

Returning All or Some of the Selected Shapes on a Document

Use the [ShapeRange](#) property of the **Selection** object to return all the shapes in the selection. The following example sets the fill foreground color for all the shapes in the selection in window one, assuming that there's at least one shape in the selection.

```
Windows(1).Selection.ShapeRange.Fill.ForeColor _  
    .RGB = RGB(255, 0, 255)
```

Use **Selection.ShapeRange(index)**, where *index* is the shape name or the index number, to return a single shape within the selection. The following example sets the fill foreground color for shape two in the collection of selected shapes in window one, assuming that there are at least two shapes in the selection.

```
Windows(1).Selection.ShapeRange(2).Fill.ForeColor _  
    .RGB = RGB(255, 0, 255)
```



Slide Object

Multiple objects [SlideRange](#)
└─ [Slide](#)
└─ Multiple objects

Represents a slide. The [Slides](#) collection contains all the **Slide** objects in a presentation.

Note Don't be confused if you're trying to return a reference to a single slide but you end up with a [SlideRange](#) object. A single slide can be represented either by a **Slide** object or by a [SlideRange](#) collection that contains only one slide, depending on how you return a reference to the slide. For example, if you create and return a reference to a slide by using the [Add](#) method, the slide is represented by a **Slide** object. However, if you create and return a reference to a slide by using the [Duplicate](#) method, the slide is represented by a **SlideRange** collection that contains a single slide. Because all the properties and methods that apply to a **Slide** object also apply to a **SlideRange** collection that contains a single slide, you can work with the returned slide in the same way, regardless of whether it's represented by a **Slide** object or a **SlideRange** collection.

Using the Slide Object

This section describes how to:

- Return a slide that you specify by name, index number, or slide ID number
- Return a slide in the selection
- Return the slide that's currently displayed in any document window or slide show window you specify
- Create a new slide

Returning a slide that you specify by name, index number, or slide ID number

Use **Slides**(*index*), where *index* is the slide name or index number, or use **Slides.FindBySlideID**(*index*), where *index* is the slide ID number, to return a single **Slide** object. The following example sets the layout for slide one in the active presentation.

```
ActivePresentation.Slides(1).Layout = ppLayoutTitle
```

The following example sets the layout for the slide with the ID number 265.

```
ActivePresentation.Slides.FindBySlideID(265).Layout = ppLayoutTitle
```

Returning a slide in the selection

Use **Selection.SlideRange**(*index*), where *index* is the slide name or index number within the selection, to return a single **Slide** object. The following example sets the layout for slide one in the selection in the active window, assuming that there's at least one slide selected.

```
ActiveWindow.Selection.SlideRange(1).Layout = ppLayoutTitle
```

If there's only one slide selected, you can use **Selection.SlideRange** to return a **SlideRange** collection that contains the selected slide. The following example sets the layout for slide one in the current selection in the active window, assuming that there's exactly one slide selected.

```
ActiveWindow.Selection.SlideRange.Layout = ppLayoutTitle
```

Returning the slide that's currently displayed in any document window or slide show window you specify

Use the **Slide** property to return the slide that's currently displayed in the specified document window or slide show window view. The following example copies the slide that's currently displayed in document window two to the Clipboard.

```
Windows(2).View.Slide.Copy
```

Creating a new slide

Use the **Add** method to create a new slide and add it to the presentation. The following example adds a title slide to the beginning of the active presentation.

```
ActivePresentation.Slides.Add 1, ppLayoutTitleOnly
```



SlideShowSettings Object

[Presentation](#) └ [SlideShowSettings](#)

└ Multiple objects

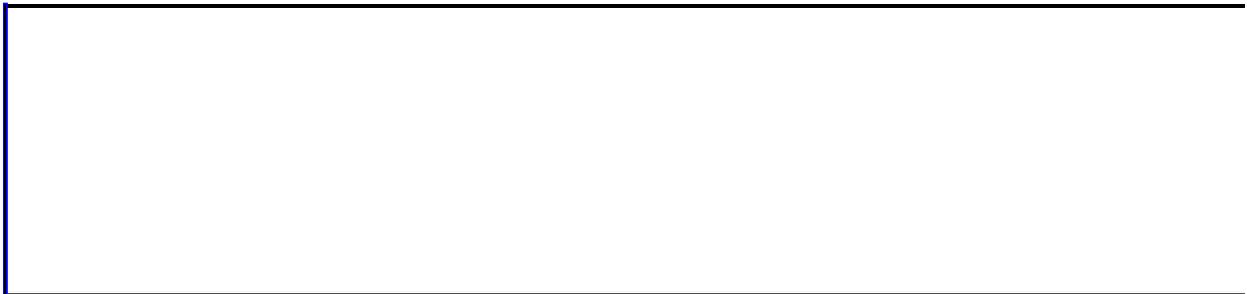
Represents the slide show setup for a presentation.

Using the SlideShowSettings Object

Use the [SlideShowSettings](#) property to return the **SlideShowSettings** object. The first section in the following example sets all the slides in the active presentation to advance automatically after five seconds. The second section sets the slide show to start on slide two, end on slide four, advance slides by using the timings set in the first section, and run in a continuous loop until the user presses ESC. Finally, the example runs the slide show.

```
For Each s In ActivePresentation.Slides
    With s.SlideShowTransition
        .AdvanceOnTime = True
        .AdvanceTime = 5
    End With
Next

With ActivePresentation.SlideShowSettings
    .RangeType = ppShowSlideRange
    .StartingSlide = 2
    .EndingSlide = 4
    .AdvanceMode = ppSlideShowUseSlideTimings
    .LoopUntilStopped = True
    .Run
End With
```



SlideShowTransition Object

Multiple objects [SlideShowTransition](#)
[SoundEffect](#)

Contains information about how the specified slide advances during a slide show.

Using the SlideShowTransition Object

Use the [SlideShowTransition](#) property to return the **SlideShowTransition** object. The following example specifies a Fast Strips Down-Left transition accompanied by the Bass.wav sound for slide one in the active presentation and specifies that the slide advance automatically five seconds after the previous animation or slide transition.

```
With ActivePresentation.Slides(1).SlideShowTransition
    .Speed = ppTransitionSpeedFast
    .EntryEffect = ppEffectStripsDownLeft
    .SoundEffect.ImportFromFile "c:\sndsys\bass.wav"
    .AdvanceOnTime = True
    .AdvanceTime = 5
End With
ActivePresentation.SlideShowSettings.AdvanceMode = _
    ppSlideShowUseSlideTimings
```



[Show All](#)

SlideShowView Object

[SlideShowWindow](#) └ [SlideShowView](#)

└ Multiple objects

Represents the [view](#) in a slide show window.

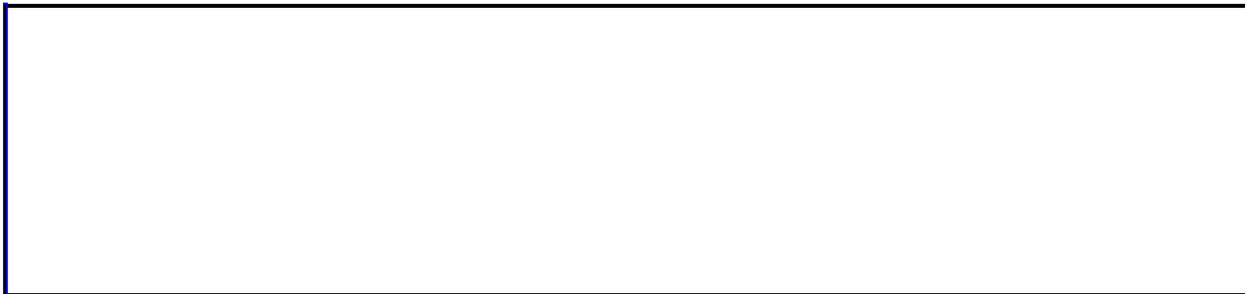
Using the SlideShowView Object

Use the [View](#) property of the **SlideShowWindow** object to return the **SlideShowView** object. The following example sets slide show window one to display the first slide in the presentation.

```
SlideShowWindows(1).View.First
```

Use the [Run](#) method of the **SlideShowSettings** object to create a **SlideShowWindow** object, and then use the **View** property to return the **SlideShowView** object the window contains. The following example runs a slide show of the active presentation, changes the pointer to a pen, and sets the pen color for the slide show to red.

```
With ActivePresentation.SlideShowSettings.Run.View  
    .PointerColor.RGB = RGB(255, 0, 0)  
    .PointerType = ppSlideShowPointerPen  
End With
```



SlideShowWindow Object

[Application](#) └ [SlideShowWindows](#)
└ [SlideShowWindow](#)
└ Multiple objects

Represents a window in which a slide show runs.

Using the SlideShowWindow Object

Use **SlideShowWindows**(*index*), where *index* is the slide show window index number, to return a single **SlideShowWindow** object. The following example activates slide show window two.

```
SlideShowWindows(2).Activate
```

Use the [Run](#) method to create a new slide show window and return a reference to this slide show window. The following example runs a slide show of the active presentation and reduces the height of the slide show window just enough so that you can see the taskbar (for screens with a resolution of 800 by 600).

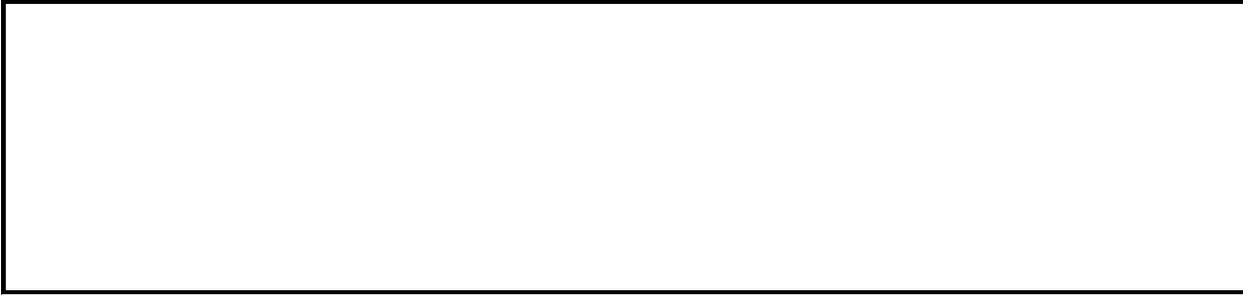
```
With ActivePresentation.SlideShowSettings
    .ShowType = ppShowTypeSpeaker
    With .Run
        .Height = 300
        .Width = 400
    End With
End With
```

Use the [View](#) property to return the view in the specified slide show window. The following example sets the view in slide show window one to display slide three in the presentation.

```
SlideShowWindows(1).View.GotoSlide 3
```

Use the [Presentation](#) property to return the presentation that's currently running in the specified slide show window. The following example displays the name of the presentation that's currently running in slide show window one.

```
MsgBox SlideShowWindows(1).Presentation.Name
```



SoundEffect Object

Multiple objects [↳ SoundEffect](#)

Represents the sound effect that accompanies an animation or slide transition in a slide show.

Using the SoundEffect Object

Use the [SoundEffect](#) property of the **AnimationSettings** object to return the **SoundEffect** object that represents the sound effect that accompanies an animation. The following example specifies that the animation of the title on slide one in the active presentation be accompanied by the sound in the Bass.wav file.

```
With ActivePresentation.Slides(1).Shapes(1).AnimationSettings
    .TextLevelEffect = ppAnimateByAllLevels
    .SoundEffect.ImportFromFile "c:\sndsys\bass.wav"
End With
```

Use the **SoundEffect** property of the **SlideShowTransition** object to return the **SoundEffect** object that represents the sound effect that accompanies a slide transition.

The following example specifies that the transition to slide one in the active presentation be accompanied by the sound in the Bass.wav file.

```
ActivePresentation.Slides(1).SlideShowTransition.SoundEffect _
    .ImportFromFile "c:\sndsys\bass.wav"
```

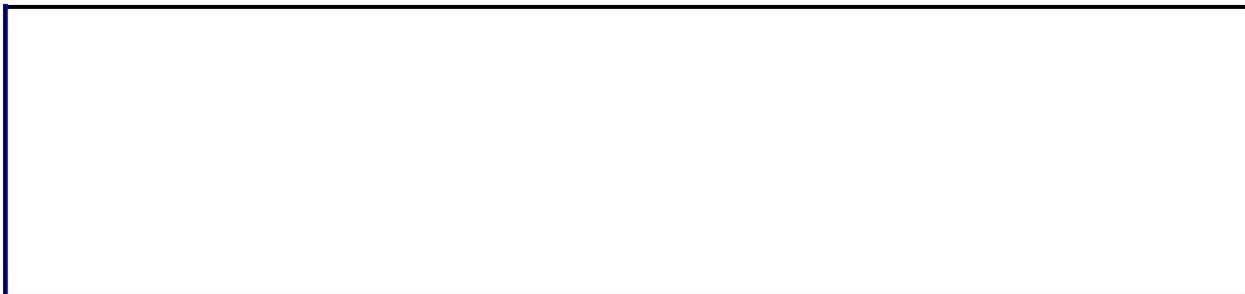


Table Object

Multiple objects [Table](#)
└─ Multiple objects

Represents a table shape on a slide. The **Table** object is a member of the **Shapes** collection. The **Table** object contains the [Columns](#) collection and the [Rows](#) collection.

Using the Table Object

Use **Shapes**(*index*), where *index* is a number, to return a shape containing a table. Use the [HasTable](#) property to see if a shape contains a table. This example walks through the shapes on slide one, checks to see if each shape has a table, and then sets the mouse click action for each table shape to advance to the next slide.

```
With ActivePresentation.Slides(2).Shapes
    For i = 1 To .Count
        If .Item(i).HasTable Then
            .Item(i).ActionSettings(ppMouseClicked) _
                .Action = ppActionNextSlide
        End If
    Next
End With
```

Use the [Cell](#) method of the **Table** object to access the contents of each cell. This example inserts the text "Cell 1" in the first cell of the table in shape five on slide three.

```
ActivePresentation.Slides(3).Shapes(5).Table _
    .Cell(1, 1).Shape.TextFrame.TextRange _
    .Text = "Cell 1"
```

Use the [AddTable](#) method to add a table to a slide. This example adds a 3x3 table on slide two in the active presentation.

```
ActivePresentation.Slides(2).Shapes.AddTable(3, 3)
```



TabStop Object

[TabStops](#) | [TabStop](#)

Represents a single tab stop. The **TabStop** object is a member of the [TabStops](#) collection. The **TabStops** collection represents all the tab stops on one ruler.

Using The Tabstop Object

Use **TabStops**(*index*), where *index* is the tab stop index number, to return a single **TabStop** object. The following example clears tab stop one for the text in shape two on slide one in the active presentation.

```
ActivePresentation.Slides(1).Shapes(2).TextFrame _  
    .Ruler.TabStops(1).Clear
```



Tags Object

Multiple objects [Tags](#)

Represents a tag or a custom property that you can create for a shape, slide, or presentation. Each **Tags** object contains the name of a custom property and a value for that property.

Create tags when you want to be able to selectively work with specific members of a collection, based on an attribute that isn't already represented by a built-in property. For example, if you want to be able to categorize slides in a presentation based on what region of the country/region they apply to, you could create a Region tag and assign a Region value to each slide in the presentation. You could then selectively perform an operation on some of the slides, based on the values of their Region tags, such as hiding all the slides with the Region value "East."

Using the Tags Object

Use the [Add](#) method to add a tag to an object. The following example adds a tag with the name "Region" and with the value "East" to slide one in the active presentation.

```
ActivePresentation.Slides(1).Tags.Add "Region", "East"
```

Use **Tags(index)**, where *index* is the name of a tag, to return a the tag value. The following example tests the value of the Region tag for all slides in the active presentation and hides any slides that don't pertain to the East Coast (denoted by the value "East").

```
For Each s In ActivePresentation.Slides
    If s.Tags("region") <> "east" Then
        s.SlideShowTransition.Hidden = True
    End If
Next
```



TextEffectFormat Object

Multiple objects [↳TextEffectFormat](#)

Contains properties and methods that apply to WordArt objects.

Using the TextEffectFormat Object

Use the **TextEffect** property to return a **TextEffectFormat** object. The following example sets the font name and formatting for shape one on myDocument. For this example to work, shape one must be a WordArt object.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).TextEffect
    .FontName = "Courier New"
    .FontBold = True
    .FontItalic = True
End With
```



[Show All](#)

TextFrame Object

Multiple objects [TextFrame](#)
└─Multiple objects

Represents the [text frame](#) in a **Shape** object. Contains the text in the text frame as well as the properties and methods that control the alignment and anchoring of the text frame.

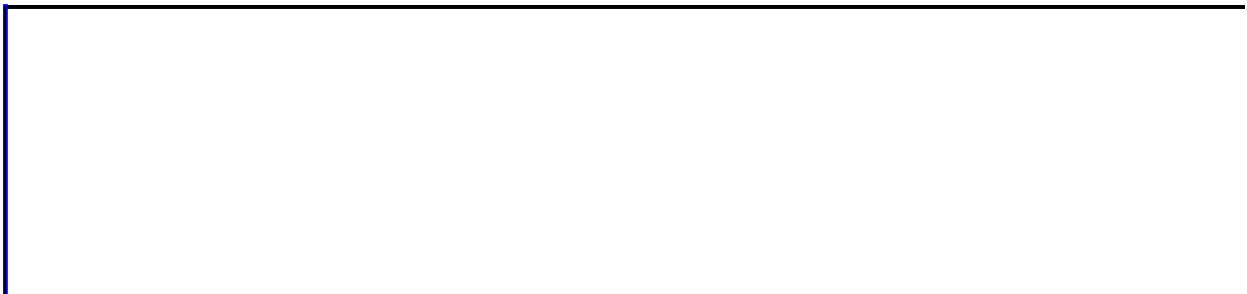
Using the TextFrame Object

Use the **TextFrame** property to return a **TextFrame** object. The following example adds a rectangle to myDocument, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 0, 0, 250, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 10
    .MarginLeft = 10
    .MarginRight = 10
    .MarginTop = 10
End With
```

Use the [HasTextFrame](#) property to determine whether a shape has a text frame, and use the [HasText](#) property to determine whether the text frame contains text, as shown in the following example.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.HasTextFrame Then
        With s.TextFrame
            If .HasText Then MsgBox .TextRange.Text
        End With
    End If
Next
```



TextRange Object

Multiple objects [↳ TextRange](#)
↳ Multiple objects

Contains the text that's attached to a shape, as well as properties and methods for manipulating the text.

Using the TextRange Object

This section describes how to:

- Return the text range in any shape you specify.
- Return a text range from the selection.
- Return particular characters, words, lines, sentences, or paragraphs from a text range.
- Find and replace text in a text range.
- Insert text, the date and time, or the slide number into a text range.
- Position the insertion point wherever you want in a text range.

Returning a Text Range from Any Shape You Specify

Use the [TextRange](#) property of the [TextFrame](#) object to return a **TextRange** object for any shape you specify. Use the [Text](#) property to return the string of text in the **TextRange** object. The following example adds a rectangle to myDocument and sets the text it contains.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250, 140) _
    .TextFrame.TextRange.Text = "Here is some test text"
```

Because the **Text** property is the default property of the **TextRange** object, the following two statements are equivalent.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame _
    .TextRange.Text = "Here is some test text"
ActivePresentation.Slides(1).Shapes(1).TextFrame _
    .TextRange = "Here is some test text"
```

Use the [HasTextFrame](#) property to determine whether a shape has a text frame, and use the [HasText](#) property to determine whether the text frame contains text.

Returning a Text Range from the Selection

Use the **TextRange** property of the **Selection** object to return the currently selected text. The following example copies the selection to the Clipboard.

```
ActiveWindow.Selection.TextRange.Copy
```

Returning Particular Characters, Words, Lines, Sentences, or Paragraphs from a Text Range

Use one of the following methods to return a portion of the text of a **TextRange** object: [Characters](#), [Lines](#), [Paragraphs](#), [Runs](#), [Sentences](#), or [Words](#).

Finding and Replacing Text in a Text Range

Use the [Find](#) and [Replace](#) methods to find and replace text in a text range.

Inserting Text, the Date and Time, or the Slide Number into a Text Range

Use one of the following methods to insert characters into a **TextRange** object: [InsertAfter](#), [InsertBefore](#), [InsertDateTime](#), [InsertSlideNumber](#), or [InsertSymbol](#).



TextStyle Object

[TextStyles](#) └ [TextStyle](#)
└ Multiple objects

Represents one of three text styles: title text, body text, or default text. Each text style contains a [TextFrame](#) object that describes how text is placed within the text bounding box, a [Ruler](#) object that contains tab stops and outline indent formatting information, and a [TextStyleLevels](#) collection that contains outline text formatting information. The **TextStyle** object is a member of the [TextStyles](#) collection.

Using the TextStyle Object

Use **TextStyles**(*index*), where *index* is either **ppBodyStyle**, **ppDefaultStyle**, or **ppTitleStyle**, to return a single **TextStyle** object. The following example sets the font name and font size for level-one body text on all the slides in the active presentation.

```
With ActivePresentation.SlideMaster _  
    .TextStyles(ppBodyStyle).Levels(1)  
    With .Font  
        .Name = "Arial"  
        .Size = 36  
    End With  
End With
```



TextStyleLevel Object

[TextStyleLevels](#) ↳ [TextStyleLevel](#)

↳ Multiple objects

Contains character and paragraph formatting information for an outline level. The **TextStyleLevel** object is a member of the [TextStyleLevels](#) collection. The **TextStyleLevels** collection contains one **TextStyleLevel** object for each of the five outline levels

Using the TextStyleLevel Object

Use **Levels(index)**, where *index* is a number from 1 through 5 that corresponds to the outline level, to return a single **TextStyleLevel** object. The following example sets the font name and font size, the space before paragraphs, and the paragraph alignment for level-one body text on all the slides in the active presentation.

```
With ActivePresentation.SlideMaster _  
    .TextStyles(ppBodyStyle).Levels(1)  
    With .Font  
        .Name = "Arial"  
        .Size = 36  
    End With  
    With .ParagraphFormat  
        .LineRuleBefore = False  
        .SpaceBefore = 14  
        .Alignment = ppAlignJustify  
    End With  
End With
```



ThreeDFormat Object

Multiple objects [└ThreeDFormat](#)
[└ColorFormat](#)

Represents a shape's three-dimensional formatting.

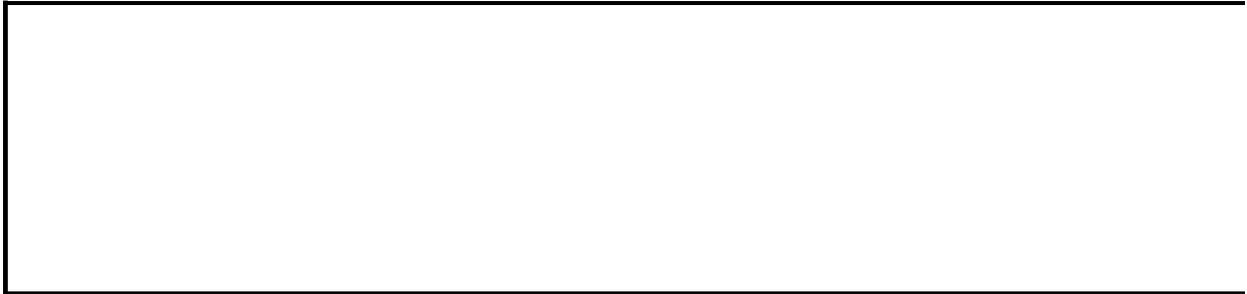
Using The ThreeDFormat Object

Use the **ThreeD** property to return a **ThreeDFormat** object. The following example adds an oval to myDocument and then specifies that the oval be extruded to a depth of 50 points and that the extrusion be purple.

```
Set myDocument = ActivePresentation.Slides(1)
Set myShape = myDocument.Shapes _
    .AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    'RGB value for purple
    .ExtrusionColor.RGB = RGB(255, 100, 255)
End With
```

Remarks

You cannot apply three-dimensional formatting to some kinds of shapes, such as beveled shapes or multiple-disjoint paths. Most of the properties and methods of the **ThreeDFormat** object for such a shape will fail.



TimeLine Object

Multiple objects [TimeLine](#)
└ Multiple objects

Stores animation information for a **Master**, **Slide** or **SlideRange** object.

Using the TimeLine object

Use the [TimeLine](#) property of the [Master](#), [Slide](#), or [SlideRange](#) object to return a **TimeLine** object.

The **TimeLine** object's [MainSequence](#) property gains access to the main animation sequence, while the [InteractiveSequences](#) property gains access to the collection of interactive animation sequences of a slide or slide range. To reference a timeline object, use syntax similar to these code examples:

```
ActivePresentation.Slides(1).TimeLine.MainSequence  
ActivePresentation.SlideMaster.TimeLine.InteractiveSequences  
ActiveWindow.Selection.SlideRange.TimeLine.InteractiveSequences
```



Timing Object

Multiple objects [Timing](#)
[Shape](#)

Represents timing properties for an animation effect.

Using the Timing object

To return a **Timing** object, use the **Timing** property of the **AnimationBehavior** or **Effect** object. The following example sets timing duration information for the main animation.

```
ActiveWindow.Selection.SlideRange(1).TimeLine _  
    .MainSequence(1).Timing.Duration = 5
```

Use the following read/write properties of the **Timing** object to manipulate animation timing effects.

Use this...	To change this...
Accelerate	Percentage of the duration over which acceleration should take place
AutoReverse	Whether an effect should play forward and then reverse, thereby doubling the duration
Decelerate	Percentage of the duration over which acceleration should take place
Duration	Length of animation (in seconds)
RepeatCount	Number of times to repeat the animation
RepeatDuration	How long should the repeats last (in seconds)
Restart	Restart behavior of an animation node
RewindAtEnd	Whether an objects return to its beginning position after an effect has ended
SmoothStart	Whether an effect accelerates when it starts
SmoothEnd	Whether an effect decelerates when it ends
TriggerDelayTime	Delay time from when the trigger is enabled (in seconds)
TriggerShape	Which shape is associated with the timing effect
TriggerType	How the timing effect is triggered



[Show All](#)

View Object

[DocumentWindow](#) └ [View](#)
└ [PrintOptions](#)

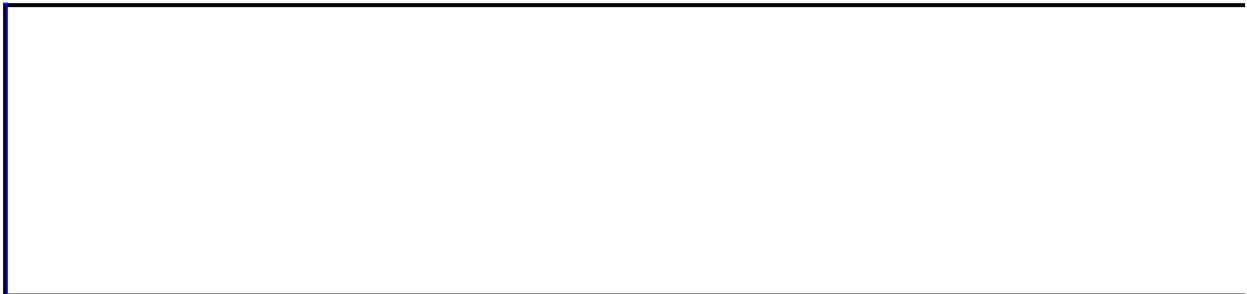
Represents the current editing [view](#) in the specified document window.

Using the View Object

Use the **View** property of the **DocumentWindow** object to return the **View** object. The following example sets the size of window one and then sets the zoom to fit the new window size.

```
With Windows(1)
    .Height = 200
    .Width = 250
    .View.ZoomToFit = True
End With
```

Note The **View** object can represent any of the document window views: normal view, slide view, outline view, slide sorter view, notes page view, slide master view, handout master view, or notes master view. Some properties and methods of the **View** object work only in certain views. If you try to use a property or method that's inappropriate for a **View** object, an error occurs.



WebOptions Object

[Presentation](#) | [WebOptions](#)

Contains presentation-level attributes used by Microsoft PowerPoint when you save or publish a complete or partial presentation as a Web page or open a Web page. You can return or set attributes either at the application (global) level or at the presentation level. (Note that attribute values can be different from one presentation to another, depending on the attribute value at the time the presentation was saved.) Presentation-level attribute settings override application-level attribute settings. Application-level attributes are contained in the [DefaultWebOptions](#) object.

Using the WebOptions Object

Use the [WebOptions](#) property to return the **WebOptions** object. The following example checks to see whether Portable Network Graphics (PNG) is allowed as an image format for presentation one. If PNG is allowed, it sets the text color for the outline pane to white and the background color for the outline and slide panes to black.

```
Set objAppWebOptions = Presentations(1).WebOptions
With objAppWebOptions
    If .AllowPNG = True Then
        .FrameColors = ppFrameColorsWhiteTextOnBlack
    End If
End With
```



Activate Method

Activates the specified object.

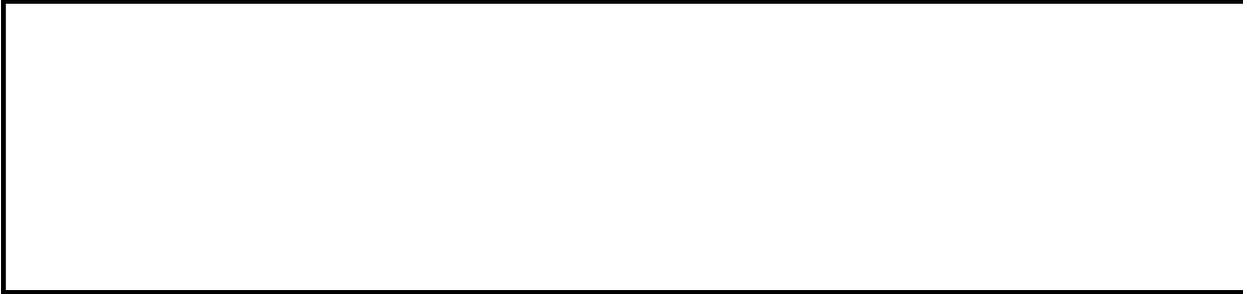
expression.**Activate**

expression Required. An expression that returns a **DocumentWindow**, **Pane**, **OLEFormat**, **Application**, or **SlideShowWindow** object.

Example

This example activates the document window immediately following the active window in the document window order.

```
Windows(2).Activate
```



[Show All](#)

Add Method

[Add method as it applies to the **AddIns** object.](#)

Returns an [AddIn](#) object that represents an add-in file added to the list of add-ins.

expression.**Add**(*Filename*)

expression Required. An expression that returns an [AddIns](#) object.

Filename Required **String**. The full name of the file (including the path and file name extension) that contains the add-in you want to add to the list of add-ins.

Remarks

This method doesn't load the new add-in. You must set the [Loaded](#) property to load the add-in.

[Add method as it applies to the **AnimationBehaviors** object.](#)

Returns an [AnimationBehavior](#) object that represents a new animation behavior.

expression.**Add**(*Type*, *Index*)

expression Required. An expression that returns an [AnimationBehaviors](#) object.

Type Required [MsoAnimType](#). The behavior of the animation.

MsoAnimType can be one of these MsoAnimType constants.

msoAnimTypeColor

msoAnimTypeMixed

msoAnimTypeMotion

msoAnimTypeNone

msoAnimTypeProperty

msoAnimTypeRotation

msoAnimTypeScale

Index Optional **Long**. The placement of the animation in relation to other animation behaviors. The default value is -1 which means that if the **Index** argument is omitted, the new animation behavior is added to the end of existing animation behaviors.

[Add method as it applies to the **AnimationPoints** and **Sequences** objects.](#)

Returns an [AnimationPoint](#) or [Sequence](#) object that represents a new animation point or sequence.

expression.**Add(*Index*)**

expression Required. An expression that returns one of the above objects.

Index Optional **Long**. The position of the animation point or sequence in relation to other animation points or sequences. The default value is -1 which means that if the **Index** argument is omitted, the new animation point or sequence is added to the end of existing animation points or sequence.



[Add method as it applies to the **ColorSchemes** object.](#)

Adds a color scheme to the collection of available schemes. Returns a **ColorScheme** object that represents the added color scheme.

expression.**Add(*Scheme*)**

expression Required. An expression that returns a **ColorSchemes** object.

Scheme Optional **ColorScheme** object. The color scheme to add. Can be a **ColorScheme** object from any slide or master, or an item in the **ColorSchemes** collection from any open presentation. If this argument is omitted, the first **ColorScheme** object (the first standard color scheme) in the specified presentation's **ColorSchemes** collection is used.

Remarks

The new color scheme is based on the colors used on the specified slide or master or on the colors in the specified color scheme from an open presentation.

The **ColorSchemes** collection can contain up to 16 color schemes. If you need to add another color scheme and the **ColorSchemes** collection is already full, use the **Delete** method to remove an existing color scheme.

Note that although Microsoft PowerPoint automatically checks whether a color scheme is a duplicate before adding it through the user interface, it doesn't check before adding a color scheme through a Visual Basic procedure. Your procedure must do its own checking to avoid adding redundant color schemes.

[Add method as it applies to the **Columns** object.](#)

Adds a new column to an existing table. Returns a **Column** object that represents the new table column.

expression.**Add(BeforeColumn)**

expression Required. An expression that returns a **Columns** object.

BeforeColumn Optional **Long**. The index number specifying the table column before which the new column will be inserted. This argument must be a **Long** from 1 to the number of columns in the table. The default value is -1 which means that if the **BeforeColumn** argument is omitted, then the new column is added as the last column in the table.

[Add method as it applies to the **Comments** object.](#)

Returns a **Comment** object that represents a new comment added to a slide.

expression.**Add(Left, Top, Author, AuthorInitials, Text)**

expression Required. An expression that returns a **Comments** object.

Left Required **Single**. The position, measured in points, of the left edge of the

comment, relative to the left edge of the presentation.

Top Required **Single**. The position, measured in points, of the top edge of the comment, relative to the top edge of the presentation.

Author Required **String**. The author of the comment.

AuthorInitials Required **String**. The author's initials.

Text Required **String**. The comment's text.



[Add method as it applies to the **Designs** object.](#)

Returns a **Design** object that represents a new slide design.

expression.Add(designName, Index)

expression Required. An expression that returns a **Designs** object.

designName Required **String**. The name of the design.

Index Optional **Integer**. The index number of the design. The default value is -1 which means that if the **Index** argument is omitted, then the new slide design is added at the end of existing slide designs.



[Add method as it applies to the **ExtraColors** object.](#)

Adds a color to the [extra colors](#) available to a presentation if the color hasn't already been added.

expression.Add(Type)

expression Required. An expression that returns an **ExtraColors** object.

Type Required **MsoRGBType**. The [red-green-blue \(RGB\) value](#) of the color to be added.



[Add method as it applies to the **NamedSlideShows** object.](#)

Creates a new named slide show and adds it to the collection of named slide

shows in the specified presentation. Returns a [NamedSlideShow](#) object that represents the new named slide show.

expression.**Add**(*Name*, *SafeArrayOfSlideIDs*)

expression Required. An expression that returns a [NamedSlideShows](#) object.

Name Required **String**. The name of the slide show.

SafeArrayOfSlideIDs Required **Variant**. Contains the unique slide IDs of the slides to be displayed in a slide show.

Remarks

The name you specify when you add a named slide show is the name you use as an argument to the [Run](#) method to run the named slide show.

[Add method as it applies to the **Presentations** object.](#)

Creates a presentation. Returns a [Presentation](#) object that represents the new presentation.

expression.**Add**(*WithWindow*)

expression Required. An expression that returns a [Presentations](#) collection.

WithWindow Optional [MsoTriState](#). **MsoTrue** creates the presentation in a visible window.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The new presentation isn't visible.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Creates the presentation in a visible window.

[Add method as it applies to the **PrintRanges** object.](#)

Returns a [PrintRange](#) object that represents a consecutive run of slides to be printed.

expression.**Add**(*Start*, *End*)

expression Required. An expression that returns a [PrintRanges](#) object.

Start Required **Long**. The first slide in the range of slides to be printed. Must be less than or equal to the value of the **End** argument.

End Required **Long**. The last slide in the range of slides to be printed. Must be

greater than or equal to the value of the ***Start*** argument.

Remarks

The [RangeType](#) property of the [PrintOptions](#) object must be set to `ppPrintSlideRange` for the ranges in the [PrintRanges](#) collection to be applied.

If you don't want to print an entire presentation, add print ranges to specify which slides you want to print. You must add one print range for each consecutive run of slides to be printed. For example, if you want to print slide 1, slides 3 through 5, and slides 8 and 9, you must add three print range objects. For more information, see the example for this method.

Use the [ClearAll](#) method to clear previously-defined print ranges.



[Add method as it applies to the Rows object.](#)

Returns a [Row](#) object that represents a new row added to an existing table.

expression.**Add**(*BeforeRow*)

expression Required. An expression that returns a [Rows](#) object.

BeforeRow Optional **Long**. The index number specifying the table row before which the new row will be inserted. This argument must be an integer value from 1 to the number of rows in the table. The default value is -1 which means that if the **BeforeRow** argument is omitted, then the new row is added as the last row in the table.



[Add method as it applies to the Slides object.](#)

Creates a new slide and adds it to the collection of slides in the specified presentation. Returns a [Slide](#) object that represents the new slide.

expression.**Add**(*Index*, *Layout*)

expression Required. An expression that returns a [Slides](#) collection.

Index Required **Long**. The index number the new slide will have within the [Slides](#) collection. This value cannot exceed the number of existing slides + 1. If

set to 1, the new slide becomes the first slide in the presentation.

Layout Required [PpSlideLayout](#). The type of slide to create.

PpSlideLayout can be one of these PpSlideLayout constants.

ppLayoutBlank

ppLayoutChart

ppLayoutChartAndText

ppLayoutClipartAndText

ppLayoutClipArtAndVerticalText

ppLayoutFourObjects

ppLayoutLargeObject

ppLayoutMediaClipAndText

ppLayoutMixed

ppLayoutObject

ppLayoutObjectAndText

ppLayoutObjectOverText

ppLayoutOrgchart

ppLayoutTable

ppLayoutText

ppLayoutTextAndChart

ppLayoutTextAndClipart

ppLayoutTextAndMediaClip

ppLayoutTextAndObject

ppLayoutTextAndTwoObjects

ppLayoutTextOverObject

ppLayoutTitle

ppLayoutTitleOnly

ppLayoutTwoColumnText

ppLayoutTwoObjectsAndText

ppLayoutTwoObjectsOverText

ppLayoutVerticalText

ppLayoutVerticalTitleAndText

ppLayoutVerticalTitleAndTextOverChart

Remarks

To alter the layout of an existing slide, use the [Layout](#) property.

[Add method as it applies to the **TabStops** object.](#)

Adds a tab stop to the ruler for the specified text. Returns a [TabStop](#) object that represents the new tab stop.

expression.Add(Type, Position)

expression Required. An expression that returns a [TabStops](#) collection.

Type Required [PpTabStopType](#). Specifies the way text will be aligned with the new tab stop.

PpTabStopType can be one of these PpTabStopType constants.

ppTabStopCenter

ppTabStopDecimal

ppTabStopLeft

ppTabStopMixed

ppTabStopRight

Position Required **Single**. The position of the new tab stop, in points.

[Add method as it applies to the **Tags** object.](#)

Creates a tag for the specified object. If the tag already exists, this method replaces the existing tag value.

expression.Add(Name, Value)

expression Required. An expression that returns a [Tags](#) object.

Name Required **String**. The new tag name. Use "name" as the string for this argument to set the value of the name tag.

Value Required **String**. The new tag value.

Remarks

The **Tags** object contains a pair of strings— the tag name and the tag value— for each tag. Use the **Add** method to create a tag, and use the [Name](#) and [Value](#) methods to return a tag's name and value components.

Example

[As it applies to the **AddIns** object.](#)

This example adds MyTools.ppa to the list of add-ins.

```
Set myAddIn = Application.AddIns.Add(FileName:="c:\my documents\myto  
MsgBox myAddIn.Name & " has been added to the list"
```

[As it applies to the **ColorSchemes** object.](#)

This example adds a new color scheme to the collection of standard color schemes for the active presentation. The new color scheme is based on the colors used in slide two in the active presentation.

```
With ActivePresentation  
    Set newClrScheme = .Slides(2).ColorScheme  
    .ColorSchemes.Add Scheme:=newClrScheme  
End With
```

[As it applies to the **Columns** object.](#)

This example creates a new column before column one in the table represented by shape five on slide two. It then sets the width of the new column to 72 points (one inch).

```
With ActivePresentation.Slides(2).Shapes(5).Table  
    .Columns.Add(1).Width = 72  
End With
```

[As it applies to the **NamedSlideShows** object.](#)

This example adds to the active presentation a named slide show Quick Show that contains slides 2, 7, and 9. The example then runs this slide show.

```
Dim qSlides(1 To 3) As Long  
With ActivePresentation
```

```

With .Slides
    qSlides(1) = .Item(2).SlideID
    qSlides(2) = .Item(7).SlideID
    qSlides(3) = .Item(9).SlideID
End With
With .SlideShowSettings
    .NamedSlideShows.Add Name:="Quick Show", SafeArrayOfSlideIDs
    .RangeType = ppShowNamedSlideShow
    .SlideShowName = "Quick Show"
    .Run
End With
End With

```

 [As it applies to the **Presentations** object.](#)

This example creates a presentation, adds a slide to it, and then saves the presentation.

```

With Presentations.Add
    .Slides.Add Index:=1, Layout:=ppLayoutTitle
    .SaveAs "Sample"
End With

```

 [As it applies to the **PrintRanges** object.](#)

This example clears any previously defined print ranges and then prints slide 1, slides 3 through 5, and slides 8 and 9 in the active presentation.

```

With ActivePresentation.PrintOptions
    .RangeType = ppPrintSlideRange
    With .Ranges
        .ClearAll
        .Add Start:=1, End:=1
        .Add Start:=3, End:=5
        .Add Start:=8, End:=9
    End With
End With
ActivePresentation.PrintOut

```

 [As it applies to the **Rows** object.](#)

This example creates a row at the end of an existing table and sets the height of

the new row to 54 points (.75 inches).

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Rows.Add.Height = 54
End With
```

 [As it applies to the **Slides** object.](#)

This example adds a slide that contains a title placeholder at the beginning of the active presentation.

```
ActivePresentation.Slides.Add Index:=1, Layout:=ppLayoutTitleOnly
```

This example adds a blank slide at the end of the active presentation.

```
With ActivePresentation.Slides
    .Add Index:=.Count + 1, Layout:=ppLayoutBlank
End With
```

 [As it applies to the **TabStops** object.](#)

This example sets a left-aligned tab stop at 2 inches (144 points) for the text in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2).TextFrame _
    .Ruler.TabStops.Add Type:=ppTabStopLeft, Position:=144
```

 [As it applies to the **Tags** object.](#)

This example adds a tag named Priority and sets the value of the name tag for slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Tags
    'Sets value for name tag
    .Add Name:="Name", Value:="New Figures"
    'Adds "Priority" tag with value "Low"
    .Add Name:="Priority", Value:="Low"
End With
```

[As it applies to the **ExtraColors** object.](#)

This example adds an extra color to the active presentation (if the color hasn't already been added).

```
ActivePresentation.ExtraColors.Add RGB(Red:=69, Green:=32, Blue:=155
```



AddBaseline Method

Adds a base line to a presentation to allow tracking of changes for a later merge.

expression.**AddBaseline**(*FileName*)

expression Required. An expression that returns a [Presentation](#) object.

FileName Optional **String**. The full path of a file to use as the base line for this presentation. If **FileName** is not specified, then the presentation represented by *expression* is used as its own base line.

Remarks

This method generates an error if the presentation already has a baseline, or is a merged author document.

Example

The following line of code adds a base line to the active presentation.

```
Sub SetBaseline()  
    ActivePresentation.AddBaseline  
End Sub
```



[Show All](#)

AddCallout Method

Creates a borderless line callout. Returns a [Shape](#) object that represents the new callout.

expression.**AddCallout**(*Type*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Required [MsoCalloutType](#). The type of callout line.

MsoCalloutType can be one of these MsoCalloutType constants.

msoCalloutOne A single-segment callout line that can be either horizontal or vertical.

msoCalloutTwo A single-segment callout line that rotates freely.

msoCalloutThree A two-segment line.

msoCalloutFour A three-segment line.

Left Required **Single**. The position, measured in points, of the left edge of the callout's bounding box relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the callout's bounding box relative to the top edge of the slide.

Width Required **Single**. The width of the callout's bounding box, measured in points.

Height Required **Single**. The height of the callout's bounding box, measured in points.

Remarks

You can insert a greater variety of callouts by using the [AddShape](#) method.

Example

This example adds a borderless callout with a freely-rotating one-segment callout line to myDocument and then sets the callout angle to 30 degrees.

```
Sub NewCallout()  
    Dim sldOne As Slide  
    Set sldOne = ActivePresentation.Slides(1)  
    sldOne.Shapes.AddCallout(Type:=msoCalloutTwo, Left:=50, Top:=50,  
        Width:=200, Height:=100).Callout.Angle = msoCalloutAngle30  
End Sub
```



AddComment Method

Adds a comment. Returns a [Shape](#) object that represents the new comment.

expression.**AddComment**(*Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns a **Shapes** object.

Left , **Top** Optional **Single**. The position (in points) of the upper-left corner of the comment bounding box relative to the upper-left corner of the document. By default, the comment is placed in the upper-left corner of the document.

Width , **Height** Optional **Single**. The width and height of the comment, in points. By default, the comment is 100 points high and 100 points wide.

Example

This example adds a comment that contains the text "Test Comment" to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddComment(100, 100, 150, 150)
    .TextFrame.TextRange.Text = .TextFrame _
        .TextRange.Text + "Test Comment"
End With
```



[Show All](#)

AddConnector Method

Creates a connector. Returns a [Shape](#) object that represents the new connector. When a connector is added, it's not connected to anything. Use the [BeginConnect](#) and [EndConnect](#) methods to attach the beginning and end of a connector to other shapes in the document.

expression.AddConnector(*Type*, *BeginX*, *BeginY*, *EndX*, *EndY*)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Required [MsoConnectorType](#). The type of connector.

MsoConnectorType can be one of these MsoConnectorType constants.

msoConnectorCurve

msoConnectorElbow

msoConnectorStraight

msoConnectorTypeMixed

BeginX Required **Single**. The horizontal position, measured in points, of the connector's starting point relative to the left edge of the slide.

BeginY Required **Single**. The vertical position, measured in points, of the connector's starting point relative to the top edge of the slide.

EndX Required **Single**. The horizontal position, measured in points, of the connector's ending point relative to the left edge of the slide.

EndY Required **Single**. The vertical position, measured in points, of the connector's ending point relative to the top edge of the slide.

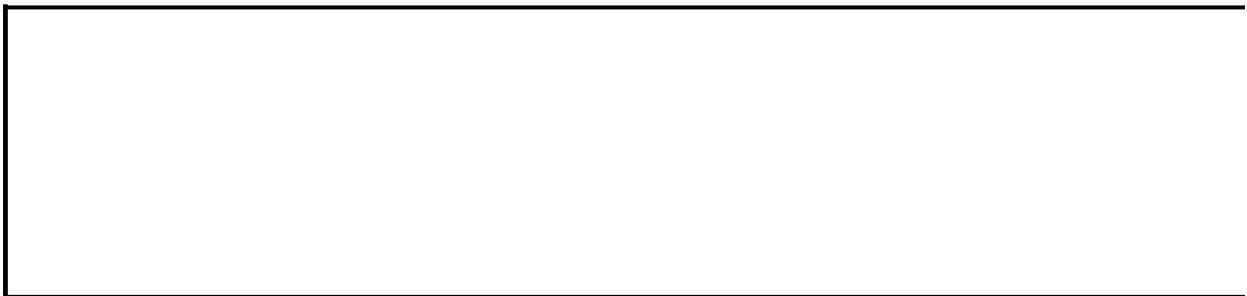
Remarks

When you attach a connector to a shape, the size and position of the connector are automatically adjusted, if necessary. Therefore, if you're going to attach a connector to other shapes, the position and dimensions you specify when adding the connector are irrelevant.

Example

This example adds two rectangles to myDocument and connects them with a curved connector. Note that when you attach the connector to the rectangles, the size and position of the connector are automatically adjusted; therefore, the position and dimensions you specify when adding the callout are irrelevant (dimensions must be nonzero).

```
Sub NewConnector()  
  
    Dim shpShapes As Shapes  
    Dim shpFirst As Shape  
    Dim shpSecond As Shape  
  
    Set shpShapes = ActivePresentation.Slides(1).Shapes  
    Set shpFirst = shpShapes.AddShape(Type:=msoShapeRectangle, _  
        Left:=100, Top:=50, Width:=200, Height:=100)  
    Set shpSecond = shpShapes.AddShape(Type:=msoShapeRectangle, _  
        Left:=300, Top:=300, Width:=200, Height:=100)  
    With shpShapes.AddConnector(Type:=msoConnectorCurve, BeginX:=0,  
        BeginY:=0, EndX:=100, EndY:=100).ConnectorFormat  
        .BeginConnect ConnectedShape:=shpFirst, ConnectionSite:=1  
        .EndConnect ConnectedShape:=shpSecond, ConnectionSite:=1  
        .Parent.RerouteConnections  
    End With  
  
End Sub
```



[Show All](#)

AddCurve Method

Creates a Bézier curve. Returns a [Shape](#) object that represents the new curve.

expression.**AddCurve**(*SafeArrayOfPoints*)

expression Required. An expression that returns one of the objects in the Applies To list.

SafeArrayOfPoints Required **Variant**. An array of [coordinate pairs](#) that specifies the vertices and control points of the curve. The first point you specify is the starting vertex, and the next two points are control points for the first Bézier segment. Then, for each additional segment of the curve, you specify a vertex and two control points. The last point you specify is the ending vertex for the curve. Note that you must always specify $3n + 1$ points, where n is the number of segments in the curve.

Example

The following example adds a two-segment Bézier curve to myDocument.

```
Dim pts(1 To 7, 1 To 2) As Single
pts(1, 1) = 0
pts(1, 2) = 0
pts(2, 1) = 72
pts(2, 2) = 72
pts(3, 1) = 100
pts(3, 2) = 40
pts(4, 1) = 20
pts(4, 2) = 50
pts(5, 1) = 90
pts(5, 2) = 120
pts(6, 1) = 60
pts(6, 2) = 30
pts(7, 1) = 150
pts(7, 2) = 90
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddCurve SafeArrayOfPoints:=pts
```



[Show All](#)

AddDiagram Method

Returns a [Shape](#) object that represents a diagram added to a slide, slide master, or slide range.

expression.**AddDiagram**(*Type*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Required [MsoDiagramType](#). The type of diagram.

MsoDiagramType can be one of these **MsoDiagramType** constants.

msoDiagramCycle Shows a process with a continuous cycle.

msoDiagramMixed Not used with this method.

msoDiagramOrgChart Shows hierarchical relationships.

msoDiagramPyramid Show foundation-based relationships.

msoDiagramRadial Shows relationships of a core element.

msoDiagramTarget Shows steps toward a goal.

msoDiagramVenn Shows areas of overlap between elements.

Left Required **Single**. The position, measured in points, of the left edge of the diagram canvas's bounding box, relative to the left edge of the page.

Top Required **Single**. The position, measured in points, of the top edge of the diagram canvas's bounding box, relative to the top edge of the page.

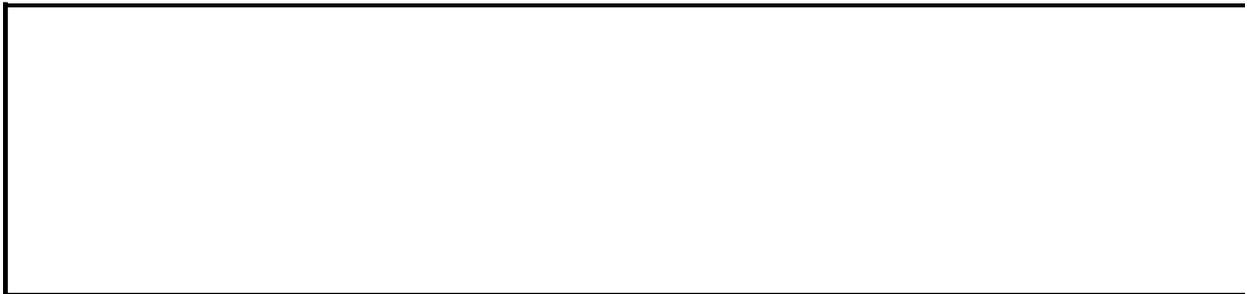
Width Required **Single**. The width, measured in points, of the diagram canvas's bounding box.

Height Required **Single**. The height, measured in points, of the diagram canvas's bounding box.

Example

The following example adds a pyramid diagram with four nodes to the first slide in the active presentation.

```
Sub CreatePyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three more child nodes to pyramid diagram  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
End Sub
```



[Show All](#)

AddEffect Method

Returns an [Effect](#) object that represents a new animation effect added to a sequence of animation effects.

expression.AddEffect(Shape, effectId, Level, trigger, Index)

expression Required. An expression that returns one of the objects in the Applies To list.

Shape Required [Shape](#) object. The shape to which the animation effect is added.

effectId Required [MsoAnimEffect](#). The animation effect to be applied.

MsoAnimEffect can be one of these **MsoAnimEffect** constants.

msoAnimEffectAppear

msoAnimEffectArcUp

msoAnimEffectAscend

msoAnimEffectBlast

msoAnimEffectBlinds

msoAnimEffectBoldFlash

msoAnimEffectBoldReveal

msoAnimEffectBoomerang

msoAnimEffectBounce

msoAnimEffectBox

msoAnimEffectBrushOnColor

msoAnimEffectBrushOnUnderline

msoAnimEffectCenterRevolve

msoAnimEffectChangeFillColor

msoAnimEffectChangeFont

msoAnimEffectChangeFontColor

msoAnimEffectChangeFontSize

msoAnimEffectChangeFontStyle

msoAnimEffectChangeLineColor
msoAnimEffectCheckerboard
msoAnimEffectCircle
msoAnimEffectColorBlend
msoAnimEffectColorReveal
msoAnimEffectColorWave
msoAnimEffectComplementaryColor
msoAnimEffectComplementaryColor2
msoAnimEffectContrastingColor
msoAnimEffectCrawl
msoAnimEffectCredits
msoAnimEffectCustom
msoAnimEffectDarken
msoAnimEffectDesaturate
msoAnimEffectDescend
msoAnimEffectDiamond
msoAnimEffectDissolve
msoAnimEffectEaseIn
msoAnimEffectExpand
msoAnimEffectFade
msoAnimEffectFadedAscend
msoAnimEffectFadedSwivel
msoAnimEffectFadedZoom
msoAnimEffectFlashBulb
msoAnimEffectFlashOnce
msoAnimEffectFlicker
msoAnimEffectFlip
msoAnimEffectFloat
msoAnimEffectFly
msoAnimEffectFold
msoAnimEffectGlide
msoAnimEffectGrowAndTurn
msoAnimEffectGrowShrink

msoAnimEffectGrowWithColor
msoAnimEffectLighten
msoAnimEffectLightSpeed
msoAnimEffectMediaPause
msoAnimEffectMediaPlay
msoAnimEffectMediaStop
msoAnimEffectPath4PointStar
msoAnimEffectPath5PointStar
msoAnimEffectPath6PointStar
msoAnimEffectPath8PointStar
msoAnimEffectPathArcDown
msoAnimEffectPathArcLeft
msoAnimEffectPathArcRight
msoAnimEffectPathArcUp
msoAnimEffectPathBean
msoAnimEffectPathBounceLeft
msoAnimEffectPathBounceRight
msoAnimEffectPathBuzzsaw
msoAnimEffectPathCircle
msoAnimEffectPathCrescentMoon
msoAnimEffectPathCurvedSquare
msoAnimEffectPathCurvedX
msoAnimEffectPathCurvyLeft
msoAnimEffectPathCurvyRight
msoAnimEffectPathCurvyStar
msoAnimEffectPathDecayingWave
msoAnimEffectPathDiagonalDownRight
msoAnimEffectPathDiagonalUpRight
msoAnimEffectPathDiamond
msoAnimEffectPathDown
msoAnimEffectPathEqualTriangle
msoAnimEffectPathFigure8Four
msoAnimEffectPathFootball

msoAnimEffectPathFunnel
msoAnimEffectPathHeart
msoAnimEffectPathHeartbeat
msoAnimEffectPathHexagon
msoAnimEffectPathHorizontalFigure8
msoAnimEffectPathInvertedSquare
msoAnimEffectPathInvertedTriangle
msoAnimEffectPathLeft
msoAnimEffectPathLoopdeLoop
msoAnimEffectPathNeutron
msoAnimEffectPathOctagon
msoAnimEffectPathParallelogram
msoAnimEffectPathPeanut
msoAnimEffectPathPentagon
msoAnimEffectPathPlus
msoAnimEffectPathPointyStar
msoAnimEffectPathRightTriangle
msoAnimEffectPathSCurve1
msoAnimEffectPathSCurve2
msoAnimEffectPathSineWave
msoAnimEffectPathSpiralLeft
msoAnimEffectPathSpiralRight
msoAnimEffectPathSpring
msoAnimEffectPathSquare
msoAnimEffectPathStairsDown
msoAnimEffectPathSwoosh
msoAnimEffectPathTeardrop
msoAnimEffectPathTrapezoid
msoAnimEffectPathTurnDown
msoAnimEffectPathTurnRight
msoAnimEffectPathTurnUp
msoAnimEffectPathTurnUpRight
msoAnimEffectPathVerticalFigure8

msoAnimEffectPathWave
msoAnimEffectPathZigzag
msoAnimEffectPeek
msoAnimEffectPinwheel
msoAnimEffectPlus
msoAnimEffectRandomBars
msoAnimEffectRandomEffects
msoAnimEffectRiseUp
msoAnimEffectShimmer
msoAnimEffectSling
msoAnimEffectSpin
msoAnimEffectSpinner
msoAnimEffectSpiral
msoAnimEffectSplit
msoAnimEffectStretch
msoAnimEffectStretchy
msoAnimEffectStrips
msoAnimEffectStyleEmphasis
msoAnimEffectSwish
msoAnimEffectSwivel
msoAnimEffectTeeter
msoAnimEffectThinLine
msoAnimEffectTransparency
msoAnimEffectUnfold
msoAnimEffectVerticalGrow
msoAnimEffectWave
msoAnimEffectWedge
msoAnimEffectWheel
msoAnimEffectWhip
msoAnimEffectWipe
msoAnimEffectZip
msoAnimEffectZoom

Level Optional [MsoAnimateByLevel](#). For charts, diagrams, or text, the level to which the animation effect will be applied. The default value is **msoAnimationLevelNone**.

MsoAnimateByLevel can be one of these **MsoAnimateByLevel** constants.

msoAnimateChartAllAtOnce
msoAnimateChartByCategory
msoAnimateChartByCategoryElements
msoAnimateChartBySeries
msoAnimateChartBySeriesElements
msoAnimateDiagramAllAtOnce
msoAnimateDiagramBreadthByLevel
msoAnimateDiagramBreadthByNode
msoAnimateDiagramClockwise
msoAnimateDiagramClockwiseIn
msoAnimateDiagramClockwiseOut
msoAnimateDiagramCounterClockwise
msoAnimateDiagramCounterClockwiseIn
msoAnimateDiagramCounterClockwiseOut
msoAnimateDiagramDepthByBranch
msoAnimateDiagramDepthByNode
msoAnimateDiagramDown
msoAnimateDiagramInByRing
msoAnimateDiagramOutByRing
msoAnimateDiagramUp
msoAnimateLevelMixed
msoAnimateTextByAllLevels
msoAnimateTextByFifthLevel
msoAnimateTextByFirstLevel
msoAnimateTextByFourthLevel
msoAnimateTextBySecondLevel
msoAnimateTextByThirdLevel
msoAnimationLevelNone

trigger Optional [MsoAnimTriggerType](#). The action that triggers the animation effect. The default value is **msoAnimTriggerOnPageClick**.

MsoAnimTriggerType can be one of these **MsoAnimTriggerType** constants.

msoAnimTriggerAfterPrevious

msoAnimTriggerMixed

msoAnimTriggerNone

msoAnimTriggerOnPageClick

msoAnimTriggerOnShapeClick

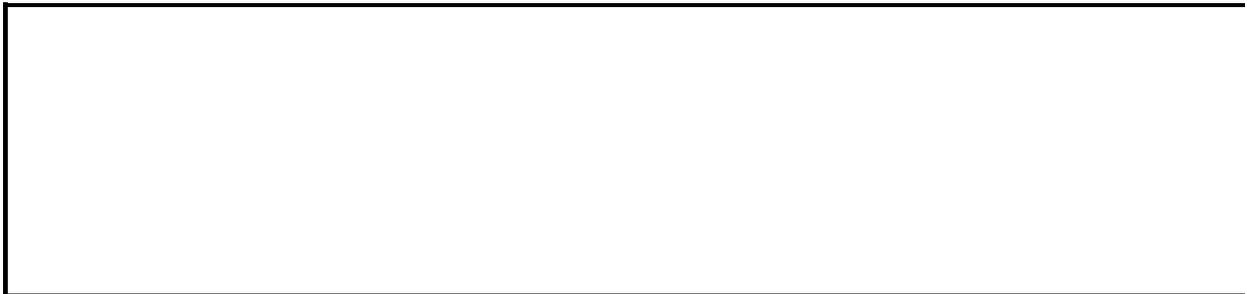
msoAnimTriggerWithPrevious

Index Optional **Long**. The position at which the effect will be placed in the collection of animation effects. The default value is -1 (added to the end).

Example

The following example adds a bouncing animation to the first shape range on the first slide. This example assumes a shape range containing one or more shapes is selected on the first slide.

```
Sub AddBouncingAnimation()  
  
    Dim sldActive As Slide  
    Dim shpSelected As Shape  
  
    Set sldActive = ActiveWindow.Selection.SlideRange(1)  
    Set shpSelected = ActiveWindow.Selection.ShapeRange(1)  
  
    ' Add a bouncing animation.  
    sldActive.TimeLine.MainSequence.AddEffect _  
        Shape:=shpSelected, effectId:=msoAnimEffectBounce  
  
End Sub
```



[Show All](#)

AddLabel Method

Creates a label. Returns a [Shape](#) object that represents the new label.

expression.AddLabel(***Orientation***, ***Left***, ***Top***, ***Width***, ***Height***)

expression Required. An expression that returns one of the objects in the Applies To list.

Orientation Required [MsoTextOrientation](#). The text orientation. Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.

MsoTextOrientation can be one of these MsoTextOrientation constants.

msoTextOrientationDownward

msoTextOrientationHorizontal

msoTextOrientationHorizontalRotatedFarEast

msoTextOrientationMixed

msoTextOrientationUpward

msoTextOrientationVertical

msoTextOrientationVerticalFarEast

Left Required **Single**. The position, measured in points, of the left edge of the label relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the label relative to the top edge of the slide.

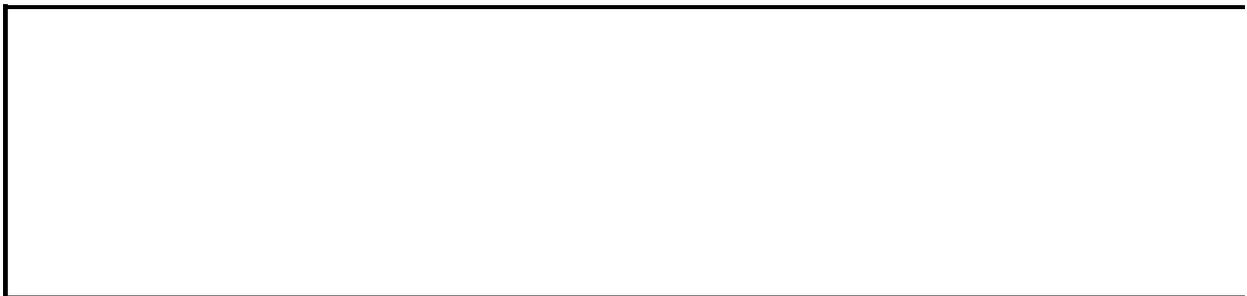
Width Required **Single**. The width of the label, measured in points.

Height Required **Single**. The height of the label, measured in points.

Example

This example adds a vertical label that contains the text "Test Label" to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddLabel(Orientation:=msoTextOrientationVerticalFa
    Left:=100, Top:=100, Width:=60, Height:=150).TextFrame _
    .TextRange.Text = "Test Label"
```



AddLine Method

Creates a line. Returns a [Shape](#) object that represents the new line.

expression.**AddLine**(*BeginX*, *BeginY*, *EndX*, *EndY*)

expression Required. An expression that returns one of the objects in the Applies To list.

BeginX Required **Single**. The horizontal position, measured in points, of the line's starting point relative to the left edge of the slide.

BeginY Required **Single**. The vertical position, measured in points, of the line's starting point relative to the top edge of the slide.

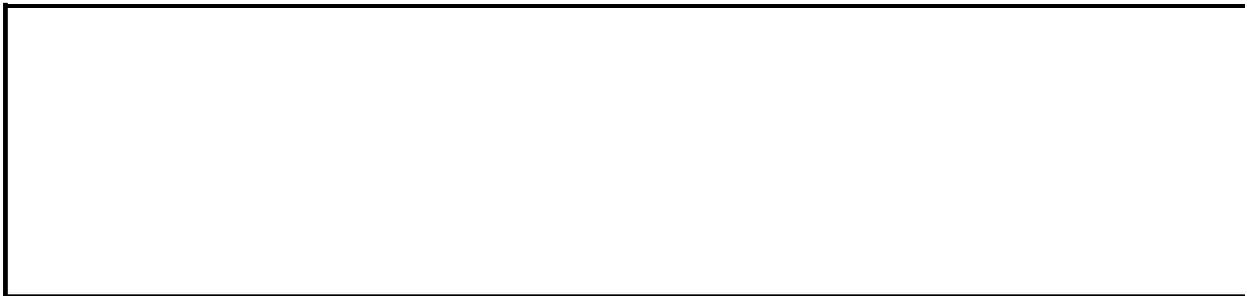
EndX Required **Single**. The horizontal position, measured in points, of the line's ending point relative to the left edge of the slide.

EndY Required **Single**. The vertical position, measured in points, of the line's ending point relative to the top edge of the slide.

Example

This example adds a blue dashed line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(BeginX:=10, BeginY:=10, _
    EndX:=250, EndY:=250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(50, 0, 128)
End With
```



AddMediaObject Method

Creates a media object. Returns a [Shape](#) object that represents the new media object.

expression.**AddMediaObject**(*FileName*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns a **Shapes** object.

FileName Required **String**. The file from which the media object is to be created. If the path isn't specified, the current working folder is used.

Left , **Top** Optional **Single**. The position (in points) of the upper-left corner of the media object's bounding box relative to the upper-left corner of the document.

Width , **Height** Optional **Single**. The width and height of the media object's bounding box, in points.

Example

This example adds the movie named "Clock.avi" to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddMediaObject FileName:="C:\WINNT\clock.avi", _
    Left:=5, Top:=5, Width:=100, Height:=100
```



[Show All](#)

AddNode Method

 [AddNode method as it applies to the **DiagramNodeChildren** object.](#)

Adds a **DiagramNode** object to a collection of child diagram nodes.

expression.**AddNode**(*Index*)

expression Required. An expression that returns a **DiagramNodeChildren** object.

Index Optional **Variant**. The index location of where to add the new diagram node; 0 adds before all nodes; -1 adds after all nodes; any other **Index** will add after that node in the collection.

 [AddNode method as it applies to the **DiagramNode** object.](#)

Returns a **DiagramNode** object that represents a node added to a diagram.

expression.**AddNode**(*Pos*)

expression Required. An expression that returns a **DiagramNode** object.

Pos Optional [MsoRelativeNodePosition](#). Specifies where the node will be added, relative to the calling node.

MsoRelativeNodePosition can be one of these **MsoRelativeNodePosition** constants.

msoAfterLastSibling

msoAfterNode *default*

msoBeforeFirstSibling

msoBeforeNode

Example

The following example adds nodes to a newly-created diagram.

```
Sub CreatePyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds the pyramid diagram and first node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three more nodes to pyramid diagram  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
End Sub
```



[Show All](#)

AddNodes Method

Inserts a new segment at the end of the freeform that's being created, and adds the nodes that define the segment. You can use this method as many times as you want to add nodes to the freeform you're creating. When you finish adding nodes, use the [ConvertToShape](#) method to create the freeform you've just defined. To add nodes to a freeform after it's been created, use the [Insert](#) method of the [ShapeNodes](#) collection.

expression.AddNodes(*SegmentType*, *EditingType*, *X1*, *Y1*, *X2*, *Y2*, *X3*, *Y3*)

expression Required. An expression that returns a **FreeformBuilder** object.

SegmentType Required [MsoSegmentType](#). The type of segment to be added.

MsoSegmentType can be one of these MsoSegmentType constants.

msoSegmentCurve

msoSegmentLine

EditingType Required [MsoEditingType](#). The editing property of the vertex. If *SegmentType* is **msoSegmentLine**, *EditingType* must be **msoEditingAuto**.

MsoEditingType can be one of these MsoEditingType constants (cannot be **msoEditingSmooth** or **msoEditingSymmetric**).

msoEditingAuto

msoEditingCorner

X1 Required **Single**. If the *EditingType* of the new segment is **msoEditingAuto**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new node is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the first control point for the new segment.

Y1 Required **Single**. If the *EditingType* of the new segment is **msoEditingAuto**, this argument specifies the vertical distance (in points) from

the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new node is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the first control point for the new segment.

X2 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the second control point for the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Y2 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the second control point for the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

X3 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Y3 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Example

This example adds a freeform with five vertices to the first slide in the active presentation.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.BuildFreeform(msoEditingCorner, 360, 200)
    .AddNodes SegmentType:=msoSegmentCurve, EditingType:=msoEditingC
        X1:=380, Y1:=230, X2:=400, Y2:=250, X3:=450, Y3:=300
    .AddNodes SegmentType:=msoSegmentCurve, EditingType:=msoEditingA
        X1:=480, Y1:=200
    .AddNodes SegmentType:=msoSegmentLine, EditingType:=msoEditingAu
        X1:=480, Y1:=400
    .AddNodes SegmentType:=msoSegmentLine, EditingType:=msoEditingAu
        X1:=360, Y1:=200
    .ConvertToShape
End With
```



AddPeriods Method

Adds a period at the end of each paragraph in the specified text.

expression.**AddPeriods**

expression Required. An expression that returns a **TextRange** object.

Remarks

This method doesn't add another period at the end of a paragraph that already ends with a period.

Example

This example adds a period at the end of each paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2).TextFrame _  
    .TextRange.AddPeriods
```



[Show All](#)

AddPicture Method

Creates a picture from an existing file. Returns a [Shape](#) object that represents the new picture.

expression.**AddPicture**(*FileName*, *LinkToFile*, *SaveWithDocument*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns one of the objects in the Applies To list.

FileName Required **String**. The file from which the OLE object is to be created.

LinkToFile Required [MsoTriState](#). Determines whether the picture will be linked to the file from which it was created.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Makes the picture an independent copy of the file.

msoTriStateMixed

msoTriStateToggle

msoTrue Links the picture to the file from which it was created.

SaveWithDocument Required [MsoTriState](#). Determines whether the linked picture will be saved with the document into which it's inserted. This argument must be **msoTrue** if **LinkToFile** is **msoFalse**.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Stores only the link information in the document.

msoTriStateMixed

msoTriStateToggle

msoTrue Saves the linked picture with the document into which it's inserted.

Left Required **Single**. The position, measured in points, of the left edge of the picture relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the picture relative to the top edge of the slide.

Width Optional **Single**. The width of the picture, measured in points.

Height Optional **Single**. The height of the picture, measured in points.

Example

This example adds a picture created from the file Music.bmp to myDocument. The inserted picture is linked to the file from which it was created and is saved with myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddPicture FileName="c:\microsoft office\" & _
    "clipart\music.bmp", LinkToFile:=msoTrue, SaveWithDocument:=msoT
    Left:=100, Top:=100, Width:=70, Height:=70
```



[Show All](#)

AddPlaceholder Method

Restores a previously deleted placeholder on a slide. Returns a [Shape](#) object that represents the restored placeholder.

Note If you haven't previously deleted the specified placeholder, this method causes an error.

expression.AddPlaceholder(*Type*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns a **Shapes** object.

Type Required [PpPlaceholderType](#). The type of placeholder. Placeholders of type **ppPlaceholderVerticalBody** or **ppPlaceholderVerticalTitle** are found only on slides of layout type **ppLayoutVerticalText**, **ppLayoutClipArtAndVerticalText**, **ppLayoutVerticalTitleAndText**, or **ppLayoutVerticalTitleAndTextOverChart**. You cannot create slides with any of these layouts from the user interface; you must create them programmatically by using the [Add](#) method or by setting the [Layout](#) property of an existing slide.

PpPlaceholderType can be one of these PpPlaceholderType constants.

ppPlaceholderBitmap

ppPlaceholderBody

ppPlaceholderCenterTitle

ppPlaceholderChart

ppPlaceholderDate

ppPlaceholderFooter

ppPlaceholderHeader

ppPlaceholderMediaClip

ppPlaceholderMixed

ppPlaceholderObject

ppPlaceholderOrgChart

ppPlaceholderSlideNumber

ppPlaceholderSubtitle

ppPlaceholderTable

ppPlaceholderTitle

ppPlaceholderVerticalBody

ppPlaceholderVerticalTitle

Left , ***Top*** Optional **Single**. The position (in points) of the upper-left corner of the placeholder relative to the upper-left corner of the document.

Width , ***Height*** Optional **Single**. The width and height of the placeholder, in points.

Remarks

If more than one placeholder of a specified type has been deleted from the slide, the **AddPlaceholder** method will add them back to the slide, one by one, starting with the placeholder that has the lowest original index number.

Example

Suppose that slide two in the active presentation originally had a title at the top of the slide that's been deleted, either manually or with the following line of code.

```
ActivePresentation.Slides(2).Shapes.Placeholders(1).Delete
```

This example restores the deleted placeholder to slide two.

```
Application.ActivePresentation.Slides(2) _  
    .Shapes.AddPlaceholder ppPlaceholderTitle
```



[Show All](#)

AddPolyline Method

Creates an open polyline or a closed polygon drawing. Returns a [Shape](#) object that represents the new polyline or polygon.

expression.**AddPolyline**(*SafeArrayOfPoints*)

expression Required. An expression that returns one of the objects in the Applies To list.

SafeArrayOfPoints Required **Variant**. An array of [coordinate pairs](#) that specifies the polyline drawing's vertices.

Remarks

To form a closed polygon, assign the same coordinates to the first and last vertices in the polyline drawing.

Example

This example adds a triangle to myDocument. Because the first and last points have the same coordinates, the polygon is closed and filled. The color of the triangle's interior will be the same as the default shape's fill color.

```
Dim triArray(1 To 4, 1 To 2) As Single
triArray(1, 1) = 25
triArray(1, 2) = 100
triArray(2, 1) = 100
triArray(2, 2) = 150
triArray(3, 1) = 150
triArray(3, 2) = 50
triArray(4, 1) = 25      ' Last point has same coordinates as first
triArray(4, 2) = 100
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddPolyline SafeArrayOfPoints:=triArray
```



[Show All](#)

AddShape Method

Creates an AutoShape. Returns a [Shape](#) object that represents the new AutoShape.

expression.**AddShape**(*Type*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Required [MsoAutoShapeType](#). Specifies the type of AutoShape to create.

MsoAutoShapeType can be one of these MsoAutoShapeType constants.

msoShapeFlowchartConnector

msoShapeFlowchartData

msoShapeFlowchartDecision

msoShapeFlowchartDelay

msoShapeFlowchartDirectAccessStorage

msoShapeFlowchartDisplay

msoShapeFlowchartDocument

msoShapeFlowchartExtract

msoShapeFlowchartInternalStorage

msoShapeFlowchartMagneticDisk

msoShapeFlowchartManualInput

msoShapeFlowchartManualOperation

msoShapeFlowchartMerge

msoShapeFlowchartMultidocument

msoShapeFlowchartOffpageConnector

msoShapeFlowchartOr

msoShapeFlowchartPredefinedProcess

msoShapeFlowchartPreparation

msoShapeFlowchartProcess

msoShapeFlowchartPunchedTape
msoShapeFlowchartSequentialAccessStorage
msoShapeFlowchartSort
msoShapeFlowchartStoredData
msoShapeFlowchartSummingJunction
msoShapeFlowchartTerminator
msoShapeFoldedCorner
msoShapeHeart
msoShapeHexagon
msoShapeHorizontalScroll
msoShapeIsoscelesTriangle
msoShapeLeftArrow
msoShapeLeftArrowCallout
msoShapeLeftBrace
msoShapeLeftBracket
msoShapeLeftRightArrow
msoShapeLeftRightArrowCallout
msoShapeLeftRightUpArrow
msoShapeLeftUpArrow
msoShapeLightningBolt
msoShapeLineCallout1
msoShapeLineCallout1AccentBar
msoShapeLineCallout1BorderandAccentBar
msoShapeLineCallout1NoBorder
msoShapeLineCallout2
msoShapeLineCallout2AccentBar
msoShapeLineCallout2BorderandAccentBar
msoShapeLineCallout2NoBorder
msoShapeLineCallout3
msoShapeLineCallout3AccentBar
msoShapeLineCallout3BorderandAccentBar
msoShapeLineCallout3NoBorder
msoShapeLineCallout4

msoShapeLineCallout4AccentBar
msoShapeLineCallout4BorderandAccentBar
msoShapeLineCallout4NoBorder
msoShapeMixed
msoShapeMoon
msoShapeNoSymbol
msoShapeNotchedRightArrow
msoShapeNotPrimitive
msoShapeOctagon
msoShapeOval
msoShapeOvalCallout
msoShapeParallelogram
msoShapePentagon
msoShapePlaque
msoShapeQuadArrow
msoShapeQuadArrowCallout
msoShapeRectangle
msoShapeRectangularCallout
msoShapeRegularPentagon
msoShapeRightArrow
msoShapeRightArrowCallout
msoShapeRightBrace
msoShapeRightBracket
msoShapeRightTriangle
msoShapeRoundedRectangle
msoShapeRoundedRectangularCallout
msoShapeSmileyFace
msoShapeStripedRightArrow
msoShapeSun
msoShapeTrapezoid
msoShapeUpArrow
msoShapeUpArrowCallout
msoShapeUpDownArrow

msoShapeUpDownArrowCallout
msoShapeUpRibbon
msoShapeUTurnArrow
msoShapeVerticalScroll
msoShapeWave
msoShapeFlowchartCollate
msoShape16pointStar
msoShape24pointStar
msoShape32pointStar
msoShape4pointStar
msoShape5pointStar
msoShape8pointStar
msoShapeActionButtonBackorPrevious
msoShapeActionButtonBeginning
msoShapeActionButtonCustom
msoShapeActionButtonDocument
msoShapeActionButtonEnd
msoShapeActionButtonForwardorNext
msoShapeActionButtonHelp
msoShapeActionButtonHome
msoShapeActionButtonInformation
msoShapeActionButtonMovie
msoShapeActionButtonReturn
msoShapeActionButtonSound
msoShapeArc
msoShapeBalloon
msoShapeBentArrow
msoShapeBentUpArrow
msoShapeBevel
msoShapeBlockArc
msoShapeCan
msoShapeChevron
msoShapeCircularArrow

msoShapeCloudCallout
msoShapeCross
msoShapeCube
msoShapeCurvedDownArrow
msoShapeCurvedDownRibbon
msoShapeCurvedLeftArrow
msoShapeCurvedRightArrow
msoShapeCurvedUpArrow
msoShapeCurvedUpRibbon
msoShapeDiamond
msoShapeDonut
msoShapeDoubleBrace
msoShapeDoubleBracket
msoShapeDoubleWave
msoShapeDownArrow
msoShapeDownArrowCallout
msoShapeDownRibbon
msoShapeExplosion1
msoShapeExplosion2
msoShapeFlowchartAlternateProcess
msoShapeFlowchartCard

Left Required **Single**. The position, measured in points, of the left edge of the AutoShape relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the AutoShape relative to the top edge of the slide.

Width Required **Single**. The width of the AutoShape, measured in points.

Height Required **Single**. The height of the AutoShape, measured in points.

Remarks

To change the type of an AutoShape that you've added, set the [AutoShapeType](#) property.

Example

This example adds a rectangle to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddShape Type:=msoShapeRectangle, _
    Left:=50, Top:=50, Width:=100, Height:=200
```



AddTable Method

Adds a table shape to a slide.

expression.**AddTable**(*NumRows*, *NumColumns*, *Left*, *Top*, *Width*, *Height*)

expression Required. An expression that returns a **Shape** object.

NumRows Required **Long**. The number of rows in the table.

NumColumns Required **Long**. The number of columns in the table.

Left Optional **Single**. The distance (in points) from the left edge of the slide to the left edge of the table.

Top Optional **Single**. The distance (in points) from the top edge of the slide to the top edge of the table.

Width Optional **Single**. The width (in points) of the new table.

Height Optional **Single**. The height (in points) of the new table.

Example

This example creates a new table on slide two of the active presentation. The table has three rows and four columns. It is 10 points from the left edge of the slide, and 10 points from the top edge. The width of the new table is 288 points, which makes each of the four columns one inch wide (there are 72 points per inch). The height is set to 216 points, which makes each of the three rows one inch tall.

```
ActivePresentation.Slides(2).Shapes _  
    .AddTable(3, 4, 10, 10, 288, 216)
```



[Show All](#)

AddTextbox Method

Creates a text box. Returns a [Shape](#) object that represents the new text box.

expression.AddTextbox(*Orientation, Left, Top, Width, Height*)

expression Required. An expression that returns one of the objects in the Applies To list.

Orientation Required [MsoTextOrientation](#). The text orientation. Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.

MsoTextOrientation can be one of these MsoTextOrientation constants.

msoTextOrientationDownward

msoTextOrientationHorizontal

msoTextOrientationHorizontalRotatedFarEast

msoTextOrientationMixed

msoTextOrientationUpward

msoTextOrientationVertical

msoTextOrientationVerticalFarEast

Left Required **Single**. The position, measured in points, of the left edge of the text box relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the text box relative to the top edge of the slide.

Width Required **Single**. The width of the text box, measured in points.

Height Required **Single**. The height of the text box, measured in points.

Example

This example adds a text box that contains the text "Test Box" to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddTextbox(Type:=msoTextOrientationHorizontal, _
    Left:=100, Top:=100, Width:=200, Height:=50).TextFrame _
    .TextRange.Text = "Test Box"
```



[Show All](#)

AddTextEffect Method

Creates a WordArt object. Returns a [Shape](#) object that represents the new WordArt object.

expression.AddTextEffect(PresetTextEffect, Text, FontName, FontSize, FontBold, FontItalic, Left, Top)

expression Required. An expression that returns one of the objects in the Applies To list.

PresetTextEffect Required [MsoPresetTextEffect](#). The preset text effect.

MsoPresetTextEffect can be one of these MsoPresetTextEffect constants.

msoTextEffect1

msoTextEffect2

msoTextEffect3

msoTextEffect4

msoTextEffect5

msoTextEffect6

msoTextEffect7

msoTextEffect8

msoTextEffect9

msoTextEffect10

msoTextEffect11

msoTextEffect12

msoTextEffect13

msoTextEffect14

msoTextEffect15

msoTextEffect16

msoTextEffect17

msoTextEffect18

msoTextEffect19

msoTextEffect20
msoTextEffect21
msoTextEffect22
msoTextEffect23
msoTextEffect24
msoTextEffect25
msoTextEffect26
msoTextEffect27
msoTextEffect28
msoTextEffect29
msoTextEffect30
msoTextEffectMixed

Text Required **String**. The text in the WordArt.

FontName Required **String**. The name of the font used in the WordArt.

FontSize Required **Single**. The size (in points) of the font used in the WordArt.

FontBold Required **MsoTriState**. Determines whether the font used in the WordArt is set to bold.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Sets the font used in the WordArt to bold.

FontItalic Required **MsoTriState**. Determines whether the font used in the WordArt is set to italic.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Sets the font used in the WordArt to italic.

Left Required **Single**. The position, measured in points, of the left edge of the WordArt's bounding box relative to the left edge of the slide.

Top Required **Single**. The position, measured in points, of the top edge of the WordArt's bounding box relative to the top edge of the slide.

Remarks

When you add WordArt to a document, the height and width of the WordArt are automatically set based on the size and amount of text you specify.

Example

This example adds WordArt that contains the text "Test" to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
Set newWordArt = myDocument.Shapes _
    .AddTextEffect(PresetTextEffect:=msoTextEffect1, _
        Text:="Test", FontName:="Arial Black", FontSize:=36, _
        FontBold:=msoFalse, FontItalic:=msoFalse, Left:=10, Top:=10)
```



AddTitle Method

Restores a previously deleted title placeholder to a slide. Returns a [Shape](#) object that represents the restored title.

Note This method will cause an error if you haven't previously deleted the title placeholder from the specified slide. Use the [HasTitle](#) property to determine whether the title placeholder has been deleted.

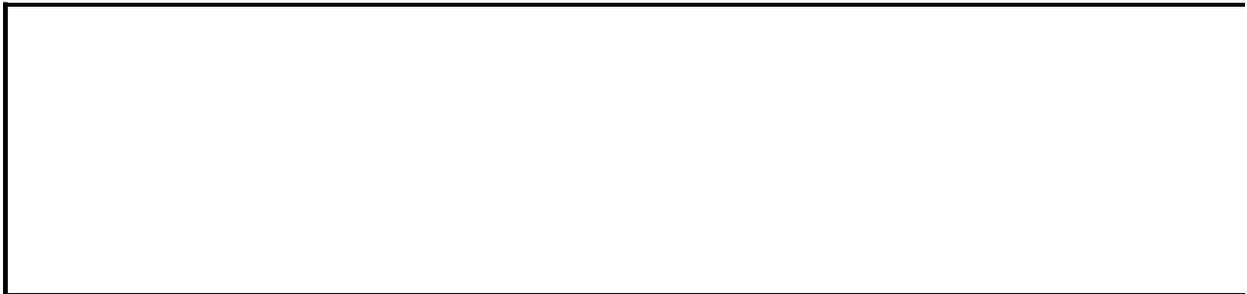
expression.**AddTitle**

expression Required. An expression that returns a **Shapes** object.

Example

This example restores the title placeholder to slide one in the active presentation if this placeholder has been deleted. The text of the restored title is "Restored title."

```
With ActivePresentation.Slides(1)
  If .Layout <> ppLayoutBlank Then
    With .Shapes
      If Not .HasTitle Then
        .AddTitle.TextFrame.TextRange _
          .Text = "Restored title"
      End If
    End With
  End If
End With
```



AddTitleMaster Method

Adds a title master to the specified presentation. Returns a [Master](#) object that represents the title master. If the presentation already has a title master, an error occurs.

expression.**AddTitleMaster**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a title master to the active presentation if it doesn't already have one.

```
With Application.ActivePresentation  
    If Not .HasTitleMaster Then .AddTitleMaster  
End With
```



AddToFavorites Method

Adds a shortcut to the Favorites folder in the Windows program folder representing either the current selection in the specified presentation (for the [Presentation](#) object) or the specified hyperlink's target document (for the [Hyperlink](#) object).

expression.**AddToFavorites**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The shortcut name is the friendly name of the document, if that's available; otherwise, the shortcut name is as calculated in HLINK.DLL.

Example

This example adds a hyperlink to the active presentation to the Favorites folder in the Windows program folder.

`Application.ActivePresentation.AddToFavorites`



[Show All](#)

Align Method

Aligns the shapes in the specified range of shapes.

expression.**Align**(*AlignCmd*, *RelativeTo*)

expression Required. An expression that returns a **ShapeRange** object.

AlignCmd Required [MsoAlignCmd](#). Specifies the way the shapes in the specified shape range are to be aligned.

MsoAlignCmd can be one of these MsoAlignCmd constants.

msoAlignBottoms

msoAlignCenters

msoAlignLefts

msoAlignMiddles

msoAlignRights

msoAlignTops

RelativeTo Required [MsoTriState](#). Determines whether shapes are aligned relative to the edge of the slide.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Aligns shapes relative to one another.

msoTriStateMixed

msoTriStateToggle

msoTrue Aligns shapes relative to the edge of the slide.

Example

This example aligns the left edges of all the shapes in the specified range in myDocument with the left edge of the leftmost shape in the range.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.Range.Align msoAlignLefts, msoFalse
```



Apply Method

Applies to the specified shape formatting that's been copied by using the [PickUp](#) method.

expression.**Apply**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Example

This example copies the formatting of shape one on myDocument, and then applies the copied formatting to shape two.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument
    .Shapes(1).PickUp
    .Shapes(2).Apply
End With
```



ApplyTemplate Method

Applies a design template to the specified presentation.

expression.**ApplyTemplate**(*FileName*)

expression Required. An expression that returns a **Presentation** object.

FileName Required **String**. Specifies the name of the design template.

Note If you refer to an uninstalled presentation design template in a string, a run-time error is generated. The template is not installed automatically regardless of your [FeatureInstall](#) property setting. To use the **ApplyTemplate** method for a template that is not currently installed, you first must install the additional design templates. To do so, install the Additional Design Templates for PowerPoint by running the Microsoft Office installation program (available through the **Add/Remove Programs** icon in Windows Control Panel).

Example

This example applies the "Professional" design template to the active presentation.

```
Application.ActivePresentation.ApplyTemplate _  
    "c:\program files\microsoft office\templates" & _  
    "\presentation designs\professional.pot"
```



[Show All](#)

Arrange Method

Arranges all open document windows in the workspace.

expression.**Arrange**(*arrangeStyle*)

expression Required. An expression that returns one of the objects in the Applies To list.

arrangeStyle Optional [PpArrangeStyle](#). Specifies whether to cascade or tile the windows.

PpArrangeStyle can be one of these PpArrangeStyle constants.

ppArrangeCascade

ppArrangeTiled *default*

Example

This example creates a new window and then arranges all open document windows.

```
Application.ActiveWindow.NewWindow  
Windows.Arrange ppArrangeCascade
```



AutomaticLength Method

Specifies that the first segment of the callout line (the segment attached to the text callout box) be scaled automatically when the callout is moved. Use the [CustomLength](#) method to specify that the first segment of the callout line retain the fixed length returned by the [Length](#) property whenever the callout is moved. Applies only to callouts whose lines consist of more than one segment (types **msoCalloutThree** and **msoCalloutFour**).

expression.**AutomaticLength**

expression Required. An expression that returns a **CalloutFormat** object.

Remarks

Applying this method sets the [AutoLength](#) property to **True**.

Example

This example toggles between an automatically scaling first segment and one with a fixed length for the callout line for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```



Background Method

Specifies that the shape's fill should match the slide background. If you change the slide background after applying this method to a fill, the fill will also change.

Remarks

Note that applying the **Background** method to a shape's fill isn't the same as setting a transparent fill for the shape, nor is it always the same as applying the same fill to the shape as you apply to the background. The second example demonstrates this.

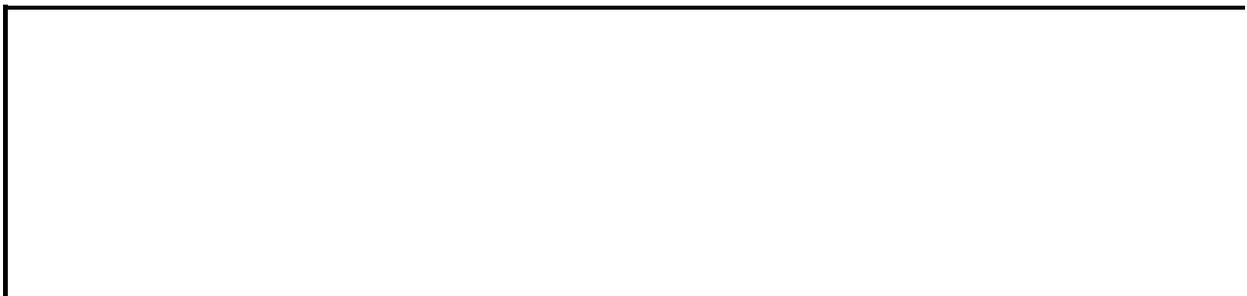
Example

This example sets the fill of shape one on slide one in the active presentation to match the slide background.

```
ActivePresentation.Slides(1).Shapes(1).Fill.Background
```

This example sets the background for slide one in the active presentation to a preset gradient, adds a rectangle to the slide, and then places three ovals in front of the rectangle. The first oval has a fill that matches the slide background, the second has a transparent fill, and the third has the same fill applied to it as was applied to the background. Notice the difference in the appearances of these three ovals.

```
With ActivePresentation.Slides(1)
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient _
        msoGradientHorizontal, 1, msoGradientDaybreak
    With .Shapes
        .AddShape msoShapeRectangle, 50, 200, 600, 100
        .AddShape(msoShapeOval, 75, 150, 150, 100) _
            .Fill.Background
        .AddShape(msoShapeOval, 275, 150, 150, 100).Fill _
            .Transparency = 1
        .AddShape(msoShapeOval, 475, 150, 150, 100) _
            .Fill.PresetGradient _
                msoGradientHorizontal, 1, msoGradientDaybreak
    End With
End With
```



BeginConnect Method

Attaches the beginning of the specified connector to a specified shape. If there's already a connection between the beginning of the connector and another shape, that connection is broken. If the beginning of the connector isn't already positioned at the specified connecting site, this method moves the beginning of the connector to the connecting site and adjusts the size and position of the connector. Use the [EndConnect](#) method to attach the end of the connector to a shape.

expression.**BeginConnect**(*ConnectedShape*, *ConnectionSite*)

expression Required. An expression that returns a **ConnectorFormat** object.

ConnectedShape Required **Shape** object. The shape to attach the beginning of the connector to. The specified **Shape** object must be in the same **Shapes** collection as the connector.

ConnectionSite Required **Long**. A connection site on the shape specified by **ConnectedShape**. Must be an integer between 1 and the integer returned by the **ConnectionSiteCount** property of the specified shape. If you want the connector to automatically find the shortest path between the two shapes it connects, specify any valid integer for this argument and then use the [RerouteConnections](#) method after the connector is attached to shapes at both ends.

Remarks

When you attach a connector to an object, the size and position of the connector are automatically adjusted, if necessary.

Example

This example adds two rectangles to the first slide in the active presentation and connects them with a curved connector. Notice that the **RerouteConnections** method makes it irrelevant what values you supply for the **ConnectionSite** arguments used with the **BeginConnect** and **EndConnect** methods.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 100, 100) _
    .ConnectorFormat
        .BeginConnect ConnectedShape:=firstRect, ConnectionSite:=1
        .EndConnect ConnectedShape:=secondRect, ConnectionSite:=1
        .Parent.RerouteConnections
End With
```



BeginDisconnect Method

Detaches the beginning of the specified connector from the shape it's attached to. This method doesn't alter the size or position of the connector: the beginning of the connector remains positioned at a connection site but is no longer connected. Use the [EndDisconnect](#) method to detach the end of the connector from a shape.

expression.**BeginDisconnect**

expression Required. An expression that returns a **ConnectorFormat** object.

Example

This example adds two rectangles to the first slide in the active presentation, attaches them with a connector, automatically reroutes the connector along the shortest path, and then detaches the connector from the rectangles.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 0, 0).ConnectorFormat
    .BeginConnect firstRect, 1
    .EndConnect secondRect, 1
    .Parent.RerouteConnections
    .BeginDisconnect
    .EndDisconnect
End With
```



[Show All](#)

BuildFreeform Method

Builds a freeform object. Returns a [FreeformBuilder](#) object that represents the freeform as it is being built. Use the [AddNodes](#) method to add segments to the freeform. After you have added at least one segment to the freeform, you can use the [ConvertToShape](#) method to convert the **FreeformBuilder** object into a [Shape](#) object that has the geometric description you've defined in the **FreeformBuilder** object.

expression.**BuildFreeform**(*EditingType*, *X1*, *Y1*)

expression Required. An expression that returns one of the objects in the Applies To list.

EditingType Required [MsoEditingType](#). The editing property of the first node.

MsoEditingType can be one of the following MsoEditingType constants (cannot be **msoEditingSmooth** or **msoEditingSymmetric**).

msoEditingAuto

msoEditingCorner

X1 Required **Single**. The horizontal position, measured in points, of the first node in the freeform drawing relative to the left edge of the slide.

Y1 Required **Single**. The vertical position, measured in points, of the first node in the freeform drawing relative to the top edge of the slide.

Example

This example adds a freeform with four segments to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.BuildFreeform(EditingType:=msoEditingCorner,
    X1=360, Y1=200)
    .AddNodes SegmentType:=msoSegmentCurve, EditingType:=msoEditingC
        X1:=380, Y1:=230, X2:=400, Y2:=250, X3:=450, Y3:=300
    .AddNodes SegmentType:=msoSegmentCurve, EditingType:=msoEditingA
        X1:=480, Y1:=200
    .AddNodes SegmentType:=msoSegmentLine, EditingType:=msoEditingAu
        X1:=480, Y1:=400
    .AddNodes SegmentType:=msoSegmentLine, EditingType:=msoEditingAu
        X1:=360, Y1:=200
    .ConvertToShape
End With
```



CanCheckIn Method

True if Microsoft PowerPoint can check in a specified presentation to a server.
Read/write **Boolean**.

expression.**CanCheckIn**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

To take advantage of the collaboration features built into PowerPoint, presentations must be stored on a Microsoft SharePoint Portal Server.

Example

This example checks the server to see if the specified presentation can be checked in and, if it can be, closes the presentation and checks it back into server.

```
Sub CheckInPresentation(strPresentation As String)
    If Presentations(strPresentation).CanCheckIn = True Then
        Presentations(strPresentation).CheckIn
        MsgBox strPresentation & " has been checked in."
    Else
        MsgBox strPresentation & " cannot be checked in " & _
            "at this time. Please try again later."
    End If
End Sub
```

To call the subroutine above, use the following subroutine and replace the "http://servername/workspace/report.ppt" file name with an actual file located on a server mentioned in the Remarks section above.

```
Sub CheckPPTIn()
    Call CheckInPresentation(strPresentation:= _
        "http://servername/workspace/report.ppt")
End Sub
```



CanCheckOut Method

True if Microsoft PowerPoint can check out a specified presentation from a server. Read/write **Boolean**.

expression.**CanCheckOut**(*FileName*)

expression Required. An expression that returns one of the objects in the Applies To list.

FileName Required **String**. The server path and name of the presentation .

Remarks

To take advantage of the collaboration features built into PowerPoint, presentations must be stored on a Microsoft SharePoint Portal Server.

Example

This example verifies that a presentation is not checked out by another user and that it can be checked out. If the presentation can be checked out, it copies the presentation to the local computer for editing.

```
Sub CheckOutPresentation(strPresentation As String)
    If Presentations.CanCheckOut(strPresentation) = True Then
        Presentations.CheckOut FileName:=strPresentation
    Else
        MsgBox "You are unable to check out this " & _
            "presentation at this time."
    End If
End Sub
```

To call the subroutine above, use the following subroutine and replace the "http://servername/workspace/report.ppt" file name with an actual file located on a server mentioned in the Remarks section above.

```
Sub CheckPPTOut()
    Call CheckOutPresentation(strPresentation:= _
        "http://servername/workspace/report.doc")
End Sub
```



Cell Method

Returns a [Cell](#) object that represents a cell in a table.

expression.**Cell**(*Row*, *Column*)

expression Required. An expression that returns a **Table** object.

Row Required **Long**. The number of the row in the table to return. Can be an integer between 1 and the number of rows in the table.

Column Required **Long**. The number of the column in the table to return. Can be an integer between 1 and the number of columns in the table.

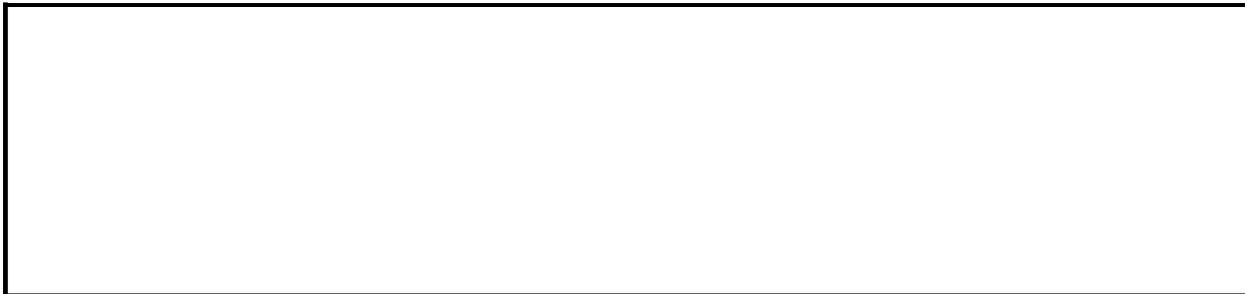
Example

This example creates a 3x3 table on a new slide in a new presentation and inserts text into the first cell of the table.

```
With Presentations.Add
  With .Slides.Add(1, ppLayoutBlank)
    .Shapes.AddTable(3, 3).Select
    .Shapes(1).Table.Cell(1, 1).Shape.TextFrame _
      .TextRange.Text = "Cell 1"
  End With
End With
```

This example sets the thickness of the bottom border of the cell in row 2, column 1 to two points.

```
ActivePresentation.Slides(2).Shapes(5).Table _
  .Cell(2, 1).Borders(ppBorderBottom).Weight = 2
```



[Show All](#)

ChangeCase Method

Changes the case of the specified text.

expression.**ChangeCase**(*Type*)

expression Required. An expression that returns a **TextRange** object.

Type Required [PpChangeCase](#). Specifies the way the case will be changed.

PpChangeCase can be one of these PpChangeCase constants.

ppCaseLower

ppCaseSentence

ppCaseTitle

ppCaseToggle

ppCaseUpper

Example

This example sets title case capitalization for the title on slide one in the the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes.Title.TextFrame _  
    .TextRange.ChangeCase ppCaseTitle
```



Characters Method

Returns a **TextRange** object that represents the specified subset of text characters. For information about counting or looping through the characters in a text range, see the [TextRange](#) object.

expression.**Characters**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first character in the returned range.

Length Optional **Long**. The number of characters to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first character and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one character.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first character in the specified range.

If ***Start*** is greater than the number of characters in the specified text, the returned range starts with the last character in the specified range.

If ***Length*** is greater than the number of characters from the specified starting character to the end of the text, the returned range contains all those characters.

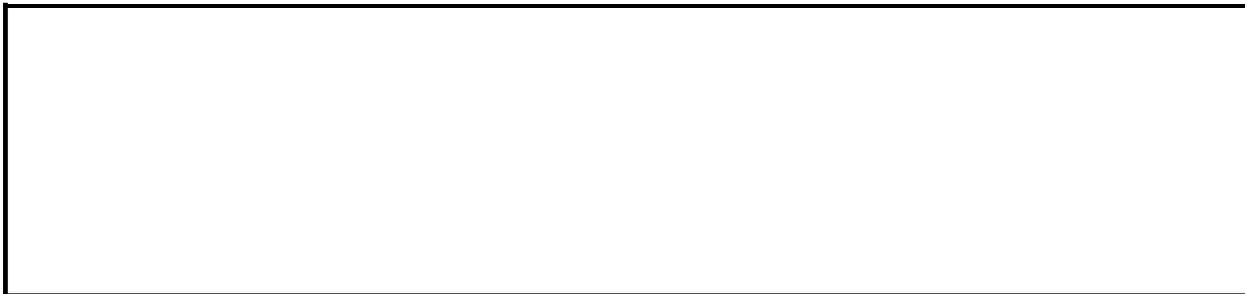
Example

This example sets the text for shape two on slide one in the active presentation and then makes the second character a subscript character with a 20-percent offset.

```
Dim charRange As TextRange
With Application.ActivePresentation.Slides(1).Shapes(2)
    Set charRange = .TextFrame.TextRange.InsertBefore("H2O")
    charRange.Characters(2).Font.BaselineOffset = -0.2
End With
```

This example formats every subscript character in shape two on slide one as bold.

```
With Application.ActivePresentation.Slides(1).Shapes(2) _
    .TextFrame.TextRange
    For i = 1 To .Characters.Count
        With .Characters(i).Font
            If .Subscript Then .Bold = True
        End With
    Next
End With
```



CheckIn Method

Returns a presentation from a local computer to a server, and sets the local file to read-only so that it cannot be edited locally.

expression.**CheckIn**(*SaveChanges*, *Comments*, *MakePublic*)

expression Required. An expression that returns one of the objects in the Applies To list.

SaveChanges Optional **Boolean**. **True** saves the presentation to the server location. The default value is **False**.

Comments Optional **Variant**. Comments for the revision of the presentation being checked in (only applies if **SaveChanges** equals **True**).

MakePublic Optional **Variant**. **True** allows the user to perform a publish on the presentation after being checked in. This submits the document for the approval process, which can eventually result in a version of the presentation being published to users with read-only rights to the presentation (only applies if **SaveChanges** equals **True**).

Remarks

To take advantage of the collaboration features built into Microsoft PowerPoint, presentations must be stored on a Microsoft SharePoint Portal Server.

Example

This example checks the server to see if the specified presentation can be checked in and, if so, closes the presentation and checks it back into server.

```
Sub CheckInPresentation(strPresentation As String)
    If Presentations(strPresentation).CanCheckIn = True Then
        Presentations(strPresentation).CheckIn
        MsgBox strPresentation & " has been checked in."
    Else
        MsgBox strPresentation & " cannot be checked in " & _
            "at this time. Please try again later."
    End If
End Sub
```

To call the subroutine above, use the following subroutine and replace the "http://servername/workspace/report.ppt" file name with an actual file located on a server mentioned in the Remarks section above.

```
Sub CheckInPresentation()
    Call CheckInPresentation(strPresentation:= _
        "http://servername/workspace/report.ppt")
End Sub
```



CheckOut Method

Copies a specified presentation from a server to a local computer for editing. Returns a **String** that represents the local path and filename of the presentation checked out.

expression.**CheckOut**(*FileName*)

expression Required. An expression that returns one of the objects in the Applies To list.

FileName Required **String**. The server path and name of the presentation.

Remarks

To take advantage of the collaboration features built into Microsoft PowerPoint, presentations must be stored on a Microsoft SharePoint Portal Server.

Example

This example verifies that a presentation is not checked out by another user and that it can be checked out. If the presentation can be checked out, it copies the presentation to the local computer for editing.

```
Sub CheckOutPresentation(strPresentation As String)
    Dim strFileName As String
    With Presentations
        If .CanCheckOut(strPresentation) = True Then
            .CheckOut FileName:=strPresentation
            .Open FileName:=strFileName
        Else
            MsgBox "You are unable to check out this " & _
                "presentation at this time."
        End If
    End Sub
```

To call the subroutine above, use the following subroutine and replace the "http://servername/workspace/report.ppt" file name for an actual file located on a server mentioned in the Remarks section above.

```
Sub CheckPPTOut()
    Call CheckOutPresentation(strPresentation:= _
        "http://servername/workspace/report.doc")
End Sub
```



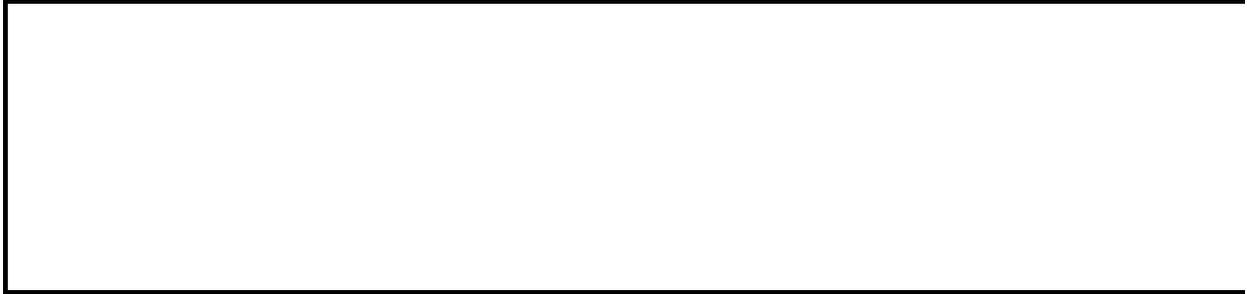
Clear Method

Clears the specified tab stop from the ruler and deletes it from the **TabStops** collection.

Example

This example clears all tab stops for the text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame _  
    .Ruler.TabStops  
    For i = .Count To 1 Step -1  
        .Item(i).Clear  
    Next  
End With
```



ClearAll Method

Clears all the print ranges from the **PrintRanges** collection. Use the [Add](#) method of the **PrintRanges** collection to add print ranges to the collection.

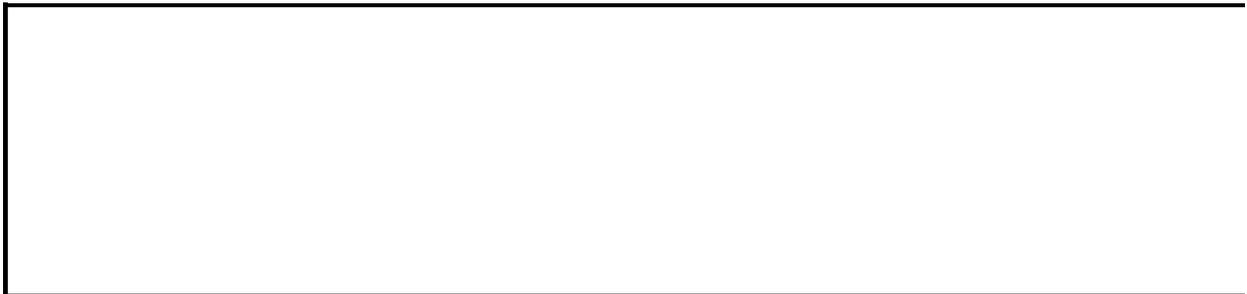
expression.**ClearAll**

expression Required. An expression that returns a **PrintRange** object.

Example

This example clears any previously defined print ranges in the active presentation; creates new print ranges that contain slide 1, slides 3 through 5, and slides 8 and 9; prints the newly defined slide ranges; and then clears the new print ranges.

```
With ActivePresentation.PrintOptions
    .RangeType = ppPrintSlideRange
    With .Ranges
        .ClearAll
        .Add 1, 1
        .Add 3, 5
        .Add 8, 9
        .Parent.Parent.PrintOut
        .ClearAll
    End With
End With
```



[Show All](#)

Clone Method

 [Clone method as it applies to the **Designs** object.](#)

Creates a copy of a [Design](#) object.

expression.Clone(pOriginal, Index)

expression Required. An expression that returns a [Designs](#) object.

pOriginal Required **Design** object. The original design.

Index Optional **Long**. The index location in the **Designs** collection into which the design will be copied. If *Index* is omitted, the cloned design is added to the end of the **Designs** collection.

 [Clone method as it applies to the **Sequence** object.](#)

Creates a copy of an [Effect](#) object, and adds it to the [Sequences](#) collection at the specified index.

expression.Clone(Effect, Index)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The animation effect to be cloned.

Index Optional **Long**. The position at which the cloned animation effect will be added to the **Sequences** collection. The default value is -1 (added to the end).

Example

 [As it applies to the **Designs** object.](#)

This example creates a design and clones the newly created design.

```
Sub CloneDesign()  
    Dim dsndesign1 As Design  
    Dim dsndesign2  
  
    Set dsndesign1 = ActivePresentation.Designs _  
        .Add(designName:="Design1")  
  
    Set dsndesign2 = ActivePresentation.Designs _  
        .Clone(pOriginal:=dsndesign1, Index:=1)  
  
End Sub
```

 [As it applies to the **Sequence** object.](#)

This example copies an animation effect. This example assumes an animation effect named "effDiamond" exists.

```
Sub CloneEffect()  
  
    ActivePresentation.Slides(1).TimeLine.MainSequence _  
        .Clone Effect:=effDiamond, Index:=-1  
  
End Sub
```



[Show All](#)

CloneNode Method

Clones a diagram node.

expression.CloneNode(CopyChildren, TargetNode, Pos)

expression Required. An expression that returns one of the objects in the Applies To list.

CopyChildren Required **Boolean**. **True** to include the diagram node's children.

TargetNode Required **DiagramNode** object. An expression that returns a **DiagramNode** that will be the source for the cloned diagram node.

Pos Optional [MsoRelativeNodePosition](#). If **TargetNode** is specified, where the node will be added, relative to **TargetNode**.

MsoRelativeNodePosition can be one of these **MsoRelativeNodePosition** constants.

msoAfterLastSibling

msoAfterNode *default*

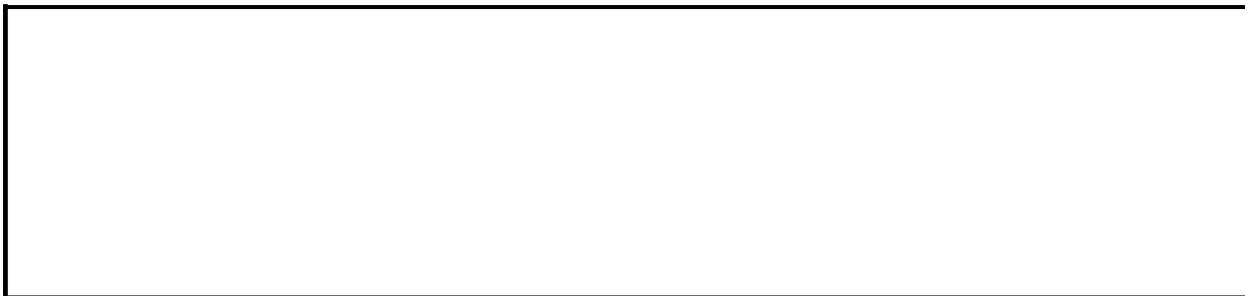
msoBeforeFirstSibling

msoBeforeNode

Example

The following example creates a diagram and clones the newest-created node.

```
Sub CloneANode()  
  
    Dim dgnNode As DiagramNode  
    Dim TdgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds cycle diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramCycle, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
    Set TdgnNode = new DiagramNode  
  
    'Adds three additional nodes to diagram  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically formats the diagram  
    dgnNode.Diagram.AutoFormat = msoTrue  
  
    'Clones the first child node without cloning associated child no  
    dgnNode.CloneNode CopyChildren:=False, TdgnNode  
  
End Sub
```



Close Method

Closes the specified document window, presentation, or open freeform drawing.

Caution When you use this method, PowerPoint will close an open presentation without prompting the user to save their work. To prevent the loss of work, use the **Save** method or the **SaveAs** method before you use the **Close** method.

expression.**Close**

expression Required. An expression that returns a **DocumentWindow** or **Presentation** object.

Example

This example closes all windows except the active window.

```
With Application.Windows
  For i = 2 To .Count
    .Item(i).Close
  Next
End With
```

This example closes Pres1.ppt without saving changes.

```
With Application.Presentations("pres1.ppt")
  .Saved = True
  .Close
End With
```

This example closes all open presentations.

```
With Application.Presentations
  For i = .Count To 1 Step -1
    .Item(i).Close
  Next
End With
```



[Show All](#)

Colors Method

Returns an [RGBColor](#) object that represents a single color in a color scheme.

expression.**Colors**(*SchemeColor*)

expression Required. An expression that returns a **ColorScheme** object.

SchemeColor Required [PpColorSchemeIndex](#). The individual color in the specified color scheme.

PpColorSchemeIndex can be one of these PpColorSchemeIndex constants.

ppAccent1

ppAccent2

ppAccent3

ppBackground

ppFill

ppForeground

ppNotSchemeColor

ppSchemeColorMixed

ppShadow

ppTitle

Example

This example sets the title color for slides one and three in the active presentation.

```
Set mySlides = ActivePresentation.Slides.Range(Array(1, 3))  
mySlides.ColorScheme.Colors(ppTitle).RGB = RGB(0, 255, 0)
```



[Show All](#)

Convert Method

Converts a diagram to a different diagram type.

expression.**Convert**(*Type*)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Required [MsoDiagramType](#). The type of diagram to convert to.

MsoDiagramType can be one of these **MsoDiagramType** constants.

msoDiagramCycle

msoDiagramMixed

msoDiagramOrgChart

msoDiagramPyramid

msoDiagramRadial

msoDiagramTarget

msoDiagramVenn

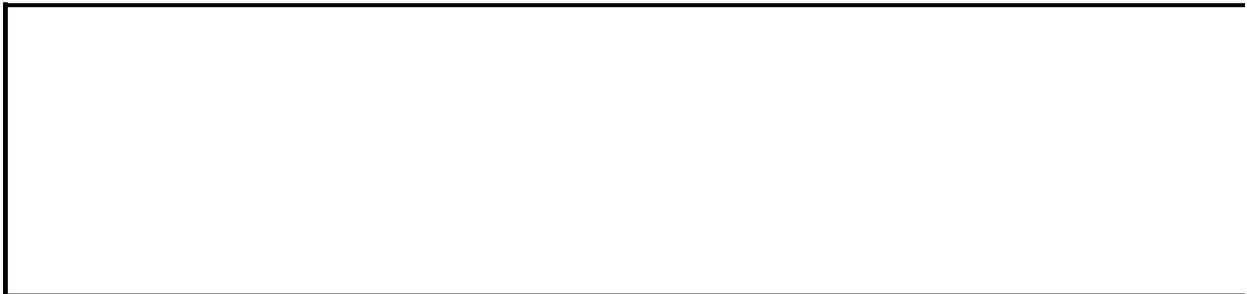
Remarks

This method generates an error if the value of the target diagram's [Type](#) property is an organization chart (**msoDiagramTypeOrgChart**).

Example

The following example adds a pyramid diagram to a slide and converts it to a radial diagram.

```
Sub ConvertPyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramPyramid, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional child nodes  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically formats the diagram and converts it to a radial d  
    With dgnNode.Diagram  
        .AutoFormat = msoTrue  
        .Convert Type:=msoDiagramRadial  
    End With  
  
End Sub
```



[Show All](#)

ConvertToAfterEffect Method

Specifies what an effect should do after it is finished. Returns an [Effect](#) object that represents an after effect.

expression.**ConvertToAfterEffect**(*Effect*, *After*, *DimColor*, *DimSchemeColor*)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The effect to which the after effect will be added.

After Required [MsoAnimAfterEffect](#). The behavior of the after effect.

MsoAnimAfterEffect can be one of these MsoAnimAfterEffect constants.

msoAnimAfterEffectDim

msoAnimAfterEffectHide

msoAnimAfterEffectHideOnNextClick

msoAnimAfterEffectMixed

msoAnimAfterEffectNone

DimColor Optional **MsoRGBType**. A single color to apply the after effect.

DimSchemeColor Optional [PpColorSchemeIndex](#). A predefined color scheme to apply to the after effect.

PpColorSchemeIndex can be one of these PpColorSchemeIndex constants.

ppAccent1

ppAccent2

ppAccent3

ppBackground

ppFill

ppForeground

ppNotSchemeColor *default*

ppSchemeColorMixed

ppShadow
ppTitle

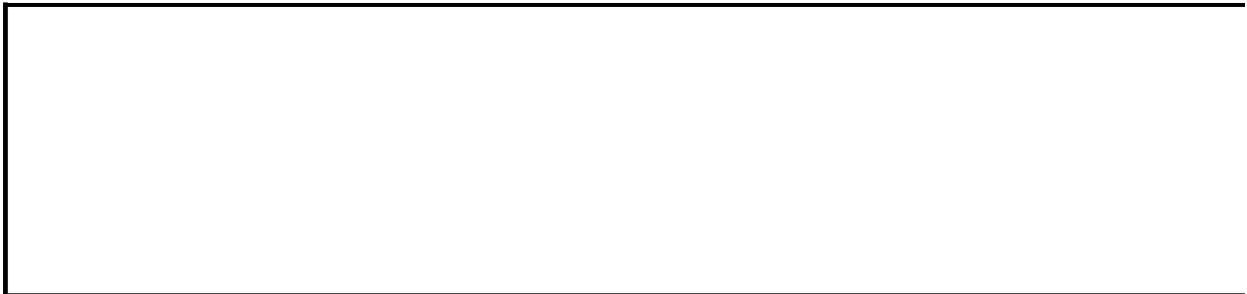
Remarks

Do not use both the *DimColor* and *DimSchemeColor* arguments in the same call to this method. An after effect may have one color, or use a predefined color scheme, but not both.

Example

The following example sets a dim color for an after effect on the first shape on the first slide in the active presentation. This example assume there is a shape on the first slide.

```
Sub ConvertToDim()  
  
    Dim shpSelected As Shape  
    Dim sldActive As Slide  
    Dim effConvert As Effect  
  
    Set sldActive = ActivePresentation.Slides(1)  
    Set shpSelected = sldActive.Shapes(1)  
  
    ' Add an animation effect.  
    Set effConvert = sldActive.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpSelected, effectId:=msoAnimEffectBounce)  
  
    ' Add a dim after effect.  
    Set effConvert = sldActive.TimeLine.MainSequence.ConvertToAfterE  
        (Effect:=effConvert, After:=msoAnimAfterEffectDim, _  
        DimColor:=RGB(Red:=255, Green:=255, Blue:=255))  
  
End Sub
```



[Show All](#)

ConvertToAnimateBackground Method

Determines whether the background will animate separately from, or in addition to, its accompanying text. Returns an [Effect](#) object representing the newly-modified animation effect.

expression.**ConvertToAnimateBackground**(*Effect*, *AnimateBackground*)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The animation effect to be applied to the background.

AnimateBackground Required [MsoTriState](#). Determines whether the text will animate separately from the background.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Animates text separately from the background.

msoTriStateMixed

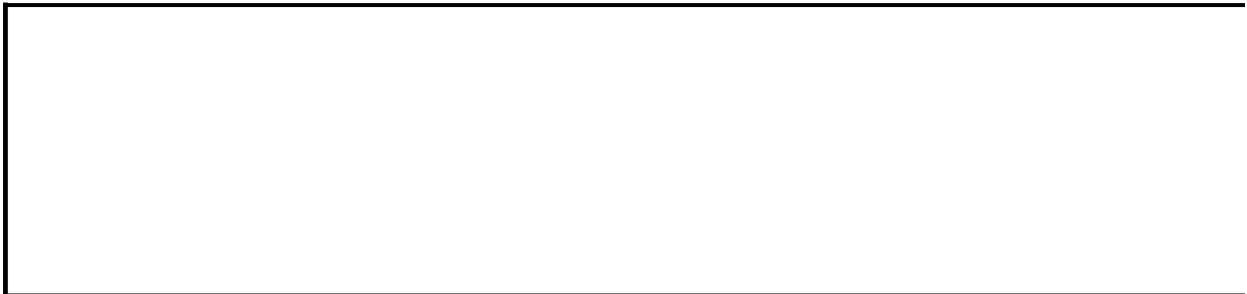
msoTriStateToggle

msoTrue Animates text along with the background.

Example

This example creates a text effect for the first shape on the first slide in the active presentation, and animates the text in the shape separately from the background. This example assumes there is a shape on the first slide, and that the shape has text inside of it.

```
Sub AnimateText()  
  
    Dim timeMain As TimeLine  
    Dim shpActive As Shape  
  
    Set shpActive = ActivePresentation.Slides(1).Shapes(1)  
    Set timeMain = ActivePresentation.Slides(1).TimeLine  
  
    ' Add a blast effect to the text, and animate the text separatel  
    ' from the background.  
    timeMain.MainSequence.ConvertToAnimateBackground _  
        Effect:=timeMain.MainSequence.AddEffect(Shape:=shpActive, _  
            effectid:=msoAnimEffectBlast), _  
        AnimateBackGround:=msoFalse  
  
End Sub
```



[Show All](#)

ConvertToAnimateInReverse Method

Determines whether text will be animated in reverse order. Returns an [Effect](#) object representing the text animation.

expression.**ConvertToAnimateInReverse**(*Effect*, *animateInReverse*)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The animation effect to which the reversal will apply.

animateInReverse Required [MsoTriState](#). Determines the text animation order.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The text animates in normal order.

msoTriStateMixed

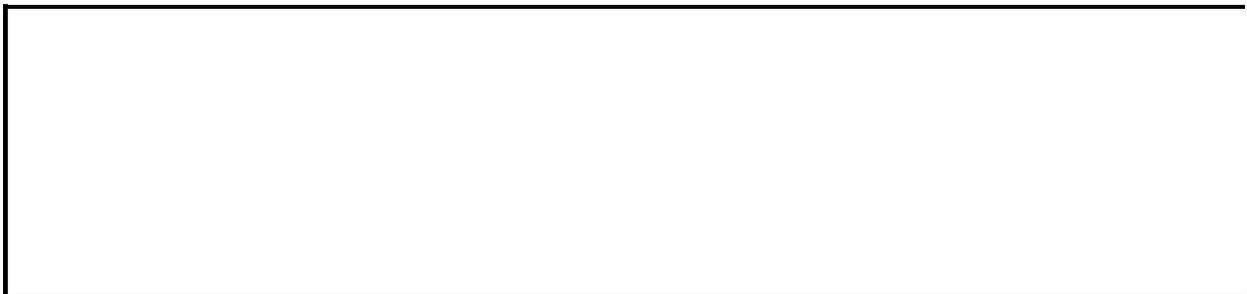
msoTriStateToggle

msoTrue The text animates in reverse order.

Example

This example creates a shape with text on a slide and adds a random animation to the shape, ensuring the shape's text animates in reverse.

```
Sub AnimateInReverse()  
  
    Dim sldActive As Slide  
    Dim timeMain As TimeLine  
    Dim shpRect As Shape  
  
    ' Create a slide, add a rectangular shape to the slide, and  
    ' access the slide's animation timeline.  
    With ActivePresentation  
        Set sldActive = .Slides.Add(Index:=1, Layout:=ppLayoutBlank)  
        Set shpRect = sldActive.Shapes.AddShape(Type:=msoShapeRectan  
            Left:=100, Top:=100, Width:=300, Height:=150)  
        Set timeMain = sldActive.TimeLine  
    End With  
  
    shpRect.TextFrame.TextRange.Text = "This is a rectangle."  
  
    ' Add a random animation effect to the rectangle,  
    ' and animate the text in reverse.  
    With timeMain.MainSequence  
        .ConvertToAnimateInReverse _  
            Effect:=.AddEffect(Shape:=shpRect, effectId:=msoAnimEffe  
                AnimateInReverse:=msoTrue  
    End With  
  
End Sub
```



[Show All](#)

ConvertToBuildLevel Method

Changes the build level information for a specified animation effect. Returns an **Effect** object that represents the build level information.

expression.**ConvertToBuildLevel**(*Effect*, *Level*)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The specified animation effect.

Level Required [MsoAnimateByLevel](#). The animation build level.

MsoAnimateByLevel can be one of these MsoAnimateByLevel constants.

msoAnimateChartAllAtOnce

msoAnimateChartByCategory

msoAnimateChartByCategoryElements

msoAnimateChartBySeries

msoAnimateChartBySeriesElements

msoAnimateDiagramAllAtOnce

msoAnimateDiagramBreadthByLevel

msoAnimateDiagramBreadthByNode

msoAnimateDiagramClockwise

msoAnimateDiagramClockwiseIn

msoAnimateDiagramClockwiseOut

msoAnimateDiagramCounterClockwise

msoAnimateDiagramCounterClockwiseIn

msoAnimateDiagramCounterClockwiseOut

msoAnimateDiagramDepthByBranch

msoAnimateDiagramDepthByNode

msoAnimateDiagramDown

msoAnimateDiagramInByRing

msoAnimateDiagramOutByRing

msoAnimateDiagramUp
msoAnimateLevelMixed
msoAnimateTextByAllLevels
msoAnimateTextByFifthLevel
msoAnimateTextByFirstLevel
msoAnimateTextByFourthLevel
msoAnimateTextBySecondLevel
msoAnimateTextByThirdLevel
msoAnimationLevelNone

Remarks

Changing build level information for an effect invalidates any existing effects.

Example

The following example changes the build level information for an animation effect, making the original effect invalid.

```
Sub ConvertBuildLevel()  
    Dim sldFirst As Slide  
    Dim shpFirst As Shape  
    Dim effFirst As Effect  
    Dim effConvert As Effect  
  
    Set sldFirst = ActiveWindow.Selection.SlideRange(1)  
    Set shpFirst = sldFirst.Shapes(1)  
    Set effFirst = sldFirst.TimeLine.MainSequence _  
        .AddEffect(Shape:=shpFirst, EffectID:=msoAnimEffectAscend)  
    Set effConvert = sldFirst.TimeLine.MainSequence _  
        .ConvertToBuildLevel(Effect:=effFirst, _  
            Level:=msoAnimateTextByFirstLevel)  
End Sub
```



ConvertToShape Method

Creates a shape that has the geometric characteristics of the specified **FreeformBuilder** object. Returns a [Shape](#) object that represents the new shape.

Note You must apply the [AddNodes](#) method to a **FreeformBuilder** object at least once before you use the **ConvertToShape** method.

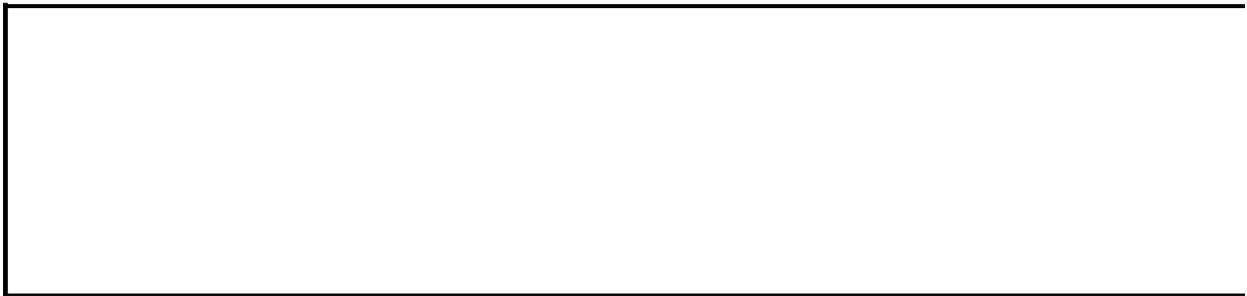
expression.**ConvertToShape**

expression Required. An expression that returns a **FreeformBuilder** object.

Example

This example adds a freeform with five vertices to the first slide in the active presentation.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.BuildFreeform(msoEditingCorner, 360, 200)
    .AddNodes msoSegmentCurve, _
        msoEditingCorner, 380, 230, 400, 250, 450, 300
    .AddNodes msoSegmentCurve, msoEditingAuto, 480, 200
    .AddNodes msoSegmentLine, msoEditingAuto, 480, 400
    .AddNodes msoSegmentLine, msoEditingAuto, 360, 200
    .ConvertToShape
End With
```



[Show All](#)

ConvertToTextUnitEffect Method

Returns an [Effect](#) object that represents how text should animate.

expression.**ConvertToTextUnitEffect**(*Effect*, *unitEffect*)

expression Required. An expression that returns a [Sequence](#) object.

Effect Required **Effect** object. The animation effect to which the text unit effect applies.

unitEffect Required [MsoAnimTextUnitEffect](#). How the text should animate.

MsoAnimTextUnitEffect can be one of these MsoAnimTextUnitEffect constants.

msoAnimTextUnitEffectByCharacter

msoAnimTextUnitEffectByParagraph

msoAnimTextUnitEffectByWord

msoAnimTextUnitEffectMixed

Example

This example adds an animation to a given shape and animates its accompanying text by character.

```
Sub NewTextUnitEffect()  
    Dim shpFirst As Shape  
    Dim tmlMain As TimeLine  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlMain = ActivePresentation.Slides(1).TimeLine  
  
    tmlMain.MainSequence.ConvertToTextUnitEffect _  
        Effect:=tmlMain.MainSequence.AddEffect(Shape:=shpFirst, _  
            EffectID:=msoAnimEffectRandomEffects), _  
        unitEffect:=msoAnimTextUnitEffectByCharacter  
End Sub
```



Copy Method

Copies the specified object to the Clipboard.

expression.**Copy**

expression Required. An expression that returns a **Selection**, **Shape**, **ShapeRange**, **Slide**, **SlideRange**, or **TextRange** object.

Remarks

Use the [Paste](#) method to paste the contents of the Clipboard.

Example

This example copies the selection in window one to the Clipboard and then pastes it into the view in window two. If the Clipboard contents cannot be pasted into the view in window two— for example, if you try to paste a shape into slide sorter view— this example fails.

```
Windows(1).Selection.Copy  
Windows(2).View.Paste
```

This example copies shapes one and two on slide one in the active presentation to the Clipboard and then pastes the copies onto slide two.

```
With ActivePresentation  
    .Slides(1).Shapes.Range(Array(1, 2)).Copy  
    .Slides(2).Shapes.Paste  
End With
```

This example copies slide one in the active presentation to the Clipboard.

```
ActivePresentation.Slides(1).Copy
```

This example copies the text in shape one on slide one in the active presentation to the Clipboard.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame.TextRange.Copy
```



[Show All](#)

CreateNewDocument Method

Creates a new Web presentation associated with the specified hyperlink.

expression.CreateNewDocument(**FileName**, **EditNow**, **Overwrite**)

expression Required. An expression that returns a **Hyperlink** object.

FileName Required **String**. The path and file name of the document.

EditNow Required **MsoTriState**. Determines whether the document is opened immediately in its associated editor.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Do not open the document immediately.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Open the document immediately in its associated editor to modify it.

Overwrite Required **MsoTriState**. Determines whether any existing file of the same name in the same folder is overwritten.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. Preserve any existing file of the same name in the same folder, requiring a new file name to be specified in the **FileName** argument.

msoTriStateMixed

msoTriStateToggle

msoTrue Overwrite any existing file of the same name in the same folder.

Example

This example creates a new Web presentation to be associated with hyperlink one on slide one. The new presentation is called Brittany.ppt, and it overwrites any file of the same name in the HTMLPres folder. The new presentation document is loaded into Microsoft PowerPoint immediately for editing.

```
ActivePresentation.Slides(1).Hyperlinks(1).CreateNewDocument _  
    FileName:="C:\HTMLPres\Brittany.ppt", _  
    EditNow:=msoTrue, _  
    Overwrite:=msoTrue
```



CustomDrop Method

Sets the vertical distance (in points) from the edge of the text bounding box to the place where the callout line attaches to the text box. This distance is measured from the top of the text box unless the **AutoAttach** property is set to **True** and the text box is to the left of the origin of the callout line (the place that the callout points to). In this case the drop distance is measured from the bottom of the text box.

expression.**CustomDrop**(*Drop*)

expression Required. An expression that returns a **CalloutFormat** object.

Drop Required **Single**. The drop distance, in points.

Example

This example sets the custom drop distance to 14 points, and specifies that the drop distance always be measured from the top. For the example to work, shape three must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Callout
    .CustomDrop 14
    .AutoAttach = False
End With
```



CustomLength Method

Specifies that the first segment of the callout line (the segment attached to the text callout box) retain a fixed length whenever the callout is moved. Use the [AutomaticLength](#) method to specify that the first segment of the callout line be scaled automatically whenever the callout is moved. Applies only to callouts whose lines consist of more than one segment (types **msoCalloutThree** and **msoCalloutFour**).

expression.**CustomLength**(*Length*)

expression Required. An expression that returns a **CalloutFormat** object.

Length Required **Single**. The length of the first segment of the callout, in points.

Remarks

Applying this method sets the [AutoLength](#) property to **False** and sets the [Length](#) property to the value specified for the **Length** argument.

Example

This example toggles between an automatically scaling first segment and one with a fixed length for the callout line for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```



Cut Method

Deletes the specified object and places it on the Clipboard.

expression.**Cut**

expression Required. An expression that returns a **Selection**, **Shape**, **ShapeRange**, **Slide**, **SlideRange**, or **TextRange** object.

Example

This example deletes the selection in window one and places a copy of it on the Clipboard.

```
Windows(1).Selection.Cut
```

This example deletes shapes one and two from slide one in the active presentation, places copies of them on the Clipboard, and then pastes the copies onto slide two.

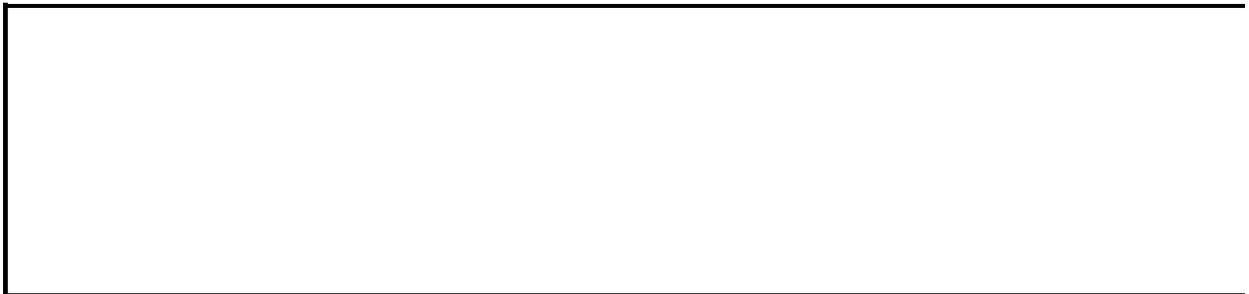
```
With ActivePresentation  
    .Slides(1).Shapes.Range(Array(1, 2)).Cut  
    .Slides(2).Shapes.Paste  
End With
```

This example deletes slide one from the active presentation and places a copy of it on the Clipboard.

```
ActivePresentation.Slides(1).Cut
```

This example deletes the text in shape one on slide one in the active presentation and places a copy of it on the Clipboard.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame.TextRange.Cut
```



[Show All](#)

Delete Method

 [Delete method as it applies to the **ShapeNodes** object.](#)

Deletes a shape node.

expression.**Delete**(*Index*)

expression Required. An expression that returns a [ShapeNodes](#) object.

Index Required **Long**. Specifies the node to be deleted. The segment following that node will also be deleted. If the node is a control point of a curve, the curve and all of its nodes will be deleted.

 [Delete method as it applies to the **Tags** object.](#)

Deletes a tag.

expression.**Delete**(*Name*)

expression Required. An expression that returns a [Tags](#) object.

Name Required **String**. Specifies the name of the tag to be deleted.

 [Delete method as it applies to the all other objects in the **Applies To** list.](#)

Deletes the specified object.

expression.**Delete**

expression Required. An expression that returns one of the objects in the **Applies To** list except for the **ShapeNodes** and **Tags** objects.

Remarks

Attempting to delete the only existing row or column in a table will result in a run-time error.

Example

 [As it applies to the **Shape** object.](#)

This example deletes all freeform shapes from slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes
  For intShape = .Count To 1 Step -1
    With .Item(intShape)
      If .Type = msoFreeform Then .Delete
    End With
  Next
End With
```



DeleteText Method

Deletes the text associated with the specified shape.

expression.**DeleteText**

expression Required. An expression that returns a **TextFrame** object.

Example

If shape two on myDocument contains text, this example deletes the text.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(2).TextFrame.DeleteText
```



[Show All](#)

Distribute Method

Evenly distributes the shapes in the specified range of shapes. You can specify whether you want to distribute the shapes horizontally or vertically and whether you want to distribute them over the entire slide or just over the space they originally occupy.

expression.**Distribute**(*DistributeCmd*, *RelativeTo*)

expression Required. An expression that returns a **ShapeRange** object.

DistributeCmd Required [MsoDistributeCmd](#). Specifies whether shapes in the range are to be distributed horizontally or vertically.

MsoDistributeCmd can be one of these MsoDistributeCmd constants.

msoDistributeHorizontally

msoDistributeVertically

RelativeTo Required [MsoTriState](#). Determines whether shapes are distributed evenly over the entire horizontal or vertical space on the slide.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Distributes the shapes within the horizontal or vertical space that the range of shapes originally occupies.

msoTriStateMixed

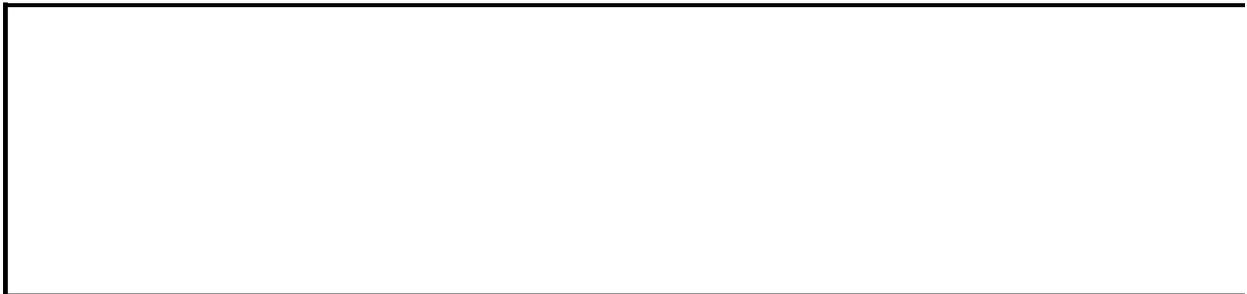
msoTriStateToggle

msoTrue Distributes the shapes evenly over the entire horizontal or vertical space on the slide.

Example

This example defines a shape range that contains all the AutoShapes on the myDocument and then horizontally distributes the shapes in this range.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    numShapes = .Count
    If numShapes > 1 Then
        numAutoShapes = 0
        ReDim autoShpArray(1 To numShapes)
        For i = 1 To numShapes
            If .Item(i).Type = msoAutoShape Then
                numAutoShapes = numAutoShapes + 1
                autoShpArray(numAutoShapes) = .Item(i).Name
            End If
        Next
        If numAutoShapes > 1 Then
            ReDim Preserve autoShpArray(1 To numAutoShapes)
            Set asRange = .Range(autoShpArray)
            asRange.Distribute msoDistributeHorizontally, msoFalse
        End If
    End If
End With
```



[Show All](#)

DoVerb Method

Requests that an OLE object perform one of its [verbs](#). Use the [ObjectVerbs](#) property to determine the available verbs for an OLE object.

expression.**DoVerb**(*Index*)

expression Required. An expression that returns an **OLEFormat** object.

Index Optional **Integer**. The verb to perform. If this argument is omitted, the default verb is performed.

Example

This example performs the default verb for shape three on slide one in the active presentation if shape three is a linked or embedded OLE object.

```
With ActivePresentation.Slides(1).Shapes(3)
  If .Type = msoEmbeddedOLEObject Or _
    .Type = msoLinkedOLEObject Then
    .OLEFormat.DoVerb
  End If
End With
```

This example performs the verb "Open" for shape three on slide one in the active presentation if shape three is an OLE object that supports the verb "Open."

```
With ActivePresentation.Slides(1).Shapes(3)
  If .Type = msoEmbeddedOLEObject Or _
    .Type = msoLinkedOLEObject Then
    For Each sVerb In .OLEFormat.ObjectVerbs
      nCount = nCount + 1
      If sVerb = "Open" Then
        .OLEFormat.DoVerb nCount
        Exit For
      End If
    Next
  End If
End With
```



DrawLine Method

Draws a line in the specified slide show view.

expression.**DrawLine**(*BeginX*, *BeginY*, *EndX*, *Height*)

expression Required. An expression that returns a **SlideShowView** object.

BeginX , *BeginY* Required **Single**. The position (in points) of the line's starting point relative to the upper-left corner of the slide.

EndX , *EndY* Required **Single**. The position (in points) of the line's ending point relative to the upper-left corner of the slide.

Example

This example draws a line in slide show window one.

```
SlideShowWindows(1).View.DrawLine 5, 5, 250, 250
```



[Show All](#)

Duplicate Method

 [Duplicate method as it applies to the **Shape** and **ShapeRange** objects.](#)

Creates a duplicate of the specified **Shape** or **ShapeRange** object, adds the new shape or range of shapes to the **Shapes** collection immediately after the shape or range of shapes specified originally, and then returns the new **Shape** or **ShapeRange** object.

expression.**Duplicate**

expression Required. An expression that returns one of the above objects.

 [Duplicate method as it applies to the **Slide** and **SlideRange** objects.](#)

Creates a duplicate of the specified **Slide** or **SlideRange** object, adds the new slide or range of slides to the **Slides** collection immediately after the slide or slide range specified originally, and then returns a **Slide** or **SlideRange** object that represents the duplicate slide or slides.

expression.**Duplicate**

expression Required. An expression that returns one of the above objects.

Example

[As it applies to the **Shape** and **ShapeRange** objects.](#)

This example adds a new, blank slide at the end of the active presentation, adds a diamond shape to the new slide, duplicates the diamond, and then sets properties for the duplicate. The first diamond will have the default fill color for the active color scheme; the second diamond will be offset from the first one and will have the default shadow color.

```
Set mySlides = ActivePresentation.Slides
Set newSlide = mySlides.Add(mySlides.Count + 1, ppLayoutBlank)
Set firstObj = newSlide.Shapes _
    .AddShape(msoShapeDiamond, 10, 10, 250, 350)
With firstObj.Duplicate
    .Left = 150
    .Fill.ForeColor.SchemeColor = ppShadow
End With
```



[As it applies to the **Slide** and **SlideRange** objects.](#)

This example creates a duplicate of slide one in the active presentation and then sets the background shading and the title text of the new slide. The new slide will be slide two in the presentation.

```
Set newSlide = ActivePresentation.Slides(1).Duplicate
With newSlide
    .Background.Fill.PresetGradient msoGradientVertical, _
        1, msoGradientGold
    .Shapes.Title.TextFrame.TextRange _
        .Text = "Second Quarter Earnings"
End With
```



EndConnect Method

Attaches the end of the specified connector to a specified shape. If there's already a connection between the end of the connector and another shape, that connection is broken. If the end of the connector isn't already positioned at the specified connecting site, this method moves the end of the connector to the connecting site and adjusts the size and position of the connector. Use the **BeginConnect** method to attach the beginning of the connector to a shape.

expression.**EndConnect**(*ConnectedShape*, *ConnectionSite*)

expression Required. An expression that returns a **ConnectorFormat** object.

ConnectedShape Required **Shape** object. The shape to attach the end of the connector to. The specified **Shape** object must be in the same **Shapes** collection as the connector.

ConnectionSite Required **Long**. A connection site on the shape specified by **ConnectedShape**. Must be an integer between 1 and the integer returned by the **ConnectionSiteCount** property of the specified shape. If you want the connector to automatically find the shortest path between the two shapes it connects, specify any valid integer for this argument and then use the [RerouteConnections](#) method after the connector is attached to shapes at both ends.

Remarks

When you attach a connector to an object, the size and position of the connector are automatically adjusted, if necessary.

Example

This example adds two rectangles to the first slide in the active presentation and connects them with a curved connector. Notice that the **RerouteConnections** method makes it irrelevant what values you supply for the *ConnectionSite* arguments used with the **BeginConnect** and **EndConnect** methods.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 100, 100) _
    .ConnectorFormat
        .BeginConnect ConnectedShape:=firstRect, ConnectionSite:=1
        .EndConnect ConnectedShape:=secondRect, ConnectionSite:=1
    .Parent.RerouteConnections
End With
```



EndDisconnect Method

Detaches the end of the specified connector from the shape it's attached to. This method doesn't alter the size or position of the connector: the end of the connector remains positioned at a connection site but is no longer connected. Use the [BeginDisconnect](#) method to detach the beginning of the connector from a shape.

expression.**EndDisconnect**

expression Required. An expression that returns a **ConnectorFormat** object.

Example

This example adds two rectangles to the first slide in the active presentation, attaches them with a connector, automatically reroutes the connector along the shortest path, and then detaches the connector from the rectangles.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 0, 0).ConnectorFormat
    .BeginConnect firstRect, 1
    .EndConnect secondRect, 1
    .Parent.RerouteConnections
    .BeginDisconnect
    .EndDisconnect
End With
```



EndNamedShow Method

Switches from running a custom, or named, slide show to running the entire presentation of which the custom show is a subset. When the slide show advances from the current slide, the next slide displayed will be the next one in the entire presentation, not the next one in the custom slide show.

expression.**EndNamedShow**

expression Required. An expression that returns a **SlideShowView** object.

Example

If a custom slide show is currently running in slide show window one, this example redefines the slide show to include all the slides in the presentation from which the slides in the custom show were selected.

```
SlideShowWindows(1).View.EndNamedShow
```



EndReview Method

Terminates a review of a file that has been sent for review using the [SendForReview](#) method or that has been automatically placed in a review cycle.

expression.**EndReview**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example terminates the review of the active presentation. When executed, this procedure displays a message asking if you want to end the review. This example assumes the active presentation is in a review cycle.

```
Sub EndPPTRev()  
    ActivePresentation.EndReview  
End Sub
```



EraseDrawing Method

Removes lines drawn during a slide show using either the [DrawLine](#) method or the pen tool.

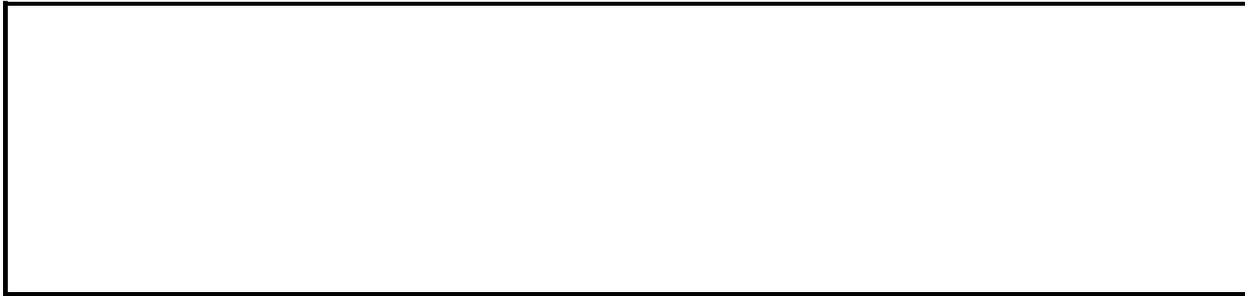
expression.**EraseDrawing**

expression Required. An expression that returns a **SlideShowView** object.

Example

This example erases any lines that have been drawn in slide show window one using either the **DrawLine** method or the pen tool.

```
SlideShowWindows(1).View.EraseDrawing
```



Exit Method

Ends the specified slide show.

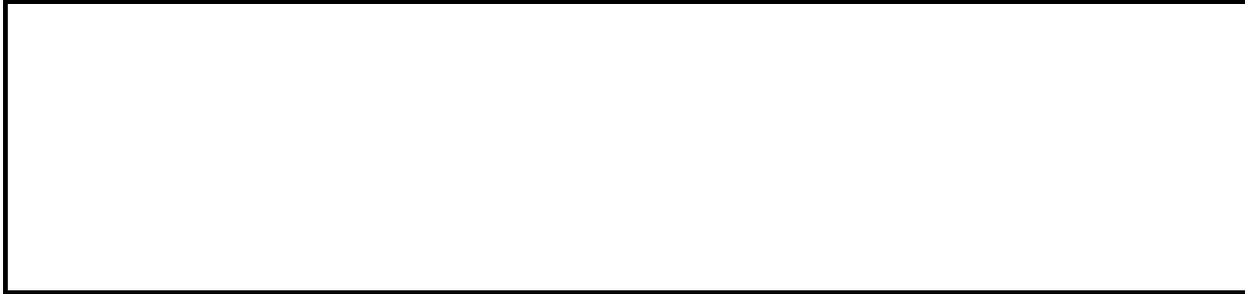
expression.**Exit**

expression Required. An expression that returns a **SlideShowView** object.

Example

This example ends the slide show that's running in slide show window one.

```
SlideShowWindows(1).View.Exit
```



[Show All](#)

Export Method

 [Export method as it applies to the **Slide** and **SlideRange** objects.](#)

Exports a slide or range of slides using the specified graphics filter, and saves the exported file under the specified file name.

expression.Export(FileName, FilterName, ScaleWidth, ScaleHeight)

expression Required. An expression that returns one of the above objects.

FileName Required **String**. The name of the file to be exported and saved to disk. You can include a full path; if you don't, Microsoft PowerPoint creates a file in the current folder.

FilterName Required **String**. The graphics format in which you want to export slides. The specified graphics format must have an export filter registered in the Windows registry. You can specify either the registered extension or the registered filter name. PowerPoint will first search for a matching extension in the registry. If no extension that matches the specified string is found, PowerPoint will look for a filter name that matches.

ScaleWidth Optional **Long**. The width in pixels of an exported slide.

ScaleHeight Optional **Long**. The height in pixels of an exported slide.

Remarks

Exporting a presentation doesn't set the [Saved](#) property of a presentation to **True**.

PowerPoint uses the specified graphics filter to save each individual slide. The names of the slides exported and saved to disk are determined by PowerPoint. They're typically saved as Slide1.wmf, Slide2.wmf, and so on. The path of the saved files is specified in the *FileName* argument.



[Export method as it applies to the **Presentation** object.](#)

Exports each slide in the presentation, using the specified graphics filter, and saves the exported files in the specified folder.

expression.Export(Path, FilterName, ScaleWidth, ScaleHeight)

expression Required. An expression that returns a [Presentation](#) object.

Path Required **String**. The path of the folder where you want to save the exported slides. You can include a full path; if you don't do this, Microsoft PowerPoint creates a subfolder in the current folder for the exported slides.

FilterName Required **String**. The graphics format in which you want to export slides. The specified graphics format must have an export filter registered in the Windows registry. You can specify either the registered extension or the registered filter name. PowerPoint will first search for a matching extension in the registry. If no extension that matches the specified string is found, PowerPoint will look for a filter name that matches.

ScaleWidth Optional **Long**. The width in pixels of an exported slide.

ScaleHeight Optional **Long**. The height in pixels of an exported slide.

Remarks

Exporting a presentation doesn't set the [Saved](#) property of a presentation to **True**.

PowerPoint uses the specified graphics filter to save each individual slide in the presentation. The names of the slides exported and saved to disk are determined by PowerPoint. They're typically saved as Slide1.wmf, Slide2.wmf, and so on. The path of the saved files is specified in the ***Path*** argument.

Example

 [As it applies to the **Presentation** object.](#)

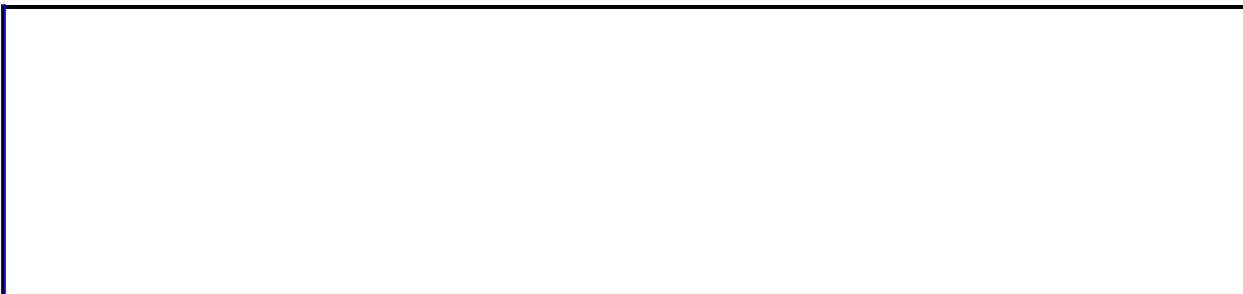
This example saves the active presentation as a Microsoft PowerPoint presentation and then exports each slide in the presentation as a Portable Network Graphics (PNG) file that will be saved in the Current Work folder. The example also exports each slide with a height of 100 pixels and a width of 100 pixels.

```
With ActivePresentation
    .SaveAs FileName:="c:\Current Work\Annual Sales", _
        FileFormat:=ppSaveAsPresentation
    .Export Path:="c:\Current Work", FilterName:="png", _
        ScaleWidth:=100, ScaleHeight:=100
End With
```

 [As it applies to the **Slide** object.](#)

This example exports slide three in the active presentation to disk in the JPEG graphic format. The slide is saved as Slide 3 of Annual Sales.jpg.

```
With Application.ActivePresentation.Slides(3)
    .Export "c:\my documents\Graphic Format\" & _
        "Slide 3 of Annual Sales", "JPG"
End With
```



[Show All](#)

Find Method

Finds the specified text in a text range, and returns a [TextRange](#) object that represents the first text range where the text is found. Returns **Nothing** if no match is found.

expression.**Find**(*FindWhat*, *After*, *MatchCase*, *WholeWords*)

expression Required. An expression that returns a **TextRange** object.

FindWhat Required **String**. The text to search for.

After Optional **Long**. The position of the character (in the specified text range) after which you want to search for the next occurrence of **FindWhat**. For example, if you want to search from the fifth character of the text range, specify 4 for **After**. If this argument is omitted, the first character of the text range is used as the starting point for the search.

MatchCase Optional [MsoTriState](#). **MsoTrue** for the search to distinguish between uppercase and lowercase characters.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Search matches the case of the letters in the **FindWhat** argument.

WholeWords Optional [MsoTriState](#). **MsoTrue** for the search to find only whole words and not parts of larger words as well.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Search finds only whole words, not parts of larger words.

Example

This example finds every occurrence of "CompanyX" in the active presentation and formats it as bold.

```
For Each sld In Application.ActivePresentation.Slides
  For Each shp In sld.Shapes
    If shp.HasTextFrame Then
      Set txtRng = shp.TextFrame.TextRange
      Set foundText = txtRng.Find(FindWhat:="CompanyX")
      Do While Not (foundText Is Nothing)
        With foundText
          .Font.Bold = True
          Set foundText = _
            txtRng.Find(FindWhat:="CompanyX", _
              After:=.Start + .Length - 1)
        End With
      Loop
    End If
  Next
Next
```



FindBySlideID Method

Returns a [Slide](#) object that represents the slide with the specified slide ID number. Each slide is automatically assigned a unique slide ID number when it's created. Use the [SlideID](#) property to return a slide's ID number.

expression.**FindBySlideID**(*SlideID*)

expression Required. An expression that returns a **Slides** collection.

SlideID Required **Long**. Specifies the ID number of the slide you want to return. PowerPoint assigns this number when the slide is created.

Remarks

Unlike the [SlideIndex](#) property, the **SlideID** property of a **Slide** object won't change when you add slides to the presentation or rearrange the slides in the presentation. Therefore, using the **FindBySlideID** method with the slide ID number can be a more reliable way to return a specific **Slide** object from a **Slides** collection than using the [Item](#) method with the slide's index number.

Example

This example demonstrates how to retrieve the unique ID number for a **Slide** object and then use this number to return that **Slide** object from the **Slides** collection.

```
Set gslides = ActivePresentation.Slides
```

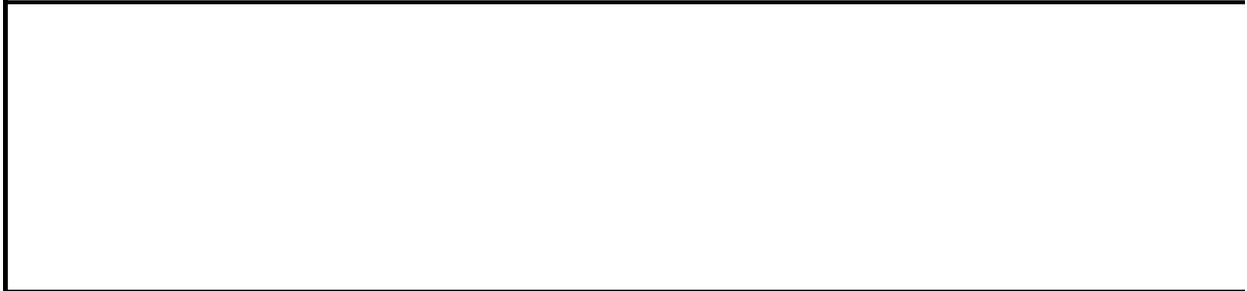
```
'Get slide ID
```

```
graphSlideID = gslides.Add(2, ppLayoutChart).SlideID
```

```
gslides.FindBySlideID(graphSlideID) _
```

```
.SlideShowTransition.EntryEffect = _
```

```
ppEffectCoverLeft      'Use ID to return specific slide
```



FindFirstAnimationFor Method

Returns an [Effect](#) object that represents the first animation for a given shape.

expression.**FindFirstAnimationFor**(*Shape*)

expression Required. An expression that returns one of the objects in the Applies To list.

Shape Required **Shape** object. The shape for which to find the first animation.

Example

The following example finds and deletes the first animation for a the first shape on the first slide. This example assumes that at least one animation effect exists for the specified shape.

```
Sub FindFirstAnimation()  
    Dim sldFirst As Slide  
    Dim shpFirst As Shape  
    Dim effFirst As Effect  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpFirst = sldFirst.Shapes(1)  
  
    Set effFirst = sldFirst.TimeLine.MainSequence _  
        .FindFirstAnimationFor(Shape:=shpFirst)  
    effFirst.Delete  
End Sub
```



FindFirstAnimationForClick Method

Returns an [Effect](#) object that represents the first animation started by the specified click number.

expression.**FindFirstAnimationForClick**(*Click*)

expression Required. An expression that returns one of the objects in the Applies To list.

Click Required **Long**. The specified click number.

Example

The following example finds the first animation for the first click on the first slide and changes the effect to a bounce.

```
Sub FindFirstAnimationClick()  
    Dim sldFirst As Slide  
    Dim effClick As Effect  
  
    Set sldFirst = ActivePresentation.Slides(1)  
  
    Set effClick = sldFirst.TimeLine.MainSequence _  
        .FindFirstAnimationForClick(Click:=1)  
    effClick.EffectType = msoAnimEffectBounce  
End Sub
```



First Method

Sets the specified slide show view to display the first slide in the presentation.

expression.**First**

expression Required. An expression that returns a **SlideShowView** object.

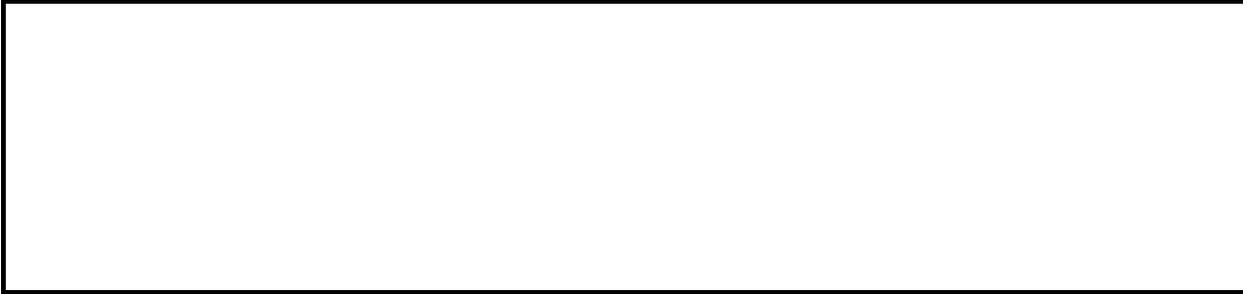
Remarks

If you use the **First** method to switch from one slide to another during a slide show, when you return to the original slide, its animation picks up where it left off.

Example

This example sets slide show window one to display the first slide in the presentation.

```
SlideShowWindows(1).View.First
```



FitToPage Method

Adjusts the size of the specified document window to accommodate the information that's currently displayed.

expression.**FitToPage**

expression Required. An expression that returns a **DocumentWindow** object.

Example

This example exits the current slide show, sets the view in the active window to slide view, sets the zoom to 25 percent, and adjusts the size of the window to fit the slide displayed there.

```
Application.SlideShowWindows(1).View.Exit  
With Application.ActiveWindow  
    .ViewType = ppViewSlide  
    .View.Zoom = 25  
    .FitToPage  
End With
```



[Show All](#)

Flip Method

Flips the specified shape around it's horizontal or vertical axis.

expression.**Flip**(*FlipCmd*)

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

FlipCmd Required [MsoFlipCmd](#). Specifies whether the shape is to be flipped horizontally or vertically.

MsoFlipCmd can be one of these MsoFlipCmd constants.

msoFlipHorizontal

msoFlipVertical

Example

This example adds a triangle to myDocument, duplicates the triangle, and then flips the duplicate triangle vertically and makes it red.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRightTriangle, 10, 10, 50, 50).Duplicate
    .Fill.ForeColor.RGB = RGB(255, 0, 0)
    .Flip msoFlipVertical
End With
```



Follow Method

Displays the HTML document associated with the specified hyperlink in a new Web browser window.

expression.**Follow**

expression Required. An expression that returns a **Hyperlink** object.

Example

This example loads the document associated with the first hyperlink on slide one in a new instance of the Web browser.

```
ActivePresentation.Slides(1).Hyperlinks(1).Follow
```



[Show All](#)

FollowHyperlink Method

Displays a cached document, if it has already been downloaded. Otherwise, this method resolves the hyperlink, downloads the target document and displays it in the appropriate application.

expression.**FollowHyperlink**(*Address*, *SubAddress*, *NewWindow*, *AddHistory*, *ExtraInfo*, *Method*, *HeaderInfo*)

expression Required. An expression that returns a **Presentation** object.

Address Required **String**. The address of the target document.

SubAddress Optional **String**. The location in the target document. By default, this argument is an empty string.

NewWindow Optional **Boolean**. **True** to have the target application opened in a new window. The default value is **False**.

AddHistory Optional **Boolean**. **True** to add the link to the current day's history folder.

ExtraInfo Optional **String**. String or byte array that specifies information for HTTP. This argument can be used, for example, to specify the coordinates of an image map or the contents of a form. It can also indicate a FAT file name. The **Method** argument determines how this extra information is handled.

Method Optional [MsoExtraInfoMethod](#). Specifies how **ExtraInfo** is posted or appended.

MsoExtraInfoMethod can be one of these MsoExtraInfoMethod constants.

msoMethodGet Default. **ExtraInfo** is a **String** that is appended to the address.

msoMethodPost **ExtraInfo** is posted as a **String** or byte array.

HeaderInfo Optional **String**. A string that specifies header information for the HTTP request. The default value is an empty string. You can combine several header lines into a single string by using the following syntax: "string1" & vbCr

& "string2". The specified string is automatically converted into ANSI characters. Note that the HeaderInfo argument may overwrite default HTTP header fields.

Example

This example loads the document at example.microsoft.com in a new window and adds it to the history folder.

```
ActivePresentation.FollowHyperlink _  
    Address:="http://example.microsoft.com", _  
    NewWindow:=True, AddHistory:=True
```



GotoNamedShow Method

Switches to the specified custom, or named, slide show during another slide show. When the slide show advances from the current slide, the next slide displayed will be the next one in the specified custom slide show, not the next one in current slide show.

expression.**GotoNamedShow**(*SlideShowName*)

expression Required. An expression that returns a **SlideShowView** object.

SlideShowName Required **String**. The name of the custom slide show to be switched to.

Example

This example redefines the slide show running in slide show window one to include only the slides in the custom slide show named "Quick Show."

```
SlideShowWindows(1).View.GotoNamedShow "Quick Show"
```



[Show All](#)

GotoSlide Method

 [GotoSlide method as it applies to the **View** object.](#)

Switches to the specified slide.

expression.**GotoSlide**(*Index*)

expression Required. An expression that returns a **View** object.

Index Required **Integer**. The number of the slide to switch to.

 [GotoSlide method as it applies to the **SlideShowView** object.](#)

Switches to the specified slide during a slide show. You can specify whether you want the animation effects to be rerun.

expression.**GotoSlide**(*Index*, *ResetSlide*)

expression Required. An expression that returns a SlideShowView object.

Index Required **Integer**. The number of the slide to switch to.

ResetSlide Optional **MsoTriState**. If you switch from one slide to another during a slide show with **ResetSlide** set to **msoFalse**, when you return to the first slide, its animation picks up where it left off. If you switch from one slide to another with **ResetSlide** set to **msoTrue**, when you return to the first slide, its entire animation starts over. The default value is **msoTrue**.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Default.

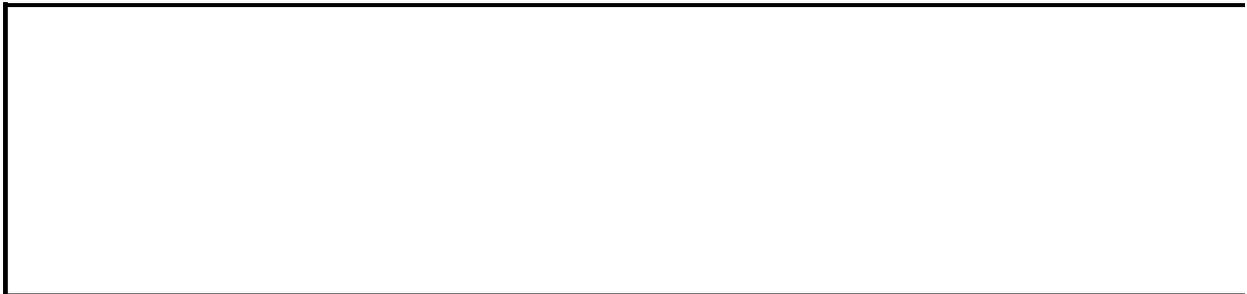
Example

This example switches from the current slide to the slide three in slide show window one. If you switch back to the current slide during the slide show, its entire animation will start over.

```
With SlideShowWindows(1).View  
    .GotoSlide 3  
End With
```

This example switches from the current slide to the slide three in slide show window one. If you switch back to the current slide during the slide show, its animation will pick up where it left off.

```
With SlideShowWindows(1).View  
    .GotoSlide 3, msoFalse  
End With
```



Group Method

Groups the shapes in the specified range. Returns the grouped shapes as a single **Shape** object.

expression.**Group**

expression Required. An expression that returns a **ShapeRange** object.

Remarks

Because a group of shapes is treated as a single shape, grouping and ungrouping shapes changes the number of items in the **Shapes** collection and changes the index numbers of items that come after the affected items in the collection.

Example

This example adds two shapes to myDocument, groups the two new shapes, sets the fill for the group, rotates the group, and sends the group to the back of the drawing layer.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeCan, 50, 10, 100, 200).Name = "shpOne"
    .AddShape(msoShapeCube, 150, 250, 100, 200).Name = "shpTwo"
    With .Range(Array("shpOne", "shpTwo")).Group
        .Fill.PresetTextured msoTextureBlueTissuePaper
        .Rotation = 45
        .ZOrder msoSendToBack
    End With
End With
```



Help Method

Displays a Help topic.

expression.**Help**(*HelpFile*, *ContextID*)

expression Required. An expression that returns an **Application** object.

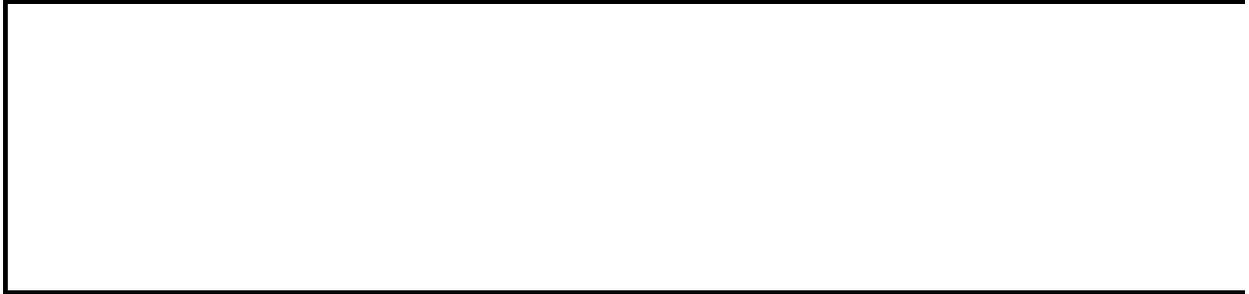
HelpFile Optional **String**. The name of the Help file you want to display. Can be either a .chm or an .hlp file. If this argument isn't specified, Microsoft PowerPoint Help is used.

ContextID Optional **Long**. The context ID number for the Help topic. If this argument isn't specified or if it specifies a context ID number that is not associated with a Help topic, the **Help Topics** dialog box is displayed.

Example

This example displays topic number 65527 in the Help file MyHelpFile.chm.

```
Application.Help "MyHelpFile.chm", 65527
```



ImportFromFile Method

Specifies the sound that will be played whenever the specified shape is clicked or animated or whenever the specified slide transition occurs.

expression.**ImportFromFile**(*FullName*)

expression Required. An expression that returns a **SoundEffect** object.

FullName Required **String**. The name of the specified sound file.

Example

This example specifies that the file `Dudududu.wav` will start to play at the transition to slide two in the active presentation and will continue to play until the next sound starts.

```
With ActivePresentation.Slides(2).SlideShowTransition  
    .SoundEffect.ImportFromFile "c:\sndsys\dudududu.wav"  
    .LoopSoundUntilNext = True  
End With
```



IncrementBrightness Method

Changes the brightness of the picture by the specified amount. Use the [Brightness](#) property to set the absolute brightness of the picture.

expression.IncrementBrightness(*Increment*)

expression Required. An expression that returns a **PictureFormat** object.

Increment Required **Single**. Specifies how much to change the value of the **Brightness** property for the picture. A positive value makes the picture brighter; a negative value makes the picture darker.

Remarks

You cannot adjust the brightness of a picture past the upper or lower limit for the **Brightness** property. For example, if the **Brightness** property is initially set to 0.9 and you specify 0.3 for the *Increment* argument, the resulting brightness level will be 1.0, which is the upper limit for the **Brightness** property, instead of 1.2.

Example

This example creates a duplicate of shape one on myDocument and then moves and darkens the duplicate. For the example to work, shape one must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Duplicate
    .PictureFormat.IncrementBrightness -0.2
    .IncrementLeft 50
    .IncrementTop 50
End With
```



IncrementContrast Method

Changes the contrast of the picture by the specified amount. Use the [Contrast](#) property to set the absolute contrast for the picture.

expression.**IncrementContrast**(*Increment*)

expression Required. An expression that returns a **PictureFormat** object.

Increment Required **Single**. Specifies how much to change the value of the **Contrast** property for the picture. A positive value increases the contrast; a negative value decreases the contrast.

Remarks

You cannot adjust the contrast of a picture past the upper or lower limit for the **Contrast** property. For example, if the **Contrast** property is initially set to 0.9 and you specify 0.3 for the **Increment** argument, the resulting contrast level will be 1.0, which is the upper limit for the **Contrast** property, instead of 1.2.

Example

This example increases the contrast for all pictures on myDocument that aren't already set to maximum contrast.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.Type = msoPicture Then
        s.PictureFormat.IncrementContrast 0.1
    End If
Next
```



IncrementLeft Method

Moves the specified shape horizontally by the specified number of points.

expression.**IncrementLeft**(*Increment*)

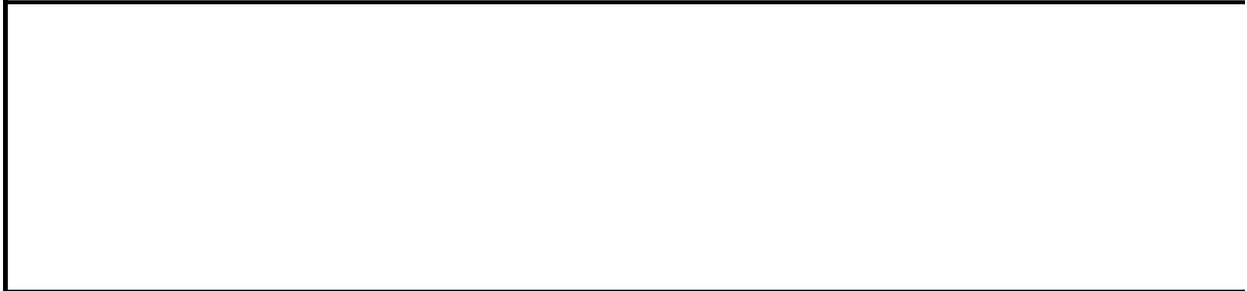
expression Required. An expression that returns a **Shape** object.

Increment Required **Single**. Specifies how far the shape is to be moved horizontally, in points. A positive value moves the shape to the right; a negative value moves it to the left.

Example

This example duplicates shape one on myDocument, sets the fill for the duplicate, moves it 70 points to the right and 50 points up, and rotates it 30 degrees clockwise.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```



IncrementOffsetX Method

Changes the horizontal offset of the shadow by the specified number of points. Use the [OffsetX](#) property to set the absolute horizontal shadow offset.

expression.**IncrementOffsetX**(*Increment*)

expression Required. An expression that returns a **ShadowFormat** object.

Increment Required **Single**. Specifies how far the shadow offset is to be moved horizontally, in points. A positive value moves the shadow to the right; a negative value moves it to the left.

Example

This example moves the shadow for shape three on myDocument to the left by 3 points.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(3).Shadow.IncrementOffsetX -3
```



IncrementOffsetY Method

Changes the vertical offset of the shadow by the specified number of points. Use the [OffsetY](#) property to set the absolute vertical shadow offset.

expression.**IncrementOffsetY**(*Increment*)

expression Required. An expression that returns a **ShadowFormat** object.

Increment Required **Single**. Specifies how far the shadow offset is to be moved vertically, in points. A positive value moves the shadow down; a negative value moves it up.

Example

This example moves the shadow for shape three on myDocument up by 3 points.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(3).Shadow.IncrementOffsetY -3
```



[Show All](#)

IncrementRotation Method

Changes the rotation of the specified shape around the [z-axis](#), by the specified number of degrees. Use the [Rotation](#) property to set the absolute rotation of the shape.

expression.IncrementRotation(*Increment*)

expression Required. An expression that returns a **Shape** object.

Increment Required **Single**. Specifies how far the shape is to be rotated horizontally, in degrees. A positive value rotates the shape clockwise; a negative value rotates it counterclockwise.

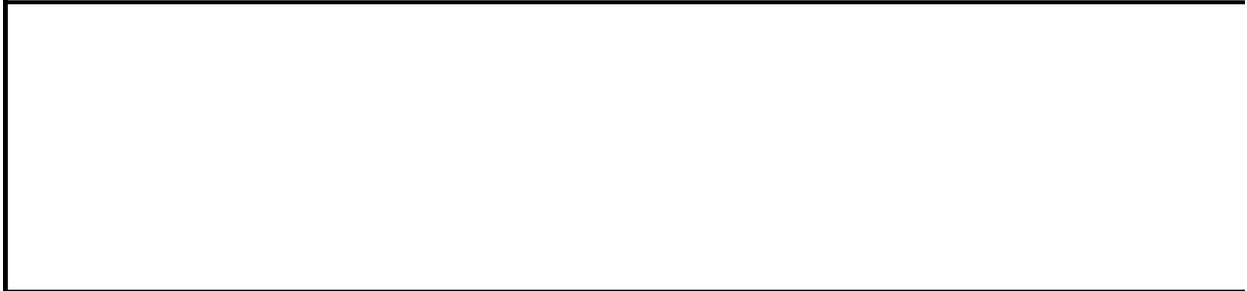
Remarks

To rotate a three-dimensional shape around the x-axis or the y-axis, use the [IncrementRotationX](#) method or the [IncrementRotationY](#) method.

Example

This example duplicates shape one on myDocument, sets the fill for the duplicate, moves it 70 points to the right and 50 points up, and rotates it 30 degrees clockwise.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```



[Show All](#)

IncrementRotationX Method

Changes the rotation of the specified shape around the x-axis by the specified number of degrees. Use the [RotationX](#) property to set the absolute rotation of the shape around the [x-axis](#).

expression.IncrementRotationX(*Increment*)

expression Required. An expression that returns a **ThreeDFormat** object.

Increment Required **Single**. Specifies how much (in degrees) the rotation of the shape around the x-axis is to be changed. Can be a value from – 90 through 90. A positive value tilts the shape up; a negative value tilts it down.

Remarks

You cannot adjust the rotation around the x-axis of the specified shape past the upper or lower limit for the **RotationX** property (90 degrees to – 90 degrees). For example, if the **RotationX** property is initially set to 80 and you specify 40 for the *Increment* argument, the resulting rotation will be 90 (the upper limit for the **RotationX** property) instead of 120.

To change the rotation of a shape around the y-axis, use the [IncrementRotationY](#) method. To change the rotation around the z-axis, use the [IncrementRotation](#) method.

Example

This example tilts shape one on myDocument up 10 degrees. Shape one must be an extruded shape for you to see the effect of this code.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).ThreeD.IncrementRotationX 10
```



[Show All](#)

IncrementRotationY Method

Changes the rotation of the specified shape around the y-axis by the specified number of degrees. Use the [RotationY](#) property to set the absolute rotation of the shape around the [y-axis](#).

expression.IncrementRotationY(*Increment*)

expression Required. An expression that returns a **ThreeDFormat** object.

Increment Required **Single**. Specifies how much (in degrees) the rotation of the shape around the y-axis is to be changed. Can be a value from – 90 through 90. A positive value tilts the shape to the left; a negative value tilts it to the right.

Remarks

To change the rotation of a shape around the x-axis, use the [IncrementRotationX](#) method. To change the rotation around the z-axis, use the [IncrementRotation](#) method.

You cannot adjust the rotation around the y-axis of the specified shape past the upper or lower limit for the **RotationY** property (90 degrees to – 90 degrees). For example, if the **RotationY** property is initially set to 80 and you specify 40 for the *Increment* argument, the resulting rotation will be 90 (the upper limit for the **RotationY** property) instead of 120.

Example

This example tilts shape one on myDocument 10 degrees to the right. Shape one must be an extruded shape for you to see the effect of this code.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).ThreeD.IncrementRotationY -10
```



IncrementTop Method

Moves the specified shape vertically by the specified number of points.

expression.**IncrementTop**(*Increment*)

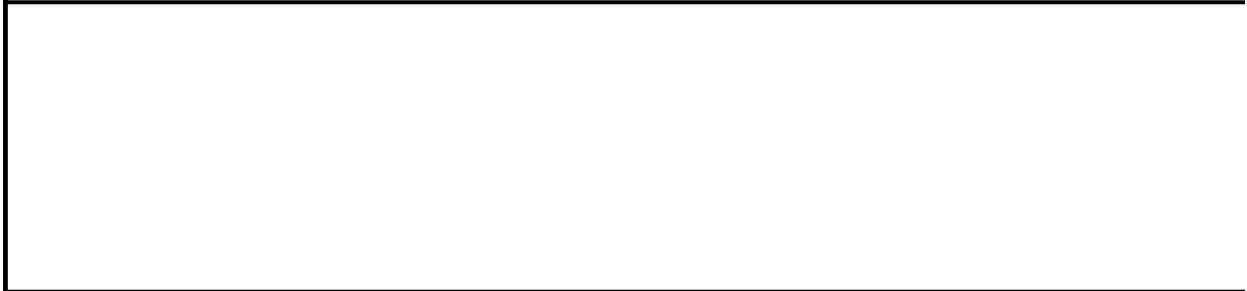
expression Required. An expression that returns a **Shape** object.

Increment Required **Single**. Specifies how far the shape object is to be moved vertically, in points. A positive value moves the shape down; a negative value moves it up.

Example

This example duplicates shape one on myDocument, sets the fill for the duplicate, moves it 70 points to the right and 50 points up, and rotates it 30 degrees clockwise.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```



[Show All](#)

Insert Method

Inserts a new segment after the specified node of the freeform.

expression.**Insert**(*Index*, *SegmentType*, *EditingType*, *X1*, *Y1*, *X2*, *Y2*, *X3*, *Y3*)

expression Required. An expression that returns a **ShapeNodes** object.

Index Required **Long**. The node that the new node is to be inserted after.

SegmentType Required [MsoSegmentType](#). The type of segment to be added.

MsoSegmentType can be one of these MsoSegmentType constants.

msoSegmentCurve

msoSegmentLine

EditingType Required [MsoEditingType](#). The editing property of the vertex.

MsoEditingType can be one of these MsoEditingType constants (cannot be **msoEditingSmooth** or **msoEditingSymmetric**).

msoEditingAuto

msoEditingCorner

X1 Required **Single**. If the **EditingType** of the new segment is **msoEditingAuto**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the end point of the new segment. If the **EditingType** of the new node is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the first control point for the new segment.

Y1 Required **Single**. If the **EditingType** of the new segment is **msoEditingAuto**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the end point of the new segment. If the **EditingType** of the new node is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the first control point for the new segment.

X2 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the second control point for the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Y2 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the second control point for the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

X3 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the horizontal distance (in points) from the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Y3 Optional **Single**. If the *EditingType* of the new segment is **msoEditingCorner**, this argument specifies the vertical distance (in points) from the upper-left corner of the document to the end point of the new segment. If the *EditingType* of the new segment is **msoEditingAuto**, don't specify a value for this argument.

Example

This example adds a smooth node with a curved segment after node four in shape three on myDocument. Shape three must be a freeform drawing with at least four nodes.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    .Insert Index:=4, SegmentType:=msoSegmentCurve, _
        EditingType:=msoEditingSmooth, X1:=210, Y1:=100
End With
```



InsertAfter Method

Appends a string to the end of the specified text range. Returns a **TextRange** object that represents the appended text. When used without an argument, this method returns a zero-length string at the end of the specified range.

expression.**InsertAfter**(*NewText*)

expression Required. An expression that returns a **TextRange** object.

NewText Optional **String**. The text to be inserted. The default value is an empty string.

Example

This example appends the string ": Test version" to the end of the title on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(1)
    .TextFrame.TextRange.InsertAfter ": Test version"
End With
```

This example appends the contents of the Clipboard to the end of the title on slide one.

```
Application.ActivePresentation.Slides(1).Shapes(1).TextFrame _
    .TextRange.InsertAfter.Paste
```



InsertBefore Method

Appends a string to the beginning of the specified text range. Returns a **TextRange** object that represents the appended text. When used without an argument, this method returns a zero-length string at the end of the specified range.

expression.**InsertBefore**(*NewText*)

expression Required. An expression that returns a **TextRange** object.

NewText Optional **String**. The text to be appended. The default value is an empty string.

Example

This example appends the string "Test version: " to the beginning of the title on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(1)  
    .TextFrame.TextRange.InsertBefore "Test version: "  
End With
```

This example appends the contents of the Clipboard to the beginning of the title on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(1).TextFrame _  
    .TextRange.InsertBefore.Paste
```



[Show All](#)

InsertDateTime Method

Inserts the date and time in the specified text range. Returns a **TextRange** object that represents the inserted text.

expression.**InsertDateTime**(*DateTimeFormat*, *InsertAsField*)

expression Required. An expression that returns a **TextRange** object.

DateTimeFormat Required [PpDateTimeFormat](#). A format for the date and time.

PpDateTimeFormat can be one of these PpDateTimeFormat constants.

ppDateTimeddddMMMMddyyyy

ppDateTimedMMMMyyyy

ppDateTimedMMMyy

ppDateTimeFormatMixed

ppDateTimeHmm

ppDateTimehmAMPM

ppDateTimeHmss

ppDateTimehmssAMPM

ppDateTimeMdyy

ppDateTimeMMddyHmm

ppDateTimeMMddyhmmAMPM

ppDateTimeMMMMyyyy

ppDateTimeMMMMyy

ppDateTimeMMyy

InsertAsField Optional [MsoTriState](#). Determines whether the inserted date and time will be updated each time the presentation is opened.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Updates the inserted date and time each time the presentation is opened.

Example

This example inserts the date and time after the first sentence of the first paragraph in shape two on slide one in the active presentation.

```
Set sh = Application.ActivePresentation.Slides(1).Shapes(2)
Set sentOne = sh.TextFrame.TextRange.Paragraphs(1).Sentences(1)
sentOne.InsertAfter.InsertDateTime ppDateTimeMdyy
```



InsertFromFile Method

Inserts slides from a file into a presentation, at the specified location. Returns an integer that represents the number of slides inserted.

expression.**InsertFromFile**(*FileName*, *Index*, *SlideStart*, *SlideEnd*)

expression Required. An expression that returns a **Slides** collection.

FileName Required **String**. The name of the file that contains the slides you want to insert.

Index Required **Long**. The index number of the **Slide** object in the specified **Slides** collection you want to insert the new slides after.

SlideStart Optional **Long**. The index number of the first **Slide** object in the **Slides** collection in the file denoted by **FileName**.

SlideEnd Optional **Long**. The index number of the last **Slide** object in the **Slides** collection in the file denoted by **FileName**.

Example

This example inserts slides three through six from C:\Ppt\Sales.ppt after slide two in the active presentation.

```
ActivePresentation.Slides.InsertFromFile _  
    "c:\ppt\sales.ppt", 2, 3, 6
```



InsertSlideNumber Method

Inserts the slide number of the current slide into the specified text range. Returns a **TextRange** object that represents the slide number.

expression.**InsertSlideNumber**

expression Required. An expression that returns a **TextRange** object.

Remarks

The inserted slide number is automatically updated when the slide number of the current slide changes.

Example

This example inserts the slide number of the current slide after the first sentence of the first paragraph in shape two on slide one in the active presentation.

```
Set sh = Application.ActivePresentation.Slides(1).Shapes(2)
Set sentOne = sh.TextFrame.TextRange.Paragraphs(1).Sentences(1)
sentOne.InsertAfter InsertSlideNumber
```



[Show All](#)

InsertSymbol Method

Returns a [TextRange](#) object that represents a symbol inserted into the specified text range.

expression.**InsertSymbol**(*FontName*, *CharNumber*, *Unicode*)

expression Required. An expression that returns a **TextRange** object.

FontName Required **String**. The font name.

CharNumber Required **Long**. The Unicode or ASCII character number.

Unicode Optional [MsoTriState](#). Specifies whether the *CharNumber* argument represents an ASCII or Unicode character.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this method.

msoFalse Default. The *CharNumber* argument represents an ASCII character number.

msoTriStateMixed Doesn't apply to this method.

msoTriStateToggle Doesn't apply to this method.

msoTrue The *CharNumber* argument represents a Unicode character.

Example

This example inserts the Registered Trademark symbol after the first sentence of the first paragraph in a new text box on the first slide in the active presentation.

```
Sub Symbol()  
    Dim txtBox As Shape  
  
    'Add text box  
    Set txtBox = Application.ActivePresentation.Slides(1) _  
        .Shapes.AddTextbox(Orientation:=msoTextOrientationHorizontal  
            Left:=100, Top:=100, Width:=100, Height:=100)  
  
    'Add symbol to text box  
    txtBox.TextFrame.TextRange.InsertSymbol _  
        FontName:="Symbol", CharNumber:=226  
  
End Sub
```



[Show All](#)

Item Method

 [Item method as it applies to the **ActionSettings** object.](#)

Returns a single action setting from the specified [ActionSettings](#) collection.

expression.**Item**(*Index*)

expression Required. An expression that returns an **ActionSettings** collection.

Index Required [PpMouseActivation](#). The action setting for a MouseClick or MouseOver event.

PpMouseActivation can be one of these PpMouseActivation constants.

ppMouseClick The action setting for when the user clicks the shape.

ppMouseOver The action setting for when the mouse pointer is positioned over the specified shape.

 [Item method as it applies to the **AddIns**, **CanvasShapes**, **Designs**, **DiagramNodeChildren**, **DiagramNodes**, **Fonts**, **GroupShapes**, **NamedSlideShows**, **Presentations**, **ShapeNodes**, **ShapeRange**, **Shapes**, **SlideRange**, and **Slides** objects.](#)

Returns a single object from the specified collection.

expression.**Item**(*Index*)

expression Required. An expression that returns one of the above objects.

Index Required **Variant**. The name or index number of the single object in the collection to be returned.

 [Item method as it applies to the **AnimationBehaviors**, **AnimationPoints**, **CellRange**, **ColorSchemes**, **Columns**, **Comments**, **DocumentWindows**, **ExtraColors**, **Hyperlinks**, **ObjectVerbs**, **Panes**, **Placeholders**, **PrintRanges**, **PublishObjects**, **Rows**, **RulerLevels**, **Sequence**, **Sequences**,](#)

[SlideShowWindows](#), [TabStops](#), and [TextStyleLevels](#) objects.

Returns a single object from the specified collection.

expression.**Item**(*Index*)

expression Required. An expression that returns an AnimationBehaviors collection.

Index Required **Long**. The index number of the single object in the collection to be returned.



[Item](#) method as it applies to the [Borders](#) object.

Returns a [LineFormat](#) object for the specified border.

expression.**Item**(*BorderType*)

expression Required. An expression that returns a [Borders](#) collection.

BorderType Required [PpBorderType](#). Specifies which border of a cell or cell range is to be returned.

PpBorderType can be one of these PpBorderType constants.

ppBorderBottom

ppBorderDiagonalDown

ppBorderDiagonalUp

ppBorderLeft

ppBorderRight

ppBorderTop



[Item](#) method as it applies to the [Tags](#) object.

Returns a single tag from the specified [Tags](#) collection.

expression.**Item**(*Name*)

expression Required. An expression that returns a **Tags** object.

Name Required **String**. The name of the single tag in the collection to be returned.

 [Item method as it applies to the **TextStyles** object.](#)

Returns a single text style from the specified **TextStyles** collection.

expression.**Item**(*Type*)

expression Required. An expression that returns a **TextStyles** collection.

Type Required **PpTextStyleType**. The text style type.

PpTextStyleType can be one of these PpTextStyleType constants.

ppBodyStyle

ppDefaultStyle

ppTitleStyle

Remarks

The **Item** method is the [default member](#) for a collection. For example, the following two lines of code are equivalent:

```
ActivePresentation.Slides.Item(1)  
ActivePresentation.Slides(1)
```

For more information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

 [As it applies to the **ActionSettings** object.](#)

This example sets shape three on slide one to play the sound of applause and uses the [AnimateAction](#) property to specify that the shape's color is to be momentarily inverted when the shape is clicked during a slide show.

```
With ActivePresentation.Slides.Item(1).Shapes _  
    .Item(3).ActionSettings.Item(ppMouseClick)  
    .SoundEffect.Name = "applause"  
    .AnimateAction = True  
End With
```

 [As it applies to the **RulerLevels** object.](#)

This example sets the first-line indent and the hanging indent for outline level one in body text on the slide master for the active presentation.

```
With ActivePresentation.SlideMaster.TextStyles.Item(ppBodyStyle)  
    With .Ruler.Levels.Item(1) ' sets indents for level 1  
        .FirstMargin = 9  
        .LeftMargin = 54  
    End With  
End With
```

 [As it applies to the **Shapes** object.](#)

This example sets the foreground color to red for the shape named "Rectangle 1" on slide one in the active presentation.

```
ActivePresentation.Slides.Item(1).Shapes.Item("rectangle 1").Fill _  
    .ForeColor.RGB = RGB(128, 0, 0)
```

 [As it applies to the **Tags** object.](#)

This example hides all slides in the active presentation that don't have the value "east" for the "region" tag.

```
For Each s In ActivePresentation.Slides
    If s.Tags.Item("region") <> "east" Then
        s.SlideShowTransition.Hidden = True
    End If
Next
```



LargeScroll Method

Scrolls through the specified document window by pages.

expression.LargeScroll(Down, Up, ToRight,ToLeft)

expression Required. An expression that returns a **DocumentWindow** object.

Down Optional **Long**. Specifies the number of pages to scroll down.

Up Optional **Long**. Specifies the number of pages to scroll up.

ToRight Optional **Long**. Specifies the number of pages to scroll right.

ToLeft Optional **Long**. Specifies the number of pages to scroll left.

Remarks

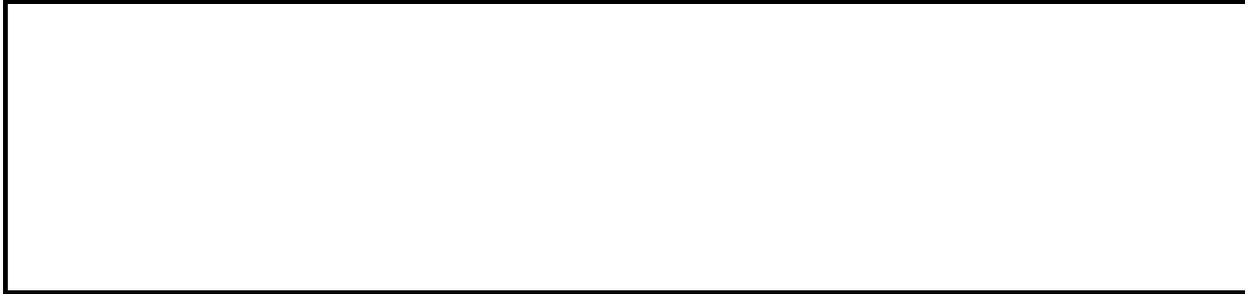
If no arguments are specified, this method scrolls down one page. If ***Down*** and ***Up*** are both specified, their effects are combined. For example, if ***Down*** is 2 and ***Up*** is 4, this method scrolls up two pages. Similarly, if ***Right*** and ***Left*** are both specified, their effects are combined.

Any of the arguments can be a negative number.

Example

This example scrolls the active window down three pages.

```
Application.ActiveWindow.LargeScroll Down:=3
```



Last Method

Sets the specified slide show view to display the last slide in the presentation.

expression.**Last**

expression Required. An expression that returns a **SlideShowView** object.

Remarks

If you use the **Last** method to switch from one slide to another during a slide show, when you return to the original slide, its animation picks up where it left off.

Example

This example sets slide show window one to display the last slide in the presentation.

```
SlideShowWindows(1).View.Last
```



Lines Method

Returns a **TextRange** object that represents the specified subset of text lines. For information about counting or looping through the lines in a text range, see the [TextRange](#) object.

expression.**Lines**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first line in the returned range.

Length Optional **Long**. The number of lines to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first line and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one line.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first line in the specified range.

If ***Start*** is greater than the number of lines in the specified text, the returned range starts with the last line in the specified range.

If ***Length*** is greater than the number of lines from the specified starting line to the end of the text, the returned range contains all those lines.

Example

This example formats as italic the first two lines of the second paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange.Paragraphs(2) _  
    .Lines(1, 2).Font.Italic = True
```



Load Method

Returns a **Design** object that represents a design loaded into the master list of the specified presentation.

expression.**Load**(*TemplateName*, *Index*)

expression Required. An expression that returns one of the objects in the Applies To list.

TemplateName Required **String**. The path to the design template.

Index Optional **Long**. The index number of the design template in the collection of design templates. The default is -1, which means the design template is added to the end of the list of designs in the presentation.

Example

This example add a design template to the beginning of the collection of design templates in the active presentation. This example assumes the "artsy.pot" template is located at the specified path.

```
Sub LoadDesign()  
    ActivePresentation.Designs.Load TemplateName:="C:\Program Files\  
        "Microsoft Office\Templates\Presentation Designs\Balance.pot  
End Sub
```



LtrRun Method

Sets the direction of text in a text range to read from left to right.

expression.**LtrRun**

expression Required. An expression that returns a **TextRange** object.

Remarks

This method makes it possible to use text from both left-to-right and right-to-left languages in the same presentation.

Example

The following example finds all of the shapes on slide one that contain text and changes the text to read from left to right.

```
ActiveWindow.ViewType = ppViewSlide
For Each sh In ActivePresentation.Slides(1).Shapes
    If sh.HasTextFrame Then
        sh.TextFrame.TextRange.LtrRun
    End If
Next
```



[Show All](#)

Merge Method

 [Merge method as it applies to the **Cell** object.](#)

Merges one table cell with another. The result is a single table cell.

expression.**Merge**(*MergeTo*)

expression Required. An expression that returns a **Cell** object.

MergeTo Required **Cell** object. **Cell** object to be merged with. Use the syntax **.Cell**(*row* , *column*).

 [Merge method as it applies to the **Presentation** object.](#)

Merges one presentation into another presentation.

expression.**Merge**(*Path*)

expression Required. An expression that returns one of the above objects.

Path Required **String**. The full path of a file to merge with this presentation.

Remarks

This method returns an error if the filename cannot be opened, or the presentation has a baseline.

Example

This example merges the first two cells of row one in the specified table.

```
With ActivePresentation.Slides(2).Shapes(5).Table
    .Cell(1, 1).Merge MergeTo:=.Cell(1, 2)
End With
```



MoveAfter Method

Moves one animation effect to after another animation effect.

expression.**MoveAfter**(*Effect*)

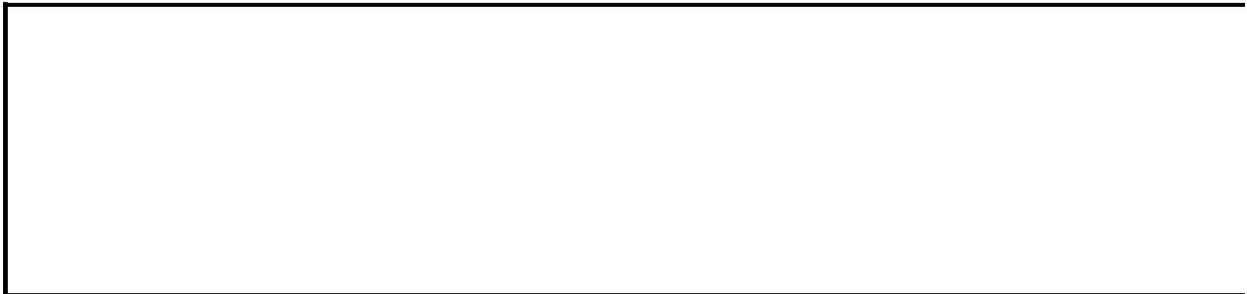
expression Required. An expression that returns one of the objects in the Applies To list.

EffectRequired **Effect** object. The effect after which the effect in *expression* will be moved.

Example

The following example moves one effect to after another.

```
Sub MoveEffect()  
    Dim effOne As Effect  
    Dim effTwo As Effect  
    Dim shpFirst As Shape  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set effOne = ActivePresentation.Slides(1).TimeLine.MainSequence.  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlinds)  
    Set effTwo = ActivePresentation.Slides(1).TimeLine.MainSequence.  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlast)  
    effOne.MoveAfter Effect:=effTwo  
End Sub
```



MoveBefore Method

Moves one animation effect to before another animation effect.

expression.**MoveBefore**(*Effect*)

expression Required. An expression that returns one of the objects in the Applies To list.

EffectRequired **Effect** object. The effect before which the effect in *expression* will be moved.

Example

The following example moves one effect in front of another one.

```
Sub MoveEffect()  
    Dim effOne As Effect  
    Dim effTwo As Effect  
    Dim shpFirst As Shape  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set effOne = ActivePresentation.Slides(1).TimeLine.MainSequence.  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlinds)  
    Set effTwo = ActivePresentation.Slides(1).TimeLine.MainSequence.  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlast)  
  
    effTwo.MoveBefore Effect:=effOne  
End Sub
```



[Show All](#)

MoveNode Method

Moves a diagram node, and any of its child nodes, within a diagram.

expression.**MoveNode**(*TargetNode*, *Pos*)

expression Required. An expression that returns one of the objects in the Applies To list.

TargetNode Required **DiagramNode** object. The source diagram node for the move.

Pos Required [MsoRelativeNodePosition](#). Specifies where the node will be added, relative to **TargetNode**.

MsoRelativeNodePosition can be one of these MsoRelativeNodePosition constants.

msoAfterLastSibling

msoAfterNode

msoBeforeFirstSibling

msoBeforeNode

Example

The following example moves the second diagram node of a newly-created diagram to after the fourth node.

```
Sub MoveDiagramNode()  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intCount As Integer  
  
    'Add pyramid diagram to the current document  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
  
    'Add four child nodes to the pyramid diagram  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    For intCount = 1 To 3  
        dgnNode.AddNode  
    Next intCount  
  
    'Move the second node to after where the  
    'fourth node is currently located.  
    dgnNode.Diagram.Nodes(2).MoveNode _  
        TargetNode:=dgnNode.Diagram.Nodes(4), _  
        Pos:=msoAfterLastSibling  
End Sub
```



MoveTo Method

Moves the specified object to a specific location within the same collection, renumbering all other items in the collection appropriately.

expression.**MoveTo**(*toPos*)

expression Required. An expression that returns one of the objects in the Applies To list.

toPos Required **Long**. The index to which to move the animation effect.

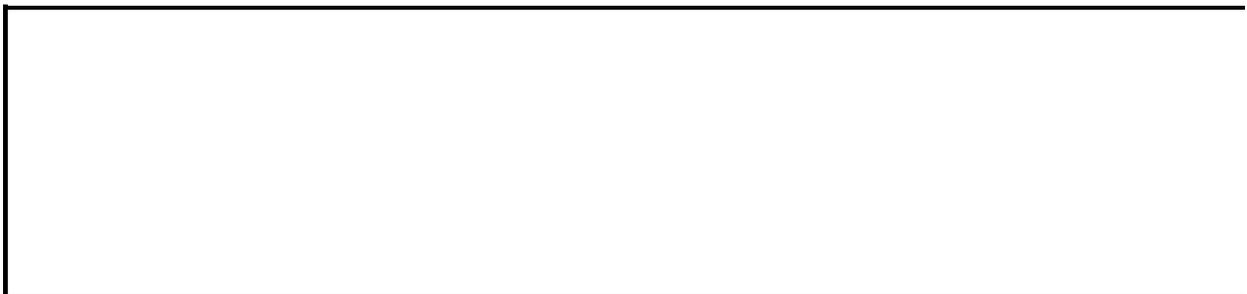
Example

This example moves an animation effect to the second in the animation effects collection for the specified shape.

```
Sub MoveEffect()  
    Dim sldFirst as Slide  
    Dim shpFirst As Shape  
    Dim effAdd As Effect  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpFirst = sldFirst.Shapes(1)  
  
    Set effAdd = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpFirst, effectId:=msoAnimEffectBlinds)  
  
    effAdd.MoveTo toPos:=2  
  
End Sub
```

This example moves the second slide in the active presentation to the first slide.

```
Sub MoveSlideToNewLocation()  
    ActivePresentation.Slides(2).MoveTo toPos:=1  
End Sub
```



Name Method

Returns the name of the specified tag as a **String**.

expression.**Name**(*Index*)

expression Required. An expression that returns a **Tags** collection.

Index Required **Long**. The tag number.

Example

This example displays the name and value for each tag associated with slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Tags
  For i = 1 To .Count
    MsgBox "Tag #" & i & ": Name = " & .Name(i)
    MsgBox "Tag #" & i & ": Value = " & .Value(i)
  Next
End With
```

This example searches through the tags for each slide in the active presentation. If there's a tag named "PRIORITY," a message box displays the tag value. If there isn't a tag named "PRIORITY," the example adds this tag with the value "Unknown."

```
For Each s In Application.ActivePresentation.Slides
  With s.Tags
    found = False
    For i = 1 To .Count
      If .Name(i) = "PRIORITY" Then
        found = True
        slNum = .Parent.SlideIndex
        MsgBox "Slide " & slNum & _
              " priority: " & .Value(i)
      End If
    Next
    If Not found Then
      slNum = .Parent.SlideIndex
      .Add "Name", "New Figures"
      .Add "Priority", "Unknown"
      MsgBox "Slide " & slNum & _
            " priority tag added: Unknown"
    End If
  End With
Next
```



NewWindow Method

Presentation object: Opens a new window that contains the specified presentation. Returns a [DocumentWindow](#) object that represents the new window.

DocumentWindow object: Opens a new window that contains the same document that's displayed in the specified window. Returns a [DocumentWindow](#) object that represents the new window.

expression.**NewWindow**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example creates a new window with the contents of the active window (this activates the new window) and then switches back to the first window.

```
Set oldW = Application.ActiveWindow  
Set newW = oldW.NewWindow  
oldW.Activate
```



Next Method

Displays the slide immediately following the slide that's currently displayed. If the last slide is displayed, the **Next** method closes the slide show in speaker mode and returns to the first slide in kiosk mode. Use the [View](#) property of the **SlideShowWindow** object to return the **SlideShowView** object.

expression.**Next**

expression Required. An expression that returns one of the items in the Applies To list.

Example

This example shows the slide immediately following the currently displayed slide on slide show window one.

```
SlideShowWindows(1).View.Next
```



NextNode Method

Returns a [DiagramNode](#) object that represents the next diagram node in a collection of diagram nodes.

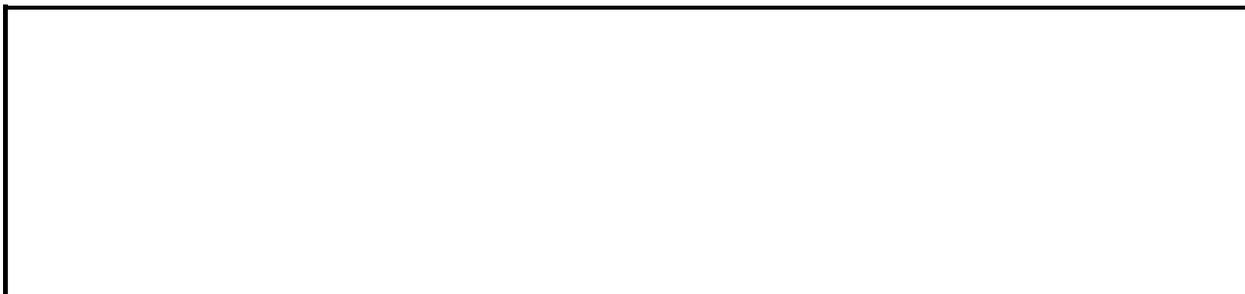
expression.**NextNode**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates an organization chart, and adds child nodes to the middle diagram node.

```
Sub AddChildrenToMiddle()  
  
    Dim dgnNode As DiagramNode  
    Dim dgnNext As DiagramNode  
    Dim shpOrgChart As Shape  
    Dim intNodes As Integer  
  
    'Add organization chart and first child node  
    Set shpOrgChart = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramOrgChart, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpOrgChart.DiagramNode.Children.AddNode  
  
    'Add three additional nodes to root node  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Set dgnNode variable to the middle node  
    Set dgnNext = dgnNode.Children.Item(1).NextNode  
  
    'Add three child nodes to middle node  
    For intNodes = 1 To 3  
        dgnNext.Children.AddNode  
    Next intNodes  
  
End Sub
```



[Show All](#)

OneColorGradient Method

Sets the specified fill to a one-color gradient.

expression.**OneColorGradient**(*Style*, *Variant*, *Degree*)

expression Required. An expression that returns a **FillFormat** object.

Style Required [MsoGradientStyle](#). The gradient style.

MsoGradientStyle can be one of these MsoGradientStyle constants.

msoGradientDiagonalDown

msoGradientDiagonalUp

msoGradientFromCenter

msoGradientFromCorner

msoGradientFromTitle

msoGradientHorizontal

msoGradientMixed

msoGradientVertical

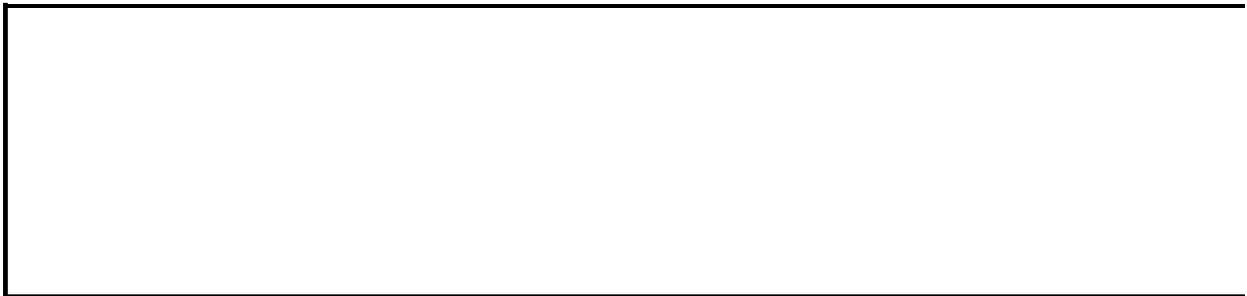
Variant Required **Long**. The gradient variant. Can be a value from 1 to 4, corresponding to the four variants on the **Gradient** tab in the **Fill Effects** dialog box. If *Style* is **msoGradientFromTitle** or **msoGradientFromCenter**, this argument can be either 1 or 2.

Degree Required **Single**. The gradient degree. Can be a value from 0.0 (dark) to 1.0 (light).

Example

This example adds a rectangle with a one-color gradient fill to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 90, 90, 90, 80).Fill
    .ForeColor.RGB = RGB(0, 128, 128)
    .OneColorGradient msoGradientHorizontal, 1, 1
End With
```



[Show All](#)

Open Method

Opens the specified presentation. Returns a [Presentation](#) object that represents the opened presentation.

expression.Open(FileName, ReadOnly, Untitled, WithWindow, OpenConflictDocument)

expression Required. An expression that returns a [Presentations](#) collection.

FileName Required **String**. The name of the file to open.

ReadOnly Optional [MsoTriState](#). Specifies whether the file is opened with read/write or read-only status.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. Opens the file with read/write status.

msoTriStateMixed

msoTriStateToggle

msoTrue Opens the file with read-only status.

Untitled Optional [MsoTriState](#). Specifies whether the file has a title.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. The file name automatically becomes the title of the opened presentation.

msoTriStateMixed

msoTriStateToggle

msoTrue Opens the file without a title. This is equivalent to creating a copy of the file.

WithWindow Optional [MsoTriState](#). Specifies whether the file is visible.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Hides the opened presentation.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Opens the file in a visible window.

OpenConflictDocument Optional [MsoTriState](#). Specifies whether to open the conflict file for a presentation with an offline conflict.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. Opens the server file and ignores the conflict document.

msoTriStateMixed

msoTriStateToggle

msoTrue Opens the conflict file and overwrites the server file.

Remarks

With the proper file converters installed, Microsoft PowerPoint opens files with the following MS-DOS file extensions: .ch3, .cht, .doc, .htm, .html, .mcw, .pot, .ppa, .pps, .ppt, .pre, .rtf, .sh3, .shw, .txt, .wk1, .wk3, .wk4, .wpd, .wpl, .wps, and .xls.

Example

This example opens a presentation with read-only status.

```
Presentations.Open FileName:="c:\My Documents\pres1.ppt", _  
    ReadOnly:=msoTrue
```



Paragraphs Method

Returns a **TextRange** object that represents the specified subset of text paragraphs. For information about counting or looping through the paragraphs in a text range, see the [TextRange](#) object.

expression.**Paragraphs**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first paragraph in the returned range.

Length Optional **Long**. The number of paragraphs to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first paragraph and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one paragraph.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first paragraph in the specified range.

If ***Start*** is greater than the number of paragraphs in the specified text, the returned range starts with the last paragraph in the specified range.

If ***Length*** is greater than the number of paragraphs from the specified starting paragraph to the end of the text, the returned range contains all those paragraphs.

Example

This example formats as italic the first two lines of the second paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange.Paragraphs(2) _  
    .Lines(1, 2).Font.Italic = True
```



[Show All](#)

Paste Method

 [Paste method as it applies to the **Shapes** object.](#)

Pastes the shapes, slides, or text on the Clipboard into the specified **Shapes** collection, at the top of the z-order. Each pasted object becomes a member of the specified **Shapes** collection. If the Clipboard contains entire slides, the slides will be pasted as shapes that contain the images of the slides. If the Clipboard contains a text range, the text will be pasted into a newly created **TextFrame** shape. Returns a [ShapeRange](#) object that represents the pasted objects.

expression.**Paste**

expression Required. An expression that returns one of the above objects.

 [Paste method as it applies to the **Slides** object.](#)

Pastes the slides on the Clipboard into the **Slides** collection for the presentation. Specify where you want to insert the slides with the **Index** argument. Returns a [SlideRange](#) object that represents the pasted objects. Each pasted slide becomes a member of the specified **Slides** collection.

expression.**Paste**(*Index*)

expression Required. An expression that returns one of the above objects.

Index Optional **Integer**. The index number of the slide that the slides on the Clipboard are to be pasted before. If this argument is omitted, the slides on the Clipboard are pasted after the last slide in the presentation.

 [Paste method as it applies to the **TextRange** object.](#)

Pastes the text on the Clipboard into the specified text range, and returns a **TextRange** object that represents the pasted text.

expression.**Paste**

expression Required. An expression that returns one of the above objects.



[Paste method as it applies to the **View** object.](#)

Pastes the contents of the Clipboard into the active view. Attempting to paste an object into a view that won't accept it causes an error. For information about views and the objects you can paste into them, see the "Remarks" section.

expression.**Paste**

expression Required. An expression that returns one of the above objects.

Remarks

Use the [ViewType](#) property to set the view for a window before pasting the Clipboard contents into it. The following table shows what you can paste into each view.

Into this view	You can paste the following from the Clipboard
Slide view or notes page view	Shapes, text, or entire slides. If you paste a slide from the Clipboard, an image of the slide will be inserted onto the slide, master, or notes page as an embedded object. If one shape is selected, the pasted text will be appended to the shape's text; if text is selected, the pasted text will replace the selection; if anything else is selected, the pasted text will be placed in its own text frame. Pasted shapes will be added to the top of the z-order and won't replace selected shapes.
Outline view	Text or entire slides. You cannot paste shapes into outline view. A pasted slide will be inserted before the slide that contains the insertion point.
Slide sorter view	Entire slides. You cannot paste shapes or text into slide sorter view. A pasted slide will be inserted at the insertion point or after the last slide selected in the presentation.

Example

[As it applies to the **Shapes** object.](#)

This example copies shape one on slide one in the active presentation to the Clipboard and then pastes it into slide two.

```
With ActivePresentation
    .Slides(1).Shapes(1).Copy
    .Slides(2).Shapes.Paste
End With
```

This example cuts the text in shape one on slide one in the active presentation, places it on the Clipboard, and then pastes it after the first word in shape two on the same slide.

```
With ActivePresentation.Slides(1)
    .Shapes(1).TextFrame.TextRange.Cut
    .Shapes(2).TextFrame.TextRange.Words(1).InsertAfter.Paste
End With
```

[As it applies to the **Slides** object.](#)

This example cuts slides three and five from the Old Sales presentation and then inserts them before slide four in the active presentation.

```
Presentations("Old Sales").Slides.Range(Array(3, 5)).Cut
ActivePresentation.Slides.Paste 4
```

[As it applies to the **View** object.](#)

This example copies the selection in window one to the Clipboard and copies it into the view in window two. If the Clipboard contents cannot be pasted into the view in window two— for example, if you try to paste a shape into slide sorter view— this example fails.

```
Windows(1).Selection.Copy  
Windows(2).View.Paste
```

This example copies the selection in window one to the Clipboard, makes sure that window one is in slide view, and then copies the Clipboard contents into the view in window two.

```
Windows(1).Selection.Copy  
With Windows(2)  
    .ViewType = ppViewSlide  
    .View.Paste  
End With
```

[Show All](#)

PasteSpecial Method

Pastes the contents of the Clipboard using a special format. Although the syntax for using this method is the same for all objects in the Applies To list, the behavior is slightly different, depending on the object calling the **PasteSpecial** method.

Object	Behavior
Shapes	Adds the shape to the collection of shapes in the specified format. If the specified data type is a text data type, then a new text box is created with the text. If the paste succeeds, the PasteSpecial method returns a ShapeRange object representing the shape range that was pasted.
TextRange	Replaces the text range with the contents of the Clipboard in the format specified. Valid data types for this object are ppPasteText , ppPasteHTML , and ppPasteRTF (any other format generates an error). If the paste succeeds, this method returns a TextRange object representing the text range that was pasted.
View	Pastes the current contents of the Clipboard into the view represented by the View object. Valid views for the PasteSpecial method are the same as those for the Paste method. If the data type can't be pasted into the view (for example, trying to paste a picture into Slide Sorter View), then an error occurs.

expression.**PasteSpecial**(*DataType*, *DisplayAsIcon*, *IconFileName*, *IconIndex*, *IconLabel*, *Link*)

expression Required. An expression that returns one of the above objects.

DataType Optional [PpPasteDataType](#). A format for the Clipboard contents when they're inserted into the document. The default value varies, depending on the contents in the Clipboard. An error occurs if the specified data type in the **DataType** argument is not supported by the clipboard contents.

PpPasteDataType can be one of these **PpPasteDataType** constants.

ppPasteBitmap

ppPasteDefault *default*
ppPasteEnhancedMetafile
ppPasteGIF
ppPasteHTML
ppPasteJPG
ppPasteMetafilePicture
ppPasteOLEObject
ppPastePNG
ppPasteRTF
ppPasteShape
ppPasteText

DisplayAsIcon Optional **MsoTriState**. **MsoTrue** to display the embedded object (or link) as an icon.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Does not apply to this method.

msoFalse *default* Does not display the embedded object (or link) as an icon.

msoTriStateMixed Does not apply to this method.

msoTriStateToggle Does not apply to this method.

msoTrue Displays the embedded object (or link) as an icon.

IconFileName Optional **String**. If **DisplayAsIcon** is set to **msoTrue**, this argument is the path and file name for the file in which the icon to be displayed is stored. If **DisplayAsIcon** is set to **msoFalse**, this argument is ignored.

IconIndex Optional **Long**. If **DisplayAsIcon** is set to **msoTrue**, this argument is a number that corresponds to the icon you want to use in the program file specified by **IconFilename**. Icons appear in the **Change Icon** dialog box, accessed from the Standard toolbar (**Insert** menu, **Object** command, **Create New** option): 0 (zero) corresponds to the first icon, 1 corresponds to the second icon, and so on. If this argument is omitted, the first (default) icon is used. If **DisplayAsIcon** is set to **msoFalse**, then this argument is ignored. If **IconIndex** is outside the valid range, then the default icon (index 0) is used.

IconLabel Optional **String**. If **DisplayAsIcon** is set to **msoTrue**, this

argument is the text that appears below the icon. If this label is missing, Microsoft PowerPoint generates an icon label based on the Clipboard contents. If **DisplayAsIcon** is set to **msoFalse**, then this argument is ignored.

Link Optional [MsoTriState](#). Determines whether to create a link to the source file of the Clipboard contents. An error occurs if the Clipboard contents do not support a link.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Does not apply to this method.

msoFalse *default* Does not create a link to the source file of the Clipboard contents.

msoTriStateMixed Does not apply to this method.

msoTriStateToggle Does not apply to this method.

msoTrue Creates a link to the source file of the Clipboard contents.

Remarks

An error occurs if there is no data on the Clipboard when the **PasteSpecial** method is called.

Example

The following example pastes a bitmap image as an icon into another window. This example assumes there are two open windows, and a bitmap image in the first window is currently selected.

```
Sub PasteOLEObject()  
    Windows(1).Selection.Copy  
    Windows(2).View.PasteSpecial DataType:=ppPasteOLEObject, _  
        DisplayAsIcon:=msoTrue, IconLabel:="New Bitmap Image"  
End Sub
```



[Show All](#)

Patterned Method

Sets the specified fill to a pattern.

expression.**Patterned**(*Pattern*)

expression Required. An expression that returns a **FillFormat** object.

Pattern Required [MsoPatternType](#). The pattern to be used for the specified fill.

MsoPatternType can be one of these MsoPatternType constants.

msoPattern10Percent

msoPattern20Percent

msoPattern25Percent

msoPattern30Percent

msoPattern40Percent

msoPattern50Percent

msoPattern5Percent

msoPattern60Percent

msoPattern70Percent

msoPattern75Percent

msoPattern80Percent

msoPattern90Percent

msoPatternDarkDownwardDiagonal

msoPatternDarkHorizontal

msoPatternDarkUpwardDiagonal

msoPatternDashedDownwardDiagonal

msoPatternDashedHorizontal

msoPatternDashedUpwardDiagonal

msoPatternDashedVertical

msoPatternDiagonalBrick

msoPatternDivot

msoPatternDottedDiamond
msoPatternDottedGrid
msoPatternHorizontalBrick
msoPatternLargeCheckerBoard
msoPatternLargeConfetti
msoPatternLargeGrid
msoPatternLightDownwardDiagonal
msoPatternLightHorizontal
msoPatternLightUpwardDiagonal
msoPatternLightVertical
msoPatternMixed
msoPatternNarrowHorizontal
msoPatternNarrowVertical
msoPatternOutlinedDiamond
msoPatternPlaid
msoPatternShingle
msoPatternSmallCheckerBoard
msoPatternSmallConfetti
msoPatternSmallGrid
msoPatternSolidDiamond
msoPatternSphere
msoPatternTrellis
msoPatternWave
msoPatternWeave
msoPatternWideDownwardDiagonal
msoPatternWideUpwardDiagonal
msoPatternZigZag
msoPatternDarkVertical

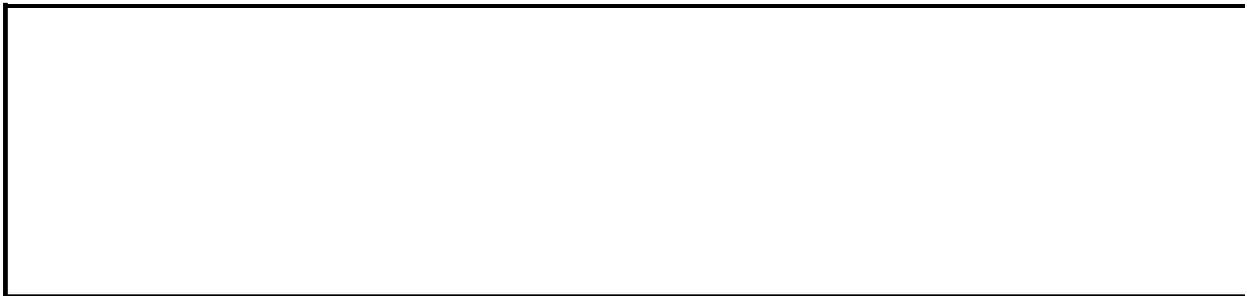
Remarks

Use the [BackColor](#) and [ForeColor](#) properties to set the colors used in the pattern.

Example

This example adds an oval with a patterned fill to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeOval, 60, 60, 80, 40).Fill
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(0, 0, 255)
    .Patterned msoPatternDarkVertical
End With
```



PickUp Method

Copies the formatting of the specified shape. Use the [Apply](#) method to apply the copied formatting to another shape.

expression.**PickUp**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Example

This example copies the formatting of shape one on myDocument, and then applies the copied formatting to shape two.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument
    .Shapes(1).PickUp
    .Shapes(2).Apply
End With
```



Picture Method

Sets the graphics file to be used for bullets in a bulleted list when the [Type](#) property of the **BulletFormat** object is set to **ppBulletPicture**.

expression.**Picture** (*Picture*)

expression Required. An expression that returns a **BulletFormat** object of type **ppBulletPicture**.

Picture Required **String**. The path and file name of a valid graphics file.

Remarks

Valid graphics files include files with the following extensions: .bmp, .cdr, .cgm, .drw, .dxf, .emf, .eps, .gif, .jpg, .jpeg, .pcd, .pct, .pcx, .pict, .png, .tga, .tiff, .wmf, and .wpg.

Example

This example sets the bullets in the text box specified by shape two on slide one to a bitmap picture of a blue rivet.

```
With ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat.Bullet
    .Type = ppBulletPicture
    .Picture ("C:\Windows\Blue Rivets.bmp")
  End With
End With
```



Play Method

Plays the specified sound effect.

expression.**Play**

expression Required. An expression that returns a **SoundEffect** object.

Example

This example plays the sound effect that's been set for the transition to slide two in the active presentation.

```
ActivePresentation.Slides(2).SlideShowTransition.SoundEffect.Play
```



PointsToScreenPixelsX Method

Converts a horizontal measurement from points to pixels. Used to return a horizontal screen location for a text frame or shape. Returns the converted measurement as a **Single**.

expression.**PointsToScreenPixelsX**(*Points*)

expression Required. An expression that returns a **DocumentWindow** object.

Points Required **Single**. The horizontal measurement (in points) to be converted to pixels.

Example

This example converts the width and height of the selected text frame bounding box from points to pixels, and returns the values to myXparm and myYparm.

```
With ActiveWindow
    myXparm = .PointsToScreenPixelsX _
        (.Selection.TextRange.Bounds.Width)
    myYparm = .PointsToScreenPixelsY _
        (.Selection.TextRange.Bounds.Height)
End With
```



PointsToScreenPixelsY Method

Converts a vertical measurement from points to pixels. Used to return a vertical screen location for a text frame or shape. Returns the converted measurement as a **Single**.

expression.**PointsToScreenPixelsY**(*Points*)

expression Required. An expression that returns a **DocumentWindow** object.

Points Required **Single**. The vertical measurement (in points) to be converted to pixels.

Example

This example converts the width and height of the selected text frame bounding box from points to pixels, and returns the values to myXparm and myYparm.

```
With ActiveWindow
    myXparm = .PointsToScreenPixelsX _
              (.Selection.TextRange.BoundsWidth)
    myYparm = .PointsToScreenPixelsY _
              (.Selection.TextRange.BoundsHeight)
End With
```



[Show All](#)

PresetDrop Method

Specifies whether the callout line attaches to the top, bottom, or center of the callout text box or whether it attaches at a point that's a specified distance from the top or bottom of the text box.

expression.**PresetDrop**(*DropType*)

expression Required. An expression that returns a **CalloutFormat** object.

DropType Required [MsoCalloutDropType](#). The starting position of the callout line relative to the text bounding box.

MsoCalloutDropType can be one of these MsoCalloutDropType constants.

msoCalloutDropBottom

msoCalloutDropCenter

msoCalloutDropCustom Specifying this constant will cause your code to fail.

msoCalloutDropMixed

msoCalloutDropTop

Example

This example specifies that the callout line attach to the top of the text bounding box for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(1).Callout.PresetDrop msoCalloutDropTop
```

This example toggles between two preset drops for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Callout
    If .DropType = msoCalloutDropTop Then
        .PresetDrop msoCalloutDropBottom
    ElseIf .DropType = msoCalloutDropBottom Then
        .PresetDrop msoCalloutDropTop
    End If
End With
```



[Show All](#)

PresetGradient Method

Sets the specified fill to a preset gradient.

expression.**PresetGradient**(*Style*, *Variant*, *PresetGradientType*)

expression Required. An expression that returns a **FillFormat** object.

Style Required [MsoGradientStyle](#). The gradient style.

MsoGradientStyle can be one of these MsoGradientStyle constants.

msoGradientDiagonalDown

msoGradientDiagonalUp

msoGradientFromCenter

msoGradientFromCorner

msoGradientFromTitle

msoGradientHorizontal

msoGradientMixed

msoGradientVertical

Variant Required **Integer**. The gradient variant. Can be a value from 1 to 4, corresponding to the four variants on the **Gradient** tab in the **Fill Effects** dialog box. If *Style* is **msoGradientFromTitle** or **msoGradientFromCenter**, this argument can be either 1 or 2.

PresetGradientType Required [MsoPresetGradientType](#). The gradient type.

MsoPresetGradientType can be one of these MsoPresetGradientType constants.

msoGradientBrass

msoGradientCalmWater

msoGradientChrome

msoGradientChromeII

msoGradientDaybreak

msoGradientDesert

msoGradientEarlySunset
msoGradientFire
msoGradientFog
msoGradientGold
msoGradientGoldII
msoGradientHorizon
msoGradientLateSunset
msoGradientMahogany
msoGradientMoss
msoGradientNightfall
msoGradientOcean
msoGradientParchment
msoGradientPeacock
msoGradientRainbow
msoGradientRainbowII
msoGradientSapphire
msoGradientSilver
msoGradientWheat
msoPresetGradientMixed

Example

This example adds a rectangle with a preset gradient fill to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddShape(msoShapeRectangle, 90, 90, 140, 80) _
    .Fill.PresetGradient msoGradientHorizontal, 1, msoGradientBrass
```



[Show All](#)

PresetTextured Method

Sets the specified fill to a preset texture.

expression.**PresetTextured**(*PresetTexture*)

expression Required. An expression that returns a **FillFormat** object.

PresetTexture Required [MsoPresetTexture](#). The preset texture.

MsoPresetTexture can be one of these MsoPresetTexture constants.

msoPresetTextureMixed

msoTextureBlueTissuePaper

msoTextureBouquet

msoTextureBrownMarble

msoTextureCanvas

msoTextureCork

msoTextureDenim

msoTextureFishFossil

msoTextureGranite

msoTextureGreenMarble

msoTextureMediumWood

msoTextureNewsprint

msoTextureOak

msoTexturePaperBag

msoTexturePapyrus

msoTextureParchment

msoTexturePinkTissuePaper

msoTexturePurpleMesh

msoTextureRecycledPaper

msoTextureSand

msoTextureStationery

msoTextureWalnut

msoTextureWaterDroplets
msoTextureWhiteMarble
msoTextureWovenMat

Example

This example adds a rectangle with a green-marble textured fill to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddShape(msoShapeCan, 90, 90, 40, 80) _
    .Fill.PresetTextured msoTextureGreenMarble
```



Previous Method

Show the slide immediately preceding the slide that's currently displayed. If you are currently on the first slide in a kiosk slide show, the **Previous** method takes you to the last slide in a slide show; otherwise, it has no effect if the first slide in the presentation is currently displayed. Use the [View](#) property of the **SlideShowWindow** object to return the **SlideShowView** object.

expression.**Previous**

expression Required. An expression that returns one of the items in the Applies To list.

Example

This example shows the slide immediately preceding the currently displayed slide on slide show window one.

```
SlideShowWindows(1).View.Previous
```



PrevNode Method

Returns a [DiagramNode](#) object that represents the previous diagram node in a collection of diagram nodes.

expression.**PrevNode**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds additional child nodes to the first child node, which is the node previous to the second node, in a newly-created diagram.

```
Sub AddNodeToFirstChild()  
  
    Dim dgnNode As DiagramNode  
    Dim dgnPrev As DiagramNode  
    Dim shpOrgChart As Shape  
    Dim intNodes As Integer  
  
    'Adds org chart and root node  
    Set shpOrgChart = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramOrgChart, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpOrgChart.DiagramNode.Children.AddNode  
  
    'Adds three child nodes to root node  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Sets dgnPrev equal to first child node (the node  
    'previous to the second child node)  
    Set dgnPrev = dgnNode.Children.Item(2).PrevNode  
  
    'Adds three child nodes to first child node  
    For intNodes = 1 To 3  
        dgnPrev.Children.AddNode  
    Next intNodes  
  
End Sub
```



[Show All](#)

PrintOut Method

Prints the specified presentation.

expression.**PrintOut**(*From*, *To*, *PrintToFile*, *Copies*, *Collate*)

expression Required. An expression that returns a **Presentation** object.

From Optional **Integer**. The number of the first page to be printed. If this argument is omitted, printing starts at the beginning of the presentation. Specifying the **To** and **From** arguments sets the contents of the [PrintRanges](#) object and sets the value of the [RangeType](#) property for the presentation.

To Optional **Integer**. The number of the last page to be printed. If this argument is omitted, printing continues to the end of the presentation. Specifying the **To** and **From** arguments sets the contents of the [PrintRanges](#) object and sets the value of the [RangeType](#) property for the presentation.

PrintToFile Optional **String**. The name of the file to print to. If you specify this argument, the file is printed to a file rather than sent to a printer. If this argument is omitted, the file is sent to a printer.

Copies Optional **Integer**. The number of copies to be printed. If this argument is omitted, only one copy is printed. Specifying this argument sets the value of the [NumberOfCopies](#) property.

Collate Optional [MsoTriState](#). If this argument is omitted, multiple copies are collated. Specifying this argument sets the value of the [Collate](#) property.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Prints a complete copy of the presentation before the first page of the next copy is printed.

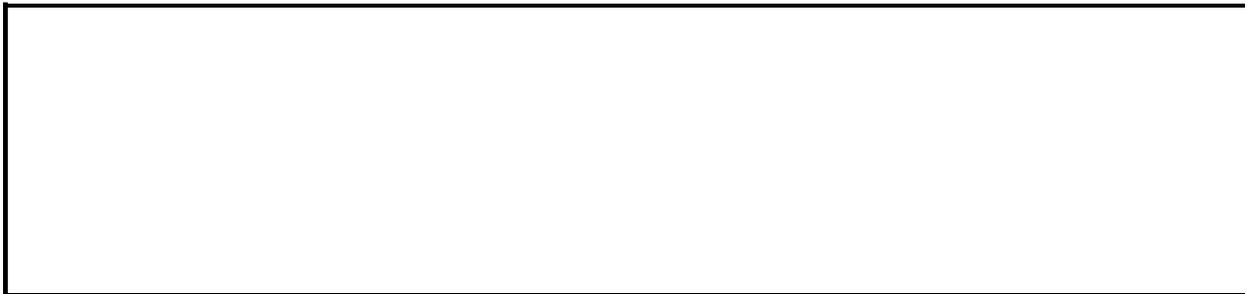
Example

This example prints two uncollated copies of each slide— whether visible or hidden— from slide two to slide five in the active presentation.

```
With Application.ActivePresentation
    .PrintOptions.PrintHiddenSlides = True
    .PrintOut From:=2, To:=5, Copies:=2, Collate:=msoFalse
End With
```

This example prints a single copy of all slides in the active presentation to the file Testprnt.prn.

```
Application.ActivePresentation.PrintOut _
    PrintToFile:="TestPrnt"
```



Publish Method

Creates a Web presentation (HTML format) from any loaded presentation. You can view the published presentation in a Web browser.

expression.**Publish**

expression Required. An expression that returns a **PublishObject** object.

Remarks

You can specify the content and attributes of the published presentation by setting various properties of the **PublishObject** object. For example, the **SourceType** property defines the portion of a loaded presentation to be published. The **RangeStart** property and the **RangeEnd** property specify the range of slides to publish, and the **SpeakerNotes** property designates whether or not to publish the speaker's notes.

Example

This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)  
    .FileName = "C:\Test\Mallard.htm"  
    .SourceType = ppPublishSlideRange  
    .RangeStart = 3  
    .RangeEnd = 5  
    .Publish  
End With
```



Quit Method

Quits PowerPoint. This is equivalent to clicking **Exit** on the **File** menu.

expression.**Quit**

expression Required. An expression that returns an **Application** object.

Remarks

To avoid being prompted to save changes, use either the **Save** or **SaveAs** method to save all open presentations before calling the **Quit** method.

Example

This example saves all open presentations and then quits PowerPoint.

```
With Application
  For Each w In .Presentations
    w.Save
  Next w
  .Quit
End With
```



[Show All](#)

Range Method

 [Range method as it applies to the **Shapes** object.](#)

Returns a **ShapeRange** object that represents a subset of the shapes in a **Shapes** collection.

expression.**Range**(*Index*)

expression Required. An expression that returns a **Shapes** collection object.

Index Optional **Variant**. The individual shapes that are to be included in the range. Can be an **Integer** that specifies the index number of the shape, a **String** that specifies the name of the shape, or an array that contains either integers or strings. If this argument is omitted, the **Range** method returns all the objects in the specified collection.

 [Range method as it applies to the **GroupShapes** object.](#)

Returns a **ShapeRange** object.

expression.**Range**(*Index*)

expression Required. An expression that returns one of the above objects.

Index Required **Variant**. The individual shapes that are to be included in the range. Can be an **Integer** that specifies the index number of the shape, a **String** that specifies the name of the shape, or an array that contains either integers or strings. If this argument is omitted, the **Range** method returns all the objects in the specified collection.

 [Range method as it applies to the **Slides** object.](#)

Returns a **SlideRange** object that represents a subset of the slides in a **Slides** collection.

expression.**Range**(*Index*)

expression Required. An expression that returns a **Slides** collection object.

Index Optional **Variant**. The individual slides that are to be included in the range. Can be an **Integer** that specifies the index number of the slide, a **String** that specifies the name of the slide, or an array that contains either integers or strings. If this argument is omitted, the **Range** method returns all the objects in the specified collection.

Remarks

Although you can use the **Range** method to return any number of shapes or slides, it's simpler to use the **Item** method if you only want to return a single member of the collection. For example, `Shapes(1)` is simpler than `Shapes.Range(1)`, and `Slides(2)` is simpler than `Slides.Range(2)`.

To specify an array of integers or strings for **Index**, you can use the **Array** function. For example, the following instruction returns two shapes specified by name.

```
Dim myArray() As Variant, myRange As Object
myArray = Array("Oval 4", "Rectangle 5")
Set myRange = ActivePresentation.Slides(1).Shapes.Range(myArray)
```

Example

 [As it applies to the **Shapes** object.](#)

This example sets the fill pattern for shapes one and three on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.Range(Array(1, 3)).Fill _
    .Patterned msoPatternHorizontalBrick
```

This example sets the fill pattern for the shapes named Oval 4 and Rectangle 5 on the first slide.

```
Dim myArray() As Variant, myRange As Object
myArray = Array("Oval 4", "Rectangle 5")
Set myRange = ActivePresentation.Slides(1).Shapes.Range(myArray)
myRange.Fill.Patterned msoPatternHorizontalBrick
```

This example sets the fill pattern for all shapes on the first slide.

```
ActivePresentation.Slides(1).Shapes.Range.Fill _
    .Patterned Pattern:=msoPatternHorizontalBrick
```

This example sets the fill pattern for shape one on the first slide.

```
Set myDocument = ActivePresentation.Slides(1)
Set myRange = myDocument.Shapes.Range(1)
myRange.Fill.Patterned msoPatternHorizontalBrick
```

This example creates an array that contains all the AutoShapes on the first slide, uses that array to define a shape range, and then distributes all the shapes in that range horizontally.

```
With myDocument.Shapes
    numShapes = .Count

    'Continues if there are shapes on the slide
    If numShapes > 1 Then
        numAutoShapes = 0
    End If
End With
```

```

ReDim autoShpArray(1 To numShapes)
For i = 1 To numShapes

    'Counts the number of AutoShapes on the Slide
    If .Item(i).Type = msoAutoShape Then
        numAutoShapes = numAutoShapes + 1
        autoShpArray(numAutoShapes) = .Item(i).Name
    End If
Next

'Adds AutoShapes to ShapeRange
If numAutoShapes > 1 Then
    ReDim Preserve autoShpArray(1 To numAutoShapes)
    Set asRange = .Range(autoShpArray)
    asRange.Distribute msoDistributeHorizontally, False
End If
End If
End With

```

 [As it applies to the **Slides** object.](#)

This example sets the title color for slides one and three.

```

Set mySlides = ActivePresentation.Slides.Range(Array(1, 3))
mySlides.ColorScheme.Colors(ppTitle).RGB = RGB(0, 255, 0)

```

This example sets the title color for the slides named Slide6 and Slide8.

```

Set mySlides = ActivePresentation.Slides _
    .Range(Array("Slide6", "Slide8"))
mySlides.ColorScheme.Colors(ppTitle).RGB = RGB(0, 255, 0)

```

This example sets the title color for all the slides in the active presentation.

```

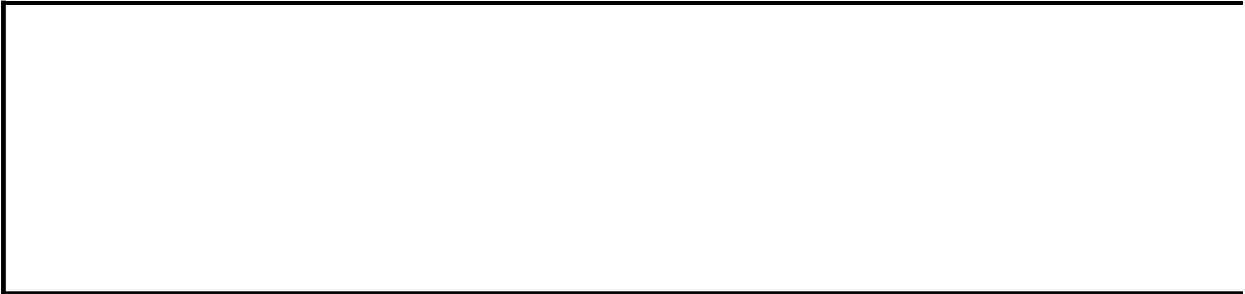
Set mySlides = ActivePresentation.Slides.Range
mySlides.ColorScheme.Colors(ppTitle).RGB = RGB(255, 0, 0)

```

This example creates an array that contains all the title slides in the active presentation, uses that array to define a slide range, and then sets the title color for all slides in that range.

```
Dim MyTitleArray() As Long
Set pSlides = ActivePresentation.Slides
ReDim MyTitleArray(1 To pSlides.Count)
For Each pSlide In pSlides
    If pSlide.Layout = ppLayoutTitle Then
        nCounter = nCounter + 1
        MyTitleArray(nCounter) = pSlide.SlideIndex
    End If
Next pSlide
ReDim Preserve MyTitleArray(1 To nCounter)

Set rngTitleSlides = ActivePresentation.Slides.Range(MyTitleArray)
rngTitleSlides.ColorScheme.Colors(ppTitle).RGB = RGB(255, 123, 99)
```



RangeFromPoint Method

Returns the **Shape** object that is located at the point specified by the screen position coordinate pair. If no shape is located at the coordinate pair specified, then the method returns **Nothing**.

expression.**RangeFromPoint**(*x*, *y*)

expression Required. An expression that returns a **DocumentWindow** object.

x Required **Long**. The horizontal distance (in pixels) from the left edge of the screen to the point.

y Required **Long**. The vertical distance (in pixels) from the top of the screen to the point.

Example

This example adds a new five-point star to slide one using the coordinates (288, 100). It then converts those coordinates from points to pixels, uses the **RangeFromPoint** method to return a reference to the new object, and changes the fill color of the star.

```
Dim myPointX As Integer, myPointY As Integer  
Dim myShape As Object
```

```
ActivePresentation.Slides(1).Shapes _  
    .AddShape(msoShape5pointStar, 288, 100, 100, 72).Select  
myPointX = ActiveWindow.PointsToScreenPixelsX(288)  
myPointY = ActiveWindow.PointsToScreenPixelsY(100)  
Set myShape = ActiveWindow.RangeFromPoint(myPointX, myPointY)  
myShape.Fill.ForeColor.RGB = RGB(80, 160, 130)
```



Regroup Method

Regroups the group that the specified shape range belonged to previously. Returns the regrouped shapes as a single **Shape** object.

expression.**Regroup**

expression Required. An expression that returns a **ShapeRange** object.

Remarks

The **Regroup** method only restores the group for the first previously grouped shape it finds in the specified **ShapeRange** collection. Therefore, if the specified shape range contains shapes that previously belonged to different groups, only one of the groups will be restored.

Note that because a group of shapes is treated as a single shape, grouping and ungrouping shapes changes the number of items in the **Shapes** collection and changes the index numbers of items that come after the affected items in the collection.

Example

This example regroups the shapes in the selection in the active window. If the shapes haven't been previously grouped and ungrouped, this example will fail.

```
ActiveWindow.Selection.ShapeRange.Regroup
```



[Show All](#)

ReloadAs Method

Reloads a presentation based on a specified HTML document [encoding](#).

expression.**ReloadAs** (*cp*)

expression Required. An expression that returns one of the objects in the Applies To list.

cp Required [MsoEncoding](#). The document encoding to use when reloading the Web page.

MsoEncoding can be one of these MsoEncoding constants.

msoEncodingArabicAutoDetect

msoEncodingAutoDetect

msoEncodingCyrillicAutoDetect

msoEncodingGreekAutoDetect

msoEncodingJapaneseAutoDetect

msoEncodingKoreanAutoDetect

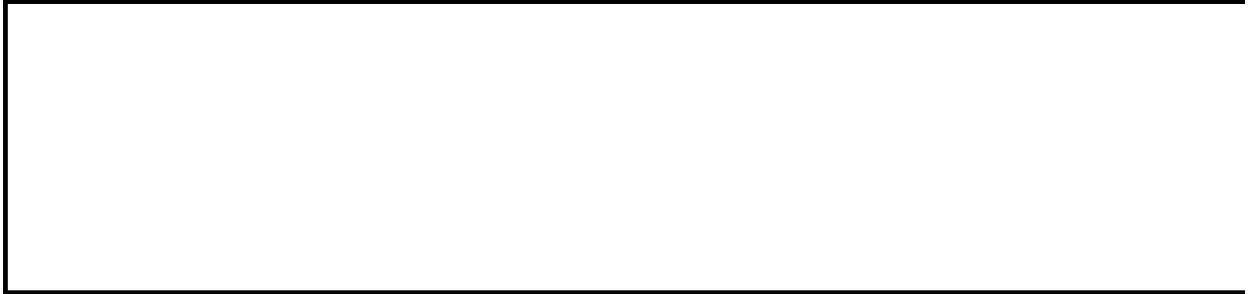
msoEncodingSimplifiedChineseAutoDetect

msoEncodingTraditionalChineseAutoDetect

Example

This example reloads the active presentation using Western encoding.

ActivePresentation.**ReloadAs** (msoEncodingWestern)



Remove Method

Removes an add-in from the collection of add-ins.

expression.**Remove**(*Index*)

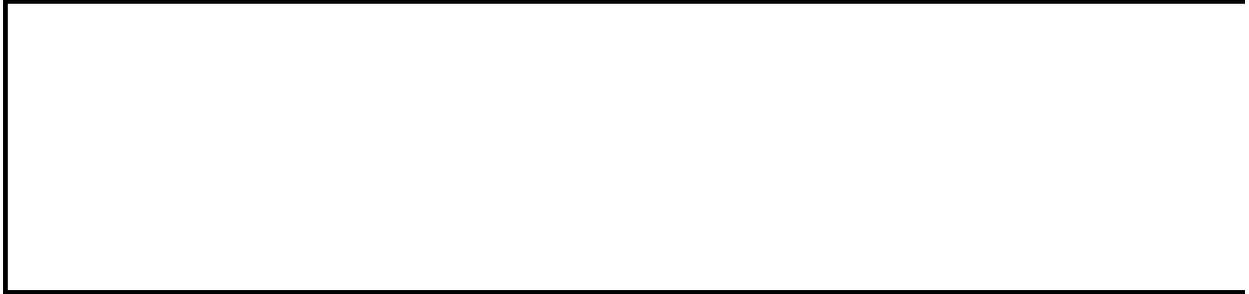
expression Required. An expression that returns an **AddIns** object.

Index Required **Variant**. The name of the add-in to be removed from the collection.

Example

This example removes the add-in named "MyTools" from the list of available add-ins.

```
AddIns.Remove "mytools"
```



RemoveBaseline Method

Removes the base line from the presentation.

expression.**RemoveBaseline**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This method generates an error if the presentation is an author presentation or if there is no base line.

Example

The following line of code removes the base line from the active presentation.

```
Sub RmvBaseline()  
    ActivePresentation.RemoveBaseline  
End Sub
```



RemovePeriods Method

Removes the period at the end of each paragraph in the specified text.

expression.**RemovePeriods**

expression Required. An expression that returns a **TextRange** object.

Example

This example removes the period at the end of each paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1) _  
    .Shapes(2).TextFrame.TextRange _  
    .RemovePeriods
```



[Show All](#)

Replace Method

 [Replace method as it applies to the **TextRange** object.](#)

Finds specific text in a text range, replaces the found text with a specified string, and returns a **TextRange** object that represents the first occurrence of the found text. Returns **Nothing** if no match is found.

expression.**Replace**(*FindWhat*, *ReplaceWhat*, *After*, *MatchCase*, *WholeWords*)

expression Required. An expression that returns one of the above objects.

FindWhat Required **String**. The text to search for.

ReplaceWhat Required **String**. The text you want to replace the found text with.

After Optional **Integer**. The position of the character (in the specified text range) after which you want to search for the next occurrence of **FindWhat**. For example, if you want to search from the fifth character of the text range, specify 4 for **After**. If this argument is omitted, the first character of the text range is used as the starting point for the search.

MatchCase Optional [MsoTriState](#). Determines whether a distinction is made on the basis of case.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Distinguish between uppercase and lowercase characters.

WholeWords Optional [MsoTriState](#). Determines whether only whole words are found.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Find only whole words, and not parts of larger words.

 [Replace method as it applies to the **Fonts** object.](#)

Replaces a font in the **Fonts** collection.

expression.**Replace(Original, Replacement)**

expression Required. An expression that returns one of the above objects.

Original Required **String**. The name of the font to replace.

Replacement Required **String**. The the name of the replacement font.

Example

 [As it applies to the **TextRange** object.](#)

This example replaces every whole-word occurrence of "like" in all of the shapes in the active presentation with "NOT LIKE".

```
Sub ReplaceText()  
  
    Dim oSld As Slide  
    Dim oShp As Shape  
    Dim oTxtRng As TextRange  
    Dim oTmpRng As TextRange  
  
    Set oSld = Application.ActivePresentation.Slides(1)  
  
    For Each oShp In oSld.Shapes  
        Set oTxtRng = oShp.TextFrame.TextRange  
        Set oTmpRng = oTxtRng.Replace(FindWhat:="like", _  
            Replacewhat:="NOT LIKE", WholeWords:=True)  
        Do While Not oTmpRng Is Nothing  
            Set oTxtRng = oTxtRng.Characters(oTmpRng.Start + oTmpRng.  
                oTxtRng.Length)  
            Set oTmpRng = oTxtRng.Replace(FindWhat:="like", _  
                Replacewhat:="NOT LIKE", WholeWords:=True)  
        Loop  
    Next oShp  
  
End Sub
```

 [As it applies to the **Fonts** object.](#)

This example replaces the Times New Roman font with the Courier font in the active presentation.

```
Application.ActivePresentation.Fonts _  
    .Replace Original:="Times New Roman", Replacement:="Courier"
```



ReplaceNode Method

Replaces a target diagram node with the source diagram node. The target diagram node is deleted, and the source diagram node, including any of its child nodes, are moved to where the target diagram node was.

expression.**ReplaceNode**(*TargetNode*)

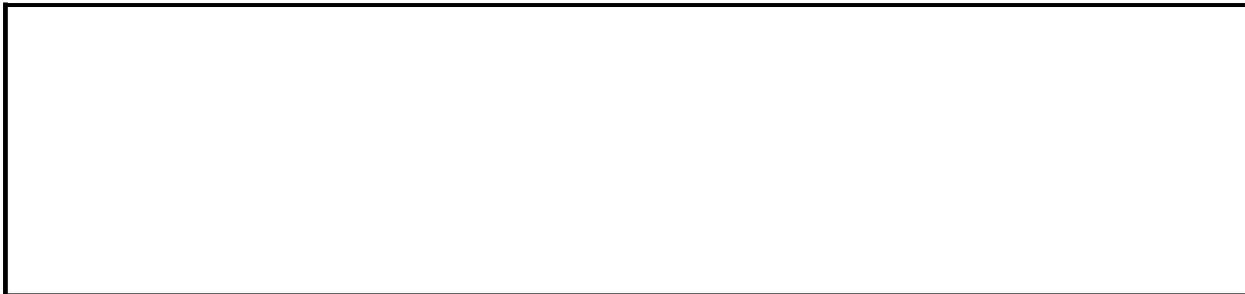
expression Required. An expression that returns one of the objects in the Applies To list.

TargetNode Required **DiagramNode** object. The diagram node to be replaced.

Example

The following example replaces the last diagram node of a newly-created diagram with the second node.

```
Sub ReplaceLastNode()  
  
    Dim dgnNode As DiagramNode  
    Dim shpRadial As Shape  
    Dim intNodes As Integer  
  
    'Adds radial diagram and root node  
    Set shpRadial = ActivePresentation.Slides(1).Shapes.AddDiagram _  
        (Type:=msoDiagramRadial, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpRadial.DiagramNode.Children.AddNode  
  
    'Adds three additional child nodes  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Displays the number of nodes in the diagram  
    MsgBox "The number of nodes in the diagram : " & _  
        dgnNode.Diagram.Nodes.Count  
  
    'Second node replaces the last node.  
    dgnNode.Diagram.Nodes(2).ReplaceNode _  
        TargetNode:=dgnNode.Diagram.Nodes(4)  
  
    'Node count is three because the replaced node was deleted  
    MsgBox "The number of nodes in the diagram : " & _  
        dgnNode.Diagram.Nodes.Count  
  
End Sub
```



ReplyWithChanges Method

Sends an e-mail message to the author of a presentation that has been sent out for review, notifying them that a reviewer has completed review of the presentation.

expression.**ReplyWithChanges**(*ShowMessage*)

expression Required. An expression that returns one of the objects in the Applies To list.

ShowMessage Optional **Boolean**. **True** to display the message prior to sending. **False** to automatically send the message without displaying it first. The default is **True**.

Remarks

Use the [SendForReview](#) method to start a collaborative review of a presentation. If the **ReplyWithChanges** method is executed on a presentation that is not part of a collaborative review cycle, the user will receive an error message.

Example

This example sends a message to the author of a review document that a reviewer has completed a review, without first displaying the e-mail message to the reviewer. This example assumes that the active presentation is part of a collaborative review cycle.

```
Sub ReplyMsg()  
    ActivePresentation.ReplyWithChanges ShowMessage:=False  
End Sub
```



RerouteConnections Method

Reroutes connectors so that they take the shortest possible path between the shapes they connect. To do this, the **RerouteConnections** method may detach the ends of a connector and reattach them to different connecting sites on the connected shapes.

This method reroutes all connectors attached to the specified shape; if the specified shape is a connector, it's rerouted.

expression.**RerouteConnections**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Remarks

If this method is applied to a connector, only that connector will be rerouted. If this method is applied to a connected shape, all connectors to that shape will be rerouted.

Example

This example adds two rectangles to myDocument, connects them with a curved connector, and then reroutes the connector so that it takes the shortest possible path between the two rectangles. Note that the **RerouteConnections** method adjusts the size and position of the connector and determines which connecting sites it attaches to, so the values you initially specify for the **ConnectionSite** arguments used with the **BeginConnect** and **EndConnect** methods are irrelevant.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
Set newConnector = s _
    .AddConnector(msoConnectorCurve, 0, 0, 100, 100)
With newConnector.ConnectorFormat
    .BeginConnect firstRect, 1
    .EndConnect secondRect, 1
End With
newConnector.RerouteConnections
```



ResetRotation Method

Resets the extrusion rotation around the x-axis and the y-axis to 0 (zero) so that the front of the extrusion faces forward. This method doesn't reset the rotation around the z-axis.

expression.**ResetRotation**

expression Required. An expression that returns a **ThreeDFormat** object.

Remarks

To set the extrusion rotation around the x-axis and the y-axis to anything other than 0 (zero), use the [RotationX](#) and [RotationY](#) properties of the **ThreeDFormat** object. To set the extrusion rotation around the z-axis, use the [Rotation](#) property of the [Shape](#) object that represents the extruded shape.

Example

This example resets the rotation around the x-axis and the y-axis to 0 (zero) for the extrusion of shape one on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).ThreeD.ResetRotation
```



ResetSlideTime Method

Resets the elapsed time (represented by the [SlideElapsedTime](#) property) for the slide that's currently displayed to 0 (zero).

expression.**ResetSlideTime**

expression Required. An expression that returns a **SlideShowView** object.

Example

This example resets the elapsed time for the slide that's currently displayed in slide show window one to 0 (zero).

```
SlideShowWindows(1).View.ResetSlideTime
```



RotatedBounds Method

Returns the coordinates of the vertices of the text bounding box for the specified text range.

expression.**RotatedBounds**(*X1*, *Y1*, *X2*, *Y2*, *X3*, *Y3*, *X4*, *Y4*)

expression Required. An expression that returns a **TextRange** object.

X1, Y1 Required **Single**. Returns the position (in points) of the first vertex of the bounding box for the text within the specified text range.

X2, Y2 Required **Single**. Returns the position (in points) of the second vertex of the bounding box for the text within the specified text range.

X3, Y3 Required **Single**. Returns the position (in points) of the third vertex of the bounding box for the text within the specified text range.

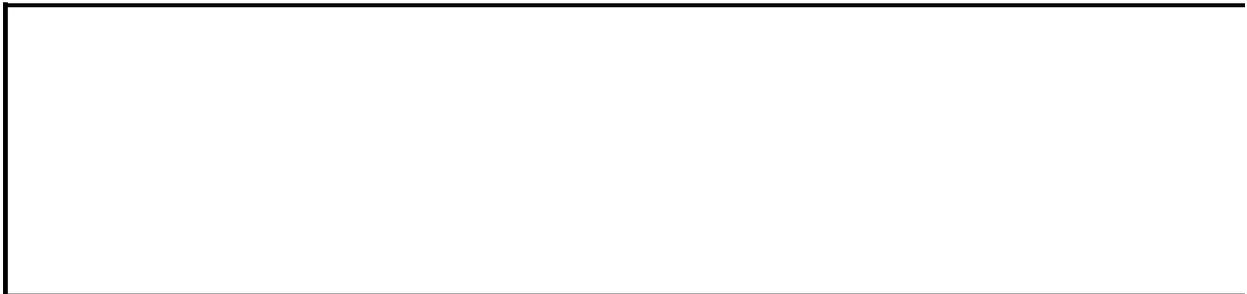
X4, Y4 Required **Single**. Returns the position (in points) of the fourth vertex of the bounding box for the text within the specified text range.

Example

This example uses the values returned by the arguments of the **RotatedBounds** method to draw a freeform that has the dimensions of the text bounding box for the third word in the text range in shape one on slide one in the active presentation.

```
Dim x1 As Single, y1 As Single
Dim x2 As Single, y2 As Single
Dim x3 As Single, y3 As Single
Dim x4 As Single, y4 As Single
Dim myDocument As Slide

Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(1).TextFrame.TextRange.Words(3).RotatedBounds _
    x1, y1, x2, y2, x3, y3, x4, y4
With myDocument.Shapes.BuildFreeform(msoEditingCorner, x1, y1)
    .AddNodes msoSegmentLine, msoEditingAuto, x2, y2
    .AddNodes msoSegmentLine, msoEditingAuto, x3, y3
    .AddNodes msoSegmentLine, msoEditingAuto, x4, y4
    .AddNodes msoSegmentLine, msoEditingAuto, x1, y1
    .ConvertToShape.ZOrder msoSendToBack
End With
```



RtlRun Method

Sets the direction of text in a text range to read from right to left.

expression.**RtlRun**

expression Required. An expression that returns a **TextRange** object.

Remarks

This method makes it possible to use text from both left-to-right and right-to-left languages in the same presentation.

Example

The following example finds all of the shapes on slide one that contain text and changes the text to read from right to left.

```
ActiveWindow.ViewType = ppViewSlide
For Each sh In ActivePresentation.Slides(1).Shapes
    If sh.HasTextFrame Then
        sh.TextFrame.TextRange.Rt1Run
    End If
Next
```



[Show All](#)

Run Method

 [Run method as it applies to the **SlideShowSettings** object.](#)

Runs a slide show of the specified presentation. Returns a [SlideShowWindow](#) object.

expression.**Run**

expression Required. An expression that returns one of the above objects.

Remarks

To run a custom slide show, set the **RangeType** property to **ppShowNamedSlideShow**, and set the **SlideShowName** property to the name of the custom show you want to run.



[Run method as it applies to the **Application** object.](#)

Runs a Visual Basic procedure.

Note Because macros can contain viruses, be careful about running them. Take the following precautions: run up-to-date antivirus software on your computer; set your macro security level to high; clear the **Trust all installed add-ins and templates** check box; use digital signatures; maintain a list of trusted publishers.

expression.**Run**(*MacroName*, *safeArrayOfParams*)

expression Required. An expression that returns one of the above objects.

MacroName Required **String**. The name of the procedure to be run. The string can contain the following: a loaded presentation or add-in file name followed by an exclamation point (!), a valid module name followed by a period (.), and the procedure name. For example, the following is a valid **MacroName** value:
"MyPres.ppt!Module1.Test."

safeArrayOfParams Required **Variant**. The argument to be passed to the procedure. You cannot specify an object for this argument, and you cannot use named arguments with this method. Arguments must be passed by position.

Example

[As it applies to the **SlideShowSettings** object.](#)

This example starts a full-screen slide show of the active presentation, with shortcut keys disabled.

```
With ActivePresentation.SlideShowSettings
    .ShowType = ppShowSpeaker
    .Run.View.AcceleratorsEnabled = False
End With
```

This example runs the named slide show "Quick Show."

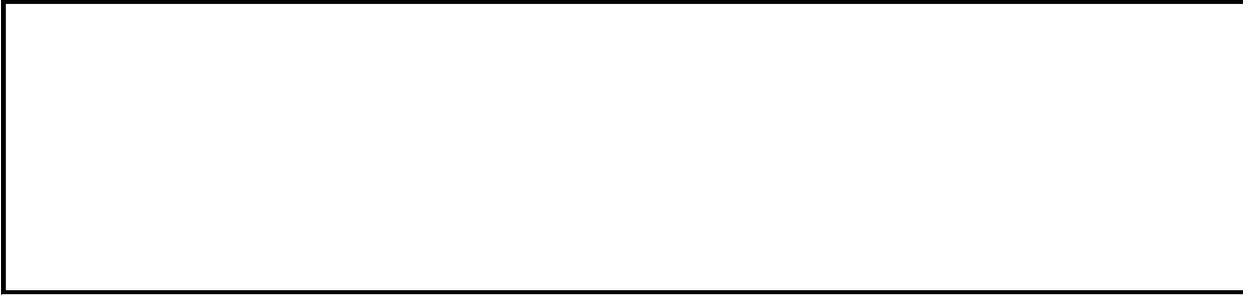
```
With ActivePresentation.SlideShowSettings
    .RangeType = ppShowNamedSlideShow
    .SlideShowName = "Quick Show"
    .Run
End With
```

[As it applies to the **Application** object.](#)

In this example, the Main procedure defines an array and then runs the macro TestPass, passing the array as an argument.

```
Sub Main()
    Dim x(1 To 2)
    x(1) = "hi"
    x(2) = 7
    Application.Run "TestPass", x
End Sub

Sub TestPass(x)
    MsgBox x(1)
    MsgBox x(2)
End Sub
```



Runs Method

Returns a **TextRange** object that represents the specified subset of text runs. A text run consists of a range of characters that share the same font attributes. For information about counting or looping through the runs in a text range, see the [TextRange](#) object.

expression.**Runs**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first run in the returned range.

Length Optional **Long**. The number of runs to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first run and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one run.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first run in the specified range.

If ***Start*** is greater than the number of runs in the specified text, the returned range starts with the last run in the specified range.

If ***Length*** is greater than the number of runs from the specified starting run to the end of the text, the returned range contains all those runs.

A run consists of all characters from the first character after a font change to the second-to-last character with the same font attributes. For example, consider the following sentence:

This *italic* word is not **bold**.

In the preceding sentence, the first run consists of the word "This" only if the space after the word "This" isn't formatted as italic (if the space is italic, the first run is only the first three characters, or "Thi"). Likewise, the second run contains the word "italic" only if the space after the word is formatted as italic.

Example

This example formats the second run in shape two on slide one in the active presentation as bold italic if it's already italic.

```
With Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange  
    With .Runs(2).Font  
        If .Italic Then  
            .Bold = True  
        End If  
    End With  
End With
```



Save Method

Saves the specified presentation.

expression.**Save**

expression Required. An expression that returns a **Presentation** object.

Remarks

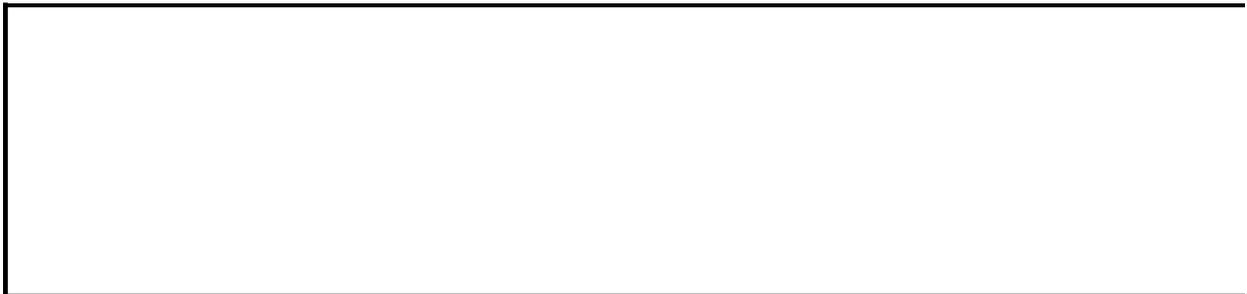
Use the [SaveAs](#) method to save a presentation that hasn't been previously saved. To determine whether a presentation has been saved, test for a nonempty value for the [FullName](#) or [Path](#) property. If a document with the same name as the specified presentation already exists on disk, that document will be overwritten. No warning message will be displayed.

To mark the presentation as saved without writing it to disk, set the [Saved](#) property to **True**.

Example

This example saves the active presentation if it's been changed since the last time it was saved.

```
With Application.ActivePresentation  
    If Not .Saved And .Path <> "" Then .Save  
End With
```



[Show All](#)

SaveAs Method

Saves a presentation that's never been saved, or saves a previously saved presentation under a different name.

expression.**SaveAs**(*Filename*, *FileFormat*, *EmbedFonts*)

expression Required. An expression that returns a **Presentation** object.

Filename Required **String**. Specifies the name to save the file under. If you don't include a full path, PowerPoint saves the file in the current folder.

FileFormat Optional [PpSaveAsFileType](#). Specifies the saved file format. If this argument is omitted, the file is saved in the format of a presentation in the current version of PowerPoint (**ppSaveAsPresentation**).

PpSaveAsFileType can be one of these PpSaveAsFileType constants.

ppSaveAsHTMLv3

ppSaveAsAddIn

ppSaveAsBMP

ppSaveAsDefault

ppSaveAsGIF

ppSaveAsHTML

ppSaveAsHTMLDual

ppSaveAsJPG

ppSaveAsMetaFile

ppSaveAsPNG

ppSaveAsPowerPoint3

ppSaveAsPowerPoint4

ppSaveAsPowerPoint4FarEast

ppSaveAsPowerPoint7

ppSaveAsPresentation Default.

ppSaveAsRTF

ppSaveAsShow

ppSaveAsTemplate

ppSaveAsTIF

ppSaveAsWebArchive

EmbedFonts Optional [MsoTriState](#). Specifies whether PowerPoint embeds TrueType fonts in the saved presentation.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed Default.

msoTriStateToggle

msoTrue PowerPoint embeds TrueType fonts in the saved presentation.

Example

This example saves a copy of the active presentation under the name "New Format Copy.ppt." By default, this copy is saved in the format of a presentation in the current version of PowerPoint. The presentation is then saved as a PowerPoint 4.0 file named "Old Format Copy."

```
With Application.ActivePresentation
    .SaveCopyAs "New Format Copy"
    .SaveAs "Old Format Copy", ppSaveAsPowerPoint4
End With
```



[Show All](#)

SaveCopyAs Method

Saves a copy of the specified presentation to a file without modifying the original.

expression.**SaveCopyAs**(*FileName*, *FileFormat*, *EmbedTrueTypeFonts*)

expression Required. An expression that returns a **Presentation** object.

FileName Required **String**. Specifies the name to save the file under. If you don't include a full path, PowerPoint saves the file in the current folder.

FileFormat Optional [PpSaveAsFileType](#). The file format.

PpSaveAsFileType can be one of these PpSaveAsFileType constants.

ppSaveAsHTMLv3

ppSaveAsAddIn

ppSaveAsBMP

ppSaveAsDefault *default*

ppSaveAsGIF

ppSaveAsHTML

ppSaveAsHTMLDual

ppSaveAsJPG

ppSaveAsMetaFile

ppSaveAsPNG

ppSaveAsPowerPoint3

ppSaveAsPowerPoint4

ppSaveAsPowerPoint4FarEast

ppSaveAsPowerPoint7

ppSaveAsPresentation

ppSaveAsRTF

ppSaveAsShow

ppSaveAsTemplate

ppSaveAsTIF

ppSaveAsWebArchive

EmbedTrueTypeFonts Optional [MsoTriState](#). Specifies whether TrueType fonts are embedded.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed *default*

msoTriStateToggle

msoTrue

Example

This example saves a copy of the active presentation under the name "New Format Copy.ppt." By default, this copy is saved in the format of a presentation in the current version of PowerPoint. The presentation is then saved as a PowerPoint 4.0 file named "Old Format Copy."

```
With Application.ActivePresentation
    .SaveCopyAs "New Format Copy"
    .SaveAs "Old Format Copy", ppSaveAsPowerPoint4
End With
```



[Show All](#)

ScaleHeight Method

Scales the height of the shape by a specified factor. For pictures and OLE objects, you can indicate whether you want to scale the shape relative to the original size or relative to the current size. Shapes other than pictures and OLE objects are always scaled relative to their current height.

expression.**ScaleHeight**(*Factor*, *RelativeToOriginalSize*, *fScale*)

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Factor Required **Single**. Specifies the ratio between the height of the shape after you resize it and the current or original height. For example, to make a rectangle 50 percent larger, specify 1.5 for this argument.

RelativeToOriginalSize Required **MsoTriState**. Specifies whether the shape is scaled relative to its current or original size.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Scales the shape relative to its current size.

msoTriStateMixed

msoTriStateToggle

msoTrue Scales the shape relative to its original size. You can specify **msoTrue** for this argument only if the specified shape is a picture or an OLE object.

fScale Optional **MsoScaleFrom**. The part of the shape that retains its position when the shape is scaled.

MsoScaleFrom can be one of these MsoScaleFrom constants.

msoScaleFromBottomRight

msoScaleFromMiddle

msoScaleFromTopLeft Default.

Example

This example scales all pictures and OLE objects on myDocument to 175 percent of their original height and width, and it scales all other shapes to 175 percent of their current height and width.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  Select Case s.Type
    Case msoEmbeddedOLEObject, msoLinkedOLEObject, _
          msoOLEControlObject, msoLinkedPicture, msoPicture
      s.ScaleHeight 1.75, msoTrue
      s.ScaleWidth 1.75, msoTrue
    Case Else
      s.ScaleHeight 1.75, msoFalse
      s.ScaleWidth 1.75, msoFalse
  End Select
Next
```



[Show All](#)

ScaleWidth Method

Scales the width of the shape by a specified factor. For pictures and OLE objects, you can indicate whether you want to scale the shape relative to the original size or relative to the current size. Shapes other than pictures and OLE objects are always scaled relative to their current width.

expression.**ScaleWidth**(*Factor*, *RelativeToOriginalSize*, *fScale*)

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Factor Required **Single**. Specifies the ratio between the width of the shape after you resize it and the current or original width. For example, to make a rectangle 50 percent larger, specify 1.5 for this argument.

RelativeToOriginalSize Required **MsoTriState**. Specifies whether a shape is scaled relative to its current or original size.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Scales the shape relative to its current size.

msoTriStateMixed

msoTriStateToggle

msoTrue Scales the shape relative to its original size. You can specify **msoTrue** for this argument only if the specified shape is a picture or an OLE object.

fScale Optional **MsoScaleFrom**. The part of the shape that retains its position when the shape is scaled.

MsoScaleFrom can be one of these MsoScaleFrom constants.

msoScaleFromBottomRight

msoScaleFromMiddle

msoScaleFromTopLeft Default.

Example

This example scales all pictures and OLE objects on myDocument to 175 percent of their original height and width, and it scales all other shapes to 175 percent of their current height and width.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  Select Case s.Type
    Case msoEmbeddedOLEObject, msoLinkedOLEObject, _
         msoOLEControlObject, msoLinkedPicture, msoPicture
      s.ScaleHeight 1.75, msoTrue
      s.ScaleWidth 1.75, msoTrue
    Case Else
      s.ScaleHeight 1.75, msoFalse
      s.ScaleWidth 1.75, msoFalse
  End Select
```



[Show All](#)

ScrollIntoView Method

Scrolls the document window so that items within a specified rectangular area are displayed in the document window or pane.

expression.**ScrollIntoView**(*Left, Top, Width, Height, Start*)

expression Required. An expression that returns a **DocumentWindow** object.

Left Required **Long**. The horizontal distance (in points) from the left edge of the document window to the rectangle.

Top Required **Long**. The vertical distance (in points) from the top of the document window to the rectangle.

Width Required **Long**. The width of the rectangle (in points).

Height Required **Long**. The height of the rectangle (in points).

Start Optional [MsoTriState](#). Determines the starting position of the rectangle in relation to the document window.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The bottom right of the rectangle is to appear at the bottom right of the document window.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. The top left of the rectangle is to appear at the top left of the document window.

Remarks

If the bounding rectangle is larger than the document window, the ***Start*** argument specifies which end of the rectangle displays or gets initial focus. This method cannot be used with outline or slide sorter views.

Example

This example brings into view a 100x200 point area beginning 50 points from the left edge of the slide, and 20 points from the top of the slide. The top left corner of the rectangle is positioned at the top left corner of the active document window.

```
ActiveWindow.ScrollIntoView Left:=50, Top:=20, _  
    Width:=100, Height:=200
```



[Show All](#)

Select Method

 [Select method as it applies to the **Cell**, **Column**, **Row**, **Slide**, **SlideRange**, and **TextRange** objects.](#)

Selects the specified object.

expression.**Select**

expression Required. An expression that returns one of the above objects.

 [Select method as it applies to the **Shape** and **ShapeRange** objects.](#)

Selects the specified object.

expression.**Select(Replace)**

expression Required. An expression that returns one of the above objects.

Replace Optional [MsoTriState](#). Specifies whether the selection replaces any previous selection.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The selection is added to the previous selection.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. The selection replaces any previous selection.

Remarks

If you try to make a selection that isn't appropriate for the view, your code will fail. For example, you can select a slide in slide sorter view but not in slide view.

Example

 [As it applies to the **Cell**, **Column**, **Row**, **Slide**, **SlideRange**, and **TextRange** objects](#)

This example selects the first five characters in the title of slide one in the active presentation.

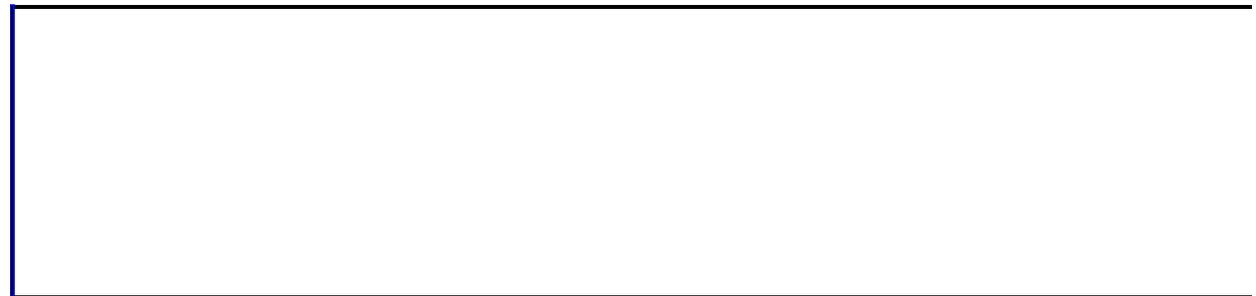
```
ActivePresentation.Slides(1).Shapes.Title.TextFrame _  
    .TextRange.Characters(1, 5).Select
```

This example selects slide one in the active presentation.

```
ActivePresentation.Slides(1).Select
```

This example selects a table that has been added to a new slide in a new presentation. The table has three rows and three columns.

```
With Presentations.Add.Slides  
    .Add(1, ppLayoutBlank).Shapes.AddTable(3, 3).Select  
End With
```



 [As it applies to the **Shape** and **ShapeRange** objects.](#)

This example selects shapes one and three on slide one in the active presentation.

```
ActivePresentation.Slides(1).Shapes.Range(Array(1, 3)).Select
```

This example adds shapes two and four on slide one in the active presentation to

the previous selection.

```
ActivePresentation.Slides(1).Shapes.Range(Array(2, 4)).Select False
```

SelectAll Method

Selects all shapes (in a [Shapes](#) collection) or all diagram nodes (in a [DiagramNodes](#) or [DiagramNodeChildren](#) collection).

expression.**SelectAll**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example selects all the shapes on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes.SelectAll
```



SendFaxOverInternet Method

Sends a presentation as a fax to the specified recipients.

expression.**SendFaxOverInternet**(*Recipients*, *Subject*, *ShowMessage*)

expression Required. An expression that returns a **Presentation** object.

Recipients Optional **Variant**. A **String** that represents the fax numbers and e-mail addresses of the people to whom to send the fax. Separate multiple recipients with a semicolon.

Subject Optional **Variant**. A **String** that represents the subject line for the faxed presentation.

ShowMessage Optional **Variant**. **True** displays the fax message before sending it. **False** sends the fax without displaying the fax message.

Remarks

Using the **SendFaxOverInternet** method requires that the fax service be enabled on a user's computer.

The format used for specifying fax numbers in the **Recipients** parameter is either *recipientsfaxnumber@usersfaxprovider* or *recipientsname@recipientsfaxnumber*. You can access the user's fax provider information using the following registry path:

```
HKEY_CURRENT_USER\Software\Microsoft\Office\11.0\Common\Services\Fax
```

Use the FaxAddress key value under the above registry path to determine the format to use for a user.

Example

The following example sends a fax to the fax service provider, who will fax the message to the recipient.

```
ActivePresentation.SendFaxOverInternet _  
    "14255550101@consolidatedmessenger.com", _  
    "For your review", True
```



SendForReview Method

Sends a presentation in an e-mail message for review to the specified recipients.

expression.**SendForReview**(*Recipients*, *Subject*, *ShowMessage*, *IncludeAttachment*)

expression Required. An expression that returns one of the objects in the Applies To list.

Recipients Optional **String**. A string that lists the people to whom to send the message. These can be unresolved names and aliases in an e-mail phone book or full e-mail addresses. Separate multiple recipients with a semicolon (;). If left blank and **ShowMessage** is **False**, you will receive an error message, and the message will not be sent.

Subject Optional **String**. A string for the subject of the message. If left blank, the subject will be: Please review "*filename*".

ShowMessage Optional **Boolean**. A **Boolean** value that indicates whether the message should be displayed when the method is executed. The default value is **True**. If set to **False**, the message is automatically sent to the recipients without first showing the message to the sender.

IncludeAttachment Optional **Variant**. A **Boolean** value that indicates whether the message should include an attachment or a link to a server location. The default value is **True**. If set to **False**, the document must be stored at a shared location.

Remarks

The **SendForReview** method starts a collaborative review cycle. Use the [EndReview](#) method to end a review cycle.

Example

This example automatically sends the active presentation as an attachment in an e-mail message to the specified recipients.

```
Sub WebReview()  
    ActivePresentation.SendForReview _  
        Recipients:="someone@example.com; Dan Wilson", _  
        Subject:="Please review this document.", _  
        ShowMessage:=False, _  
        IncludeAttachment:=True  
End Sub
```



Sentences Method

Returns a **TextRange** object that represents the specified subset of text sentences. For information about counting or looping through the sentences in a text range, see the [TextRange](#) object.

expression.**Sentences**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first sentence in the returned range.

Length Optional **Long**. The number of sentences to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first sentence and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one sentence.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first sentence in the specified range.

If ***Start*** is greater than the number of sentences in the specified text, the returned range starts with the last sentence in the specified range.

If ***Length*** is greater than the number of sentences from the specified starting sentence to the end of the text, the returned range contains all those sentences.

Example

This example formats as bold the second sentence in the second paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange.Paragraphs(2).Sentences(2).Font _  
    .Bold = True
```



[Show All](#)

SetEditingType Method

Sets the editing type of the node specified by ***Index***. If the node is a control point for a curved segment, this method sets the editing type of the node adjacent to it that joins two segments. Note that, depending on the editing type, this method may affect the position of adjacent nodes.

expression.**SetEditingType**(*Index*, *EditingType*)

expression Required. An expression that returns a **ShapeNodes** object.

Index Required **Long**. The node whose editing type is to be set.

EditingType Required [MsoEditingType](#). The editing type.

MsoEditingType can be one of these MsoEditingType constants.

msoEditingAuto

msoEditingCorner

msoEditingSmooth

msoEditingSymmetric

Example

This example changes all corner nodes to smooth nodes in shape three on myDocument. Shape three must be a freeform drawing.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    For n = 1 to .Count
        If .Item(n).EditingType = msoEditingCorner Then
            .SetEditingType n, msoEditingSmooth
        End If
    Next
End With
```



[Show All](#)

SetExtrusionDirection Method

Sets the direction that the extrusion's sweep path takes away from the extruded shape.

expression.**SetExtrusionDirection**(*PresetExtrusionDirection*)

expression Required. An expression that returns a **ThreeDFormat** object.

PresetExtrusionDirection Required [MsoPresetExtrusionDirection](#). Specifies the extrusion direction.

MsoPresetExtrusionDirection can be one of these MsoPresetExtrusionDirection constants.

msoExtrusionBottom

msoExtrusionBottomLeft

msoExtrusionBottomRight

msoExtrusionLeft

msoExtrusionNone

msoExtrusionRight

msoExtrusionTop

msoExtrusionTopLeft

msoExtrusionTopRight

msoPresetExtrusionDirectionMixed

Remarks

This method sets the [PresetExtrusionDirection](#) property to the direction specified by the *PresetExtrusionDirection* argument.

Example

This example specifies that the extrusion for shape one on myDocument extend toward the top of the shape and that the lighting for the extrusion come from the left.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```



[Show All](#)

SetPasswordEncryptionOptions

Method

Sets the options Microsoft PowerPoint uses for encrypting presentations with passwords.

expression.**SetPasswordEncryptionOptions**(*PasswordEncryptionProvider*, *PasswordEncryptionFileProperties*, *PasswordEncryptionKeyLength*, *PasswordEncryptionAlgorithm*)

expression Required. An expression that returns one of the objects in the Applies To list.

PasswordEncryptionProvider Required **String**. The name of the encryption provider.

PasswordEncryptionAlgorithm Required **String**. The name of the encryption algorithm. PowerPoint supports stream-encrypted algorithms.

PasswordEncryptionKeyLength Required **Long**. The encryption key length. Must be a multiple of 8, starting at 40.

PasswordEncryptionFileProperties Required **MsoTriState**. **MsoTrue** for PowerPoint to encrypt file properties.

MsoTriState can be one of these MsoTriState constants.

msoCTrue Not used with this method.

msoFalse

msoTriStateMixed Not used with this method.

msoTriStateToggle Not used with this method.

msoTrue

Example

This example sets the password encryption options if the file properties are not encrypted for password-protected documents.

```
Sub PasswordSettings()  
  With ActivePresentation  
    If .PasswordEncryptionFileProperties = msoFalse Then  
      .SetPasswordEncryptionOptions _  
        PasswordEncryptionProvider:="Microsoft RSA SChannel  
        PasswordEncryptionAlgorithm:="RC4", _  
        PasswordEncryptionKeyLength:=56, _  
        PasswordEncryptionFileProperties:=True  
    End If  
  End With  
End Sub
```



SetPosition Method

Sets the location of the node specified by ***Index***. Note that, depending on the editing type of the node, this method may affect the position of adjacent nodes.

expression.**SetPosition**(*Index*, *X1*, *Y1*)

expression Required. An expression that returns a **ShapeNodes** object.

Index Required **Long**. The node whose position is to be set.

X1 , *Y1* Required **Single**. The position (in points) of the new node relative to the upper-left corner of the document.

Example

This example moves node two in shape three on myDocument to the right 200 points and down 300 points. Shape three must be a freeform drawing.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    pointsArray = .Item(2).Points
    currXvalue = pointsArray(1, 1)
    currYvalue = pointsArray(1, 2)
    .SetPosition 2, currXvalue + 200, currYvalue + 300
End With
```



[Show All](#)

SetSegmentType Method

Sets the segment type of the segment that follows the node specified by ***Index***. If the node is a control point for a curved segment, this method sets the segment type for that curve. Note that this may affect the total number of nodes by inserting or deleting adjacent nodes.

expression.**SetSegmentType**(***Index***, ***SegmentType***)

expression Required. An expression that returns a **ShapeNodes** object.

Index Required **Long**. The node whose segment type is to be set.

SegmentType Required **MsoSegmentType**. Specifies if the segment is straight or curved.

MsoSegmentType can be one of these MsoSegmentType constants.

msoSegmentCurve

msoSegmentLine

Example

This example changes all straight segments to curved segments in shape three on myDocument. Shape three must be a freeform drawing.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    n = 1
    While n <= .Count
        If .Item(n).SegmentType = msoSegmentLine Then
            .SetSegmentType n, msoSegmentCurve
        End If
        n = n + 1
    Wend
End With
```



[Show All](#)

SetShapesDefaultProperties Method

Applies the formatting for the specified shape to the [default shape](#). Shapes created after this method has been used will have this formatting applied to them by default.

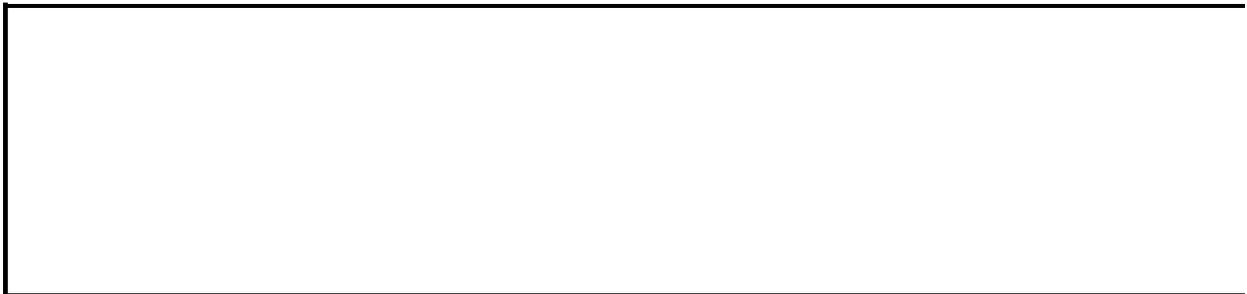
expression.**SetShapesDefaultProperties**

expression Required. An expression that returns a **Shape** object.

Example

This example adds a rectangle to myDocument, formats the rectangle's fill, applies the rectangle's formatting to the default shape, and then adds another smaller rectangle to the document. The second rectangle has the same fill as the first one.

```
Set mydocument = ActivePresentation.Slides(1)
With mydocument.Shapes
  With .AddShape(msoShapeRectangle, 5, 5, 80, 60)
    With .Fill
      .ForeColor.RGB = RGB(0, 0, 255)
      .BackColor.RGB = RGB(0, 204, 255)
      .Patterned msoPatternHorizontalBrick
    End With
  ' Sets formatting for default shapes
  .SetShapesDefaultProperties
End With
' New shape has default formatting
.AddShape msoShapeRectangle, 90, 90, 40, 30
End With
```



[Show All](#)

SetThreeDFormat Method

Sets the preset extrusion format. Each preset extrusion format contains a set of preset values for the various properties of the extrusion.

expression.**SetThreeDFormat**(*PresetThreeDFormat*)

expression Required. An expression that returns a **ThreeDFormat** object.

PresetThreeDFormat Required [MsoPresetThreeDFormat](#). Specifies a preset extrusion format that corresponds to one of the options (numbered from left to right, from top to bottom) displayed when you click the **3-D** button on the **Drawing** toolbar.

MsoPresetThreeDFormat can be one of these MsoPresetThreeDFormat constants.

msoPresetThreeDFormatMixed Specifying this constant causes an error.

msoThreeD1

msoThreeD2

msoThreeD3

msoThreeD4

msoThreeD5

msoThreeD6

msoThreeD7

msoThreeD8

msoThreeD9

msoThreeD10

msoThreeD11

msoThreeD12

msoThreeD13

msoThreeD14

msoThreeD15

msoThreeD16

msoThreeD17

msoThreeD18

msoThreeD19

msoThreeD20

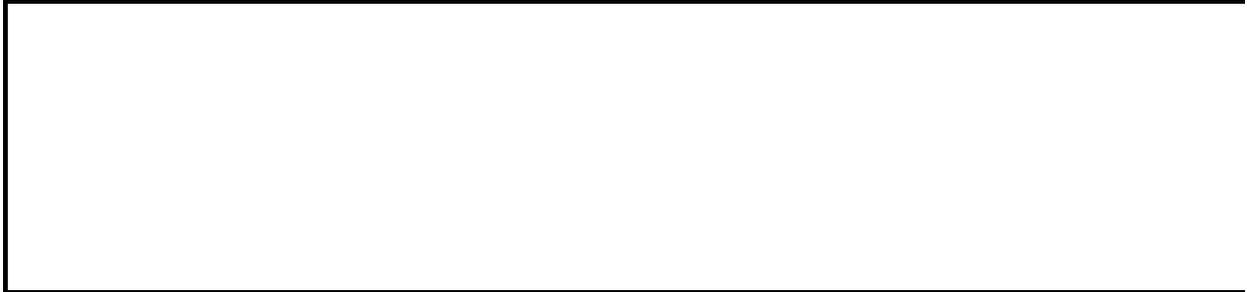
Remarks

This method sets the [PresetThreeDFormat](#) property to the format specified by the *PresetThreeDFormat* argument.

Example

This example adds an oval to myDocument and sets its extrusion format to 3D Style 12.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeOval, 30, 30, 50, 25).ThreeD
    .Visible = True
    .SetThreeDFormat msoThreeD12
End With
```



SmallScroll Method

Scrolls through the specified document window by lines and columns.

expression.**SmallScroll**(*Down*, *Up*, *ToRight*, *ToLeft*)

expression Required. An expression that returns a **DocumentWindow** object.

Down Optional **Long**. Specifies the number of lines to scroll down.

Up Optional **Long**. Specifies the number of lines to scroll up.

ToRight Optional **Long**. Specifies the number of columns to scroll right.

ToLeft Optional **Long**. Specifies the number of columns to scroll left.

Remarks

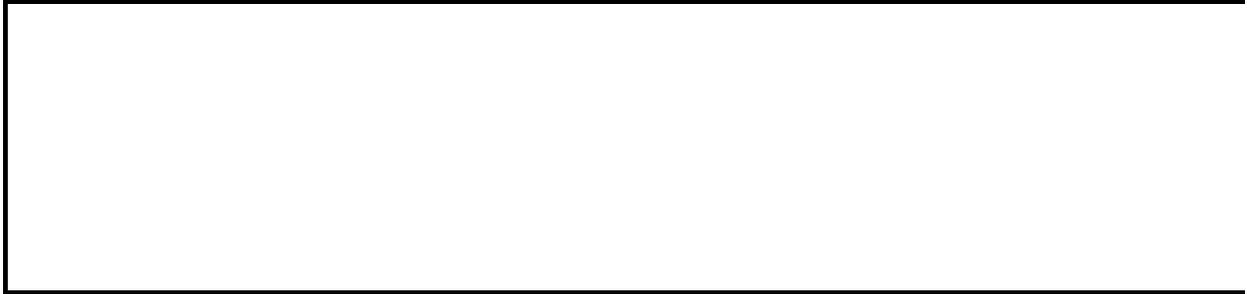
If no arguments are specified, this method scrolls down one line. If ***Down*** and ***Up*** are both specified, their effects are combined. For example, if ***Down*** is 2 and ***Up*** is 4, this method scrolls up two lines. Similarly, if ***Right*** and ***Left*** are both specified, their effects are combined.

Any of the arguments can be a negative number.

Example

This example scrolls down three lines in the active window.

```
Application.ActiveWindow.SmallScroll Down:=3
```



Solid Method

Sets the specified fill to a uniform color. Use this method to convert a gradient, textured, patterned, or background fill back to a solid fill.

expression.**Solid**

expression Required. An expression that returns a **FillFormat** object.

Example

This example converts all fills on myDocument to uniform red fills.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  With s.Fill
    .Solid
    .ForeColor.RGB = RGB(255, 0, 0)
  End With
Next
```



Split Method

Splits a single table cell into multiple cells.

expression.**Split** *NumRows*, *NumColumns*

expression Required. An expression that returns a **Cell** object.

NumRows Required **Long**. Number of rows that the cell is being split into.

NumColumns Required **Long**. Number of columns that the cell is being split into.

Example

This example splits the first cell in the referenced table into two cells, one directly above the other.

```
ActivePresentation.Slides(2).Shapes(5).Table.Cell(1, 1).Split 2, 1
```



SwapNode Method

Swaps the source diagram node with the target diagram node. Any child diagram nodes are moved along with their corresponding root nodes unless specified otherwise.

expression.**SwapNode**(*TargetNode*, *SwapChildren*)

expression Required. An expression that returns one of the objects in the Applies To list.

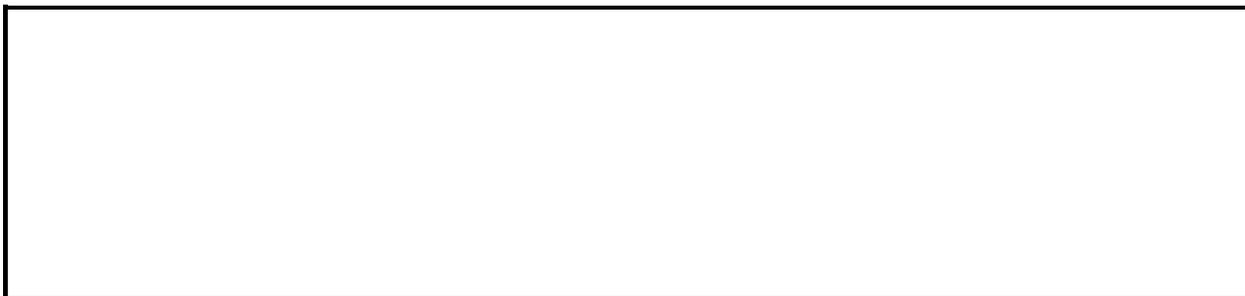
TargetNode Required **DiagramNode** object. The target diagram node.

SwapChildren Optional **Boolean**. **True** (default) if all child diagram nodes are moved along with their corresponding target or source diagram nodes. **False** to swap just the target and source diagram nodes, inheriting the other's child diagram nodes.

Example

The following example swaps the first and third nodes in a newly-created diagram.

```
Sub SwapTwoNodes()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds radial diagram and first node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramRadial, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional nodes  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Swaps the first and the third nodes  
    dgnNode.Children.Item(1).SwapNode _  
        TargetNode:=dgnNode.Children.Item(3)  
  
End Sub
```



ToggleVerticalText Method

Switches the text flow in the specified WordArt from horizontal to vertical, or vice versa.

expression.**ToggleVerticalText**

expression Required. An expression that returns a **TextEffectFormat** object.

Remarks

Using the **ToggleVerticalText** method swaps the values of the **Width** and **Height** properties of the **Shape** object that represents the WordArt and leaves the **Left** and **Top** properties unchanged.

The **Flip** method and **Rotation** property of the **Shape** object and the **RotatedChars** property and **ToggleVerticalText** method of the **TextEffectFormat** object all affect the character orientation and the direction of text flow in a **Shape** object that represents WordArt. You may have to experiment to find out how to combine the effects of these properties and methods to get the result you want.

Example

This example adds WordArt that contains the text "Test" to myDocument, and switches from horizontal text flow (the default for the specified WordArt style, **msoTextEffect1**) to vertical text flow.

```
Set myDocument = ActivePresentation.Slides(1)
Set newWordArt = myDocument.Shapes.AddTextEffect _
    (PresetTextEffect:=msoTextEffect1, Text:="Test", _
    FontName:="Arial Black", FontSize:=36, _
    FontBold:=False, FontItalic:=False, Left:=100, Top:=100)
newWordArt.TextEffect.ToggleVerticalText
```



TransferChildren Method

Moves the child nodes of one diagram node to another diagram node.

expression.**TransferChildren**(*ReceivingNode*)

expression Required. An expression that returns one of the objects in the Applies To list.

ReceivingNode Required **DiagramNode** object. The target (receiving) diagram node.

Example

The following example transfers the child nodes from the first node to the third node of a newly-created diagram.

```
Sub TransferChildNodes()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds org chart and root node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramOrgChart, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three child nodes to root node  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Adds three child nodes to first child node  
    For intNodes = 1 To 3  
        dgnNode.Children.Item(1).Children.AddNode  
    Next intNodes  
  
    ' Transfers children of the first node to the third node  
    dgnNode.Children.Item(1).TransferChildren _  
        ReceivingNode:=dgnNode.Children.Item(3)  
  
End Sub
```



TrimText Method

Returns a **TextRange** object that represents the specified text minus any trailing spaces.

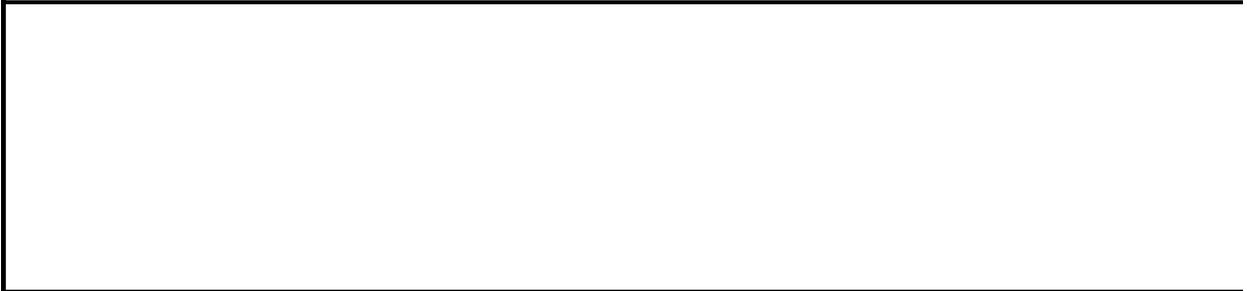
expression.**TrimText**

expression Required. An expression that returns a **TextRange** object.

Example

This example inserts the string " Text to trim " at the beginning of the text in shape two on slide one in the active presentation and then displays message boxes showing the string before and after it's trimmed.

```
With Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange  
    With .InsertBefore(" Text to trim ")  
        MsgBox "Untrimmed: " & """" & .Text & """"  
        MsgBox "Trimmed: " & """" & .TrimText.Text & """"  
    End With  
End With
```



[Show All](#)

TwoColorGradient Method

Sets the specified fill to a two-color gradient.

expression.TwoColorGradient(*Style*, *Variant*)

expression Required. An expression that returns one of the objects in the Applies To list.

Style Required [MsoGradientStyle](#). The gradient style.

MsoGradientStyle can be one of these MsoGradientStyle constants.

msoGradientDiagonalDown

msoGradientDiagonalUp

msoGradientFromCenter

msoGradientFromCorner

msoGradientFromTitle

msoGradientHorizontal

msoGradientMixed

msoGradientVertical

Variant Required **Long**. The gradient variant. Can be a value from 1 to 4, corresponding to the four variants on the **Gradient** tab in the **Fill Effects** dialog box. If *Style* is **msoGradientFromTitle** or **msoGradientFromCenter**, this argument can be either 1 or 2.

Example

This example adds a rectangle with a two-color gradient fill to myDocument, and sets the background and foreground color for the fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(Type:=msoShapeRectangle, Left:=0, _
    Top:=0, Width:=40, Height:=80).Fill
    .ForeColor.RGB = RGB(Red:=128, Green:=0, Blue:=0)
    .BackColor.RGB = RGB(Red:=0, Green:=170, Blue:=170)
    .TwoColorGradient Style:=msoGradientHorizontal, Variant:=1
End With
```



Ungroup Method

Ungroups any grouped shapes in the specified shape or range of shapes. Disassembles pictures and OLE objects within the specified shape or range of shapes. Returns the ungrouped shapes as a single [ShapeRange](#) object.

expression.**Ungroup**

expression Required. An expression that returns a **ShapeRange** object.

Remarks

Because a group of shapes is treated as a single object, grouping and ungrouping shapes changes the number of items in the **Shapes** collection and changes the index numbers of items that come after the affected items in the collection.

Example

This example ungroups any grouped shapes and disassembles any pictures or OLE objects on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    s.Ungroup
Next
```

This example ungroups any grouped shapes on myDocument without disassembling pictures or OLE objects on the slide.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.Type = msoGroup Then s.Ungroup
Next
```



Unselect Method

Cancels the current selection.

expression.**Unselect**

expression Required. An expression that returns a **Selection** object.

Example

This example cancels the current selection in window one.

```
Windows(1).Selection.Unselect
```



Update Method

Updates the specified linked OLE object. To update all the links in a presentation at once, use the [UpdateLinks](#) method.

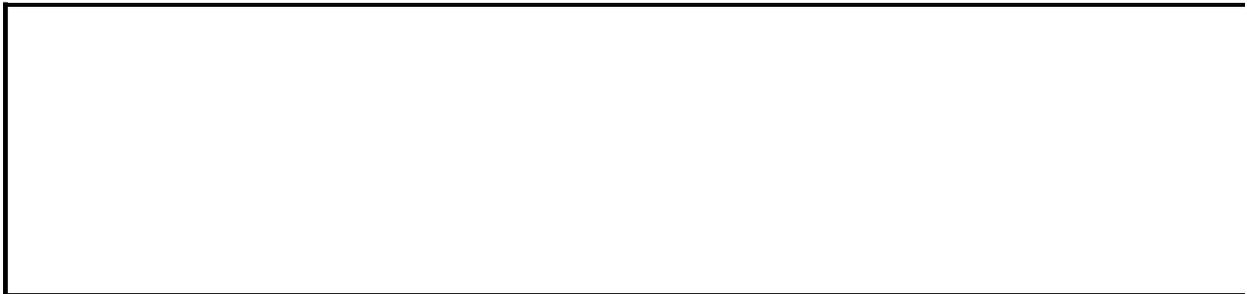
expression.**Update**

expression Required. An expression that returns a **LinkFormat** object.

Example

This example updates all linked OLE objects in the active presentation.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      sh.LinkFormat.Update
    End If
  Next
Next
```



UpdateLinks Method

Updates linked OLE objects in the specified presentation.

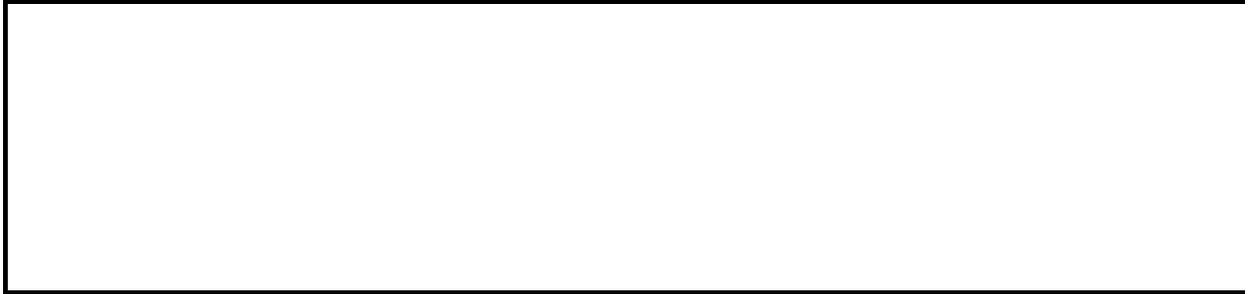
expression.**UpdateLinks**

expression Required. An expression that returns a **Presentation** object.

Example

This example updates all OLE links in the active presentation.

`ActivePresentation.UpdateLinks`



UseDefaultFolderSuffix Method

Sets the folder suffix for the specified presentation to the default suffix for the language support you have selected or installed.

expression.**UseDefaultFolderSuffix**

expression An expression that returns a **WebOptions** object.

Remarks

Microsoft PowerPoint uses the folder suffix when you save or publish a complete or partial presentation as a Web page, use long file names, and choose to save supporting files in a separate folder (that is, if the [UseLongFileNames](#) and [OrganizeInFolder](#) properties are set to **True**).

The suffix appears in the folder name after the presentation name. For example, if the presentation is called "Pres1" and the language is English, the folder name is Pres1_files. The folder suffixes for each language are listed in the [FolderSuffix](#) property topic.

Example

This example sets the folder suffix for the active presentation to the default suffix.

```
ActivePresentation.WebOptions.UseDefaultFolderSuffix
```



UserPicture Method

Fills the specified shape with one large image. If you want to fill the shape with small tiles of an image, use the [UserTextured](#) method.

expression.**UserPicture**(*PictureFile*)

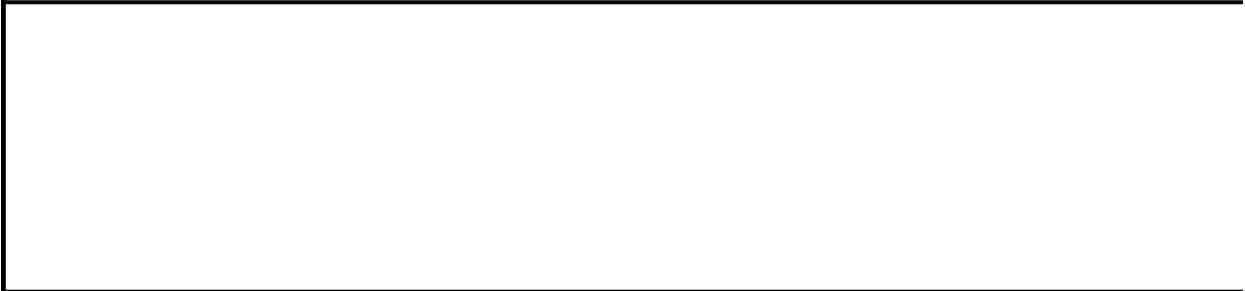
expression Required. An expression that returns a **FillFormat** object.

PictureFile Required **String**. The name of the picture file.

Example

This example adds two rectangles to myDocument. The rectangle on the left is filled with one large image of the picture in Tiles.bmp; the rectangle on the right is filled with many small tiles of the picture in Tiles.bmp

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeRectangle, 0, 0, 200, 100).Fill _
        .UserPicture "c:\windows\tiles.bmp"
    .AddShape(msoShapeRectangle, 300, 0, 200, 100).Fill _
        .UserTextured "c:\windows\tiles.bmp"
End With
```



UserTextured Method

Fills the specified shape with small tiles of an image. If you want to fill the shape with one large image, use the **UserPicture** method.

expression.**UserTextured**(*TextureFile*)

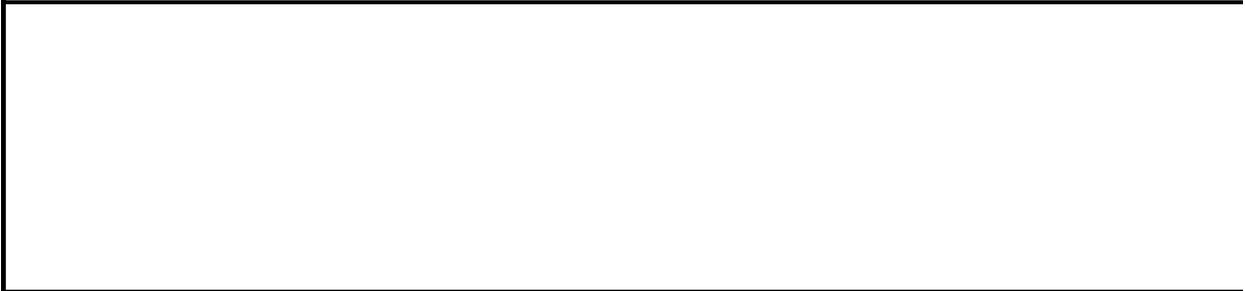
expression Required. An expression that returns a **FillFormat** object.

TextureFile Required **String**. The name of the picture file.

Example

This example adds two rectangles to myDocument. The rectangle on the left is filled with one large image of the picture in Tiles.bmp; the rectangle on the right is filled with many small tiles of the picture in Tiles.bmp

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeRectangle, 0, 0, 200, 100).Fill _
        .UserPicture "c:\windows\tiles.bmp"
    .AddShape(msoShapeRectangle, 300, 0, 200, 100).Fill _
        .UserTextured "c:\windows\tiles.bmp"
End With
```



Value Method

Returns the value of the specified tag as a **String**.

expression.**Value**(*Index*)

expression Required. An expression that returns a **Tags** collection.

Index Required **Long**. The tag number.

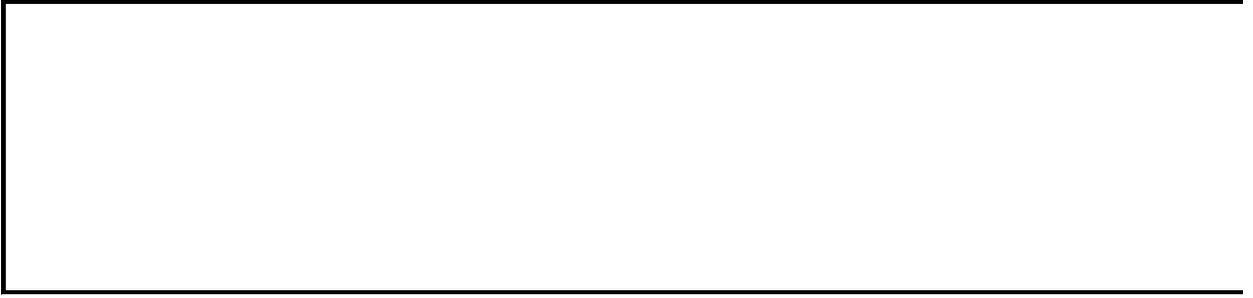
Example

This example displays the name and value for each tag associated with slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Tags
  For i = 1 To .Count
    MsgBox "Tag #" & i & ": Name = " & .Name(i)
    MsgBox "Tag #" & i & ": Value = " & .Value(i)
  Next
End With
```

This example searches through the tags for each slide in the active presentation. If there's a tag named "PRIORITY," a message box displays the tag value. If there isn't a tag named "PRIORITY," the example adds this tag with the value "Unknown."

```
For Each s In Application.ActivePresentation.Slides
  With s.Tags
    found = False
    For i = 1 To .Count
      If .Name(i) = "PRIORITY" Then
        found = True
        slNum = .Parent.SlideIndex
        MsgBox "Slide " & slNum & " priority: " & .Value(i)
      End If
    Next
    If Not found Then
      slNum = .Parent.SlideIndex
      .Add "Name", "New Figures"
      .Add "Priority", "Unknown"
      MsgBox "Slide " & slNum & _
        " priority tag added: Unknown"
    End If
  End With
Next
```



WebPagePreview Method

Shows a preview of the presentation in the active Web browser.

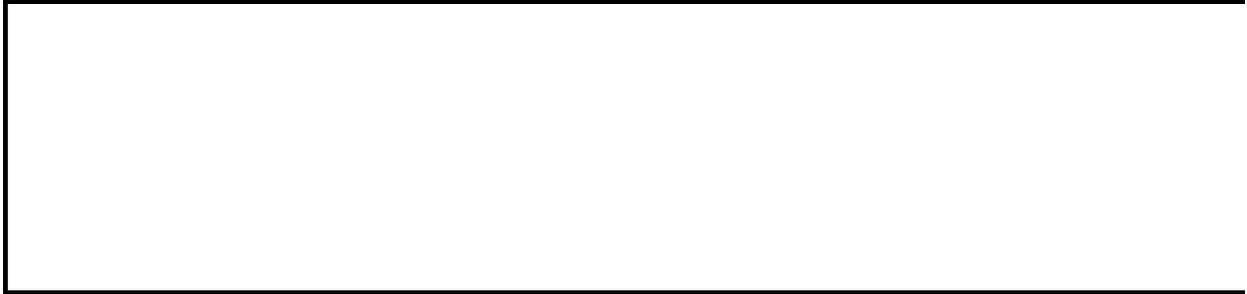
expression.**WebPagePreview**

expression Required. An expression that returns a **Presentation** object.

Example

This example previews presentation two as a Web page.

Presentations(2).WebPagePreview



Words Method

Returns a **TextRange** object that represents the specified subset of text words. For information about counting or looping through the words in a text range, see the [TextRange](#) object.

expression.**Words**(*Start*, *Length*)

expression Required. An expression that returns a **TextRange** object.

Start Optional **Long**. The first word in the returned range.

Length Optional **Long**. The number of words to be returned.

Remarks

If both ***Start*** and ***Length*** are omitted, the returned range starts with the first word and ends with the last paragraph in the specified range.

If ***Start*** is specified but ***Length*** is omitted, the returned range contains one word.

If ***Length*** is specified but ***Start*** is omitted, the returned range starts with the first word in the specified range.

If ***Start*** is greater than the number of words in the specified text, the returned range starts with the last word in the specified range.

If ***Length*** is greater than the number of words from the specified starting word to the end of the text, the returned range contains all those words.

Example

This example formats as bold the second, third, and fourth words in the first paragraph in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange.Paragraphs(1).Words(2, 3).Font _  
    .Bold = True
```



[Show All](#)

ZOrder Method

Moves the specified shape in front of or behind other shapes in the collection (that is, changes the shape's position in the z-order).

expression.ZOrder(ZOrderCmd)

expression Required. An expression that returns a **Shape** object.

ZOrderCmd Required [MsoZOrderCmd](#). Specifies where to move the specified shape relative to the other shapes.

MsoZOrderCmd can be one of these MsoZOrderCmd constants.

msoBringForward

msoBringInFrontOfText For use in Microsoft Word only.

msoBringToFront

msoSendBackward

msoSendBehindText For use in Microsoft Word only.

msoSendToBack

Remarks

Use the [ZOrderPosition](#) property to determine a shape's current position in the z-order.

Example

This example adds an oval to myDocument and then places the oval second from the back in the z-order if there is at least one other shape on the slide.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeOval, 100, 100, 100, 300)
    While .ZOrderPosition > 2
        .ZOrder msoSendBackward
    Wend
End With
```



Accelerate Property

Returns or sets a **Single** that represents the percent of the duration over which a timing acceleration should take place. For example, a value of 0.9 means that an acceleration should start slower than the default speed for 90% of the total animation time, with the last 10% of the animation at the default speed.

Read/write.

expression.**Accelerate**

expression Required. An expression that returns one of the objects in the Applies To list.

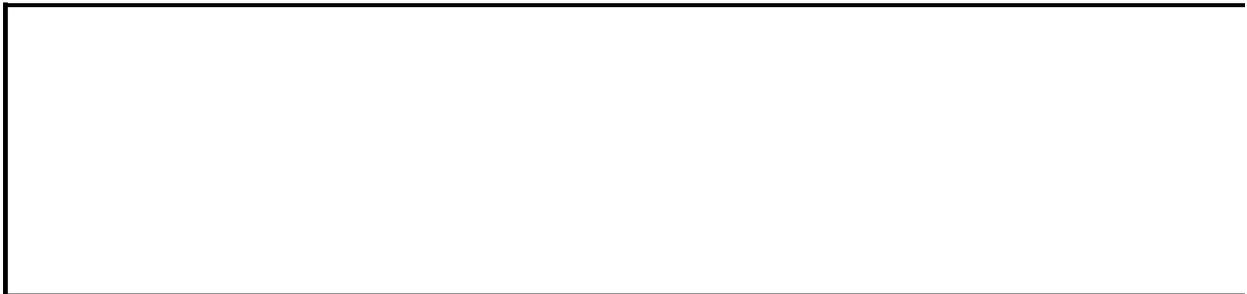
Remarks

To slow down an animation at the end, use the [Decelerate](#) property.

Example

This example adds a shape and adds an animation, starting out slow and matching the default speed after 30% of the animation sequence.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds rectangle and specifies effect to use for rectangle  
    Set shpRectangle = ActivePresentation.Slides(1) _  
        .Shapes.AddShape(Type:=msoShapeRectangle, _  
            Left:=100, Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1) _  
        .TimeLine.MainSequence.AddEffect(Shape:=shpRectangle, _  
            effectId:=msoAnimEffectPathDiamond)  
  
    'Specifies the acceleration for the effect  
    With effDiamond.Timing  
        .Accelerate = 0.3  
    End With  
  
End Sub
```



[Show All](#)

AcceleratorsEnabled Property

Determines whether shortcut key are enabled during a slide show. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants. (If shortcut keys are disabled during a slide show, you can neither use keys to navigate in the slide show nor press F1 to get a list of shortcut keys. You can still use the ESC key to exit the slide show.)

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Shortcut keys are enabled during a slide show.

expression.**AcceleratorsEnabled**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example runs a slide show of the active presentation with shortcut keys disabled.

```
ActivePresentation.SlideShowSettings.Run _  
    .View.AcceleratorsEnabled = False
```



[Show All](#)

Accent Property

Determines whether a vertical accent bar separates the callout text from the callout line. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue A vertical accent bar separates the callout text from the callout line.

expression.**Accent**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds to myDocument an oval and a callout that points to the oval. The callout text won't have a border, but it will have a vertical accent bar that separates the text from the callout line.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape msoShapeOval, 180, 200, 280, 130
    With .AddCallout(msoCalloutTwo, 420, 170, 170, 40)
        .TextFrame.TextRange.Text = "My oval"
        With .Callout
            .Accent = msoTrue
            .Border = msoFalse
        End With
    End With
End With
```



[Show All](#)

Accumulate Property

Sets or returns an [MsoAnimAccumulate](#) constant that represents whether animation behaviors accumulate. Read/write.

MsoAnimAccumulate can be one of these MsoAnimAccumulate constants.

msoAnimAccumulateAlways Repetitions start with the current value.

msoAnimAccumulateNone Default.

expression.**Accumulate**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use this property in conjunction with the [Additive](#) property to combine animation effects.

Example

The following example allows a specified animation behavior to accumulate with other animation behaviors.

```
Sub SetAccumulate()  
    Dim animBehavior As AnimationBehavior  
  
    Set animBehavior = ActiveWindow.Selection.SlideRange(1).TimeLine  
        MainSequence(1).Behaviors(1)  
  
    animBehavior.Accumulate = msoAnimAccumulateAlways  
  
End Sub
```



[Show All](#)

Action Property

Returns or sets the type of action that will occur when the specified shape is clicked or the mouse pointer is positioned over the shape during a slide show. Can be one of the following [PpActionType](#) constants. Read/write **Long**.

ppActionEndShow
ppActionFirstSlide
ppActionHyperlink
ppActionLastSlide
ppActionLastSlideViewed
ppActionMixed
ppActionNamedSlideShow
ppActionNextSlide
ppActionNone
ppActionOLEVerb
ppActionPlay
ppActionPreviousSlide
ppActionRunMacro
ppActionRunProgram

You can use the **Action** property in conjunction with other properties of the **ActionSetting** object, as shown in the following table.

If you set the Action property to this value	Use this property	To do this
ppActionHyperlink	Hyperlink	Set properties for the hyperlink that will be followed in response to a mouse action on the shape during a slide show. Return or set the name of the program to run in response to a

ppActionRunProgram	<u>Run</u>	mouse action on the shape during a slide show.
ppActionRunMacro	<u>Run</u>	Return or set the name of the macro to run in response to a mouse action on the shape during a slide show.
ppActionOLEVerb	<u>ActionVerb</u>	Set the OLE verb that will be invoked in response to a mouse action on the shape during a slide show.
ppActionNamedSlideShow	<u>SlideShowName</u>	Set the name of the custom slide show that will run in response to a mouse action on the shape during a slide show.

Example

This example sets shape three (an OLE object) on slide one in the active presentation to be played when the mouse passes over it during a slide show.

```
With ActivePresentation.Slides(1) _  
    .Shapes(3).ActionSettings(ppMouseOver)  
        .ActionVerb = "Play"  
        .Action = ppActionOLEVerb  
End With
```



ActionSettings Property

Returns an [ActionSettings](#) object that contains information about what action occurs when the user clicks or moves the mouse over the specified shape or text range during a slide show. Read-only.

Example

The following example sets the actions for clicking and moving the mouse over shape one on slide two in the active presentation.

```
Set myShape = ActivePresentation.Slides(2).Shapes(1)
myShape.ActionSettings(ppMouseClick).Action = ppActionLastSlide
myShape.ActionSettings(ppMouseOver).SoundEffect.Name = "applause"
```



[Show All](#)

ActionVerb Property

 [ActionVerb property as it applies to the **PlaySettings** object.](#)

Returns or sets a string that contains the OLE verb that will be run when the specified OLE object is animated during a slide show. The default verb specifies the action that the OLE object runs— such as playing a wave file or displaying data so that the user can modify it— after the previous animation or slide transition. Read/write **String**.

 [ActionVerb property as it applies to the **ActionSetting** object.](#)

Returns or sets a string that contains the OLE verb that will be run when the user clicks the specified shape or passes the mouse pointer over it during a slide show. The [Action](#) property must be set to **ppActionOLEVerb** first for this property to affect the slide show action. Read/write **String**.

Example

 [As it applies to the **PlaySettings** object.](#)

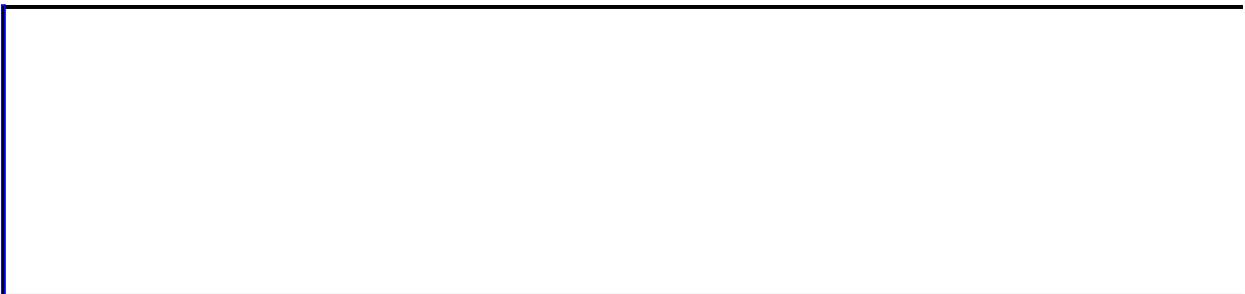
This example specifies that shape three on slide one in the active presentation will automatically open for editing when it's animated. Shape three must be an OLE object that contains a sound or movie object and that supports the "Edit" verb.

```
Set OLEobj = ActivePresentation.Slides(1).Shapes(3)
With OLEobj.AnimationSettings.PlaySettings
    .PlayOnEntry = True
    .ActionVerb = "Edit"
End With
```

 [As it applies to the **ActionSetting** object.](#)

This example sets shape three on slide one to be played whenever the mouse pointer passes over it during a slide show. Shape three must represent an OLE object that supports the "Play" verb.

```
With ActivePresentation.Slides(1).Shapes(3) _
    .ActionSettings(ppMouseOver)
    .ActionVerb = "Play"
    .Action = ppActionOLEVerb
End With
```



[Show All](#)

Active Property

Returns whether the specified pane or window is active. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified pane or window is active.

expression.**Active**

expression Required. An expression that returns of the objects in the Applies To list.

Example

This example checks to see if the presentation file "test.ppt" is in the active window. If not, it saves the name of the presentation that is currently active in the variable oldWin and activates the "test.ppt" presentation.

```
With Application.Presentations("test.ppt").Windows(1)
  If Not .Active Then
    Set oldWin = Application.ActiveWindow
    .Activate
  End If
End With
```



ActivePane Property

Returns a [Pane](#) object that represents the active pane in the document window.
Read-only.

Example

If the active pane is the slide pane, this example makes the notes pane the active pane. The notes pane is the third member of the **Panes** collection.

```
With ActiveWindow
    If .ActivePane.ViewType = ppViewSlide Then
        .Panes(3).Activate
    End If
End With
```



ActivePresentation Property

Returns a [Presentation](#) object that represents the presentation open in the active window. Read-only.

Note that if an embedded presentation is in-place active, the **ActivePresentation** property returns the embedded presentation.

Example

This example saves the loaded presentation to the application folder in a file named "TestFile."

```
MyPath = Application.Path & "\TestFile"  
Application.ActivePresentation.SaveAs MyPath
```



ActivePrinter Property

Returns the name of the active printer. Read-only **String**.

Example

This example displays the name of the active printer.

```
MsgBox "The name of the active printer is " _  
    & Application.ActivePrinter
```



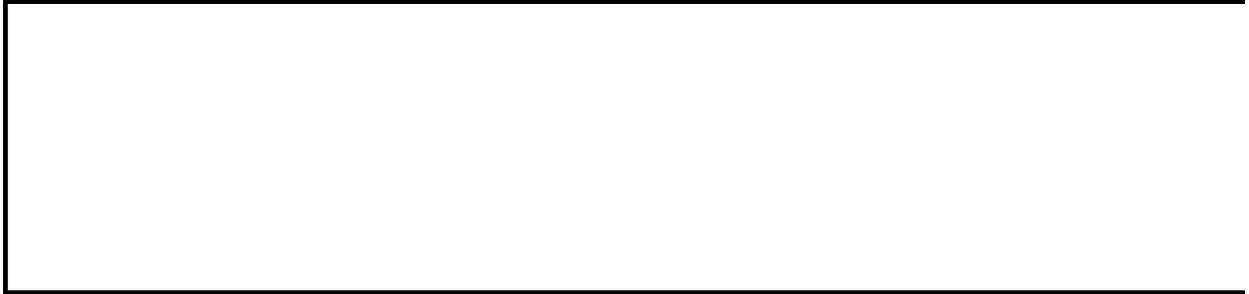
ActiveWindow Property

Returns a [DocumentWindow](#) object that represents the active document window. Read-only.

Example

This example minimizes the active window.

```
Application.ActiveWindow.WindowState = ppWindowMinimized
```



[Show All](#)

AddIns Property

Returns the program-specific [AddIns](#) collection that represents all the add-ins listed in the **Add-Ins** dialog box (**Tools** menu). Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Remarks

Microsoft PowerPoint-specific add-ins are identified by a .ppa file name extension. [Component Object Model \(COM\) add-ins](#) can be used universally across Microsoft programming products and have a .dll or .exe file name extension.

Example

This example adds the add-in named "Myaddin.ppa" to the list in the **Add-Ins** dialog box and loads the add-in automatically.

```
Set myAddIn = Application.AddIns.Add(FileName:="c:\myaddin.ppa")  
myAddIn.Loaded = True  
MsgBox myAddIn.Name & " has been added to the list"
```



[Show All](#)

Additive Property

Sets or returns an [MsoAnimAdditive](#) constant that represents whether the current animation behavior is combined with other running animations. Read/write.

MsoAnimAdditive can be one of these **MsoAnimAdditive** constants.

msoAnimAdditiveAddBase Does not combine current animation with other animations. *Default.*

msoAnimAdditiveAddSum Combines the current animation with other running animations.

expression.**Additive**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Combining animation behaviors particularly useful for rotation effects. For example, if the current animation changes rotation and another animation is also changing rotation, then if this property is set to **msoAnimAdditiveAddSum**, Microsoft PowerPoint will add together the rotations from both the animations.

Example

The following example allows the current animation behavior to be added to another animation behavior.

```
Sub SetAdditive()  
    Dim animBehavior As AnimationBehavior  
  
    Set animBehavior = ActiveWindow.Selection.SlideRange(1) _  
        .TimeLine.MainSequence(1).Behaviors(1)  
  
    animBehavior.Additive = msoAnimAdditiveAddSum  
  
End Sub
```



Address Property

Returns or sets the Internet address (URL) to the target document. Read/write **String**.

Example

This example scans all shapes on the first slide for the URL to the Microsoft Web site.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Hyperlinks
    If s.Address = "http://www.microsoft.com/" Then
        MsgBox "You have a link to the Microsoft Home Page"
    End If
Next
```



Adjustments Property

Returns an [Adjustments](#) object that contains adjustment values for all the adjustments in the specified shape. Applies to any **Shape** or **ShapeRange** object that represents an AutoShape, WordArt, or a connector. Read-only.

Example

This example sets to 0.25 the value of adjustment one for shape three on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(3).Adjustments(1) = 0.25
```



[Show All](#)

AdvanceMode Property

[AdvanceMode property as it applies to the **AnimationSettings** object.](#)

Returns or sets a value that indicates whether the specified shape animation advances only when clicked or automatically after a specified amount of time. Read/write **PpAdvanceMode**. If your shape doesn't become animated, make sure that the **TextLevelEffect** property is set to a value other than **ppAnimateLevelNone** and that the **Animate** property is set to **True**.

PpAdvanceMode can be one of these PpAdvanceMode constants.

ppAdvanceModeMixed

ppAdvanceOnClick

ppAdvanceOnTime

expression.**AdvanceMode**

expression Required. An expression that returns one of the above objects.

[AdvanceMode property as it applies to the **SlideShowSettings** object.](#)

Returns or sets a value that indicates how the slide show advances. Read/write **PpSlideShowAdvanceMode**.

PpSlideShowAdvanceMode can be one of these PpSlideShowAdvanceMode constants.

ppSlideShowManualAdvance

ppSlideShowRehearseNewTimings

ppSlideShowUseSlideTimings

expression.**AdvanceMode**

expression Required. An expression that returns one of the above objects.

[AdvanceMode property as it applies to the **SlideShowView** object.](#)

Returns a value that indicates how the slide show in the specified view advances.
Read-only [PpSlideShowAdvanceMode](#).

PpSlideShowAdvanceMode can be one of these PpSlideShowAdvanceMode constants.

ppSlideShowManualAdvance

ppSlideShowRehearseNewTimings

ppSlideShowUseSlideTimings

expression.**AdvanceMode**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **AnimationSettings** object.](#)

This example sets shape two on slide one in the active presentation to become animated automatically after five seconds.

```
With ActivePresentation.Slides(1).Shapes(2).AnimationSettings
    .AdvanceMode = ppAdvanceOnTime
    .AdvanceTime = 5
    .TextLevelEffect = ppAnimateByAllLevels
    .Animate = True
End With
```



[Show All](#)

AdvanceOnClick Property

Determines whether the specified slide advances when it's clicked during a slide show. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide advances when it's clicked during a slide show.

expression.**AdvanceOnClick**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

To set the slide to advance automatically after a certain amount of time elapses, set the [AdvanceOnTime](#) property to **True** and set the [AdvanceTime](#) property to the amount of time you want the slide to be shown. If you set both the **AdvanceOnClick** and the **AdvanceOnTime** properties to **True**, the slide will advance either when it's clicked or when the specified amount of time has elapsed— whichever comes first.

Example

This example sets slide one in the active presentation to advance after five seconds have passed or when the mouse is clicked— whichever occurs first.

```
With ActivePresentation.Slides(1).SlideShowTransition
    .AdvanceOnClick = msoTrue
    .AdvanceOnTime = msoTrue
    .AdvanceTime = 5
End With
```



[Show All](#)

AdvanceOnTime Property

Determines whether the specified slide advances automatically after a specified amount of time has elapsed. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide advances automatically after a specified amount of time has elapsed.

expression.**AdvanceOnTime**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [AdvanceTime](#) property to specify the number of seconds after which the slide will automatically advance. Set the [AdvanceMode](#) property of the **SlideShowSettings** object to **ppSlideShowUseSlideTimings** to put the slide interval settings into effect for the entire slide show.

Example

This example sets slide one in the active presentation to advance after five seconds have passed or when the mouse is clicked— whichever occurs first.

```
With ActivePresentation.Slides(1).SlideShowTransition
    .AdvanceOnClick = msoTrue
    .AdvanceOnTime = msoTrue
    .AdvanceTime = 5
End With
```



[Show All](#)

AdvanceTime Property

[As it applies to the **AnimationSettings** object.](#)

Returns or sets the amount of time, in seconds, after which the specified shape will become animated. Read/write **Single**.

[As it applies to the **SlideShowTransition** object.](#)

Returns or sets the amount of time, in seconds, after which the specified slide transition will occur. Read/write **Single**.

Remarks

The specified slide animation won't start automatically after the amount of time you've specified unless the [AdvanceMode](#) property of the animation is set to **ppAdvanceOnTime**. The specified slide transition won't advance automatically unless the **AdvanceMode** property of the slide show settings is set to **ppSlideShowUseSlideTimings**.

Example

 [As it applies to the **AnimationSettings** object.](#)

This example sets shape two on slide one in the active presentation to become animated automatically after five seconds.

```
With ActivePresentation.Slides(1).Shapes(2).AnimationSettings
    .AdvanceMode = ppAdvanceOnTime
    .AdvanceTime = 5
    .TextLevelEffect = ppAnimateByAllLevels
    .Animate = True
End With
```



[Show All](#)

AfterEffect Property

 [AfterEffect property as it applies to the **EffectInformation** object.](#)

Returns an [MsoAnimAfterEffect](#) constant that indicates whether an after effect is dimmed, hidden, or unchanged after it runs. Read-only.

MsoAnimAfterEffect can be one of these MsoAnimAfterEffect constants.

msoAnimAfterEffectDim

msoAnimAfterEffectHide

msoAnimAfterEffectHideOnNextClick

msoAnimAfterEffectMixed

msoAnimAfterEffectNone

expression.**AfterEffect**

expression Required. An expression that an [EffectInformation](#) object.

 [AfterEffect property as it applies to the **AnimationSettings** object.](#)

Returns or sets a [PpAfterEffect](#) constant that indicates whether the specified shape appears dimmed, hidden, or unchanged after it's been built. Read/write.

PpAfterEffect can be one of these PpAfterEffect constants.

ppAfterEffectDim

ppAfterEffectHide

ppAfterEffectHideOnClick

ppAfterEffectMixed

ppAfterEffectNothing

expression.**AfterEffect**

expression Required. An expression that returns an [AnimationSettings](#) object

.

Remarks

You won't see the after effect you set for a shape unless the shape gets animated and at least one other shape on the slide gets animated after it. For a shape to be animated, the [TextLevelEffect](#) property of the **AnimationSettings** object for the shape must be set to something other than **ppAnimateLevelNone**, or the [EntryEffect](#) property must be set to a constant other than **ppEffectNone**. In addition, the [Animate](#) property must be set to **True**. To change the build order of the shapes on a slide, use the [AnimationOrder](#) property.

Example

This example specifies that the title on slide one in the active presentation is to appear dimmed after the title is built. If the title is the last or only shape to be built on slide one, the text won't be dimmed.

```
With ActivePresentation.Slides(1).Shapes.Title.AnimationSettings
    .Animate = True
    .TextLevelEffect = ppAnimateByAllLevels
    .AfterEffect = ppAfterEffectDim
End With
```



[Show All](#)

Alignment Property

 [Alignment property as it applies to the **TextEffectFormat** object.](#)

Returns or sets the alignment for the specified WordArt. Read/write **MsoTextEffectAlignment**.

MsoTextEffectAlignment can be one of these MsoTextEffectAlignment constants.

msoTextEffectAlignmentCentered

msoTextEffectAlignmentLeft

msoTextEffectAlignmentMixed

msoTextEffectAlignmentRight

msoTextEffectAlignmentStretchJustify

msoTextEffectAlignmentWordJustify

msoTextEffectAlignmentLetterJustify

expression.**Alignment**

expression Required. An expression that returns a **TextEffectFormat** object.

 [Alignment property as it applies to the **ParagraphFormat** object.](#)

Returns or sets the alignment for each paragraph in the specified paragraph format. Read/write **PpParagraphAlignment**.

PpParagraphAlignment can be one of these PpParagraphAlignment constants.

ppAlignCenter

ppAlignDistribute

ppAlignJustify

ppAlignJustifyLow

ppAlignLeft

ppAlignmentMixed

ppAlignRight

ppAlignThaiDistribute

expression.**Alignment**

expression Required. An expression that returns a **ParagraphFormat** object.

Example

 [As it applies to the **TextEffectFormat** object.](#)

This example adds a WordArt object to slide one in the active presentation and then right aligns the WordArt.

```
Set mySh = Application.ActivePresentation.Slides(1).Shapes
Set myTE = mySh.AddTextEffect(PresetTextEffect:=msoTextEffect1, _
    Text:="Test Text", FontName:="Palatino", FontSize:=54, _
    FontBold:=True, FontItalic:=False, Left:=100, Top:=50)
myTE.TextEffect.Alignment = msoTextEffectAlignmentRight
```

 [As it applies to the **ParagraphFormat** object.](#)

This example left aligns the paragraphs in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2) _
    .TextFrame.TextRange.ParagraphFormat.Alignment = ppAlignLeft
```



[Show All](#)

AllowPNG Property

Determines whether PNG (Portable Network Graphics) is allowed as an output format when you save or publish a complete or partial presentation as a Web page. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. PNG is not allowed as an output format when you save or publish a complete or partial presentation as a Web page.

msoTriStateMixed

msoTriStateToggle

msoTrue PNG is allowed as an image format when you save or publish a complete or partial presentation as a Web page.

expression.**AllowPNG**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If you save images in the PNG format as opposed to any other file format, you might improve the image quality or reduce the size of those image files, and therefore decrease the download time, assuming that the Web browsers you are targeting support the PNG format.

Example

This example enables PNG as an output format for the active presentation.

```
ActivePresentation.WebOptions.AllowPNG = msoTrue
```

Alternatively, PNG can be enabled as the global default for the application for newly created presentations.

```
Application.DefaultWebOptions.AllowPNG = msoTrue
```



AlternativeText Property

Returns or sets the alternative text associated with a shape in a Web presentation.
Read/write **String**.

Example

The following example sets the alternative text for the selected shape in the active window. The selected shape is a picture of a mallard duck.

```
ActiveWindow.Selection.ShapeRange _  
    .AlternativeText = "This is a mallard duck."
```



[Show All](#)

AlwaysSaveInDefaultEncoding Property

Determines whether the default [encoding](#) is used when you save a Web page or plain text document, independent of the file's original encoding when opened. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. The original encoding of the file is used when you save a Web page or plain text document, independent of the file's original encoding when opened.

msoTriStateMixed

msoTriStateToggle

msoTrue The default encoding is used when you save a Web page or plain text document, independent of the file's original encoding when opened.

expression.AlwaysSaveInDefaultEncoding

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The [Encoding](#) property can be used to set the default encoding.

Example

This example sets the encoding to the default encoding. The encoding is used when you save the document as a Web page.

```
Application.DefaultWebOptions.AlwaysSaveInDefaultEncoding = msoTrue
```



Amount Property

Returns or sets a **Single** that represents the number of degrees an animated shape is rotated around the z-axis. A positive value indicates clockwise rotation; a negative value indicates counterclockwise rotation. Read/write.

expression.**Amount**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, and a 90-degree spin animation to the shape.

```
Sub SetAnimEffect()  
  
    Dim effSpin As Effect  
    Dim shpCube As Shape  
  
    Set shpCube = ActivePresentation.Slides(1).Shapes.AddShape _  
        (Type:=msoShapeCube, Left:=100, Top:=100, _  
        Width:=50, Height:=50)  
    Set effSpin = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpCube, _  
        effectId:=msoAnimEffectSpin)  
  
    effSpin.Timing.Duration = 3  
    effSpin.EffectParameters.Amount = -90  
  
End Sub
```



[Show All](#)

Angle Property

Returns or sets the angle of the callout line. If the callout line contains more than one line segment, this property returns or sets the angle of the segment that is farthest from the callout text box. Read/write [MsoCalloutAngleType](#).

MsoCalloutAngleType can be one of these MsoCalloutAngleType constants.

msoCalloutAngle30

msoCalloutAngle45

msoCalloutAngle60

msoCalloutAngle90

msoCalloutAngleAutomatic Callout line maintains a fixed angle as you drag the callout.

msoCalloutAngleMixed

expression.**Angle**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets to 90 degrees the callout angle for a callout named "co1" on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes("co1").Callout.Angle = msoCalloutAngle90
```



[Show All](#)

Animate Property

Determines whether the specified shape is animated during a slide show.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is animated during a slide show.

expression.**Animate**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

For a shape to be animated, the [TextLevelEffect](#) property of the **AnimationSettings** object for the shape must be set to something other than **ppAnimateLevelNone**, and either the **Animate** property must be set to **True**, or the [EntryEffect](#) property must be set to a constant other than **ppEffectNone**.

Example

This example specifies that the title on slide two in the active presentation appear dimmed after the title is built. If the title is the last or only shape to be built on slide two, the text won't be dimmed.

```
With ActivePresentation.Slides(2).Shapes.Title.AnimationSettings
    .TextLevelEffect = ppAnimateByAllLevels
    .AfterEffect = ppAfterEffectDim
    .Animate = msoTrue
End With
```



[Show All](#)

AnimateAction Property

MsoTrue if the color of the specified shape is momentarily inverted when the specified mouse action occurs. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**AnimateAction**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets shape three on slide one in the active presentation to play the sound of applause and to momentarily invert its color when it's clicked during a slide show.

```
With ActivePresentation.Slides(1) _  
    .Shapes(3).ActionSettings(ppMouseClick)  
        .SoundEffect.Name = "applause"  
        .AnimateAction = msoTrue  
End With
```



[Show All](#)

AnimateBackground Property

 [AnimateBackground property as it applies to the **AnimationSettings** object.](#)

If the specified object is an AutoShape, **msoTrue** if the shape is animated separately from the text it contains. If the specified shape is a graph object, **msoTrue** if the background (the axes and gridlines) of the specified graph object is animated. Applies only to AutoShapes with text that can be built in more than one step or to graph objects. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**AnimateBackground**

expression Required. An expression that returns an [AnimationSettings](#) object.

 [AnimateBackground property as it applies to the **EffectInformation** object.](#)

Returns **MsoTrue** if the specified effect is a background animation. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**AnimateBackground**

expression Required. An expression that returns one of the above objects.

Remarks

Use the [TextLevelEffect](#) and [TextUnitEffect](#) properties to control the animation of text attached to the specified shape.

If this property is set to **MsoTrue** and the **TextLevelEffect** property is set to **ppAnimateByAllLevels**, the shape and its text will be animated simultaneously. If this property is set to **MsoTrue** and the **TextLevelEffect** property is set to anything other than **ppAnimateByAllLevels**, the shape will be animated immediately before the text is animated.

You won't see effects of setting this property unless the specified shape is animated. For a shape to be animated, the **TextLevelEffect** property for the shape must be set to something other than **ppAnimateLevelNone**, and either the [Animate](#) property must be set to **MsoTrue**, or the [EntryEffect](#) property must be set to a constant other than **ppEffectNone**.

Example

 [As it applies to the **AnimationSettings** object.](#)

This example creates a rectangle that contains text. The example then specifies that the shape should fly in from the lower right, that the text should be built from first-level paragraphs, and that the shape should be animated separately from the text it contains. In this example, the **EntryEffect** property turns on animation.

```
Sub AnimateTextBox()  
    With ActivePresentation.Slides(1).Shapes.AddShape _  
        (Type:=msoShapeRectangle, Left:=50, Top:=200, _  
         Width:=200, Height:=200)  
        .TextFrame.TextRange = "Reason 1" & Chr(13) & _  
        "Reason 2" & Chr(13) & "Reason 3"  
        With .AnimationSettings  
            .EntryEffect = ppEffectFlyFromBottomRight  
            .TextLevelEffect = ppAnimateByFirstLevel  
            .TextUnitEffect = ppAnimateByParagraph  
            .AnimateBackground = msoTrue  
        End With  
    End With  
End Sub
```

 [As it applies to the **EffectInformation** object.](#)

This example changes the direction of the animation if the background is currently animated.

```
Sub ChangeAnimationDirection()  
    With ActivePresentation.Slides(1).TimeLine.MainSequence(1)  
        If .EffectInformation.AnimateBackground = msoTrue Then  
            .EffectParameters.Direction = msoAnimDirectionTopLeft  
        End If  
    End With  
End Sub
```



[Show All](#)

AnimateTextInReverse Property

Determines whether the specified shape is built in reverse order. Applies only to shapes (such as shapes containing lists) that can be built in more than one step. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is built in reverse order.

expression.**AnimateTextInReverse**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

You won't see effects of setting this property unless the specified shape gets animated. For a shape to be animated, the **TextLevelEffect** property of the **AnimationSettings** object for the shape must be set to something other than **ppAnimateLevelNone** and the [Animate](#) property must be set to **True**.

Example

This example adds a slide after slide one in the active presentation, sets the title text, adds a three-item list to the text placeholder, and sets the list to be built in reverse order.

```
With ActivePresentation.Slides.Add(2, ppLayoutText).Shapes
    .Item(1).TextFrame.TextRange.Text = "Top Three Reasons"
    With .Item(2)
        .TextFrame.TextRange = "Reason 1" & Chr(13) _
            & "Reason 2" & Chr(13) & "Reason 3"
        With .AnimationSettings
            .Animate = msOTrue
            .TextLevelEffect = ppAnimateByFirstLevel
            .AnimateTextInReverse = msOTrue
        End With
    End With
End With
```



AnimationOrder Property

Returns or sets an integer that represents the position of the specified shape within the collection of shapes to be animated. Read/write **Long**.

Remarks

You won't see effects of setting this property unless the specified shape gets animated. For a shape to be animated, the **TextLevelEffect** property of the **AnimationSettings** object for the shape must be set to something other than **ppAnimateLevelNone** and the [Animate](#) property must be set to **True**.

Note Setting the **AnimationOrder** property to a value that is less than the greatest existing **AnimationOrder** property value can shift the animation order.

Example

This example specifies that shape two on slide two in the active presentation be animated second.

```
ActivePresentation.Slides(2).Shapes(2) _  
    .AnimationSettings.AnimationOrder = 2
```



AnimationSettings Property

Returns an [AnimationSettings](#) object that represents all the special effects you can apply to the animation of the specified shape.

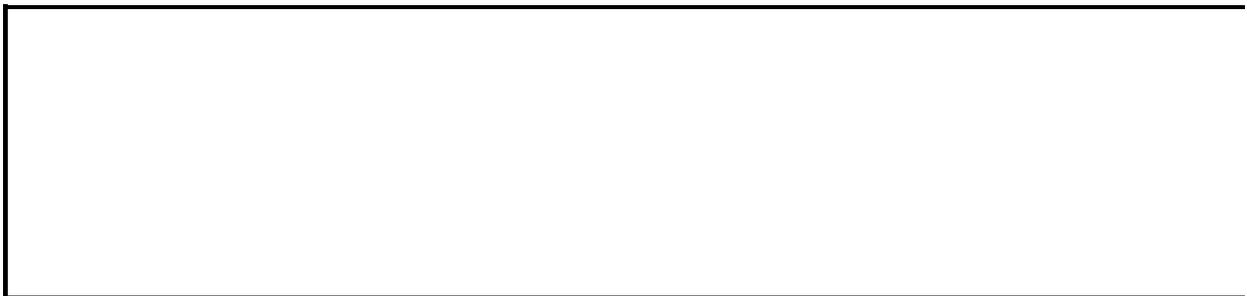
expression.**AnimationSettings**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets shape one on slide two in the active presentation to fly in from the left when the slide is built.

```
With ActivePresentation.Slides(2).Shapes(1).AnimationSettings
    .EntryEffect = ppEffectFlyFromLeft
    .TextLevelEffect = ppAnimateByAllLevels
End With
```



AnswerWizard Property

Returns the **AnswerWizard** object that contains the files used by the online Help search engine. Read-only.



Application Property

Returns an [Application](#) object that represents the creator of the specified object.

expression.**Application**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

In this example, a [Presentation](#) object is passed to the procedure. The procedure adds a slide to the presentation and then saves the presentation in the folder where Microsoft PowerPoint is running.

```
Sub AddAndSave(pptPres As Presentation)
    pptPres.Slides.Add 1, 1
    pptPres.SaveAs pptPres.Application.Path & "\Added Slide"
End Sub
```

This example displays the name of the application that created each linked OLE object on slide one in the active presentation.

```
For Each shpOle In ActivePresentation.Slides(1).Shapes
    If shpOle.Type = msoLinkedOLEObject Then
        MsgBox shpOle.OLEFormat.Application.Name
    End If
Next
```



Assistant Property

Some of the content in this topic may not be applicable to some languages.

Returns an **Assistant** object that represents the Office Assistant. Read-only.

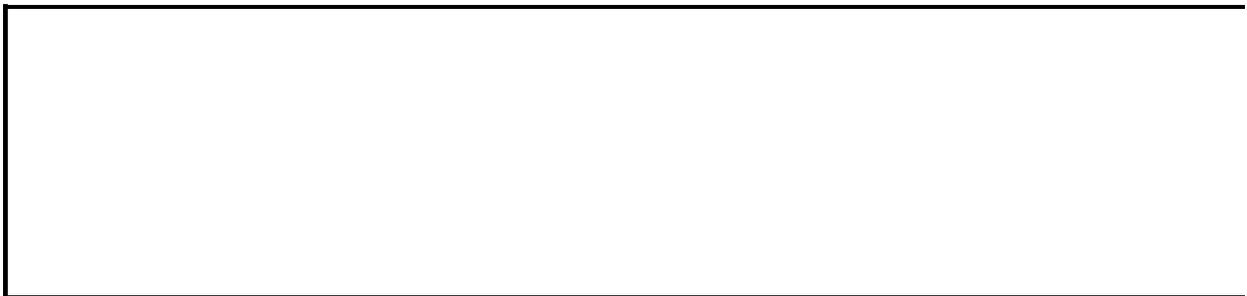
Example

This example displays the Office Assistant.

```
Application.Assistant.Visible = True
```

This example moves the Office Assistant to the upper-left region of the screen.

```
Application.Assistant.Move xLeft:=100, yTop:=100
```



Author Property

Returns a **String** that represents the author as for a specified [Comment](#) object.
Read-only.

expression.**Author**

expression Required. An expression that returns one of the objects in the Applies To list.

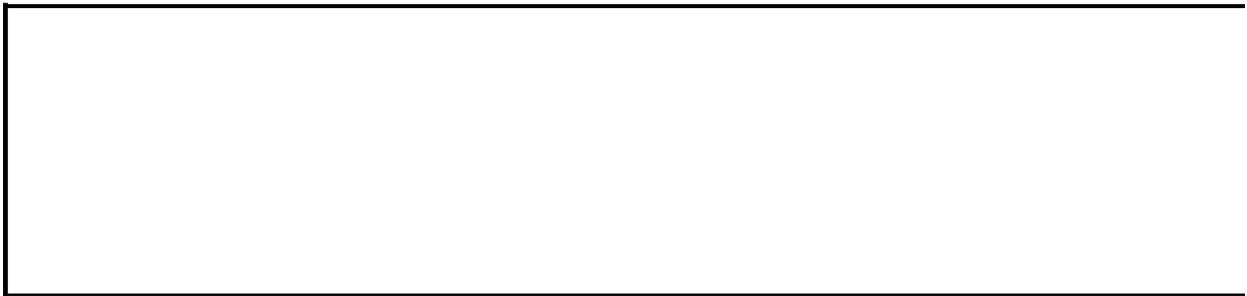
Remarks

This property only returns the author's name. To return the author's initials, use the [AuthorInitials](#) property. Specify the Author of a comment when you add a new comment to the presentation.

Example

The following example adds a comment to the first slide of the active presentation and then displays the author's name and initials in a message.

```
Sub GetAuthorName()  
  With ActivePresentation.Slides(1)  
    .Comments.Add Left:=100, Top:=100, Author:="Jeff Smith", _  
      AuthorInitials:="JS", _  
      Text:="This is a new comment added to the first slide."  
    MsgBox "This comment was created by " & _  
      .Comments(1).Author & " (" & .Comments(1).AuthorInitials  
  End With  
End Sub
```



AuthorIndex Property

Returns a **Long** representing the index number of a comment for a given author. The first comment for a given author has an index number of 1, their second comment has an index number of 2, and so on. Read-only.

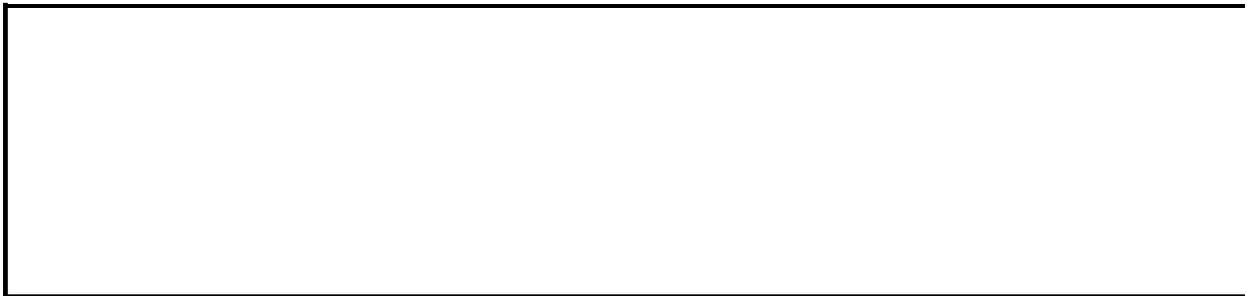
expression.**AuthorIndex**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example provide information about the authors and their comment indexes for a given slide.

```
Sub GetCommentAuthorInfo()  
  
    Dim cmtComment As Comment  
    Dim strAuthorInfo As String  
  
    With ActivePresentation.Slides(1)  
        If .Comments.Count > 0 Then  
            For Each cmtComment In .Comments  
                strAuthorInfo = strAuthorInfo & "Comment Number: "  
                    & cmtComment.AuthorIndex & vbLf & _  
                    & "Made by: " & cmtComment.Author & vbLf & _  
                    & "Says: " & cmtComment.Text & vbLf & vbLf  
            Next cmtComment  
        End If  
    End With  
  
    MsgBox "The comments for this slide are as follows: " & _  
        & vbLf & vbLf & strAuthorInfo  
  
End Sub
```



AuthorInitials Property

Returns the author's initials as a read-only **String** for a specified [Comment](#) object. Read-only.

expression.**AuthorInitials**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This property only returns the author's initials. To return the author's name use the [Author](#) property. Specify the author's initials when you add a new comment to the presentation.

Example

The following example returns the author's initials for a specified comment.

```
Sub GetAuthorName()  
  With ActivePresentation.Slides(1)  
    .Comments.Add Left:=100, Top:=100, Author:="Jeff Smith", _  
      AuthorInitials:="JS", _  
      Text:="This is a new comment added to the first slide."  
    MsgBox .Comments(1).Author & .Comments(1).AuthorInitials  
  End With  
End Sub
```



[Show All](#)

AutoAttach Property

Determines whether the place where the callout line attaches to the callout text box changes depending on whether the origin of the callout line (where the callout points to) is to the left or right of the callout text box. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The place where the callout line attaches to the callout text box changes depending on whether the origin of the callout line (where the callout points to) is to the left or right of the callout text box.

Remarks

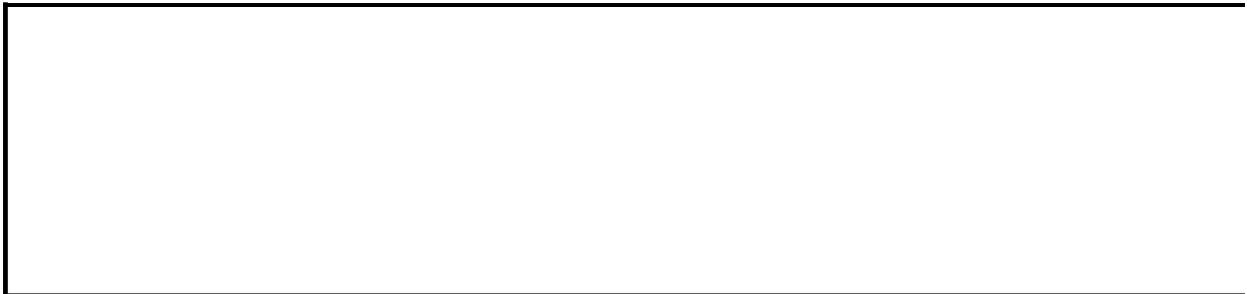
When the value of this property is **msoTrue**, the drop value (the vertical distance from the edge of the callout text box to the place where the callout line attaches) is measured from the top of the text box when the text box is to the right of the origin, and it's measured from the bottom of the text box when the text box is to the left of the origin. When the value of this property is **msoFalse**, the drop value is always measured from the top of the text box, regardless of the relative positions of the text box and the origin. Use the [CustomDrop](#) method to set the drop value, and use the [Drop](#) property to return the drop value.

Setting this property affects a callout only if it has an explicitly set drop value — that is, if the value of the [DropType](#) property is **msoCalloutDropCustom**. By default, callouts have explicitly set drop values when they're created.

Example

This example adds two callouts to the first slide. One of the callouts is automatically attached and the other is not. If you change the callout line origin for the automatically attached callout to the right of the attached text box, the position of the text box changes. The callout that is not automatically attached does not display this behavior.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    With .AddCallout(msoCalloutTwo, 420, 170, 200, 50)
        .TextFrame.TextRange.Text = "auto-attached"
        .Callout.AutoAttach = msoTrue
    End With
    With .AddCallout(msoCalloutTwo, 420, 350, 200, 50)
        .TextFrame.TextRange.Text = "not auto-attached"
        .Callout.AutoAttach = msoFalse
    End With
End With
```



AutoCorrect Property

Returns an [AutoCorrect](#) object that represents the AutoCorrect functionality in Microsoft PowerPoint.

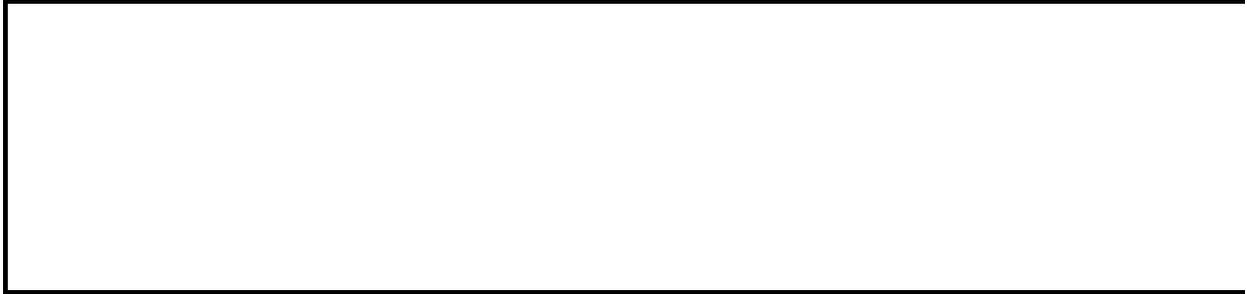
expression.**AutoCorrect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example disables display of the **AutoCorrect Options** and **AutoLayout Options** buttons.

```
Sub HideAutoCorrectOpButton()  
    With Application.AutoCorrect  
        .DisplayAutoCorrectOptions = msoFalse  
        .DisplayAutoLayoutOptions = msoFalse  
    End With  
End Sub
```



[Show All](#)

AutoFormat Property

Sets or returns an [MsoTriState](#) constant that represents a diagram's automatic formatting state. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Automatic formatting is not enabled.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Automatic formatting is enabled.

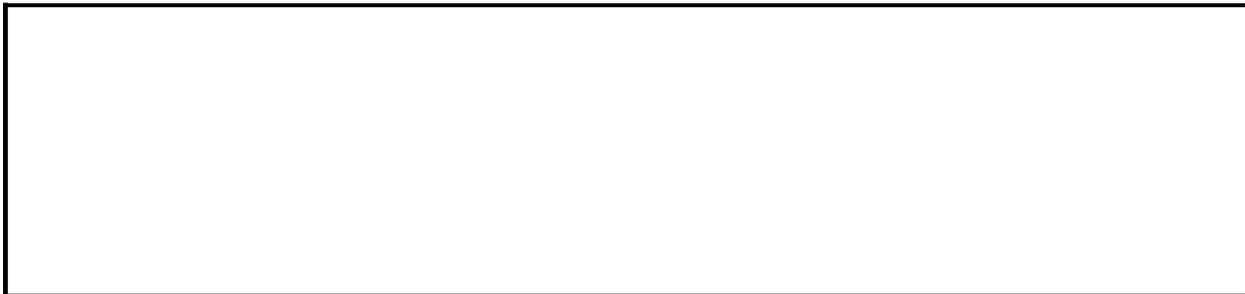
expression.**AutoFormat**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates a diagram, and enables automatic formatting.

```
Sub ConvertPyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional nodes  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically formats the diagram nodes and  
    'converts pyramid diagram to radial diagram  
    With dgnNode.Diagram  
        .AutoFormat = msoTrue  
        .Convert Type:=msoDiagramRadial  
    End With  
  
End Sub
```



[Show All](#)

AutoLayout Property

Sets or returns an [MsoTriState](#) constant that represents whether a diagram's components are automatically laid out. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse The diagram's components are not automatically laid out.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The diagram's components are automatically laid out.

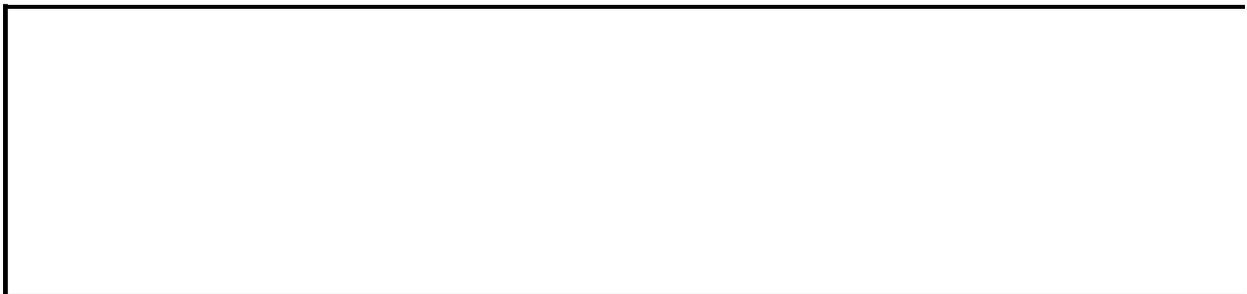
expression.**AutoLayout**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a diagram to a slide, converts it to a radial diagram, and arranges the diagram's nodes automatically.

```
Sub ConvertPyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional nodes  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically places the diagram nodes and  
    'converts pyramid diagram to radial diagram  
    With dgnNode.Diagram  
        .AutoLayout = msoTrue  
        .Convert Type:=msoDiagramRadial  
    End With  
  
End Sub
```



[Show All](#)

AutoLength Property

Determines whether the first segment of the callout retains the fixed length specified by the [Length](#) property, or is scaled automatically, whenever the callout is moved. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants. Applies only to callouts whose lines consist of more than one segment (types **msoCalloutThree** and **msoCalloutFour**).

msoCTrue

msoFalse The first segment of the callout retains the fixed length specified by the **Length** property whenever the callout is moved.

msoTriStateMixed

msoTriStateToggle

msoTrue The first segment of the callout line (the segment attached to the text callout box) is scaled automatically whenever the callout is moved.

Remarks

This property is read-only. Use the [AutomaticLength](#) method to set this property to **msoTrue**, and use the [CustomLength](#) method to set this property to **msoFalse**.

Example

This example toggles between an automatically scaling first segment and one with a fixed length for the callout line for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```



[Show All](#)

AutoLoad Property

Determines whether the specified add-in is automatically loaded each time PowerPoint is started. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified add-in is automatically loaded each time PowerPoint is started.

Remarks

Setting this property to **msoTrue** automatically sets the [Registered](#) property to **msoTrue**.

Example

This example displays the name of each add-in that's automatically loaded each time PowerPoint is started.

```
For Each myAddIn In AddIns
    If myAddIn.AutoLoad Then
        MsgBox myAddIn.Name
        afound = True
    End If
Next myAddIn
If afound <> True Then
    MsgBox "No add-ins were loaded automatically."
End If
```

This example specifies that the add-in named "MyTools" be loaded automatically each time PowerPoint is started.

```
Application.AddIns("mytools").AutoLoad = msoTrue
```



[Show All](#)

AutomationSecurity Property

Returns or sets an [MsoAutomationSecurity](#) constant that represents the security mode Microsoft PowerPoint uses when programmatically opening files. This property is automatically set to **msoAutomationSecurityLow** when the application is started. Therefore, to avoid breaking solutions that rely on the default setting, you should be careful to reset this property to **msoAutomationSecurityLow** after programmatically opening a file. Also, this property should be set immediately before and after opening a file programmatically to avoid malicious subversion. Read/write.

MsoAutomationSecurity can be one of these MsoAutomationSecurity constants. **msoAutomationSecurityByUI** Uses the security setting specified in the **Security** dialog box.

msoAutomationSecurityForceDisable Disables all macros in all files opened programmatically without showing any security alerts.

msoAutomationSecurityLow Enables all macros. This is the default value when the application is started.

expression.**AutomationSecurity**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The value of the [DisplayAlerts](#) property will not apply to security warnings. For example, if the user sets the **DisplayAlerts** property equal to **False** and the **AutomationSecurity** property to **msoAutomationSecurityByUI**, while the user is on Medium security level, then there will be security warnings while the macro is running. This allows the macro to trap file open errors, while still showing the security warning if the file open succeeds.

Example

This example captures the current automation security setting, changes the setting to disable macros, displays the **Open** dialog box, and after opening the selected presentation, sets the automation security back to its original setting.

```
Sub Security()  
    Dim secAutomation As MsoAutomationSecurity  
  
    secAutomation = Application.AutomationSecurity  
  
    Application.AutomationSecurity = msoAutomationSecurityForceDisab  
    Application.FileDialog(msoFileDialogOpen).Show  
  
    Application.AutomationSecurity = secAutomation  
End Sub
```



[Show All](#)

AutoReverse Property

Sets or returns an [MsoTriState](#) that represents whether an effect should play forward and then reverse, thereby doubling the duration. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. The effect does not play forward and then reverse.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The effect plays forward and then reverse.

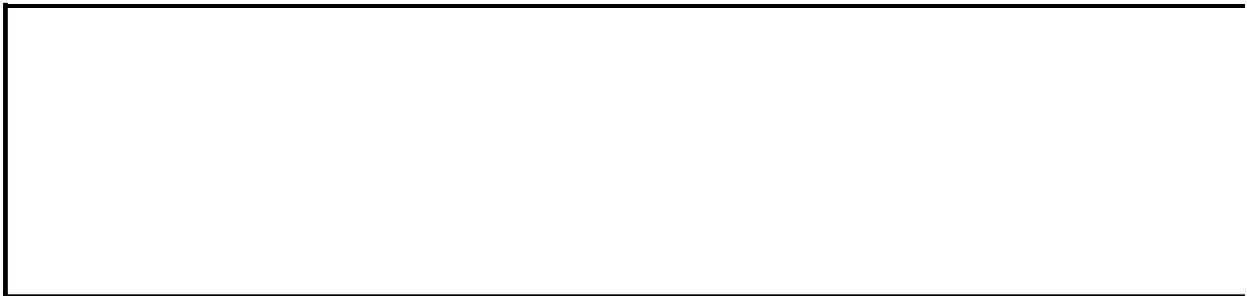
expression.**AutoReverse**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape and an animation effect to it; then sets the animation to reverse direction after finishing forward movement.

```
Sub SetEffectTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds rectangle and applies diamond effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes.AddShape  
        (Type:=msoShapeRectangle, Left:=100, _  
         Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpRectangle, _  
        effectId:=msoAnimEffectPathDiamond)  
  
    'Sets the duration of and reverses the effect  
    With effDiamond.Timing  
        .Duration = 5 ' Length of effect.  
        .AutoReverse = msoTrue  
    End With  
  
End Sub
```



[Show All](#)

AutoRotateNumbers Property

Returns or sets lateral compression. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Half-width numbers will not be compressed in lateral columns.

msoTriStateMixed

msoTriStateToggle

msoTrue Displays half-width numbers within vertical text in two-character lateral columns.

Example

This example sets the text direction of shape three on the first slide to vertical text, and sets lateral column compression.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).TextFrame
    .Orientation = msoTextOrientationVerticalFarEast
    .TextRange.Font.AutoRotateNumbers = msoTrue
End With
```



[Show All](#)

AutoShapeType Property

Returns or sets the shape type for the specified **Shape** or **ShapeRange** object, which must represent an AutoShape other than a line, freeform drawing, or connector. Read/write [MsoAutoShapeType](#).

MsoAutoShapeType can be one of these MsoAutoShapeType constants.

msoShapeFlowchartConnector

msoShapeFlowchartData

msoShapeFlowchartDecision

msoShapeFlowchartDelay

msoShapeFlowchartDirectAccessStorage

msoShapeFlowchartDisplay

msoShapeFlowchartDocument

msoShapeFlowchartExtract

msoShapeFlowchartInternalStorage

msoShapeFlowchartMagneticDisk

msoShapeFlowchartManualInput

msoShapeFlowchartManualOperation

msoShapeFlowchartMerge

msoShapeFlowchartMultidocument

msoShapeFlowchartOffpageConnector

msoShapeFlowchartOr

msoShapeFlowchartPredefinedProcess

msoShapeFlowchartPreparation

msoShapeFlowchartProcess

msoShapeFlowchartPunchedTape

msoShapeFlowchartSequentialAccessStorage

msoShapeFlowchartSort

msoShapeFlowchartStoredData

msoShapeFlowchartSummingJunction

msoShapeFlowchartTerminator

msoShapeFoldedCorner
msoShapeHeart
msoShapeHexagon
msoShapeHorizontalScroll
msoShapeIsoscelesTriangle
msoShapeLeftArrow
msoShapeLeftArrowCallout
msoShapeLeftBrace
msoShapeLeftBracket
msoShapeLeftRightArrow
msoShapeLeftRightArrowCallout
msoShapeLeftRightUpArrow
msoShapeLeftUpArrow
msoShapeLightningBolt
msoShapeLineCallout1
msoShapeLineCallout1AccentBar
msoShapeLineCallout1BorderandAccentBar
msoShapeLineCallout1NoBorder
msoShapeLineCallout2
msoShapeLineCallout2AccentBar
msoShapeLineCallout2BorderandAccentBar
msoShapeLineCallout2NoBorder
msoShapeLineCallout3
msoShapeLineCallout3AccentBar
msoShapeLineCallout3BorderandAccentBar
msoShapeLineCallout3NoBorder
msoShapeLineCallout4
msoShapeLineCallout4AccentBar
msoShapeLineCallout4BorderandAccentBar
msoShapeLineCallout4NoBorder
msoShapeMixed
msoShapeMoon
msoShapeNoSymbol

msoShapeNotchedRightArrow
msoShapeNotPrimitive
msoShapeOctagon
msoShapeOval
msoShapeOvalCallout
msoShapeParallelogram
msoShapePentagon
msoShapePlaque
msoShapeQuadArrow
msoShapeQuadArrowCallout
msoShapeRectangle
msoShapeRectangularCallout
msoShapeRegularPentagon
msoShapeRightArrow
msoShapeRightArrowCallout
msoShapeRightBrace
msoShapeRightBracket
msoShapeRightTriangle
msoShapeRoundedRectangle
msoShapeRoundedRectangularCallout
msoShapeSmileyFace
msoShapeStripedRightArrow
msoShapeSun
msoShapeTrapezoid
msoShapeUpArrow
msoShapeUpArrowCallout
msoShapeUpDownArrow
msoShapeUpDownArrowCallout
msoShapeUpRibbon
msoShapeUTurnArrow
msoShapeVerticalScroll
msoShapeWave
msoShapeFlowchartCollate

msoShape16pointStar
msoShape24pointStar
msoShape32pointStar
msoShape4pointStar
msoShape5pointStar
msoShape8pointStar
msoShapeActionButtonBackorPrevious
msoShapeActionButtonBeginning
msoShapeActionButtonCustom
msoShapeActionButtonDocument
msoShapeActionButtonEnd
msoShapeActionButtonForwardorNext
msoShapeActionButtonHelp
msoShapeActionButtonHome
msoShapeActionButtonInformation
msoShapeActionButtonMovie
msoShapeActionButtonReturn
msoShapeActionButtonSound
msoShapeArc
msoShapeBalloon
msoShapeBentArrow
msoShapeBentUpArrow
msoShapeBevel
msoShapeBlockArc
msoShapeCan
msoShapeChevron
msoShapeCircularArrow
msoShapeCloudCallout
msoShapeCross
msoShapeCube
msoShapeCurvedDownArrow
msoShapeCurvedDownRibbon
msoShapeCurvedLeftArrow

msoShapeCurvedRightArrow
msoShapeCurvedUpArrow
msoShapeCurvedUpRibbon
msoShapeDiamond
msoShapeDonut
msoShapeDoubleBrace
msoShapeDoubleBracket
msoShapeDoubleWave
msoShapeDownArrow
msoShapeDownArrowCallout
msoShapeDownRibbon
msoShapeExplosion1
msoShapeExplosion2
msoShapeFlowchartAlternateProcess
msoShapeFlowchartCard

expression.**AutoShapeType**

expression Required. An expression that returns one of the objects in the Applies To list. Read/write **Long**.

Note When you change the type of a shape, the shape retains its size, color, and other attributes.

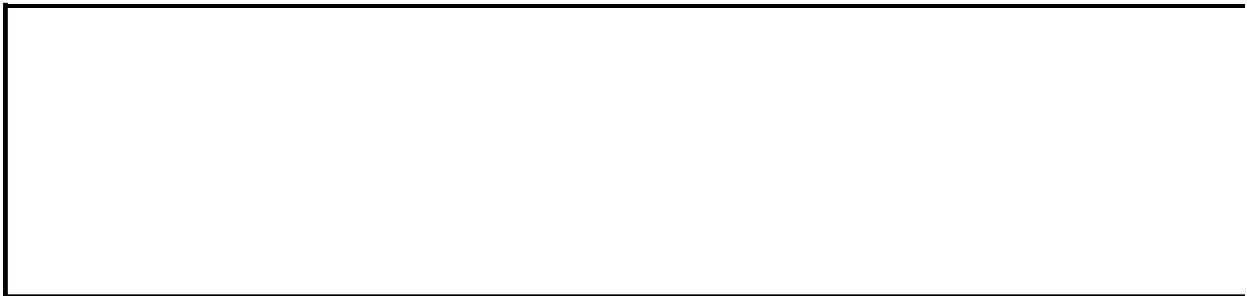
Remarks

Use the **Type** property of the [ConnectorFormat](#) object to set or return the connector type.

Example

This example replaces all 16-point stars with 32-point stars in myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.AutoShapeType = msoShape16pointStar Then
        s.AutoShapeType = msoShape32pointStar
    End If
Next
```



[Show All](#)

AutoSize Property

Returns or sets a value that indicates whether the size of the specified shape is changed automatically to fit text within its boundaries. Read/write [PpAutoSize](#).

PpAutoSize can be one of these PpAutoSize constants.

ppAutoSizeMixed

ppAutoSizeNone

ppAutoSizeShapeToFitText

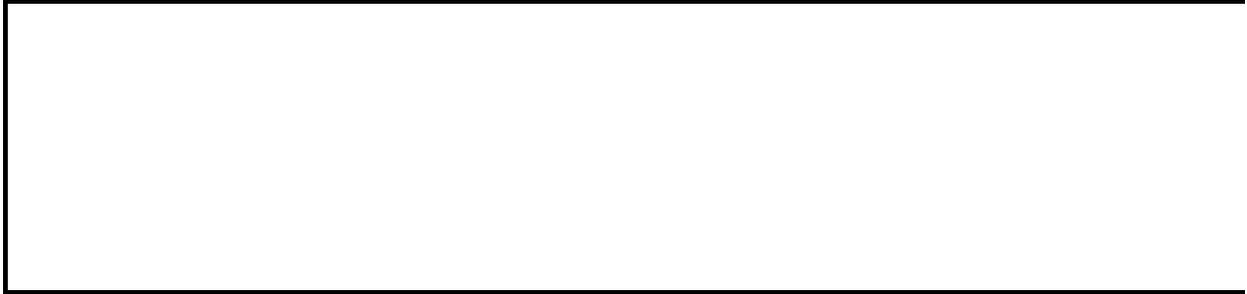
expression.**AutoSize**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adjusts the size of the title bounding box on slide one to fit the title text.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1)
    If .TextFrame.TextRange.Characters.Count < 50 Then
        .TextFrame.AutoSize = ppAutoSizeShapeToFitText
    End If
End With
```



[Show All](#)

AutoUpdate Property

Returns or sets the way the link will be updated. Read/write [PpUpdateOption](#).

PpUpdateOption can be one of these PpUpdateOption constants.

ppUpdateOptionAutomatic Link will be updated each time the presentation is opened or the source file changes.

ppUpdateOptionManual Link will be updated only when the user specifically asks to update the presentation.

ppUpdateOptionMixed

expression.**AutoUpdate**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example loops through all the shapes on all the slides in the active presentation and sets all linked Microsoft Excel worksheets to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Excel.Sheet" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
```



BackColor Property

Returns or sets a [ColorFormat](#) object that represents the background color for the specified fill or patterned line. Read/write.

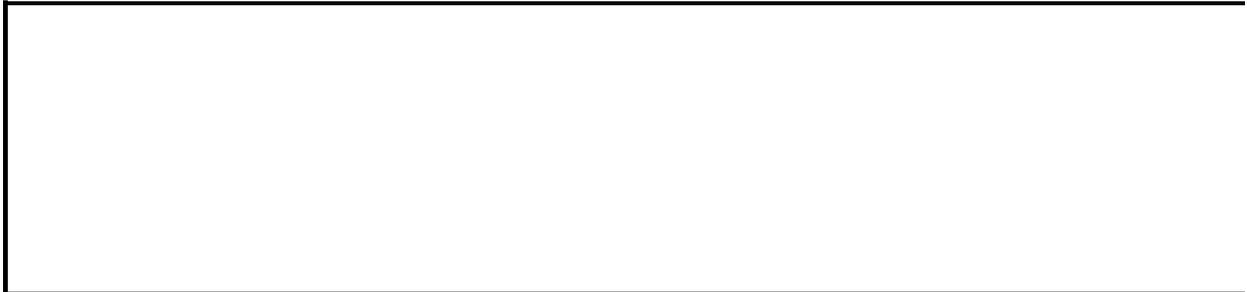
Example

This example adds a rectangle to myDocument and then sets the foreground color, background color, and gradient for the rectangle's fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    90, 90, 90, 50).Fill
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(170, 170, 170)
    .TwoColorGradient msoGradientHorizontal, 1
End With
```

This example adds a patterned line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 100, 250, 0).Line
    .Weight = 6
    .ForeColor.RGB = RGB(0, 0, 255)
    .BackColor.RGB = RGB(128, 0, 0)
    .Pattern = msoPatternDarkDownwardDiagonal
End With
```



Background Property

Returns a [ShapeRange](#) object that represents the slide background.

expression.**Background**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If you use the **Background** property to set the background for an individual slide without changing the slide master, the [FollowMasterBackground](#) property for that slide must be set to **False**.

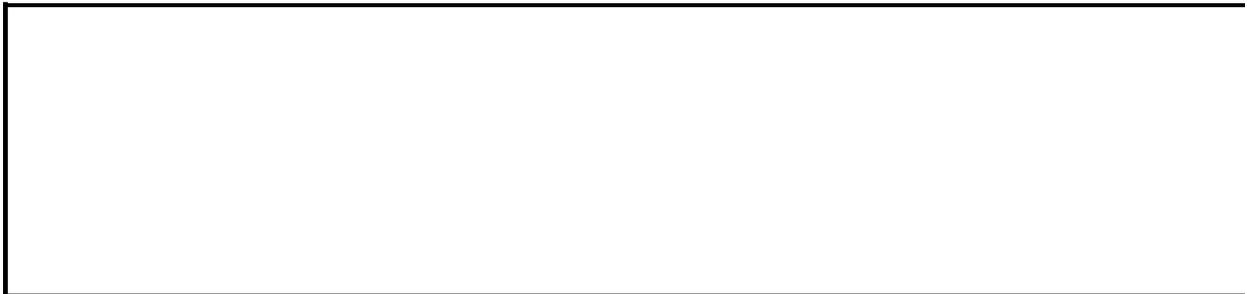
Example

This example sets the background of the slide master in the active presentation to a preset shade.

```
ActivePresentation.SlideMaster.Background.Fill.PresetGradient _  
    Style:=msoGradientHorizontal, Variant:=1, _  
    PresetGradientType:=msoGradientLateSunset
```

This example sets the background of slide one in the active presentation to a preset shade.

```
With ActivePresentation.Slides(1)  
    .FollowMasterBackground = False  
    .Background.Fill.PresetGradient Style:=msoGradientHorizontal, _  
        Variant:=1, PresetGradientType:=msoGradientLateSunset  
End With
```



[Show All](#)

BaseLineAlignment Property

Returns or sets the base line alignment for the specified paragraph. Read/write [PpBaselineAlignment](#).

PpBaselineAlignment can be one of these PpBaselineAlignment constants.

ppBaselineAlignBaseline

ppBaselineAlignCenter

ppBaselineAlignFarEast50

ppBaselineAlignMixed

ppBaselineAlignTop

expression.**BaseLineAlignment**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the base line alignment for the paragraphs in shape two on slide one in the active presentation.

```
MsgBox ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange _  
    .ParagraphFormat.BaseLineAlignment
```



BaselineOffset Property

Returns or sets the baseline offset for the specified superscript or subscript characters. Can be a floating-point value from -1 through 1 . A value of -1 represents an offset of -100 percent, and a value of 1 represents an offset of 100 percent. Read/write **Single**.

Remarks

Setting the **BaselineOffset** property to a negative value automatically sets the **Subscript** property to **True** and the **Superscript** property to **False**.

Setting the **BaselineOffset** property to a positive value automatically sets the **Subscript** property to **False** and the **Superscript** property to **True**.

Setting the **Subscript** property to **True** automatically sets the **BaselineOffset** property to 0.3 (30 percent).

Setting the **Superscript** property to **True** automatically sets the **BaselineOffset** property to -0.25 (-25 percent).

Example

This example sets the text for shape two on slide one and then makes the second character subscript with a 20-percent offset.

```
With Application.ActivePresentation.Slides(1) _  
    .Shapes(2).TextFrame.TextRange  
    .Text = "H2O"  
    .Characters(2, 1).Font.BaselineOffset = -0.2  
End With
```



[Show All](#)

BeginArrowheadLength Property

Returns or sets the length of the arrowhead at the beginning of the specified line.
Read/write [MsoArrowheadLength](#).

MsoArrowheadLength can be one of these MsoArrowheadLength constants.

msoArrowheadLengthMedium

msoArrowheadLengthMixed

msoArrowheadLong

msoArrowheadShort

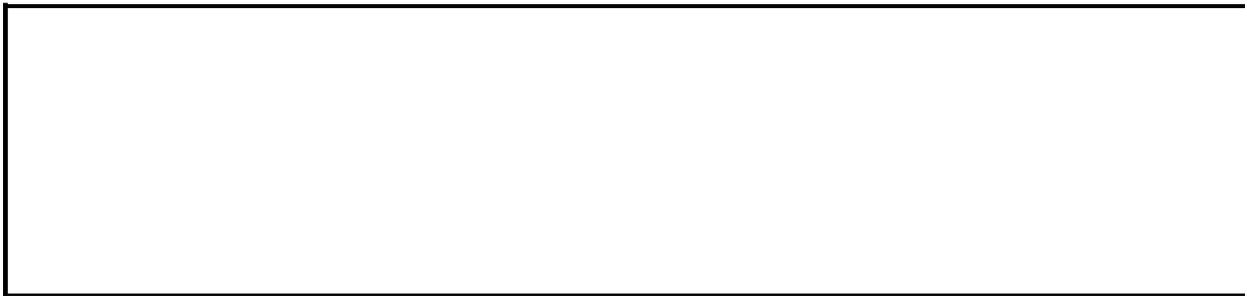
expression.**BeginArrowheadLength**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

BeginArrowheadStyle Property

Returns or sets the style of the arrowhead at the beginning of the specified line.
Read/write [MsoArrowheadStyle](#).

MsoArrowheadStyle can be one of these MsoArrowheadStyle constants.

msoArrowheadDiamond

msoArrowheadNone

msoArrowheadOpen

msoArrowheadOval

msoArrowheadStealth

msoArrowheadStyleMixed

msoArrowheadTriangle

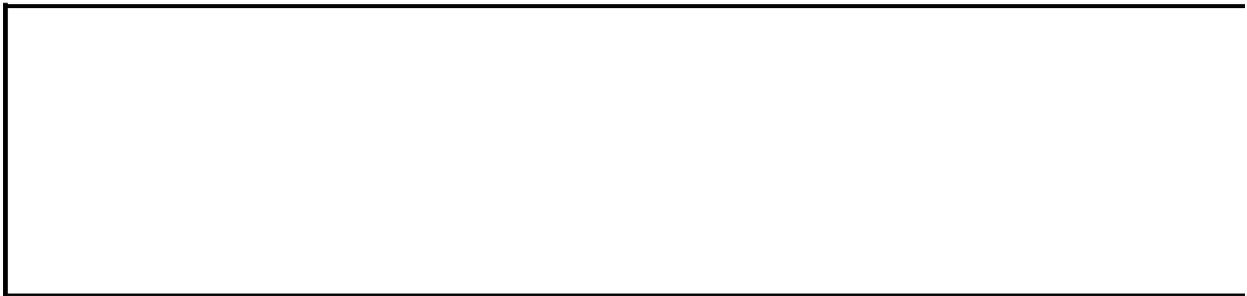
expression.**BeginArrowheadStyle**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

BeginArrowheadWidth Property

Returns or sets the width of the arrowhead at the beginning of the specified line.
Read/write [MsoArrowheadWidth](#).

MsoArrowheadWidth can be one of these MsoArrowheadWidth constants.

msoArrowheadNarrow

msoArrowheadWide

msoArrowheadWidthMedium

msoArrowheadWidthMixed

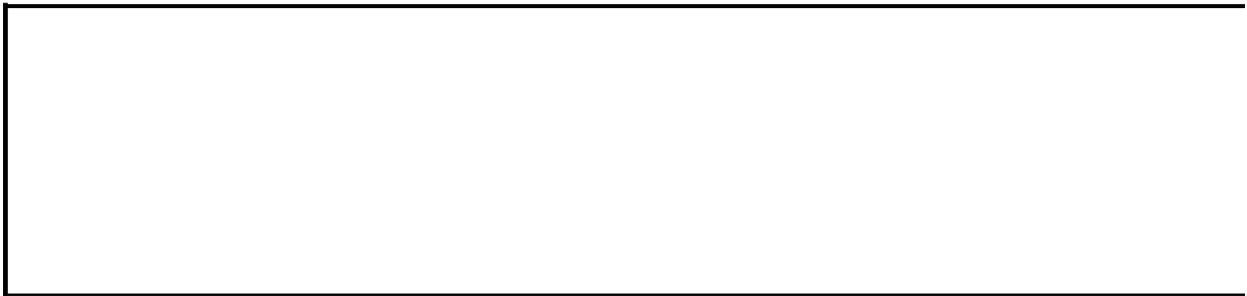
expression.**BeginArrowheadWidth**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

BeginConnected Property

Determines whether the beginning of the specified connector is connected to a shape. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

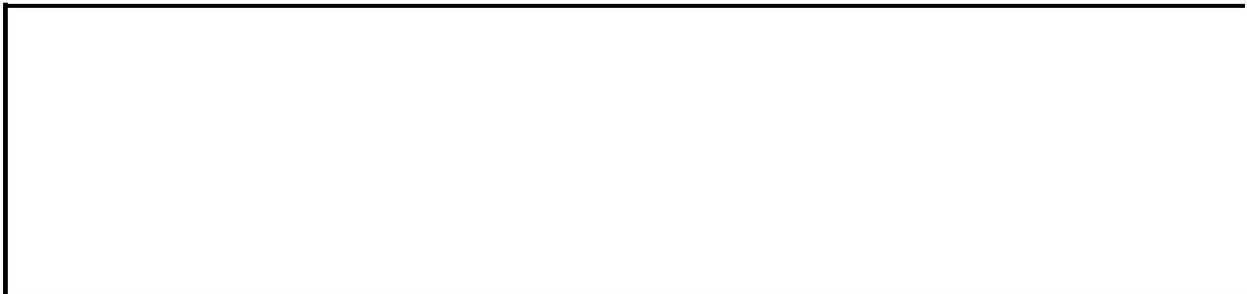
msoTriStateToggle

msoTrue The beginning of the specified connector is connected to a shape.

Example

If shape three on the first slide in the active presentation is a connector whose beginning is connected to a shape, this example stores the connection site number in the variable `oldBeginConnSite`, stores a reference to the connected shape in the object variable `oldBeginConnShape`, and then disconnects the beginning of the connector from the shape.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Connector Then
        With .ConnectorFormat
            If .BeginConnected Then
                oldBeginConnSite = .BeginConnectionSite
                Set oldBeginConnShape = .BeginConnectedShape
                .BeginDisconnect
            End If
        End With
    End If
End With
```



BeginConnectedShape Property

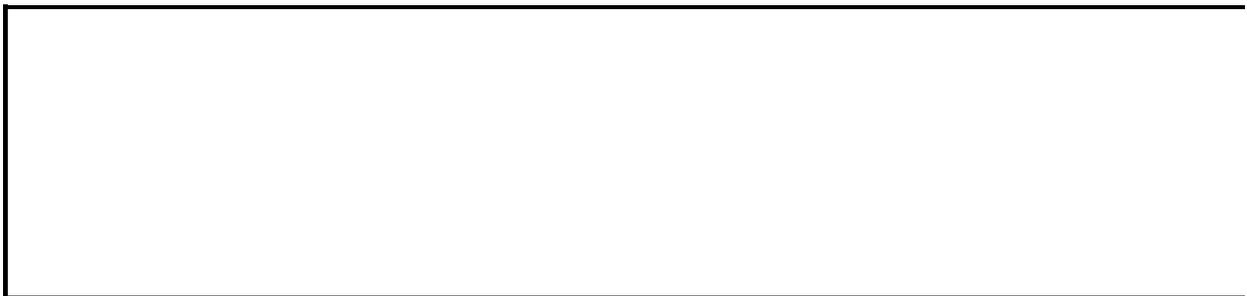
Returns a [Shape](#) object that represents the shape that the beginning of the specified connector is attached to. Read-only.

Note If the beginning of the specified connector isn't attached to a shape, this property generates an error.

Example

This example assumes that the first slide in the active presentation already contains two shapes attached by a connector named "Conn1To2." The code adds a rectangle and a connector to the first slide. The beginning of the new connector will be attached to the same connection site as the beginning of the connector named "Conn1To2," and the end of the new connector will be attached to connection site one on the new rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    Set r3 = .AddShape(msoShapeRectangle, 450, 190, 200, 100)
    .AddConnector(msoConnectorCurve, 0, 0, 10, 10) _
        .Name = "Conn1To3"
    With .Item("Conn1To2").ConnectorFormat
        beginConnSite1 = .BeginConnectionSite
        Set beginConnShape1 = .BeginConnectedShape
    End With
    With .Item("Conn1To3").ConnectorFormat
        .BeginConnect beginConnShape1, beginConnSite1
        .EndConnect r3, 1
    End With
End With
```



BeginConnectionSite Property

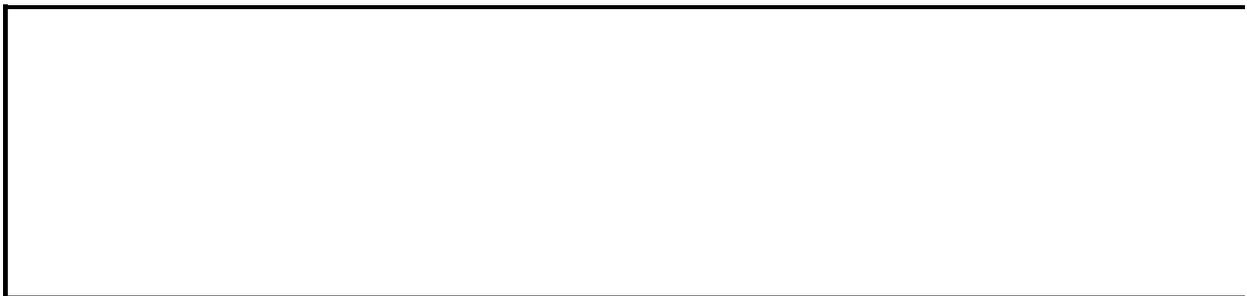
Returns an integer that specifies the connection site that the beginning of a connector is connected to. Read-only **Long**.

Note If the beginning of the specified connector isn't attached to a shape, this property generates an error.

Example

This example assumes that the first slide in the active presentation already contains two shapes attached by a connector named "Conn1To2." The code adds a rectangle and a connector to the first slide. The beginning of the new connector will be attached to the same connection site as the beginning of the connector named "Conn1To2," and the end of the new connector will be attached to connection site one on the new rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    Set r3 = .AddShape(msoShapeRectangle, 450, 190, 200, 100)
    .AddConnector(msoConnectorCurve, _
        0, 0, 10, 10).Name = "Conn1To3"
    With .Item("Conn1To2").ConnectorFormat
        beginConnSite1 = .BeginConnectionSite
        Set beginConnShape1 = .BeginConnectedShape
    End With
    With .Item("Conn1To3").ConnectorFormat
        .BeginConnect beginConnShape1, beginConnSite1
        .EndConnect r3, 1
    End With
End With
```



Behaviors Property

Returns a specified slide animation behavior as an [AnimationBehaviors](#) collection.

expression.**Behaviors**

expression Required. An expression that returns one of the objects in the Applies To list.

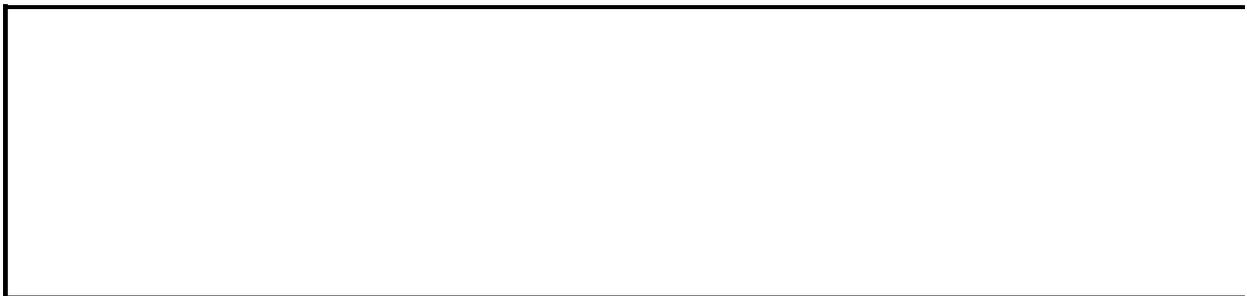
Remarks

To return a single [AnimationBehavior](#) object in the **AnimationBehaviors** collection, use the [Item](#) method or **Behaviors(*index*)**, where *index* is the index number of the **AnimationBehavior** object in the **AnimationBehaviors** collection.

Example

The following example returns a specific animation behavior type in the active presentation.

```
Sub ReturnTypeValue
    MsgBox ActiveWindow.Selection.SlideRange(1).TimeLine _
        .MainSequence(1).Behaviors.Item(1).Type
End Sub
```



[Show All](#)

BlackAndWhite Property

Determines whether the document window display is black and white.

Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

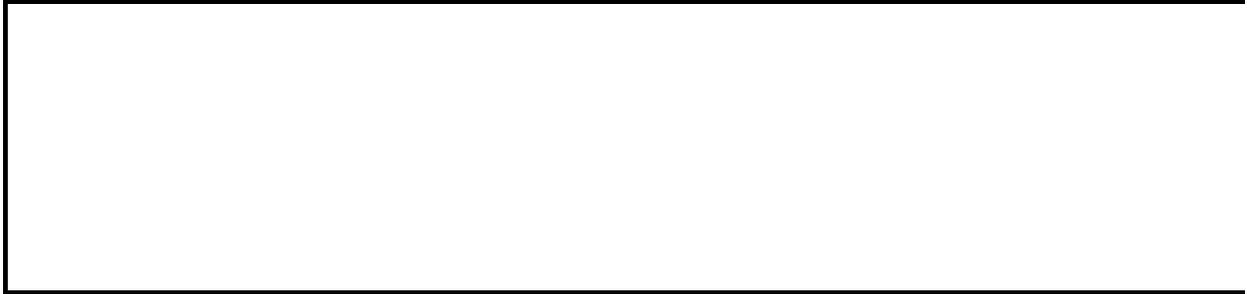
msoTriStateToggle

msoTrue The document window display is black and white.

Example

This example changes the display in window one to black and white.

```
Application.Windows(1).BlackAndWhite = msoTrue
```



[Show All](#)

BlackWhiteMode Property

Returns or sets a value that indicates how the specified shape appears when the presentation is viewed in black-and-white mode. Read/write [MsoBlackWhiteMode](#).

MsoBlackWhiteMode can be one of these MsoBlackWhiteMode constants.

msoBlackWhiteAutomatic

msoBlackWhiteBlack

msoBlackWhiteBlackTextAndLine

msoBlackWhiteDontShow

msoBlackWhiteGrayOutline

msoBlackWhiteGrayScale

msoBlackWhiteHighContrast

msoBlackWhiteInverseGrayScale

msoBlackWhiteLightGrayScale

msoBlackWhiteMixed

msoBlackWhiteWhite

expression.**BlackWhiteMode**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets shape one on myDocument to appear in black-and-white mode. When you view the presentation in black-and-white mode, shape one will appear black, regardless of what color it is in color mode.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(1).BlackWhiteMode = msoBlackWhiteBlack
```



[Show All](#)

Bold Property

Determines whether the character format is bold. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The character format is not bold.

msoTriStateMixed The specified text range contains both bold and nonbold characters.

msoTriStateToggle

msoTrue The character format is bold.

Example

This example sets characters one through five in the title on slide one to bold.

```
Set myT = Application.ActivePresentation.Slides(1).Shapes.Title  
myT.TextFrame.TextRange.Characters(1, 5).Font.Bold = msoTrue
```



[Show All](#)

Border Property

Determines whether the text in the specified callout is surrounded by a border.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The text in the specified callout is surrounded by a border.

Example

This example adds to myDocument an oval and a callout that points to the oval. The callout text won't have a border, but it will have a vertical accent bar that separates the text from the callout line.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape msoShapeOval, 180, 200, 280, 130
    With .AddCallout(msoCalloutTwo, 420, 170, 170, 40)
        .TextFrame.TextRange.Text = "My oval"
        With .Callout
            .Accent = msoTrue
            .Border = msoFalse
        End With
    End With
End With
```



Borders Property

Returns a [Borders](#) collection that represents the borders and diagonal lines for the specified **Cell** object or **CellRange** collection. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example sets the thickness of the left border for the first cell in the second row of the selected table to three points.

```
ActiveWindow.Selection.ShapeRange.Table.Rows(2) _  
    .Cells(1).Borders.Item(ppBorderLeft).Weight = 3
```



[Show All](#)

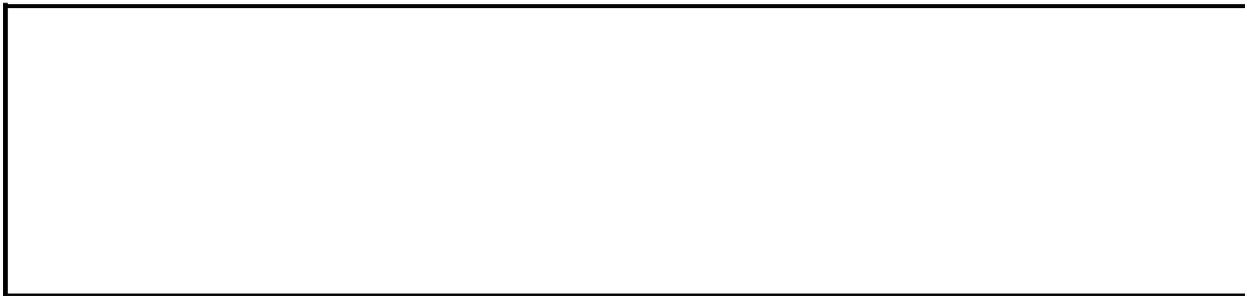
BoundHeight Property

Returns the height (in points) of the text [bounding box](#) for the specified text frame. Read-only **Single**.

Example

This example adds a rounded rectangle to slide one in the active presentation. The rectangle has the same dimensions as the text bounding box for shape one.

```
With Application.ActivePresentation.Slides(1).Shapes
    Set tr = .Item(1).TextFrame.TextRange
    Set roundRect = .AddShape(msoShapeRoundedRectangle, _
        tr.BoundLeft, tr.BoundTop, tr.BoundWidth, tr.BoundHeight)
End With
With roundRect.Fill
    .ForeColor.RGB = RGB(255, 0, 128)
    .Transparency = 0.75
End With
```



[Show All](#)

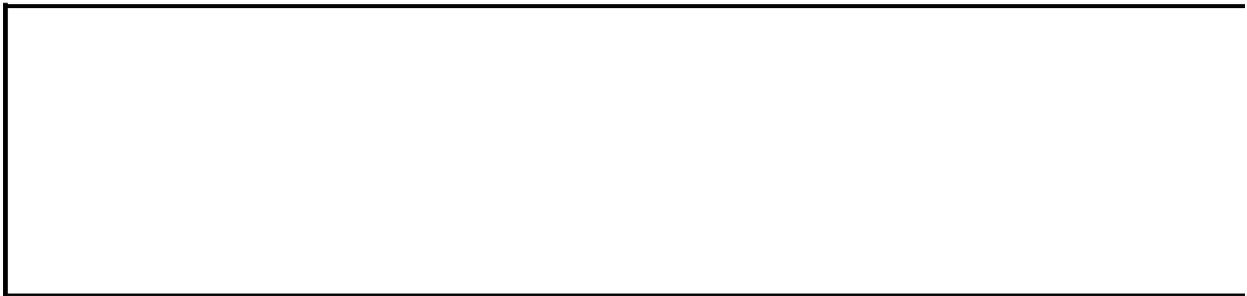
BoundLeft Property

Returns the distance (in points) from the left edge of the text [bounding box](#) for the specified text frame to the left edge of the slide. Read-only **Single**.

Example

This example adds a rounded rectangle to slide one in the active presentation. The rectangle has the same dimensions as the text bounding box for shape one.

```
With Application.ActivePresentation.Slides(1).Shapes
    Set tr = .Item(1).TextFrame.TextRange
    Set roundRect = .AddShape(msoShapeRoundedRectangle, _
        tr.BoundLeft, tr.BoundTop, tr.BoundWidth, tr.BoundHeight)
End With
With roundRect.Fill
    .ForeColor.RGB = RGB(255, 0, 128)
    .Transparency = 0.75
End With
```



[Show All](#)

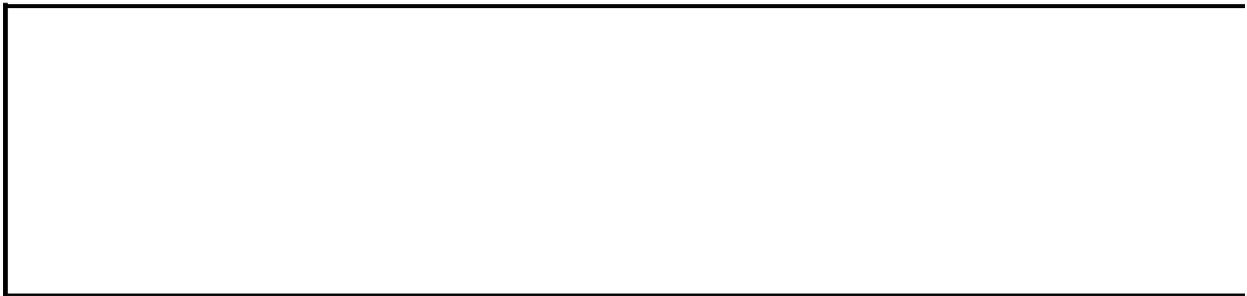
BoundTop Property

Returns the distance (in points) from the top of the of the text [bounding box](#) for the specified text frame to the top of the slide. Read-only **Single**.

Example

This example adds a rounded rectangle to slide one in the active presentation. The rectangle has the same dimensions as the text bounding box for shape one.

```
With Application.ActivePresentation.Slides(1).Shapes
    Set tr = .Item(1).TextFrame.TextRange
    Set roundRect = .AddShape(msoShapeRoundedRectangle, _
        tr.BoundLeft, tr.BoundTop, tr.BoundWidth, tr.BoundHeight)
End With
With roundRect.Fill
    .ForeColor.RGB = RGB(255, 0, 128)
    .Transparency = 0.75
End With
```



[Show All](#)

BoundWidth Property

Returns the width (in points) of the text [bounding box](#) for the specified text frame. Read-only **Single**.

Example

This example adds a rounded rectangle to slide one in the active presentation. The rectangle has the same dimensions as the text bounding box for shape one.

```
With Application.ActivePresentation.Slides(1).Shapes
    Set tr = .Item(1).TextFrame.TextRange
    Set roundRect = .AddShape(msoShapeRoundedRectangle, _
        tr.BoundLeft, tr.BoundTop, tr.BoundWidth, tr.BoundHeight)
End With
With roundRect.Fill
    .ForeColor.RGB = RGB(255, 0, 128)
    .Transparency = 0.75
End With
```



Brightness Property

Returns or sets the brightness of the specified picture or OLE object. The value for this property must be a number from 0.0 (dimkest) to 1.0 (brightest).

Read/write **Single**.

Example

This example sets the brightness for shape one on myDocument. Shape one must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).PictureFormat.Brightness = 0.3
```



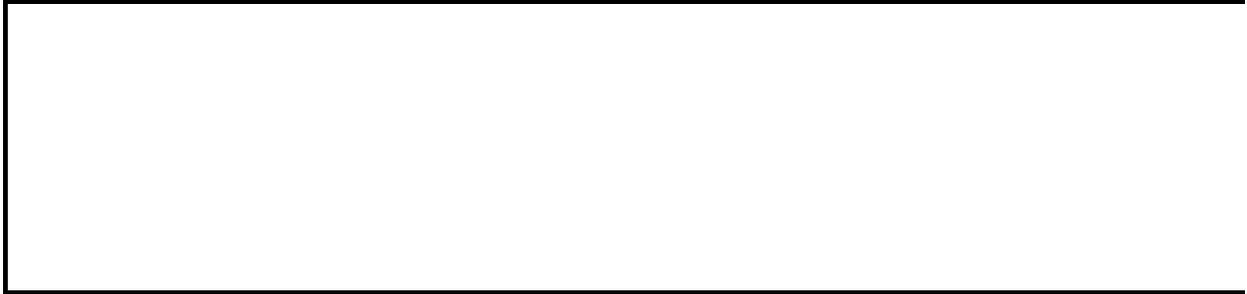
Build Property

Returns the PowerPoint build number. Read-only **String**.

Example

This example displays the PowerPoint build number.

```
MsgBox Prompt:=Application.Build, Title:="PowerPoint Build"
```



[Show All](#)

BuildByLevelEffect Property

Returns an [MsoAnimateByLevel](#) constant that represents the level of the animation build effect. Read-only.

MsoAnimateByLevel can be one of these MsoAnimateByLevel constants.

msoAnimateChartAllAtOnce

msoAnimateChartByCategory

msoAnimateChartByCategoryElements

msoAnimateChartBySeries

msoAnimateChartBySeriesElements

msoAnimateDiagramAllAtOnce

msoAnimateDiagramBreadthByLevel

msoAnimateDiagramBreadthByNode

msoAnimateDiagramClockwise

msoAnimateDiagramClockwiseIn

msoAnimateDiagramClockwiseOut

msoAnimateDiagramCounterClockwise

msoAnimateDiagramCounterClockwiseIn

msoAnimateDiagramCounterClockwiseOut

msoAnimateDiagramDepthByBranch

msoAnimateDiagramDepthByNode

msoAnimateDiagramDown

msoAnimateDiagramInByRing

msoAnimateDiagramOutByRing

msoAnimateDiagramUp

msoAnimateLevelMixed

msoAnimateTextByAllLevels

msoAnimateTextByFifthLevel

msoAnimateTextByFirstLevel

msoAnimateTextByFourthLevel

msoAnimateTextBySecondLevel

msoAnimateTextByThirdLevel
msoAnimationLevelNone

expression.**BuildByLevelEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example returns a build-by-level effect.

```
Sub QueryBuildByLevelEffect()  
    Dim effMain As Effect  
  
    Set effMain = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence(1)  
  
    If effMain.EffectInformation.BuildByLevelEffect <> msoAnimateLev  
        ActivePresentation.Slides(1).TimeLine.MainSequence _  
            .ConvertToTextUnitEffect Effect:=effMain, _  
            UnitEffect:=msoAnimTextUnitEffectByWord  
    End If  
  
End Sub
```



BuiltInDocumentProperties Property

Returns a **DocumentProperties** collection that represents all the built-in document properties for the specified presentation. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Remarks

Use the [CustomDocumentProperties](#) property to return the collection of custom document properties.

Example

This example displays the names of all the built-in document properties for the active presentation.

```
For Each p In Application.ActivePresentation _  
    .BuiltInDocumentProperties  
    bidpList = bidpList & p.Name & Chr$(13)  
Next  
MsgBox bidpList
```

This example sets the "Category" built-in property for the active presentation if the author of the presentation is Jake Jarmel.

```
With Application.ActivePresentation.BuiltInDocumentProperties  
    If .Item("author").Value = "Jake Jarmel" Then  
        .Item("category").Value = "Creative Writing"  
    End If  
End With
```



Bullet Property

Returns a [BulletFormat](#) object that represents bullet formatting for the specified paragraph format. Read-only.

Example

This example sets the bullet size and bullet color for the paragraphs in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat.Bullet
    .Visible = True
    .RelativeSize = 1.25
    .Font.Color = RGB(255, 0, 255)
  End With
End With
```



[Show All](#)

By Property

 [By property as it applies to the **ColorEffect** object.](#)

Returns a **ColorFormat** object that represents a change to the color of the object by the specified number, expressed in RGB format.

expression.**By**

expression Required. An expression that returns a **ColorEffect** object.

 [By property as it applies to the **RotationEffect** object.](#)

Sets or returns a **Single** that represents the rotation of an object by the specified number of degrees; for example, a value of 180 means to rotate the object by 180 degrees. Read/write.

expression.**By**

expression Required. An expression that returns a **RotationEffect** object.

Remarks

The specified object will be rotated with the center of the object remaining in the same position on the screen.

If both the **By** and **To** properties are set for a rotation effect, then the value of the **By** property is ignored.

Floating point numbers (for example, 55.5) are valid, but negative numbers are not.

Remarks

Do not confuse this property with the [ByX](#) or [ByY](#) properties of the [ScaleEffect](#) and [MotionEffect](#) objects, which are only used for scaling or motion effects.

Example

 [As it applies to the **ColorEffect** object.](#)

This example adds a color effect and changes its color. This example assumes there is at least one shape on the first slide of the active presentation.

```
Sub AddAndChangeColorEffect()  
    Dim effBlinds As Effect  
    Dim tmlnShape As TimeLine  
    Dim shpShape As Shape  
    Dim animBehavior As AnimationBehavior  
    Dim clrEffect As ColorEffect  
  
    'Sets shape, timing, and effect  
    Set shpShape = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlnShape = ActivePresentation.Slides(1).TimeLine  
    Set effBlinds = tmlnShape.MainSequence.AddEffect _  
        (Shape:=shpShape, effectId:=msoAnimEffectBlinds)  
  
    'Adds animation behavior and color effect  
    Set animBehavior = tmlnShape.MainSequence(1).Behaviors _  
        .Add(Type:=msoAnimTypeColor)  
    Set clrEffect = animBehavior.ColorEffect  
  
    'Specifies color  
    clrEffect.By.RGB = RGB(Red:=255, Green:=0, Blue:=0)  
End Sub
```

 [As it applies to the **RotationEffect** object.](#)

This example adds a rotation effect and changes its rotation.

```
Sub AddAndChangeRotationEffect()  
    Dim effBlinds As Effect  
    Dim tmlnShape As TimeLine  
    Dim shpShape As Shape  
    Dim animBehavior As AnimationBehavior  
    Dim rtnEffect As RotationEffect  
  
    'Sets shape, timing, and effect  
    Set shpShape = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlnShape = ActivePresentation.Slides(1).TimeLine
```

```
Set effBlinds = tmInShape.MainSequence.AddEffect _  
    (Shape:=shpShape, effectId:=msoAnimEffectBlinds)  
  
'Adds animation behavior and sets rotation effect  
Set animBehavior = tmInShape.MainSequence(1).Behaviors _  
    .Add(Type:=msoAnimTypeRotation)  
Set rtnEffect = animBehavior.RotationEffect  
rtnEffect.By = 270  
End Sub
```



ByX Property

Sets or returns a **Single** that represents scaling or moving an object horizontally by a specified percentage of the screen width, depending on whether it used in conjunction with a [ScaleEffect](#) or [MotionEffect](#) object, respectively. For example, a value of 50 for a motion effect means to move the object half the screen width to the right. Read/write.

expression.**ByX**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Negative numbers move the object horizontally to the left. Floating point numbers (for example, 55.5) are allowed.

To scale or move an object vertically, use the [ByY](#) property.

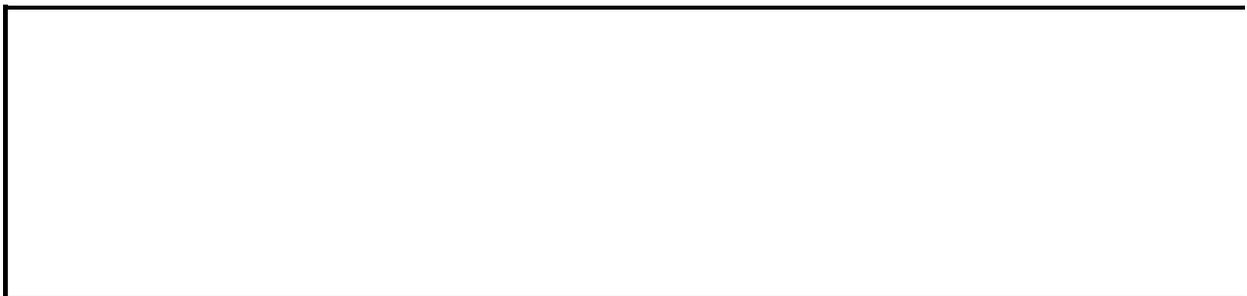
If both the **ByX** and **ByY** properties are set, then the object is scaled or moves both horizontally and vertically.

Do not confuse this property with the [By](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds an animation path; then sets the horizontal and vertical movement of the shape.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animBehavior As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds rectangle and sets effect and animation  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=300, _  
            Top:=300, Width:=300, Height:=150)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpRectangle, _  
            effectId:=msoAnimEffectCustom)  
    Set animBehavior = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Specifies animation motion  
    With animBehavior.MotionEffect  
        .ByX = 50  
        .ByY = 50  
    End With  
  
End Sub
```



ByY Property

Sets or returns a **Single** that represents scaling or moving an object vertically by a specified percentage of the screen width, depending on whether it is used in conjunction with a [ScaleEffect](#) or [MotionEffect](#) object, respectively.

Read/write.

expression.**ByY**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Negative numbers move the object horizontally to the left. Floating point numbers (for example, 55.5) are allowed.

To scale or move an object horizontally, use the [ByX](#) property.

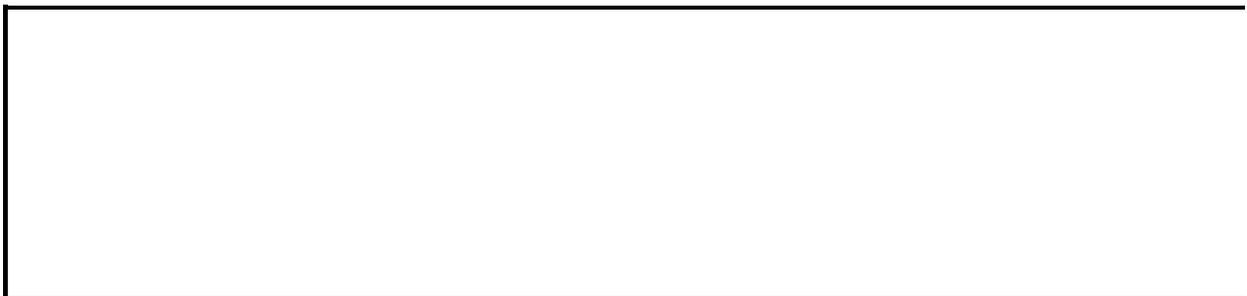
If both the **ByX** and **ByY** properties are set, then the object is scaled or moves both horizontally and vertically.

Do not confuse this property with the [By](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds an animation path; then sets the horizontal and vertical movement of the shape.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animBehavior As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds rectangle and sets effect and animation  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=300, _  
        Top:=300, Width:=300, Height:=150)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpRectangle, _  
        effectId:=msoAnimEffectCustom)  
    Set animBehavior = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Specifies animation motion  
    With animBehavior.MotionEffect  
        .ByX = 50  
        .ByY = 50  
    End With  
  
End Sub
```



Callout Property

Returns a [CalloutFormat](#) object that contains callout formatting properties for the specified shape. Applies to **Shape** or **ShapeRange** objects that represent line callouts. Read-only.

Example

This example adds to myDocument an oval and a callout that points to the oval. The callout text won't have a border, but it will have a vertical accent bar that separates the text from the callout line.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape msoShapeOval, 180, 200, 280, 130
    With .AddCallout(msoCalloutTwo, 420, 170, 170, 40)
        .TextFrame.TextRange.Text = "My oval"
        With .Callout
            .Accent = True
            .Border = False
        End With
    End With
End With
```



[Show All](#)

Caption Property

[Caption property as it applies to the **Application** object.](#)

Returns the text that appears in the title bar of the application window.
Read/write **String**.

expression.**Caption**

expression Required. An expression that returns one of the above objects.

[Caption property as it applies to the **DocumentWindow** object.](#)

Returns the text that appears in the title bar of the document window. Read-only
String.

expression.**Caption**

expression Required. An expression that returns one of the above objects.

Example

This example displays the caption for each open document window.

```
With Application.Windows
  For w = 1 To .Count
    MsgBox "Window " & w & " contains " & .Item(1).Caption
  Next
End With
```



Cells Property

Returns a [CellRange](#) collection that represents the cells in a table column or row. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example creates a new presentation, adds a slide, inserts a 3x3 table on the slide, and assigns the column and row number to each cell in the table.

```
Dim i As Integer
Dim j As Integer
With Presentations.Add
    .Slides.Add(1, ppLayoutBlank).Shapes.AddTable(3, 3).Select
    Set myTable = .Slides(1).Shapes(1).Table
    For i = 1 To myTable.Columns.Count
        For j = 1 To myTable.Columns(i).Cells.Count
            myTable.Columns(i).Cells(j).Shape.TextFrame _
                .TextRange.Text = "col. " & i & "row " & j
        Next j
    Next i
End With
```



Character Property

Returns or sets the Unicode character value that is used for bullets in the specified text. Read/write **Long**.

Example

This example sets the bullet character for shape two on slide one in the active presentation.

```
Set frame2 = ActivePresentation.Slides(1).Shapes(2).TextFrame
With frame2.TextRange.ParagraphFormat.Bullet
    .Character = 8226
    .Visible = True
End With
```



[Show All](#)

ChartUnitEffect Property

Returns or sets a value that indicates whether the graph range is animated by series, category, or element. Read/write [PpChartUnitEffect](#).

PpChartUnitEffect can be one of these PpChartUnitEffect constants.

ppAnimateByCategory

ppAnimateByCategoryElements

ppAnimateBySeries

ppAnimateBySeriesElements

ppAnimateChartAllAtOnce

ppAnimateChartMixed

expression.**ChartUnitEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If your graph doesn't become animated, make sure that the [Animate](#) property is set to **True**

Example

This example sets shape two on slide three in the active presentation to be animated by series. Shape two must be a graph for this to work.

```
With ActivePresentation.Slides(3).Shapes(2)
    With .AnimationSettings
        .ChartUnitEffect = ppAnimateBySeries
        .EntryEffect = ppEffectFlyFromLeft
        .Animate = True
    End With
End With
```



[Show All](#)

CheckIfOfficeIsHTMLEditor Property

Determines whether Microsoft PowerPoint checks to see whether an Office application is the default HTML editor when you start PowerPoint. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse PowerPoint does not check to see whether an Office application is the default HTML editor when you start PowerPoint.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. PowerPoint checks to see whether an Office application is the default HTML editor when you start PowerPoint.

Remarks

This property is used only if the Web browser you are using supports HTML editing and HTML editors.

To use a different HTML editor, you must set this property to **False** and then register the editor as the default system HTML editor.

Example

This example causes Microsoft PowerPoint not to check whether an Office application is the default HTML editor.

```
Application.DefaultWebOptions.CheckIfOfficeIsHTMLEditor = msoFalse
```



[Show All](#)

Child Property

MsoTrue if the shape is a child shape or if all shapes in a shape range are child shapes of the same parent. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue Does not apply to this property.

msoFalse The shape is not a child shape or, if a shape range, all child shapes do not belong to the same parent.

msoTriStateMixed Does not apply to this property.

msoTriStateToggle Does not apply to this property.

msoTrue The shape is a child shape or, if a shape range, all child shapes belong to the same parent.

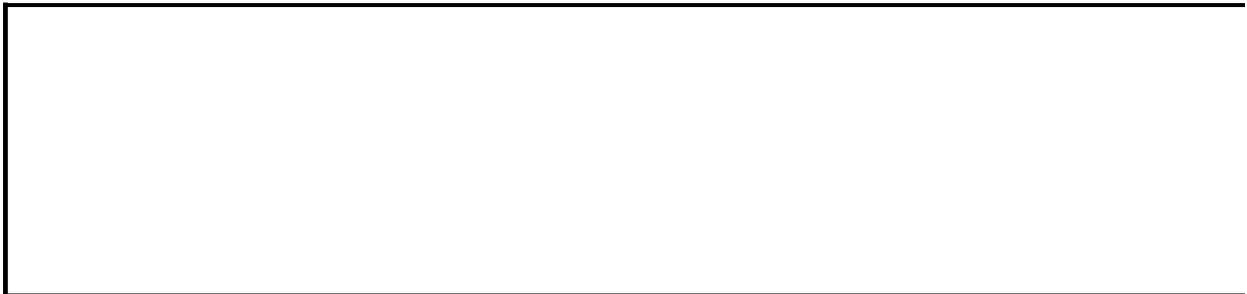
expression.**Child**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example selects the first shape in the canvas, and if the selected shape is a child shape, fills the shape with the specified color. This example assumes that the first shape in the active presentation is a drawing canvas that contains multiple shapes.

```
Sub FillChildShape()  
  
    'Select the first shape in the drawing canvas  
    ActivePresentation.Slides(1).Shapes(1).CanvasItems(1).Select  
  
    'Fill selected shape if it is a child shape  
    With ActiveWindow.Selection  
  
        If .ShapeRange.Child = msoTrue Then  
            .ShapeRange.Fill.ForeColor.RGB = RGB(Red:=100, Green:=0,  
            Else  
                MsgBox "This shape is not a child shape."  
            End If  
  
        End With  
  
    End Sub
```



Children Property

Returns a [DiagramNodeChildren](#) object that represents all of the children of the specified diagram node.

expression.**Children**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates a pyramid diagram and adds child nodes to it.

```
Sub CreatePyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional child nodes to diagram  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
End Sub
```



ChildShapeRange Property

Returns a [ShapeRange](#) object that represents the child shapes of a selection.

expression.**ChildShapeRange**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example creates a new document with a drawing canvas, populates the drawing canvas with shapes, and selects the shapes added to the canvas. Then after checking that the shapes selected are child shapes, it fills the child shapes with a pattern.

```
Sub ChildShapes()  
    Dim sldNew As Slide  
    Dim shpCanvas As Shape  
  
    'Create a new slide with a drawing canvas and shapes  
    Set sldNew = Presentations(1).Slides _  
        .Add(Index:=1, Layout:=ppLayoutBlank)  
    Set shpCanvas = sldNew.Shapes.AddCanvas( _  
        Left:=100, Top:=100, Width:=200, Height:=200)  
  
    With shpCanvas.CanvasItems  
        .AddShape msoShapeRectangle, Left:=0, Top:=0, _  
            Width:=100, Height:=100  
        .AddShape msoShapeOval, Left:=0, Top:=50, _  
            Width:=100, Height:=100  
        .AddShape msoShapeDiamond, Left:=0, Top:=100, _  
            Width:=100, Height:=100  
    End With  
  
    'Select all shapes in the canvas  
    shpCanvas.CanvasItems.SelectAll  
  
    'Fill canvas child shapes with a pattern  
    With ActiveWindow.Selection  
        If .HasChildShapeRange = True Then  
            .ChildShapeRange.Fill.Patterned Pattern:=msoPatternDivot  
        Else  
            MsgBox "This is not a range of child shapes."  
        End If  
    End With  
  
End Sub
```



[Show All](#)

Collate Property

Determines whether a complete copy of the specified presentation is printed before the first page of the next copy is printed. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Default. A complete copy of the specified presentation is printed before the first page of the next copy is printed.

Remarks

Specifying a value for the *Collate* argument of the [PrintOut](#) method sets the value of this property.

Example

This example prints three collated copies of the active presentation.

```
With ActivePresentation.PrintOptions
    .NumberOfCopies = 3
    .Collate = msoTrue
    .Parent.PrintOut
End With
```



Color Property

Returns a [ColorFormat](#) object that represents the color for the specified characters. Read-only.

Example

This example sets the color of characters one through five in the title on slide one.

```
myRed = RGB(255, 0, 0)
Set myT = Application.ActivePresentation.Slides(1).Shapes.Title
myT.TextFrame.TextRange.Characters(1, 5).Font.Color.RGB = myRed
```



Color2 Property

Returns a [ColorFormat](#) object that represents the color on which to end a color-cycle animation.

expression.**Color2**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, adds a fill animation to that shape, then reports the starting and ending fill colors.

```
Sub SetStartEndColors()  
  
    Dim effChangeFill As Effect  
    Dim shpCube As Shape  
    Dim a As AnimationBehavior  
  
    'Adds cube and set fill effect  
    Set shpCube = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeCube, Left:=300, _  
            Top:=300, Width:=100, Height:=100)  
    Set effChangeFill = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpCube, _  
            effectId:=msoAnimEffectChangeFillColor)  
  
    'Sets duration of effect and displays a message containing  
    'the starting and ending colors for the fill effect  
    effChangeFill.Timing.Duration = 3  
    MsgBox "Start Color = " & effChangeFill.EffectParameters _  
        .Color1 & vbCrLf & "End Color = " & effChangeFill _  
        .EffectParameters.Color2  
  
End Sub
```



ColorEffect Property

Returns a [ColorEffect](#) object that represents the color properties for a specified animation behavior.

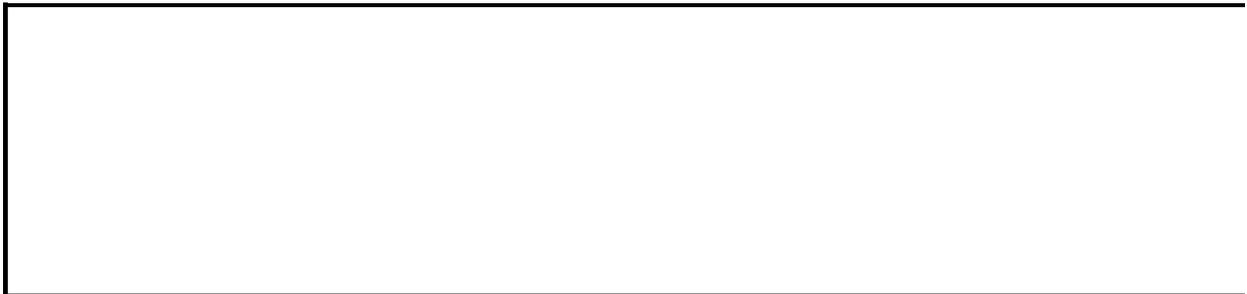
expression.**ColorEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a shape to the first slide of the active presentation and sets a color effect behavior to change the fill color of the new shape.

```
Sub ChangeColorEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(Type:=msoAnimTypeColor)  
  
    With bhvEffect.ColorEffect  
        .From.RGB = RGB(Red:=255, Green:=0, Blue:=0)  
        .To.RGB = RGB(Red:=0, Green:=0, Blue:=255)  
    End With  
End Sub
```



ColorScheme Property

Returns or sets the [ColorScheme](#) object that represents the scheme colors for the specified slide, slide range, or slide master. Read/write.

Example

This example sets the title color to green for slides one and three in the active presentation.

```
Set mySlides = ActivePresentation.Slides.Range(Array(1, 3))  
mySlides.ColorScheme.Colors(ppTitle).RGB = RGB(0, 255, 0)
```



ColorSchemes Property

Returns a [ColorSchemes](#) collection that represents the color schemes in the specified presentation. Read-only.

Example

This example sets the background color for color scheme three in the active presentation and then applies the color scheme to all slides in the presentation that are based on the slide master.

```
With ActivePresentation
  Set cs1 = .ColorSchemes(3)
  cs1.Colors(ppBackground).RGB = RGB(128, 128, 0)
  .SlideMaster.ColorScheme = cs1
End With
```



[Show All](#)

ColorType Property

Returns or sets the type of color transformation applied to the specified picture or OLE object. Read/write [MsoPictureColorType](#).

MsoPictureColorType can be one of these MsoPictureColorType constants.

msoPictureAutomatic

msoPictureBlackAndWhite

msoPictureGrayscale

msoPictureMixed

msoPictureWatermark

expression.**ColorType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the color transformation to grayscale for shape one on myDocument. Shape one must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).PictureFormat.ColorType = msoPictureGrayscale
```



Columns Property

Returns a [Columns](#) collection that represents all the columns in a table. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example displays the shape number, the slide number, and the number of columns in the first table of the active presentation.

```
Dim ColCount As Integer
Dim sl As Integer
Dim sh As Integer

With ActivePresentation
    For sl = 1 To .Slides.Count
        For sh = 1 To .Slides(sl).Shapes.Count
            If .Slides(sl).Shapes(sh).HasTable Then
                ColCount = .Slides(sl).Shapes(sh) _
                    .Table.Columns.Count
                MsgBox "Shape " & sh & " on slide " & sl & _
                    " contains the first table and has " & _
                    ColCount & " columns."
            Exit Sub
        End If
    Next
Next
End With
```

This example tests the selected shape to see if it contains a table. If it does, the code sets the width of column one to 72 points (one inch).

```
With ActiveWindow.Selection.ShapeRange
    If .HasTable = True Then
        .Table.Columns(1).Width = 72
    End If
End With
```



COMAddIns Property

Returns a reference to the Component Object Model (COM) add-ins currently loaded in Microsoft PowerPoint. These are listed in the **COM Add-Ins** dialog box. You can add the **COM Add-Ins** command to your **Tools** menu by using the **Customize** dialog box. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).



Command Property

Sets or returns a **String** that represents the command to be executed for the command effect. Read/write.

expression.**Command**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

You can send OLE verbs to embedded objects using this property.

If the shape is an OLE object, then the ole object will execute the command if it understands the verb.

If the shape is a media object (sound/video), PowerPoint understands the following verbs: play, stop, pause, togglepause, resume and playfrom. Any other command sent to the shape will be ignored.

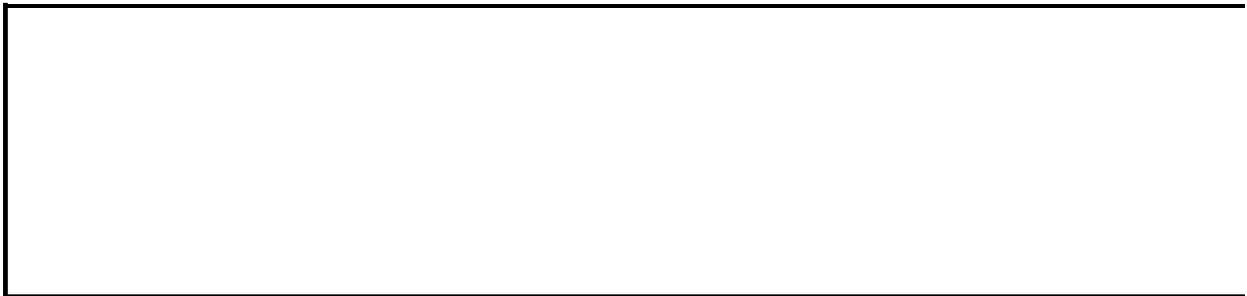
Example

The following example shows how to set a command effect animation behavior.

```
Set bhvEffect = effectNew.Behaviors.Add(msoAnimTypeCommand)
```

```
With bhvEffect.CommandEffect  
    .Type = msoAnimCommandTypeVerb  
    .Command = Play
```

```
End With
```



[Show All](#)

CommandBars Property

 [CommandBars property as it applies to the **Application** object.](#)

Returns a **CommandBars** collection that represents all the command bars in PowerPoint. Read-only.

 [CommandBars property as it applies to the **Presentation** object.](#)

Returns a **CommandBars** collection that represents the merged command bar set from the host container application and PowerPoint. This property returns a valid object only when the container is a DocObject server, like Microsoft Binder, and PowerPoint is acting as an OLE server. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

[As it applies to the **Application** object.](#)

This example enlarges all command bar buttons and enables ToolTips.

```
With Application.CommandBars  
    .LargeButtons = True  
    .DisplayTooltips = True  
End With
```

[As it applies to the **Presentation** object.](#)

This example displays the **Formatting** command bar with the merged command bar set at the top of the application window.

```
With ActivePresentation.CommandBars("Formatting")  
    .Visible = True  
    .Position = msoBarTop  
End With
```



CommandEffect Property

Returns a **CommandEffect** object for the specified animation behavior. Read-only.

expression.**CommandEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

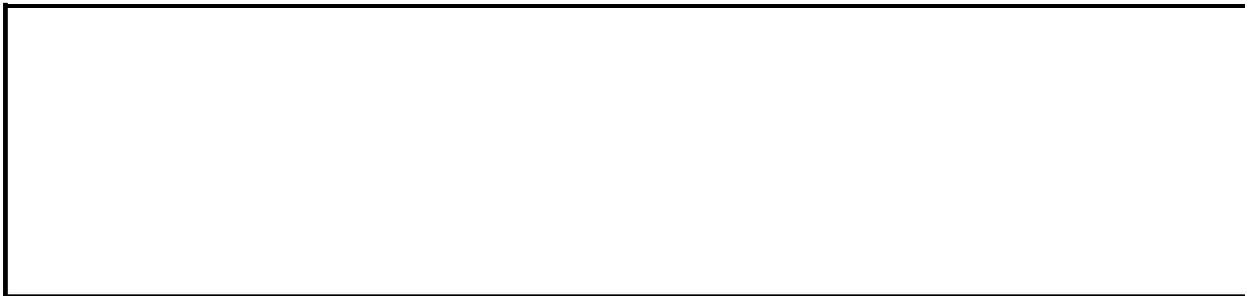
You can send events, call functions, and send OLE verbs to embedded objects using this property.

Example

The following example shows how to set a command effect animation behavior.

```
Set bhvEffect = effectNew.Behaviors.Add(msoAnimTypeCommand)
```

```
With bhvEffect.CommandEffect  
    .Type = msoAnimCommandTypeVerb  
    .Command = Play  
End With
```



Comments Property

Returns a [Comments](#) object that represents a collection of comments.

expression.**Comments**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a comment to a slide.

```
Sub AddNewComment()  
    ActivePresentation.Slides(1).Comments.Add _  
        Left:=0, Top:=0, Author:="John Doe", AuthorInitials:="jd", _  
        Text:="Please check this spelling again before the next draf  
End Sub
```



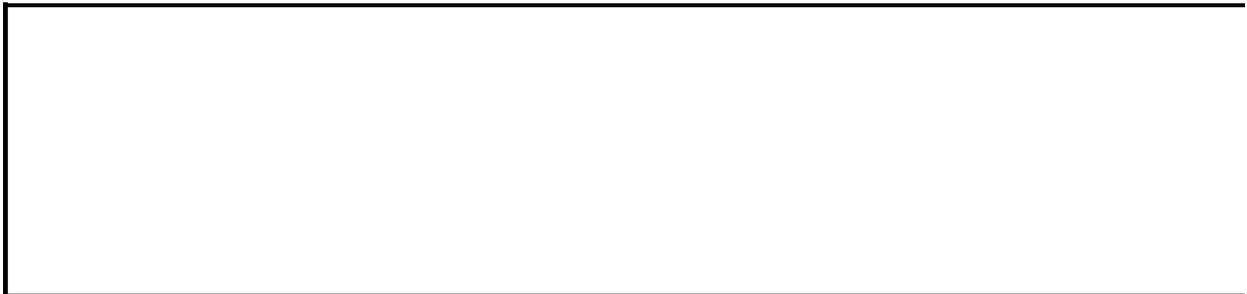
ConnectionSiteCount Property

Returns the number of connection sites on the specified shape. Read-only **Long**.

Example

This example adds two rectangles to myDocument and joins them with two connectors. The beginnings of both connectors attach to connection site one on the first rectangle; the ends of the connectors attach to the first and last connection sites of the second rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
lastsite = secondRect.ConnectionSiteCount
With s.AddConnector(msoConnectorCurve, 0, 0, 100, 100) _
    .ConnectorFormat
        .BeginConnect ConnectedShape:=firstRect, ConnectionSite:=1
        .EndConnect ConnectedShape:=secondRect, ConnectionSite:=1
End With
With s.AddConnector(msoConnectorCurve, 0, 0, 100, 100) _
    .ConnectorFormat
        .BeginConnect ConnectedShape:=firstRect, ConnectionSite:=1
        .EndConnect ConnectedShape:=secondRect, _
            ConnectionSite:=lastsite
End With
```



[Show All](#)

Connector Property

Determines whether the specified shape is a connector. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is a connector.

Example

This example deletes all connectors on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
  For i = .Count To 1 Step -1
    With .Item(i)
      If .Connector Then .Delete
    End With
  Next
End With
```



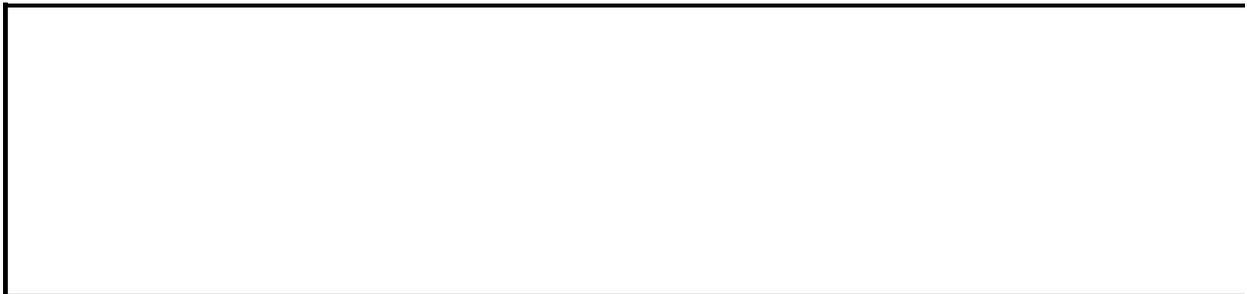
ConnectorFormat Property

Returns a [ConnectorFormat](#) object that contains connector formatting properties. Applies to **Shape** or **ShapeRange** objects that represent connectors. Read-only.

Example

This example adds two rectangles to myDocument, attaches them with a connector, automatically reroutes the connector along the shortest path, and then detaches the connector from the rectangles.

```
Set myDocument = ActivePresentation.Slides(1)
Set s = myDocument.Shapes
Set firstRect = s.AddShape(msoShapeRectangle, 100, 50, 200, 100)
Set secondRect = s.AddShape(msoShapeRectangle, 300, 300, 200, 100)
With s.AddConnector(msoConnectorCurve, 0, 0, 0, 0).ConnectorFormat
    .BeginConnect firstRect, 1
    .EndConnect secondRect, 1
    .Parent.RerouteConnections
    .BeginDisconnect
    .EndDisconnect
End With
```



Container Property

Returns the object that contains the specified embedded presentation. Read-only **Object**.

Note If the container doesn't support OLE Automation, or if the specified presentation isn't embedded in a Microsoft Binder file, this property fails.

Example

This example hides the second section of the Microsoft Binder file that contains the embedded active presentation. The **Container** property of the presentation returns a **Section** object, and the **Parent** property of the **Section** object returns a **Binder** object.

```
Application.ActivePresentation.Container.Parent.Sections(2) _  
    .Visible = False
```



Contrast Property

Returns or sets the contrast for the specified picture or OLE object. The value for this property must be a number from 0.0 (the least contrast) to 1.0 (the greatest contrast). Read/write **Single**.

Example

This example sets the contrast for shape one on myDocument. Shape one must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).PictureFormat.Contrast = 0.8
```



Count Property

Returns the number of objects in the specified collection.

Read-only **Integer** for the following objects: **Adjustments**, **CanvasShapes**, **DiagramNodeChildren**, **DiagramNodes**, **GroupShapes**, **ShapeNodes**, and **Shapes**.

Read-only **Long** for the following objects: **ActionSettings**, **AddIns**, **AnimationPoints**, **Animations10**, **Borders**, **CellRange**, **ColorScheme**, **ColorSchemes**, **Columns**, **Comments**, **Designs**, **DocumentWindows**, **ExtraColors**, **Fonts**, **Hyperlinks**, **NamedSlideShow**, **NamedSlideShows**, **ObjectVerbs**, **Panes**, **Placeholders**, **Presentations**, **PrintRanges**, **PublishObjects**, **Rows**, **RulerLevels**, **ShapeRange**, **SlideRange**, **Slides**, **SlideShowWindows**, **TabStops**, **Tags**, **TextRange**, **TextStyleLevels**, **TextStyles**, **TimeNodes**, and **Triggers**.

Example

This example closes all windows except the active window.

```
With Application.Windows
  For i = 2 To .Count
    .Item(2).Close
  Next
End With
```



Creator Property

Returns a **Long** that represents the four-character creator code for the application in which the specified object was created. For example, if the object was created in PowerPoint, this property returns the hexadecimal number 50575054. Read-only.

expression.**Creator**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The **Creator** property is designed to be used in Microsoft Office applications for the Macintosh.

Example

This example displays a message about the creator of myObject.

```
Set myObject = Application.ActivePresentation.Slides(1).Shapes(1)
If myObject.Creator = &h50575054 Then
    MsgBox "This is a PowerPoint object"
Else
    MsgBox "This is not a PowerPoint object"
End If
```



CropBottom Property

Returns or sets the number of points that are cropped off the bottom of the specified picture or OLE object. Read/write **Single**.

Note Cropping is calculated relative to the original size of the picture. For example, if you insert a picture that is originally 100 points high, rescale it so that it's 200 points high, and then set the **CropBottom** property to 50, 100 points (not 50) will be cropped off the bottom of your picture.

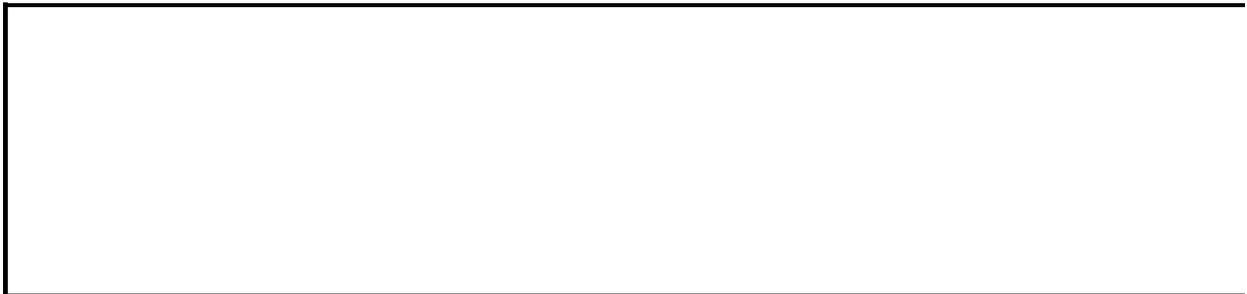
Example

This example crops 20 points off the bottom of shape three on myDocument. For the example to work, shape three must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).PictureFormat.CropBottom = 20
```

This example crops the percentage specified by the user off the bottom of the selected shape, regardless of whether the shape has been scaled. For the example to work, the selected shape must be either a picture or an OLE object.

```
percentToCrop = InputBox("What percentage do you " & _
    "want to crop off the bottom of this picture?")
Set shapeToCrop = ActiveWindow.Selection.ShapeRange(1)
With shapeToCrop.Duplicate
    .ScaleHeight 1, True
    origHeight = .Height
    .Delete
End With
cropPoints = origHeight * percentToCrop / 100
shapeToCrop.PictureFormat.CropBottom = cropPoints
```



CropLeft Property

Returns or sets the number of points that are cropped off the left side of the specified picture or OLE object. Read/write **Single**.

Note Cropping is calculated relative to the original size of the picture. For example, if you insert a picture that is originally 100 points wide, rescale it so that it's 200 points wide, and then set the **CropLeft** property to 50, 100 points (not 50) will be cropped off the left side of your picture.

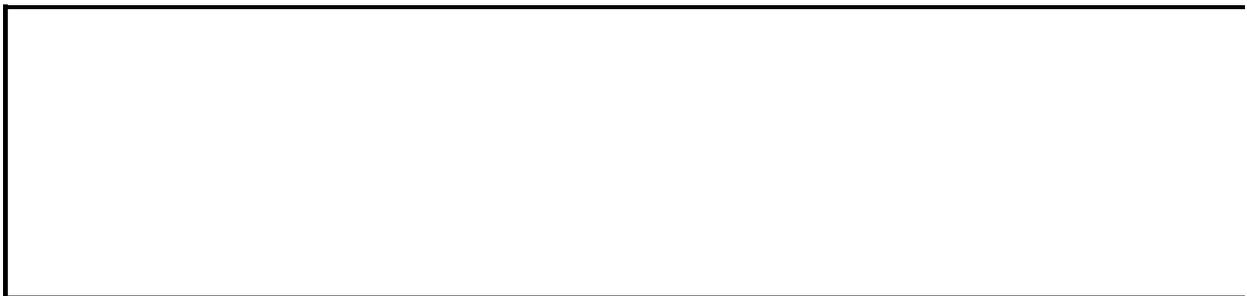
Example

This example crops 20 points off the left side of shape three on myDocument. For the example to work, shape three must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).PictureFormat.CropLeft = 20
```

This example crops the percentage specified by the user off the left side of the selected shape, regardless of whether the shape has been scaled. For the example to work, the selected shape must be either a picture or an OLE object.

```
percentToCrop = InputBox("What percentage do you " & _
    "want to crop off the left of this picture?")
Set shapeToCrop = ActiveWindow.Selection.ShapeRange(1)
With shapeToCrop.Duplicate
    .ScaleWidth 1, True
    .Width = origWidth
    .Delete
End With
cropPoints = origWidth * percentToCrop / 100
shapeToCrop.PictureFormat.CropLeft = cropPoints
```



CropRight Property

Returns or sets the number of points that are cropped off the right side of the specified picture or OLE object. Read/write **Single**.

Note Cropping is calculated relative to the original size of the picture. For example, if you insert a picture that is originally 100 points wide, rescale it so that it's 200 points wide, and then set the **CropRight** property to 50, 100 points (not 50) will be cropped off the right side of your picture.

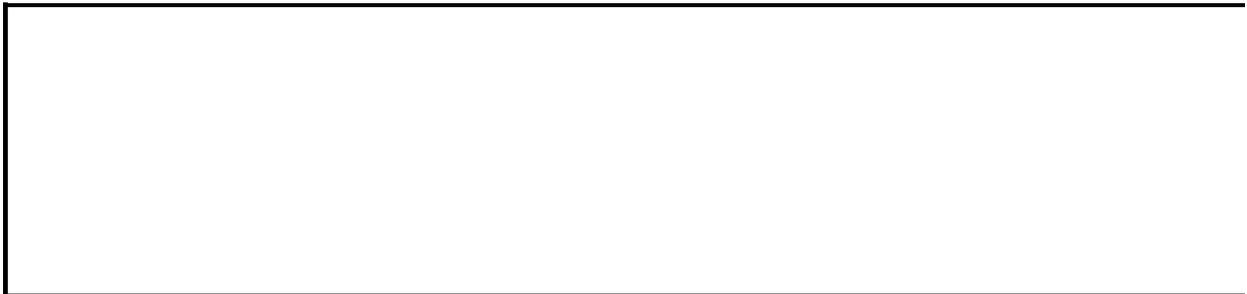
Example

This example crops 20 points off the right side of shape three on myDocument. For this example to work, shape three must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).PictureFormat.CropRight = 20
```

This example crops the percentage specified by the user off the right side of the selected shape, regardless of whether the shape has been scaled. For the example to work, the selected shape must be either a picture or an OLE object.

```
percentToCrop = InputBox("What percentage do you " & _
    "want to crop off the right of this picture?")
Set shapeToCrop = ActiveWindow.Selection.ShapeRange(1)
With shapeToCrop.Duplicate
    .ScaleWidth 1, True
    origWidth = .Width
    .Delete
End With
cropPoints = origWidth * percentToCrop / 100
shapeToCrop.PictureFormat.CropRight = cropPoints
```



CropTop Property

Returns or sets the number of points that are cropped off the top of the specified picture or OLE object. Read/write **Single**.

Note Cropping is calculated relative to the original size of the picture. For example, if you insert a picture that is originally 100 points high, rescale it so that it's 200 points high, and then set the **CropTop** property to 50, 100 points (not 50) will be cropped off the top of your picture.

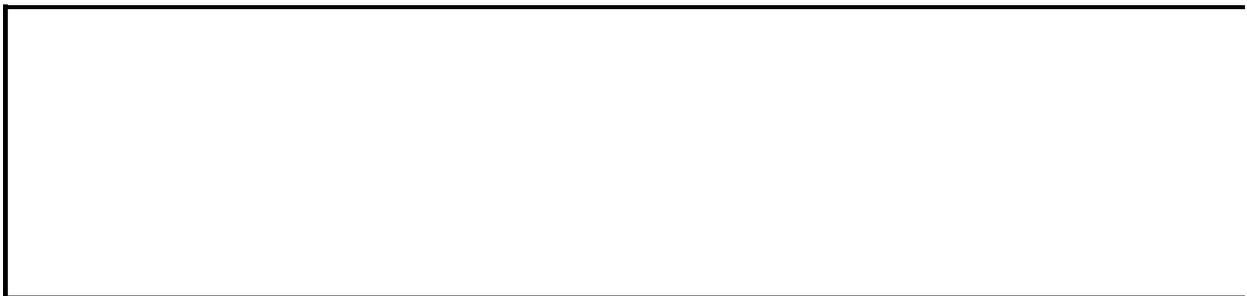
Example

This example crops 20 points off the top of shape three on myDocument. For the example to work, shape three must be either a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).PictureFormat.CropTop = 20
```

This example crops the percentage specified by the user off the top of the selected shape, regardless of whether the shape has been scaled. For the example to work, the selected shape must be either a picture or an OLE object.

```
percentToCrop = InputBox("What percentage do you " & _
    "want to crop off the top of this picture?")
Set shapeToCrop = ActiveWindow.Selection.ShapeRange(1)
With shapeToCrop.Duplicate
    .ScaleHeight 1, True
    origHeight = .Height
    .Delete
End With
cropPoints = origHeight * percentToCrop / 100
shapeToCrop.PictureFormat.CropTop = cropPoints
```



CurrentShowPosition Property

Returns the position of the current slide within the slide show that is showing in the specified view. Read-only **Long**.

Remarks

If the specified view contains a custom show, the **CurrentShowPosition** property returns the position of the current slide within the custom show, not the position of the current slide within the entire presentation.

Example

This example sets a variable to the position of the current slide in the slide show running in slide show window one.

```
lastSlideSeen = SlideShowWindows(1).View.CurrentShowPosition
```



CustomDocumentProperties Property

Returns a **DocumentProperties** collection that represents all the custom document properties for the specified presentation. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Remarks

Use the [BuiltInDocumentProperties](#) property to return the collection of built-in document properties.

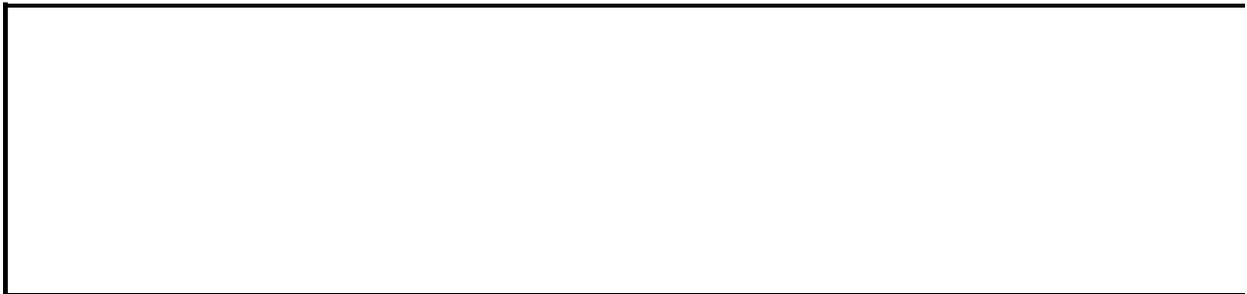
Example

This example adds a static custom property named "Complete" for the active presentation.

```
Application.ActivePresentation.CustomDocumentProperties _  
    .Add Name:="Complete", LinkToContent:=False, _  
    Type:=msoPropertyTypeBoolean, Value:=False
```

This example prints out the active presentation if the value of the "Complete" custom property is **True**.

```
With Application.ActivePresentation  
    If .CustomDocumentProperties("complete") Then .PrintOut  
End With
```



[Show All](#)

DashStyle Property

Returns or sets the dash style for the specified line. Read/write [MsoLineDashStyle](#).

MsoLineDashStyle can be one of these MsoLineDashStyle constants.

msoLineDash

msoLineDashDot

msoLineDashDotDot

msoLineDashStyleMixed

msoLineLongDash

msoLineLongDashDot

msoLineRoundDot

msoLineSolid

msoLineSquareDot

expression.DashStyle

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a blue dashed line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(50, 0, 128)
End With
```



DateAndTime Property

Returns a [HeaderFooter](#) object that represents the date and time item that appears in the lower-left corner of a slide or in the upper-right corner of a notes page, handout, or outline. Read-only.

Example

This example sets the date and time format for the slide master in the active presentation. This setting will apply to all slides that are based on this master.

```
Set myPres = Application.ActivePresentation
With myPres.SlideMaster.HeadersFooters.DateAndTime
    .Format = ppDateTimeMdyy
    .UseFormat = True
End With
```



DateTime Property

Returns a **Date** representing the date and time a comment was created.

expression.**DateTime**

expression Required. An expression that returns a [Comment](#) object.

Remarks

Don't confuse this property with the [DateAndTime](#) property, which applies to the headers and footers of a slide.

Example

The following example provides information about all the comments for a given slide.

```
Sub ListComments()  
    Dim cmtExisting As Comment  
    Dim strAuthorInfo As String  
  
    For Each cmtExisting In ActivePresentation.Slides(1).Comments  
        With cmtExisting  
            strAuthorInfo = strAuthorInfo & ".Author & "'s comment #"  
                .AuthorIndex & " (" & .Text & ") was created on " &  
                .DateTime & vbCrLf  
        End With  
    Next  
  
    If strAuthorInfo <> "" Then  
        MsgBox strAuthorInfo  
    Else  
        MsgBox "There are no comments on this slide."  
    End If  
End Sub
```



Decelerate Property

Sets or returns a **Single** that represents the percent of the duration over which a timing deceleration should take place. For example, a value of 0.9 means that an deceleration should start at the default speed, and then start to slow down after the first ten percent of the animation. Read/write.

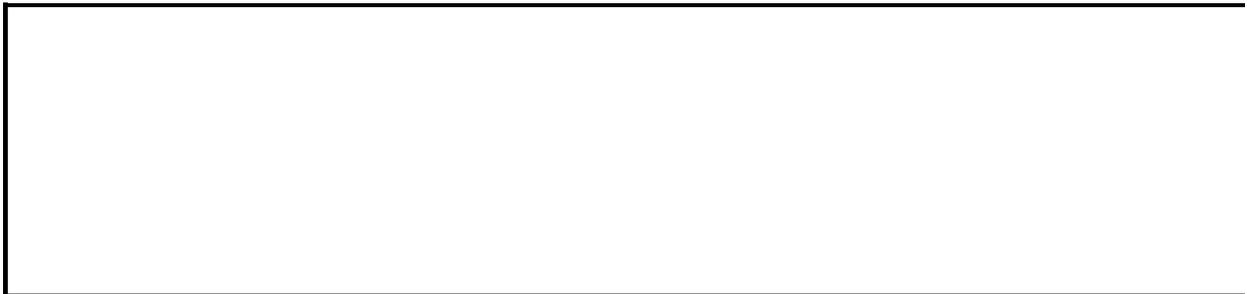
expression.**Decelerate**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a shape and adds an animation that starts at the default speed and slows down after 70% of the animation has finished.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds rectangle and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpRectangle, _  
        effectId:=msoAnimEffectPathDiamond)  
  
    'Slows the effect after seventy percent of the animation has fin  
    With effDiamond.Timing  
        .Decelerate = 0.3  
    End With  
  
End Sub
```



[Show All](#)

DefaultLanguageID Property

Returns or sets the default language of a presentation. When you set the **DefaultLanguageID** property for a presentation, you set it for all subsequent new presentations as well. Read/write [MsoLanguageID](#).

MsoLanguageID can be one of these MsoLanguageID constants.

msoLanguageIDAfrikaans

msoLanguageIDAlbanian

msoLanguageIDAmharic

msoLanguageIDArabic

msoLanguageIDArabicAlgeria

msoLanguageIDArabicBahrain

msoLanguageIDArabicEgypt

msoLanguageIDArabicIraq

msoLanguageIDArabicJordan

msoLanguageIDArabicKuwait

msoLanguageIDArabicLebanon

msoLanguageIDArabicLibya

msoLanguageIDArabicMorocco

msoLanguageIDArabicOman

msoLanguageIDArabicQatar

msoLanguageIDArabicSyria

msoLanguageIDArabicTunisia

msoLanguageIDArabicUAE

msoLanguageIDArabicYemen

msoLanguageIDArmenian

msoLanguageIDAssamese

msoLanguageIDAzeriCyrillic

msoLanguageIDAzeriLatin

msoLanguageIDBasque

msoLanguageIDBelgianDutch

msoLanguageIDBelgianFrench
msoLanguageIDBengali
msoLanguageIDBrazilianPortuguese
msoLanguageIDBulgarian
msoLanguageIDBurmese
msoLanguageIDByelorussian
msoLanguageIDCatalan
msoLanguageIDCherokee
msoLanguageIDChineseHongKong
msoLanguageIDChineseMacao
msoLanguageIDChineseSingapore
msoLanguageIDCroatian
msoLanguageIDCzech
msoLanguageIDDanish
msoLanguageIDDutch
msoLanguageIDEnglishAUS
msoLanguageIDEnglishBelize
msoLanguageIDEnglishCanadian
msoLanguageIDEnglishCaribbean
msoLanguageIDEnglishIreland
msoLanguageIDEnglishJamaica
msoLanguageIDEnglishNewZealand
msoLanguageIDEnglishPhilippines
msoLanguageIDEnglishSouthAfrica
msoLanguageIDEnglishTrinidad
msoLanguageIDEnglishUK
msoLanguageIDEnglishUS
msoLanguageIDEnglishZimbabwe
msoLanguageIDEstonian
msoLanguageIDFaeroese
msoLanguageIDFarsi
msoLanguageIDFinnish
msoLanguageIDFrench

msoLanguageIDFrenchCameroon
msoLanguageIDFrenchCanadian
msoLanguageIDFrenchCotedIvoire
msoLanguageIDFrenchLuxembourg
msoLanguageIDFrenchMali
msoLanguageIDFrenchMonaco
msoLanguageIDFrenchReunion
msoLanguageIDFrenchSenegal
msoLanguageIDFrenchWestIndies
msoLanguageIDFrenchZaire
msoLanguageIDFrisianNetherlands
msoLanguageIDGaelicIreland
msoLanguageIDGaelicScotland
msoLanguageIDGalician
msoLanguageIDGeorgian
msoLanguageIDGerman
msoLanguageIDGermanAustria
msoLanguageIDGermanLiechtenstein
msoLanguageIDGermanLuxembourg
msoLanguageIDGreek
msoLanguageIDGujarati
msoLanguageIDHebrew
msoLanguageIDHindi
msoLanguageIDHungarian
msoLanguageIDIcelandic
msoLanguageIDIndonesian
msoLanguageIDInuktitut
msoLanguageIDItalian
msoLanguageIDJapanese
msoLanguageIDKannada
msoLanguageIDKashmiri
msoLanguageIDKazakh
msoLanguageIDKhmer

msoLanguageIDKirghiz
msoLanguageIDKonkani
msoLanguageIDKorean
msoLanguageIDLao
msoLanguageIDLatvian
msoLanguageIDLithuanian
msoLanguageIDMacedonian
msoLanguageIDMalayalam
msoLanguageIDMalayBruneiDarussalam
msoLanguageIDMalaysian
msoLanguageIDMaltese
msoLanguageIDManipuri
msoLanguageIDMarathi
msoLanguageIDMexicanSpanish
msoLanguageIDMixed
msoLanguageIDMongolian
msoLanguageIDNepali
msoLanguageIDNone
msoLanguageIDNoProofing
msoLanguageIDNorwegianBokmol
msoLanguageIDNorwegianNynorsk
msoLanguageIDOriya
msoLanguageIDPolish
msoLanguageIDPunjabi
msoLanguageIDRhaetoRomanic
msoLanguageIDRomanian
msoLanguageIDRomanianMoldova
msoLanguageIDRussian
msoLanguageIDRussianMoldova
msoLanguageIDSamiLappish
msoLanguageIDSanskrit
msoLanguageIDSerbianCyrillic
msoLanguageIDSerbianLatin

msoLanguageIDSesotho
msoLanguageIDSimplifiedChinese
msoLanguageIDSindhi
msoLanguageIDSlovak
msoLanguageIDSlovenian
msoLanguageIDSorbian
msoLanguageIDSpanish
msoLanguageIDSpanishArgentina
msoLanguageIDSpanishBolivia
msoLanguageIDSpanishChile
msoLanguageIDSpanishColombia
msoLanguageIDSpanishCostaRica
msoLanguageIDSpanishDominicanRepublic
msoLanguageIDSpanishEcuador
msoLanguageIDSpanishElSalvador
msoLanguageIDSpanishGuatemala
msoLanguageIDSpanishHonduras
msoLanguageIDSpanishModernSort
msoLanguageIDSpanishNicaragua
msoLanguageIDSpanishPanama
msoLanguageIDSpanishParaguay
msoLanguageIDSpanishPeru
msoLanguageIDSpanishPuertoRico
msoLanguageIDSpanishUruguay
msoLanguageIDSpanishVenezuela
msoLanguageIDSutu
msoLanguageIDSwahili
msoLanguageIDSwedish
msoLanguageIDSwedishFinland
msoLanguageIDSwissFrench
msoLanguageIDSwissGerman
msoLanguageIDSwissItalian
msoLanguageIDTajik

msoLanguageIDTamil
msoLanguageIDTatar
msoLanguageIDTelugu
msoLanguageIDThai
msoLanguageIDTibetan
msoLanguageIDTraditionalChinese
msoLanguageIDTsonga
msoLanguageIDTswana
msoLanguageIDTurkish
msoLanguageIDTurkmen
msoLanguageIDUkrainian
msoLanguageIDUrdu
msoLanguageIDUzbekCyrillic
msoLanguageIDUzbekLatin
msoLanguageIDVenda
msoLanguageIDVietnamese
msoLanguageIDWelsh
msoLanguageIDXhosa
msoLanguageIDZulu
msoLanguageIDPortuguese

expression.**DefaultLanguageID**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

You can use the [LanguageID](#) property to set text ranges to different languages. Any text range not explicitly set to another language will be set to this default.

Example

This example sets the default language for the active presentation, and all subsequent new presentations, to German.

```
ActivePresentation.DefaultLanguageID = msoLanguageIDGerman
```



DefaultShape Property

Returns a [Shape](#) object that represents the default shape for the presentation.
Read-only.

Example

This example adds a shape to slide one in the active presentation, sets the default fill color to red for shapes in the presentation, and then adds another shape. This second shape will automatically have the new default fill color applied to it.

```
With Application.ActivePresentation
  Set sld1Shapes = .Slides(1).Shapes
  sld1Shapes.AddShape msoShape16pointStar, 20, 20, 100, 100
  .DefaultShape.Fill.ForeColor.RGB = RGB(255, 0, 0)
  sld1Shapes.AddShape msoShape16pointStar, 150, 20, 100, 100
End With
```



DefaultSpacing Property

Returns or sets the default tab-stop spacing for the specified text, in points.
Read/write **Single**.

Example

This example sets the default tab-stop spacing to 0.5 inch (36 points) for the text in shape two on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes(2).TextFrame _  
    .Ruler.TabStops.DefaultSpacing = 36
```



DefaultWebOptions Property

Returns the [DefaultWebOptions](#) object, which contains global application-level attributes used by Microsoft PowerPoint when you publish or save a complete or partial presentation as a Web page or open a Web page. Read-only.

Example

This example checks to see whether the default document encoding is Western. If it is, the string `strDocEncoding` is set accordingly.

```
Set objAppWebOptions = Application.DefaultWebOptions
With objAppWebOptions
    If .Encoding = msoEncodingWestern Then
        strDocEncoding = "Western"
    End If
End With
```



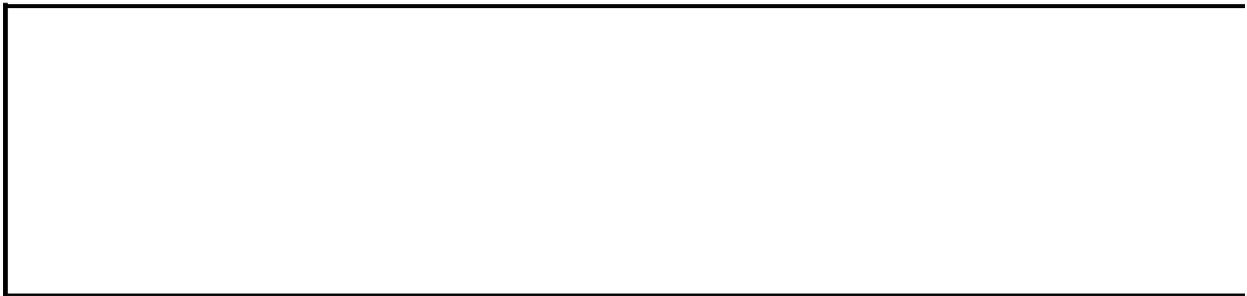
Depth Property

Returns or sets the depth of the shape's extrusion. Can be a value from – 600 through 9600 (positive values produce an extrusion whose front face is the original shape; negative values produce an extrusion whose back face is the original shape). Read/write **Single**.

Example

This example adds an oval to myDocument, and then specifies that the oval be extruded to a depth of 50 points and that the extrusion be purple.

```
Set myDocument = ActivePresentation.Slides(1)
Set myShape = myDocument.Shapes _
    .AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    'RGB value for purple
    .ExtrusionColor.RGB = RGB(255, 100, 255)
End With
```



Design Property

Returns a **Design** object representing a design.

expression.**Design**

expression Required. An expression that returns one of the above objects.

Example

The following example adds a title master.

```
Sub AddDesignMaster
    ActivePresentation.Slides(1).Design.AddTitleMaster
End Sub
```



Designs Property

Returns a [Designs](#) object, representing a collection of designs.

expression.**Designs**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example displays a message for each design in the active presentation.

```
Sub AddDesignMaster()  
    Dim desName As Design  
  
    With ActivePresentation  
  
        For Each desName In .Designs  
            MsgBox "The design name is " & .Designs.Item(desName.Index)  
        Next  
  
    End With  
  
End Sub
```



Diagram Property

Returns a **Diagram** object to which a diagram node belongs.

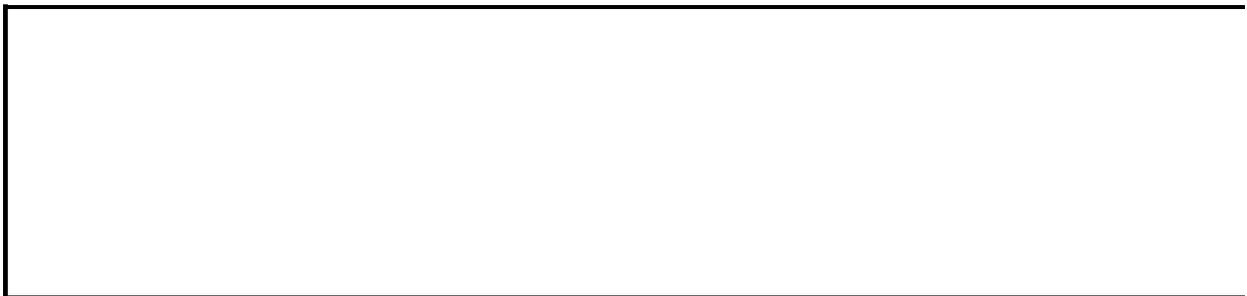
expression.**Diagram**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a diagram to a slide.

```
Sub AddADiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramCycle, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional child nodes  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically formats the diagram  
    dgnNode.Diagram.AutoFormat = msoTrue  
  
End Sub
```



DiagramNode Property

Returns a [DiagramNode](#) object that represents a node in a diagram.

expression.**DiagramNode**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a pyramid chart to the first slide in the active presentation.

```
Sub CreatePyramidDiagram()  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intCount As Integer  
  
    'Add pyramid diagram to current document  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
  
    'Add first diagram node child  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Add three more diagram child nodes  
    For intCount = 1 To 3  
        dgnNode.AddNode  
    Next intCount  
End Sub
```



Dim Property

Returns a [ColorFormat](#) object that represents the color to dim to after an animation is finished.

expression.**Dim**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the color to dim to after the animation.

```
Sub QueryDimColor()  
    Dim effDim As Effect  
    Set effDim = ActivePresentation.Slides(1).TimeLine.MainSequence(1)  
    MsgBox effDim.EffectInformation.Dim  
End Sub
```



DimColor Property

Returns or sets a [ColorFormat](#) object that represents the color of the specified shape after it's been built. Read-only.

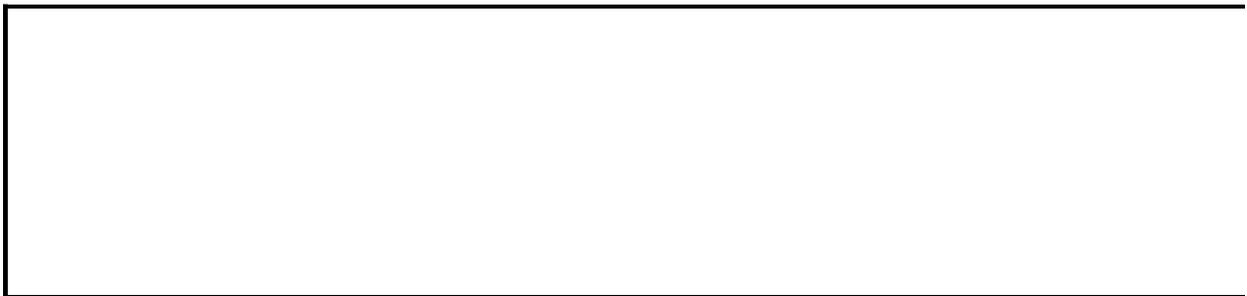
Remarks

If you don't get the effect you expect, check your other build settings. You won't see the effect of the **DimColor** property unless the [TextLevelEffect](#) property of the **AnimationSettings** object is set to something other than **ppAnimateLevelNone**, the [AfterEffect](#) property is set to **ppAfterEffectDim**, and the [Animate](#) property is set to **True**. In addition, if the specified shape is the only item or the last item to be built on the slide, the shape won't be dimmed. To change the build order of the shapes on a slide, use the [AnimationOrder](#) property.

Example

This example adds a slide that contains both a title and a three-item list to the active presentation, sets the title and list to be dimmed after being built, and sets the color that each of them will be dimmed to.

```
With ActivePresentation.Slides.Add(2, ppLayoutText).Shapes
  With .Item(1)
    .TextFrame.TextRange.Text = "Sample title"
    With .AnimationSettings
      .TextLevelEffect = ppAnimateByAllLevels
      .AfterEffect = ppAfterEffectDim
      .DimColor.SchemeColor = ppShadow
      .Animate = True
    End With
  End With
End With
With .Item(2)
  .TextFrame.TextRange.Text = "Item one" _
    & Chr(13) & "Item two"
  With .AnimationSettings
    .TextLevelEffect = ppAnimateByFirstLevel
    .AfterEffect = ppAfterEffectDim
    .DimColor.RGB = RGB(100, 150, 130)
    .Animate = True
  End With
End With
End With
```



[Show All](#)

Direction Property

Returns an [MsoAnimDirection](#) that represents the direction used for an animation effect. This property can be used only if the effect uses a direction. Read/write.

MsoAnimDirection can be one of these MsoAnimDirection constants.

msoAnimDirectionAcross

msoAnimDirectionBottom

msoAnimDirectionBottomLeft

msoAnimDirectionBottomRight

msoAnimDirectionCenter

msoAnimDirectionClockwise

msoAnimDirectionCounterclockwise

msoAnimDirectionCycleClockwise

msoAnimDirectionCycleCounterclockwise

msoAnimDirectionDown

msoAnimDirectionDownLeft

msoAnimDirectionDownRight

msoAnimDirectionFontAllCaps

msoAnimDirectionFontBold

msoAnimDirectionFontItalic

msoAnimDirectionFontShadow

msoAnimDirectionFontStrikethrough

msoAnimDirectionFontUnderline

msoAnimDirectionGradual

msoAnimDirectionHorizontal

msoAnimDirectionHorizontalIn

msoAnimDirectionHorizontalOut

msoAnimDirectionIn

msoAnimDirectionInBottom

msoAnimDirectionInCenter

msoAnimDirectionInSlightly
msoAnimDirectionInstant
msoAnimDirectionLeft
msoAnimDirectionNone
msoAnimDirectionOrdinalMask
msoAnimDirectionOut
msoAnimDirectionOutBottom
msoAnimDirectionOutCenter
msoAnimDirectionOutSlightly
msoAnimDirectionRight
msoAnimDirectionSlightly
msoAnimDirectionTop
msoAnimDirectionTopLeft
msoAnimDirectionTopRight
msoAnimDirectionUp
msoAnimDirectionUpLeft
msoAnimDirectionUpRight
msoAnimDirectionVertical
msoAnimDirectionVerticalIn
msoAnimDirectionVerticalOut

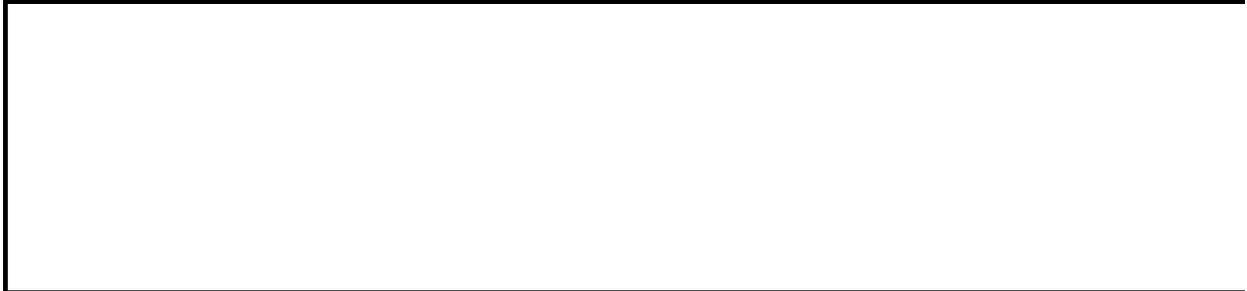
expression.**Direction**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, and animates the shape to fly in from the left.

```
Sub AddShapeSetAnimFly()  
  
    Dim effFly As Effect  
    Dim shpCube As Shape  
  
    Set shpCube = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeCube, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effFly = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpCube, effectId:=msoAnimEffectFly)  
  
    effFly.Timing.Duration = 3  
    effFly.EffectParameters.Direction = msoAnimDirectionLeft  
  
End Sub
```



[Show All](#)

DisplayAlerts Property

Sets or returns a [PpAlertLevel](#) constant that represents whether Microsoft PowerPoint displays alerts while running a macro. Read/write.

PpAlertLevel can be one of these **PpAlertLevel** constants.

ppAlertsAll All message boxes and alerts are displayed; errors are returned to the macro.

ppAlertsNone Default. No alerts or message boxes are displayed. If a macro encounters a message box, the default value is chosen and the macro continues.

expression.**DisplayAlerts**

expression Required. An expression that returns an **Application** object.

Remarks

The value of the **DisplayAlerts** property is not reset once a macro stops running; it is maintained throughout a session. It is not stored across sessions, so when PowerPoint begins, it reset to **ppAlertsNone**.

Example

The following line of code instructs PowerPoint to display all message boxes and alerts, returning errors to the macro.

```
Sub SetAlert  
    Application.DisplayAlerts = ppAlertsAll  
End Sub
```



[Show All](#)

DisplayAutoCorrectOptions Property

MsoTrue for Microsoft PowerPoint to display the **AutoCorrect Options** button.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**DisplayAutoCorrectOptions**

expression Required. An expression that returns an [AutoCorrect](#) object.

Example

This example disables display of the **AutoCorrect Options** and **AutoLayout Options** buttons.

```
Sub HideAutoCorrectOpButton()  
  With Application.AutoCorrect  
    .DisplayAutoCorrectOptions = msoFalse  
    .DisplayAutoLayoutOptions = msoFalse  
  End With  
End Sub
```



[Show All](#)

DisplayAutoLayoutOptions Property

MsoTrue for Microsoft PowerPoint to display the **AutoLayout Options** button.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**DisplayAutoLayoutOptions**

expression Required. An expression that returns an [AutoCorrect](#) object.

Example

This example disables display of the **AutoCorrect Options** and **AutoLayout Options** buttons.

```
Sub HideAutoCorrectOpButton()  
  With Application.AutoCorrect  
    .DisplayAutoCorrectOptions = msoFalse  
    .DisplayAutoLayoutOptions = msoFalse  
  End With  
End Sub
```



[Show All](#)

DisplayComments Property

Determines whether comments are displayed in the specified presentation.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

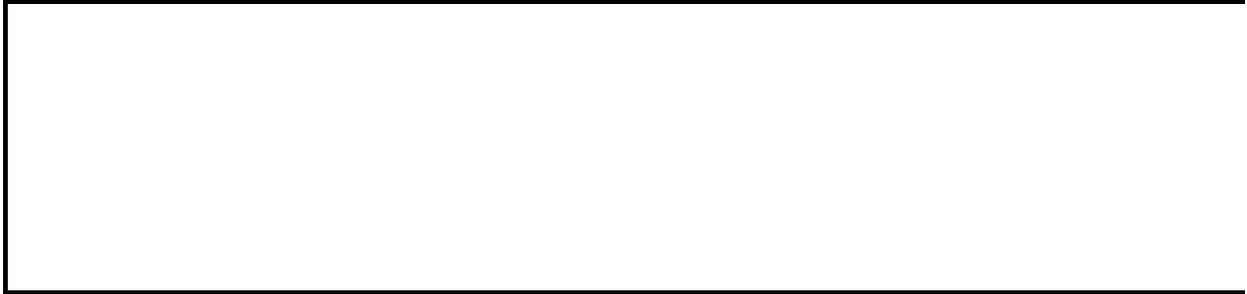
msoTriStateToggle

msoTrue Comments are displayed in the specified presentation.

Example

This example hides comments in the active presentation.

```
ActivePresentation.DisplayComments = msoFalse
```



[Show All](#)

DisplayGridLines Property

MsoTrue to display gridlines in Microsoft PowerPoint. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**DisplayGridLines**

expression Required. An expression that returns an [Application](#) object.

Example

This example toggles the display of the gridlines in PowerPoint.

```
Sub ToggleGridLines()  
  With Application  
    If .DisplayGridLines = msoTrue Then  
      .DisplayGridLines = msoFalse  
    Else  
      .DisplayGridLines = msoTrue  
    End If  
  End With  
End Sub
```



[Show All](#)

DisplayMasterShapes Property

Determines whether the specified slide or range of slides displays the background objects on the slide master. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide or range of slides displays the background objects on the slide master. These background objects can include text, drawings, OLE objects, and clip art you add to the slide master. Headers and footers aren't included.

Remarks

When you create a new slide, the default value for this property is **msoTrue**. If you copy a slide from another presentation, it retains the setting it had in the original presentation. That is, if the slide omitted slide master background objects in the original presentation, it will omit them in the new presentation as well.

Note that the look of the slide's background is determined by the color scheme and background as well as by the background objects. If setting the **DisplayMasterShapes** property alone doesn't give you the results you want, try setting the [FollowMasterBackground](#) and [ColorScheme](#) properties as well.

Example

This example copies slide one from presentation two, pastes it at the end of presentation one, and matches the slide's background, color scheme, and background objects to the rest of presentation one.

```
Presentations(2).Slides(1).Copy  
With Presentations(1).Slides.Paste  
    .FollowMasterBackground = True  
    .ColorScheme = Presentations(1).SlideMaster.ColorScheme  
    .DisplayMasterShapes = msoTrue  
End With
```



DisplayName Property

Returns a **String** that represents the name of an animation effect. Read-only.

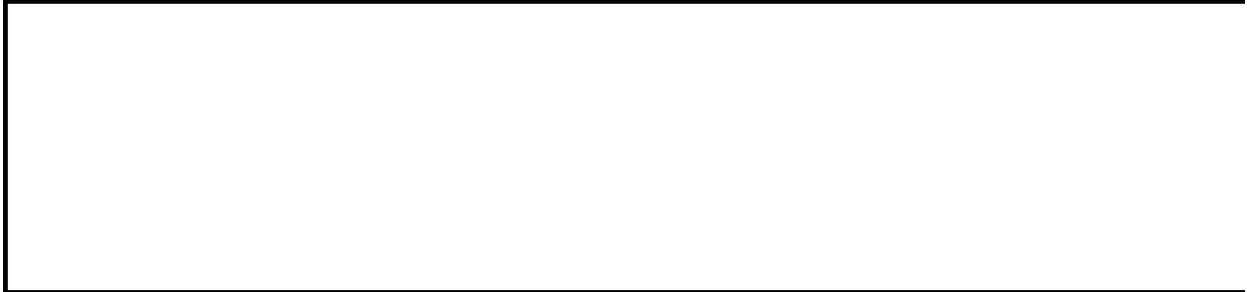
expression.**DisplayName**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the name for the first animation sequence of the first slide's main animation sequence timeline.

```
Sub DisplayEffectName()  
    Dim effMain As Effect  
    Set effMain = ActivePresentation.Slides(1).TimeLine.MainSequence  
    MsgBox effMain.DisplayName  
End Sub
```



[Show All](#)

DisplayOnTitleSlide Property

Determines whether the footer, date and time, and slide number appear on the title slide. Read/write [MsoTriState](#). Applies to slide masters.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The footer, date and time, and slide number appears on all slides except the title slide.

msoTriStateMixed

msoTriStateToggle

msoTrue The footer, date and time, and slide number appear on the title slide.

Example

This example sets the footer, date and time, and slide number to not appear on the title slide.

```
Application.ActivePresentation.SlideMaster.HeadersFooters _  
    .DisplayOnTitleSlide = msoFalse
```



[Show All](#)

DisplayPasteOptions Property

MsoTrue for Microsoft PowerPoint to display the **Paste Options** button, which displays directly under newly pasted text. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

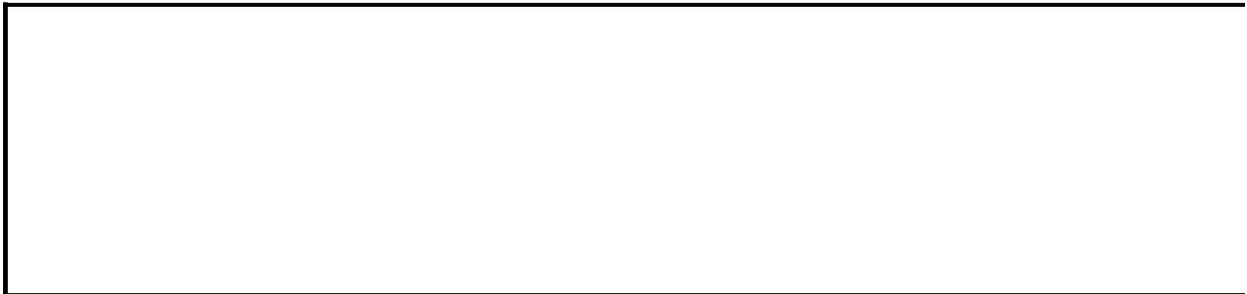
expression.**DisplayPasteOptions**

expression Required. An expression that returns an [Options](#) object.

Example

This example enables the **Paste Options** button if the option has been disabled.

```
Sub ShowPasteOptionsButton()  
  With Application.Options  
    If .DisplayPasteOptions = False Then  
      .DisplayPasteOptions = True  
    End If  
  End With  
End Sub
```



[Show All](#)

DisplaySlideMiniature Property

Determines if and when the slide miniature window is displayed automatically.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The slide miniature window is not displayed automatically.

msoTriStateMixed

msoTriStateToggle

msoTrue The slide miniature window is displayed automatically when the document window is in black-and-white view, the slide pane is zoomed to greater than 150% of the fit percentage, or a master view is visible.

Remarks

This property is not available in slide show view and slide sorter view. The slide miniature window isn't a member of either the **Windows** collection or the **SlideShowWindows** collection.

The fit percentage is determined by a combination of the size of the slide pane and the size of the presentation window. To determine the fit percentage, set the [ZoomToFit](#) property to **True** and then return the value of the [Zoom](#) property.

Example

If document window one is in slide view, this example displays the slide miniature window.

```
With Windows(1).View  
    If .Type = ppViewSlide Then .DisplaySlideMiniature = msoTrue  
End With
```



DocumentLibraryVersions Property

Returns a [DocumentLibraryVersions](#) collection that represents the collection of versions of a shared presentation that has versioning enabled and that is stored in a document library on a server.

expression.**DocumentLibraryVersions**

expression Required. An expression that returns a **Presentation** object.

Example

The following example returns the collection of versions for the active presentation. This example assumes that the active presentation has versioning enabled and is stored in a shared document library on a server.

```
Dim objVersions As DocumentLibraryVersions  
Set objVersions = ActivePresentation.DocumentLibraryVersions
```



Drop Property

For callouts with an explicitly set drop value, this property returns the vertical distance (in points) from the edge of the text bounding box to the place where the callout line attaches to the text box. This distance is measured from the top of the text box unless the **AutoAttach** property is set to **True** and the text box is to the left of the origin of the callout line (the place that the callout points to). In this case the drop distance is measured from the bottom of the text box. Read-only **Single**.

Remarks

Use the [CustomDrop](#) method to set the value of this property.

The value of this property accurately reflects the position of the callout line attachment to the text box only if the callout has an explicitly set drop value — that is, if the value of the [DropType](#) property is **msoCalloutDropCustom**.

Example

This example replaces the custom drop for shape one on myDocument with one of two preset drops, depending on whether the custom drop value is greater than or less than half the height of the callout text box. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).Callout
    If .DropType = msoCalloutDropCustom Then
        If .Drop < .Parent.Height / 2 Then
            .PresetDrop msoCalloutDropTop
        Else
            .PresetDrop msoCalloutDropBottom
        End If
    End If
End With
```



[Show All](#)

DropType Property

Returns a value that indicates where the callout line attaches to the callout text box. Read-only [MsoCalloutDropType](#).

MsoCalloutDropType can be one of these MsoCalloutDropType constants.

msoCalloutDropBottom

msoCalloutDropCenter

msoCalloutDropCustom

msoCalloutDropMixed

msoCalloutDropTop

expression.**DropType**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If the callout drop type is **msoCalloutDropCustom**, the values of the [Drop](#) and [AutoAttach](#) properties and the relative positions of the callout text box and callout line origin (the place that the callout points to) are used to determine where the callout line attaches to the text box.

This property is read-only. Use the [PresetDrop](#) method to set the value of this property.

Example

This example checks to determine whether shape three on myDocument is a callout with a custom drop. If it is, the code replaces the custom drop with one of two preset drops, depending on whether the custom drop value is greater than or less than half the height of the callout text box.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Type = msoCallout Then
        With .Callout
            If .DropType = msoCalloutDropCustom Then
                If .Drop < .Parent.Height / 2 Then
                    .PresetDrop msoCalloutDropTop
                Else
                    .PresetDrop msoCalloutDropBottom
                End If
            End If
        End With
    End If
End With
```



Duration Property

Returns or sets a **Single** that represents the length of an animation in seconds.
Read/write.

expression.**Duration**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape and an animation to that shape, then sets its animation duration.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=sh, effectId:=msoAnimEffectPathDiamond)  
  
    'Sets duration of effect  
    effDiamond.Timing.Duration = 5  
  
End Sub
```



[Show All](#)

EditingType Property

If the specified node is a vertex, this property returns a value that indicates how changes made to the node affect the two segments connected to the node. If the node is a control point for a curved segment, this property returns the editing type of the adjacent vertex. Read-only [MsoEditingType](#).

MsoEditingType can be one of these MsoEditingType constants.

msoEditingAuto

msoEditingCorner

msoEditingSmooth

msoEditingSymmetric

expression.**EditingType**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This property is read-only. Use the [SetEditingType](#) method to set the value of this property.

Example

This example changes all corner nodes to smooth nodes in shape three on myDocument. Shape three must be a freeform drawing.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
  For n = 1 to .Count
    If .Item(n).EditingType = msoEditingCorner Then
      .SetEditingType n, msoEditingSmooth
    End If
  Next
End With
```



EffectInformation Property

Returns an [EffectInformation](#) object representing information for a specified animation effect.

expression.**EffectInformation**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a sound effect to the main animation sequence for a given shape.

```
Sub AddSoundEffect()  
    Dim effMain As Effect  
    Set effMain = ActivePresentation.Slides(1).TimeLine.MainSequence  
    MsgBox effMain.EffectInformation.AfterEffect  
End Sub
```



EffectParameters Property

Returns an [EffectParameters](#) object representing animation effect properties.

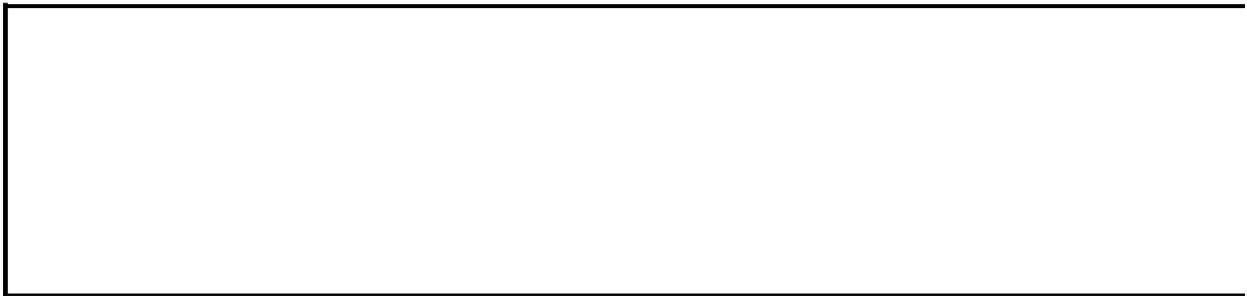
expression.**EffectParameters**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds an effect to the main animation sequence on the first slide. This effect changes the font for the first shape to Broadway.

```
Sub ChangeFontName()  
    Dim shpText As Shape  
    Dim effNew As Effect  
  
    Set shpText = ActivePresentation.Slides(1).Shapes(1)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpText, EffectId:=msoAnimEffectChangeFont  
    effNew.EffectParameters.FontName = "Broadway"  
End Sub
```



[Show All](#)

EffectType Property

Sets or returns an [MsoAnimEffect](#) constant that represents an animation effect type. Read/write.

MsoAnimEffect can be one of these MsoAnimEffect constants.

msoAnimEffectAppear

msoAnimEffectArcUp

msoAnimEffectAscend

msoAnimEffectBlast

msoAnimEffectBlinds

msoAnimEffectBoldFlash

msoAnimEffectBoldReveal

msoAnimEffectBoomerang

msoAnimEffectBounce

msoAnimEffectBox

msoAnimEffectBrushOnColor

msoAnimEffectBrushOnUnderline

msoAnimEffectCenterRevolve

msoAnimEffectChangeFillColor

msoAnimEffectChangeFont

msoAnimEffectChangeFontColor

msoAnimEffectChangeFontSize

msoAnimEffectChangeFontStyle

msoAnimEffectChangeLineColor

msoAnimEffectCheckerboard

msoAnimEffectCircle

msoAnimEffectColorBlend

msoAnimEffectColorReveal

msoAnimEffectColorWave

msoAnimEffectComplementaryColor

msoAnimEffectComplementaryColor2

msoAnimEffectContrastingColor
msoAnimEffectCrawl
msoAnimEffectCredits
msoAnimEffectCustom
msoAnimEffectDarken
msoAnimEffectDesaturate
msoAnimEffectDescend
msoAnimEffectDiamond
msoAnimEffectDissolve
msoAnimEffectEaseIn
msoAnimEffectExpand
msoAnimEffectFade
msoAnimEffectFadedAscend
msoAnimEffectFadedSwivel
msoAnimEffectFadedZoom
msoAnimEffectFlashBulb
msoAnimEffectFlashOnce
msoAnimEffectFlicker
msoAnimEffectFlip
msoAnimEffectFloat
msoAnimEffectFly
msoAnimEffectFold
msoAnimEffectGlide
msoAnimEffectGrowAndTurn
msoAnimEffectGrowShrink
msoAnimEffectGrowWithColor
msoAnimEffectLighten
msoAnimEffectMediaPause
msoAnimEffectMediaPlay
msoAnimEffectMediaStop
msoAnimEffectPath4PointStar
msoAnimEffectPath5PointStar
msoAnimEffectPath6PointStar

msoAnimEffectPath8PointStar
msoAnimEffectPathArcDown
msoAnimEffectPathArcLeft
msoAnimEffectPathArcRight
msoAnimEffectPathArcUp
msoAnimEffectPathBean
msoAnimEffectPathBounceLeft
msoAnimEffectPathBounceRight
msoAnimEffectPathBuzzsaw
msoAnimEffectPathCircle
msoAnimEffectPathCrescentMoon
msoAnimEffectPathCurvedSquare
msoAnimEffectPathCurvedX
msoAnimEffectPathCurvyLeft
msoAnimEffectPathCurvyRight
msoAnimEffectPathCurvyStar
msoAnimEffectPathDecayingWave
msoAnimEffectPathDiagonalDownRight
msoAnimEffectPathDiagonalUpRight
msoAnimEffectPathDiamond
msoAnimEffectPathDown
msoAnimEffectPathEqualTriangle
msoAnimEffectPathFigure8Four
msoAnimEffectPathFootball
msoAnimEffectPathFunnel
msoAnimEffectPathHeart
msoAnimEffectPathHeartbeat
msoAnimEffectPathHexagon
msoAnimEffectPathHorizontalFigure8
msoAnimEffectPathInvertedSquare
msoAnimEffectPathInvertedTriangle
msoAnimEffectPathLeft
msoAnimEffectPathLoopdeLoop

msoAnimEffectPathNeutron
msoAnimEffectPathOctagon
msoAnimEffectPathParallelogram
msoAnimEffectPathPeanut
msoAnimEffectPathPentagon
msoAnimEffectPathPlus
msoAnimEffectPathPointyStar
msoAnimEffectPathRightTriangle
msoAnimEffectPathSCurve1
msoAnimEffectPathSCurve2
msoAnimEffectPathSineWave
msoAnimEffectPathSpiralLeft
msoAnimEffectPathSpiralRight
msoAnimEffectPathSpring
msoAnimEffectPathSquare
msoAnimEffectPathStairsDown
msoAnimEffectPathSwoosh
msoAnimEffectPathTeardrop
msoAnimEffectPathTrapezoid
msoAnimEffectPathTurnDown
msoAnimEffectPathTurnRight
msoAnimEffectPathTurnUp
msoAnimEffectPathTurnUpRight
msoAnimEffectPathVerticalFigure8
msoAnimEffectPathWave
msoAnimEffectPathZigzag
msoAnimEffectPeek
msoAnimEffectPinwheel
msoAnimEffectPlus
msoAnimEffectRandomBars
msoAnimEffectRandomEffects
msoAnimEffectRiseUp
msoAnimEffectShimmer

msoAnimEffectSling
msoAnimEffectSpin
msoAnimEffectSpinner
msoAnimEffectSpiral
msoAnimEffectSplit
msoAnimEffectStretch
msoAnimEffectStretchy
msoAnimEffectStrips
msoAnimEffectStyleEmphasis
msoAnimEffectSwish
msoAnimEffectSwivel
msoAnimEffectTeeter
msoAnimEffectThinLine
msoAnimEffectTransparency
msoAnimEffectUnfold
msoAnimEffectVerticalGrow
msoAnimEffectWave
msoAnimEffectWedge
msoAnimEffectWheel
msoAnimEffectWhip
msoAnimEffectWipe
msoAnimEffectZip
msoAnimEffectZoom
msoAnimEffectLightSpeed

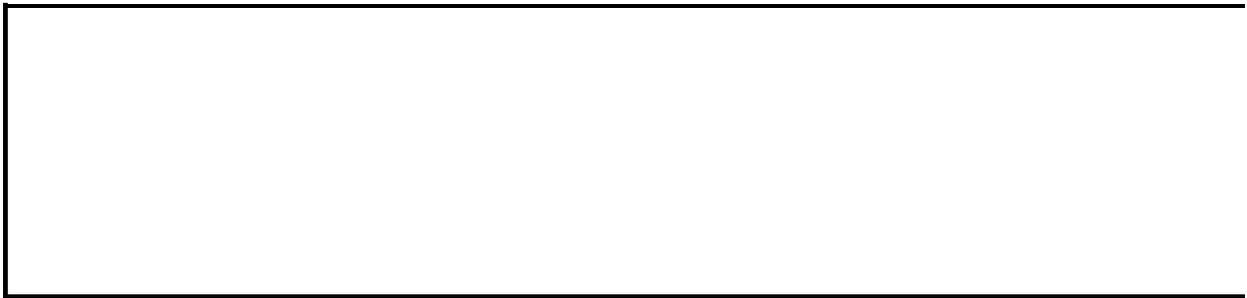
expression.**EffectType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example changes an animation effect to a random bar animation.

```
Sub ChangeEffectType()  
    Dim effRandom As Effect  
    Set effRandom = ActivePresentation.Slides(1).TimeLine.MainSequen  
    effRandom.EffectType = msoAnimEffectRandomBars  
End Sub
```



EmailSubject Property

Returns or sets the text string of the hyperlink subject line. The subject line is appended to the Internet address (URL) of the hyperlink. Read/write **String**.

Remarks

This property is commonly used with e-mail hyperlinks. The value of this property takes precedence over any e-mail subject specified in the [Address](#) property of the same **Hyperlink** object.

Example

This example sets the e-mail subject line of the first hyperlink on slide one in the active presentation.

```
ActivePresentation.Slides(1).Hyperlinks(1) _  
    .EmailSubject = "Quote Request"
```



[Show All](#)

Embeddable Property

Determines whether the specified font can be embedded in the presentation.
Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified font can be embedded in the presentation.

expression.**Embeddable**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example checks each font used in the active presentation to determine whether it's embeddable in the presentation.

```
For Each usedFont In Presentations(1).Fonts
    If usedFont.Embeddable Then
        MsgBox usedFont.Name & ": Embeddable"
    Else
        MsgBox usedFont.Name & ": Not embeddable"
    End If
Next usedFont
```



[Show All](#)

Embedded Property

Determines whether the specified font is embedded in the presentation. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified font is embedded in the presentation.

expression.**Embedded**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example checks each font used in the active presentation to determine whether it's embedded in the presentation.

```
For Each usedFont In Presentations(1).Fonts
    If usedFont.Embedded Then
        MsgBox usedFont.Name & ": Embedded"
    Else
        MsgBox usedFont.Name & ": Not embedded"
    End If
Next usedFont
```



[Show All](#)

Emboss Property

Determines whether the character format is embossed. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The character format is not embossed.

msoTriStateMixed The specified text range contains both embossed and unembossed characters.

msoTriStateToggle

msoTrue The character format is embossed.

expression.**Emboss**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the title text on slide one to embossed.

```
Application.ActivePresentation.Slides(1).Shapes.Title _  
    .TextFrame.TextRange.Font.Emboss = msoTrue
```



[Show All](#)

Encoding Property

Returns or sets the document [encoding](#) (code page or character set) to be used by the Web browser when you view the saved document. The default is the system code page. Read/write [MsoEncoding](#).

Although MsoEncoding can be one of these MsoEncoding constants, you cannot not use any of the constants that have the suffix **AutoDetect**. Those constants are used by the [ReloadAs](#) method.

msoEncodingArabic

msoEncodingArabicASMO

msoEncodingArabicAutoDetect

msoEncodingArabicTransparentASMO

msoEncodingAutoDetect

msoEncodingBaltic

msoEncodingCentralEuropean

msoEncodingCyrillic

msoEncodingCyrillicAutoDetect

msoEncodingEBCDICArabic

msoEncodingEBCDICDenmarkNorway

msoEncodingEBCDICFinlandSweden

msoEncodingEBCDICFrance

msoEncodingEBCDICGermany

msoEncodingEBCDICGreek

msoEncodingEBCDICGreekModern

msoEncodingEBCDICHebrew

msoEncodingEBCDICIcelandic

msoEncodingEBCDICInternational

msoEncodingEBCDICItaly

msoEncodingEBCDICJapaneseKatakanaExtended

msoEncodingEBCDICJapaneseKatakanaExtendedAndJapanese

msoEncodingEBCDICJapaneseLatinExtendedAndJapanese

msoEncodingEBCDICKoreanExtended
msoEncodingEBCDICKoreanExtendedAndKorean
msoEncodingEBCDICLatinAmericaSpain
msoEncodingEBCDICMultilingualROECELatin2
msoEncodingEBCDICRussian
msoEncodingEBCDICSerbianBulgarian
msoEncodingEBCDICSimplifiedChineseExtendedAndSimplifiedChinese
msoEncodingEBCDICThai
msoEncodingEBCDICTurkish
msoEncodingEBCDICTurkishLatin5
msoEncodingEBCDICUnitedKingdom
msoEncodingEBCDICUSCanada
msoEncodingEBCDICUSCanadaAndJapanese
msoEncodingEBCDICUSCanadaAndTraditionalChinese
msoEncodingEUCChineseSimplifiedChinese
msoEncodingEUCJapanese
msoEncodingEUCKorean
msoEncodingEUCTaiwaneseTraditionalChinese
msoEncodingEuropa3
msoEncodingExtAlphaLowercase
msoEncodingGreek
msoEncodingGreekAutoDetect
msoEncodingHebrew
msoEncodingHZGBSimplifiedChinese
msoEncodingIA5German
msoEncodingIA5IRV
msoEncodingIA5Norwegian
msoEncodingIA5Swedish
msoEncodingISO2022CNSimplifiedChinese
msoEncodingISO2022CNTraditionalChinese
msoEncodingISO2022JPJISX02011989
msoEncodingISO2022JPJISX02021984
msoEncodingISO2022JPNoHalfwidthKatakana

msoEncodingISO2022KR
msoEncodingISO6937NonSpacingAccent
msoEncodingISO885915Latin9
msoEncodingISO88591Latin1
msoEncodingISO88592CentralEurope
msoEncodingISO88593Latin3
msoEncodingISO88594Baltic
msoEncodingISO88595Cyrillic
msoEncodingISO88596Arabic
msoEncodingISO88597Greek
msoEncodingISO88598Hebrew
msoEncodingISO88599Turkish
msoEncodingJapaneseAutoDetect
msoEncodingJapaneseShiftJIS
msoEncodingKOI8R
msoEncodingKOI8U
msoEncodingKorean
msoEncodingKoreanAutoDetect
msoEncodingKoreanJohab
msoEncodingMacCroatia
msoEncodingMacCyrillic
msoEncodingMacGreek1
msoEncodingMacHebrew
msoEncodingMacIcelandic
msoEncodingMacJapanese
msoEncodingMacKorean
msoEncodingMacLatin2
msoEncodingMacRoman
msoEncodingMacRomania
msoEncodingMacSimplifiedChineseGB2312
msoEncodingMacTraditionalChineseBig5
msoEncodingMacTurkish
msoEncodingMacUkraine

msoEncodingOEMArabic
msoEncodingOEMBaltic
msoEncodingOEMCanadianFrench
msoEncodingOEMCyrillic
msoEncodingOEMCyrillicII
msoEncodingOEMGreek437G
msoEncodingOEMHebrew
msoEncodingOEMIcelandic
msoEncodingOEMModernGreek
msoEncodingOEMMultilingualLatinI
msoEncodingOEMMultilingualLatinII
msoEncodingOEMNordic
msoEncodingOEMPortuguese
msoEncodingOEMTurkish
msoEncodingOEMUnitedStates
msoEncodingSimplifiedChineseAutoDetect
msoEncodingSimplifiedChineseGBK
msoEncodingT61
msoEncodingTaiwanCNS
msoEncodingTaiwanEten
msoEncodingTaiwanIBM5550
msoEncodingTaiwanTCA
msoEncodingTaiwanTeleText
msoEncodingTaiwanWang
msoEncodingThai
msoEncodingTraditionalChineseAutoDetect
msoEncodingTraditionalChineseBig5
msoEncodingTurkish
msoEncodingUnicodeBigEndian
msoEncodingUnicodeLittleEndian
msoEncodingUSASCII
msoEncodingUTF7
msoEncodingUTF8

msoEncodingVietnamese

msoEncodingWestern

msoEncodingMacArabic

expression.**Encoding**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example checks to see whether the default document encoding is Western, and then it sets the string `strDocEncoding` accordingly.

```
If Application.DefaultWebOptions.Encoding = msoEncodingWestern Then
    strDocEncoding = "Western"
Else
    strDocEncoding = "Other"
End If
```



End Property

Returns the number of the last slide in the specified print range. Read-only
Long.

Example

This example displays a message that indicates the starting and ending slide numbers for print range one in the active presentation.

```
With ActivePresentation.PrintOptions.Ranges
  If .Count > 0 Then
    With .Item(1)
      MsgBox "Print range 1 starts on slide " & .Start & _
        " and ends on slide " & .End
    End With
  End If
End With
```



[Show All](#)

EndArrowheadLength Property

Returns or sets the length of the arrowhead at the end of the specified line.
Read/write [MsoArrowheadLength](#).

MsoArrowheadLength can be one of these MsoArrowheadLength constants.

msoArrowheadLengthMedium

msoArrowheadLengthMixed

msoArrowheadLong

msoArrowheadShort

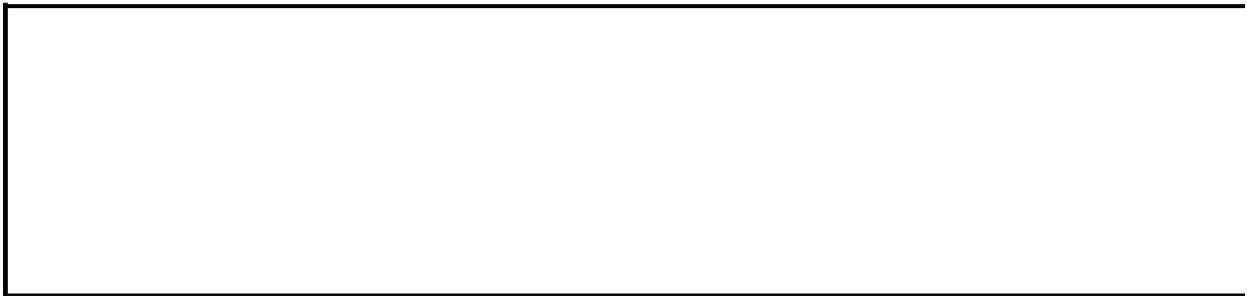
expression.**EndArrowheadLength**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

EndArrowheadStyle Property

Returns or sets the style of the arrowhead at the end of the specified line.
Read/write [MsoArrowheadStyle](#).

MsoArrowheadStyle can be one of these MsoArrowheadStyle constants.

msoArrowheadDiamond

msoArrowheadNone

msoArrowheadOpen

msoArrowheadOval

msoArrowheadStealth

msoArrowheadStyleMixed

msoArrowheadTriangle

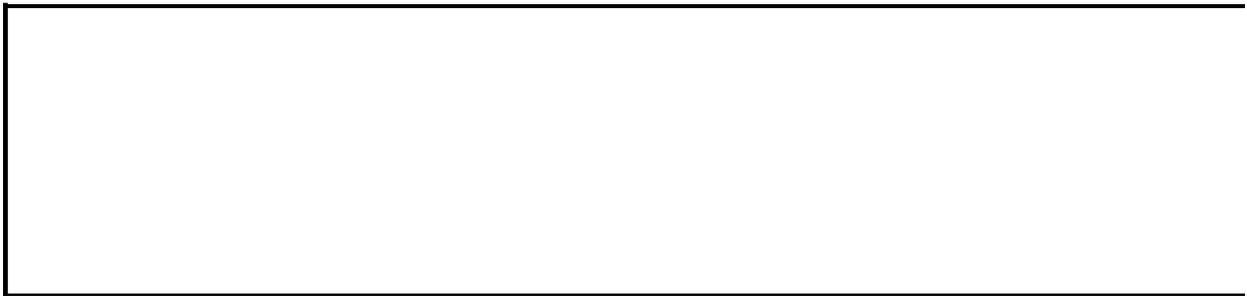
expression.**EndArrowheadStyle**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

EndArrowheadWidth Property

Returns or sets the width of the arrowhead at the end of the specified line.
Read/write [MsoArrowheadWidth](#).

MsoArrowheadWidth can be one of these MsoArrowheadWidth constants.

msoArrowheadNarrow

msoArrowheadWide

msoArrowheadWidthMedium

msoArrowheadWidthMixed

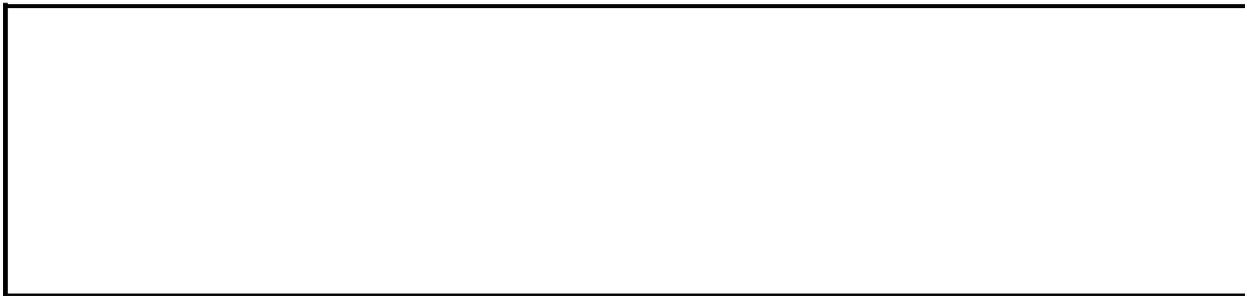
expression.**EndArrowheadWidth**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a line to myDocument. There's a short, narrow oval on the line's starting point and a long, wide triangle on its end point.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```



[Show All](#)

EndConnected Property

Determines whether the end of the specified connector is connected to a shape.
Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The end of the specified connector is connected to a shape.

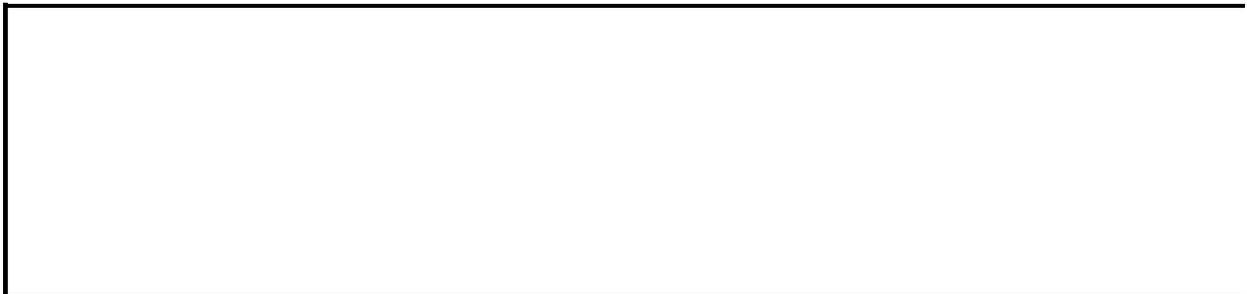
expression.**EndConnected**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

If the end of the connector represented by shape three on the first slide in the active presentation is connected to a shape, this example stores the connection site number in the variable `oldEndConnSite`, stores a reference to the connected shape in the object variable `oldEndConnShape`, and then disconnects the end of the connector from the shape.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Connector Then
        With .ConnectorFormat
            If .EndConnected Then
                oldEndConnSite = .EndConnectionSite
                Set oldEndConnShape = .EndConnectedShape
                .EndDisconnect
            End If
        End With
    End If
End With
```



EndConnectedShape Property

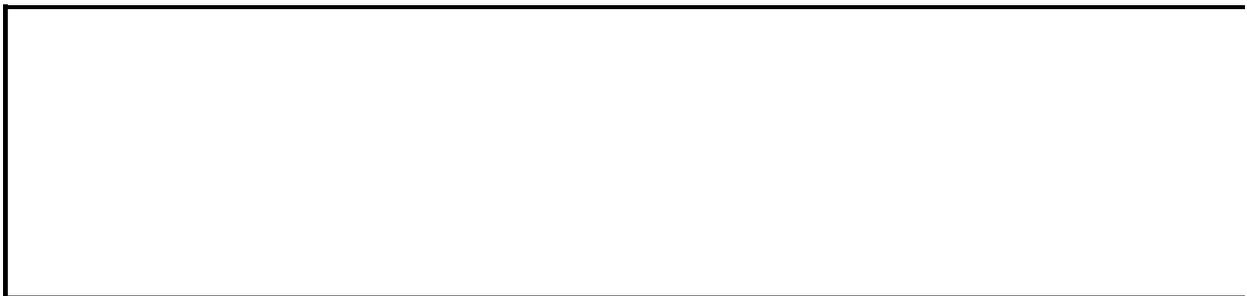
Returns a [Shape](#) object that represents the shape that the end of the specified connector is attached to. Read-only.

Note If the end of the specified connector isn't attached to a shape, this property generates an error.

Example

This example assumes that the first slide in the active presentation already contains two shapes attached by a connector named "Conn1To2." The code adds a rectangle and a connector to the first slide. The end of the new connector will be attached to the same connection site as the end of the connector named "Conn1To2," and the beginning of the new connector will be attached to connection site one on the new rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    Set r3 = .AddShape(msoShapeRectangle, 100, 420, 200, 100)
    .AddConnector(msoConnectorCurve, 0, 0, 10, 10) _
        .Name = "Conn1To3"
    With .Item("Conn1To2").ConnectorFormat
        endConnSite1 = .EndConnectionSite
        Set endConnShape1 = .EndConnectedShape
    End With
    With .Item("Conn1To3").ConnectorFormat
        .BeginConnect r3, 1
        .EndConnect endConnShape1, endConnSite1
    End With
End With
```



EndConnectionSite Property

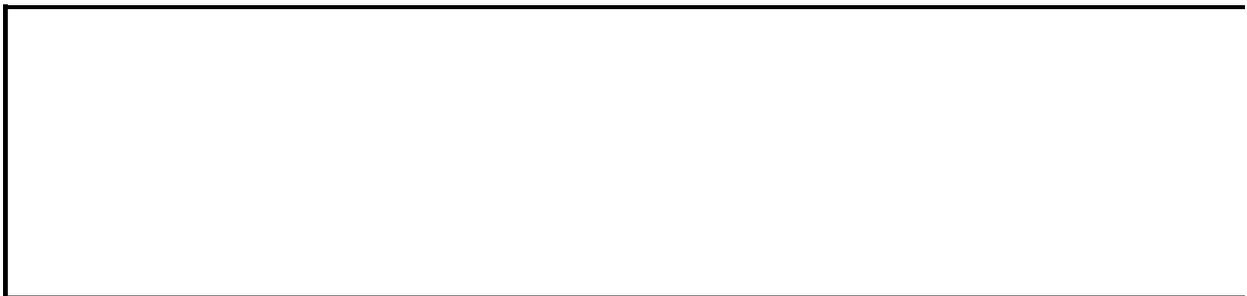
Returns an integer that specifies the connection site that the end of a connector is connected to. Read-only **Long**.

Note If the end of the specified connector isn't attached to a shape, this property generates an error.

Example

This example assumes that the first slide in the active presentation already contains two shapes attached by a connector named "Conn1To2." The code adds a rectangle and a connector to the first slide. The end of the new connector will be attached to the same connection site as the end of the connector named "Conn1To2," and the beginning of the new connector will be attached to connection site one on the new rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    Set r3 = .AddShape(msoShapeRectangle, 100, 420, 200, 100)
    .AddConnector(msoConnectorCurve, 0, 0, 10, 10) _
        .Name = "Conn1To3"
    With .Item("Conn1To2").ConnectorFormat
        endConnSite1 = .EndConnectionSite
        Set endConnShape1 = .EndConnectedShape
    End With
    With .Item("Conn1To3").ConnectorFormat
        .BeginConnect r3, 1
        .EndConnect endConnShape1, endConnSite1
    End With
End With
```



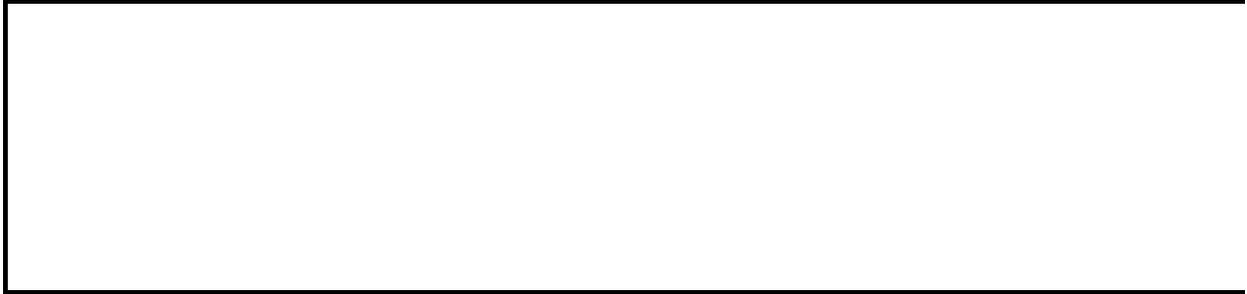
EndingSlide Property

Returns or sets the last slide to be displayed in the specified slide show.
Read/write **Long**.

Example

This example runs a slide show of the active presentation, starting with slide two and ending with slide four.

```
With ActivePresentation.SlideShowSettings
    .RangeType = ppShowSlideRange
    .StartingSlide = 2
    .EndingSlide = 4
    .Run
End With
```



[Show All](#)

EntryEffect Property

For the **AnimationSettings** object, this property returns or sets the special effect applied to the animation for the specified shape. For the **SlideShowTransition** object, this property returns or sets the special effect applied to the specified slide transition. Read/write [PpEntryEffect](#).

PpEntryEffect can be one of these PpEntryEffect constants.

ppEffectAppear

ppEffectBlindsHorizontal

ppEffectBlindsVertical

ppEffectBoxIn

ppEffectBoxOut

ppEffectCheckerboardAcross

ppEffectCheckerboardDown

ppEffectCoverDown

ppEffectCoverLeft

ppEffectCoverLeftDown

ppEffectCoverLeftUp

ppEffectCoverRight

ppEffectCoverRightDown

ppEffectCoverRightUp

ppEffectCoverUp

ppEffectCrawlFromDown

ppEffectCrawlFromLeft

ppEffectCrawlFromRight

ppEffectCrawlFromUp

ppEffectCut

ppEffectCutThroughBlack

ppEffectDissolve

ppEffectFade

ppEffectFlashOnceFast

ppEffectFlashOnceMedium
ppEffectFlashOnceSlow
ppEffectFlyFromBottom
ppEffectFlyFromBottomLeft
ppEffectFlyFromBottomRight
ppEffectFlyFromLeft
ppEffectFlyFromRight
ppEffectFlyFromTop
ppEffectFlyFromTopLeft
ppEffectFlyFromTopRight
ppEffectMixed
ppEffectNone
ppEffectPeekFromDown
ppEffectPeekFromLeft
ppEffectPeekFromRight
ppEffectPeekFromUp
ppEffectRandom
ppEffectRandomBarsHorizontal
ppEffectRandomBarsVertical
ppEffectSpiral
ppEffectSplitHorizontalIn
ppEffectSplitHorizontalOut
ppEffectSplitVerticalIn
ppEffectSplitVerticalOut
ppEffectStretchAcross
ppEffectStretchDown
ppEffectStretchLeft
ppEffectStretchRight
ppEffectStretchUp
ppEffectStripsDownLeft
ppEffectStripsDownRight
ppEffectStripsLeftDown
ppEffectStripsLeftUp

ppEffectStripsRightDown
ppEffectStripsRightUp
ppEffectStripsUpLeft
ppEffectStripsUpRight
ppEffectSwivel
ppEffectUncoverDown
ppEffectUncoverLeft
ppEffectUncoverLeftDown
ppEffectUncoverLeftUp
ppEffectUncoverRight
ppEffectUncoverRightDown
ppEffectUncoverRightUp
ppEffectUncoverUp
ppEffectWipeDown
ppEffectWipeLeft
ppEffectWipeRight
ppEffectWipeUp
ppEffectZoomBottom
ppEffectZoomCenter
ppEffectZoomIn
ppEffectZoomInSlightly
ppEffectZoomOut
ppEffectZoomOutSlightly

expression.**EntryEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If the [TextLevelEffect](#) property for the specified shape is set to **ppAnimateLevelNone** (the default value) or the [Animate](#) property is set to **False**, you won't see the special effect you've applied with the **EntryEffect** property.

Example

This example adds a title slide to the active presentation and sets the title to fly in from the right whenever it's animated during a slide show.

```
With ActivePresentation.Slides.Add(1, ppLayoutTitleOnly).Shapes(1)
    .TextFrame.TextRange.Text = "Sample title"
    With .AnimationSettings
        .TextLevelEffect = ppAnimateByAllLevels
        .EntryEffect = ppEffectFlyFromRight
        .Animate = True
    End With
End With
```



[Show All](#)

EnvelopeVisible Property

Determines whether the e-mail message header is visible in the document window. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue The e-mail message header is visible in the document window.

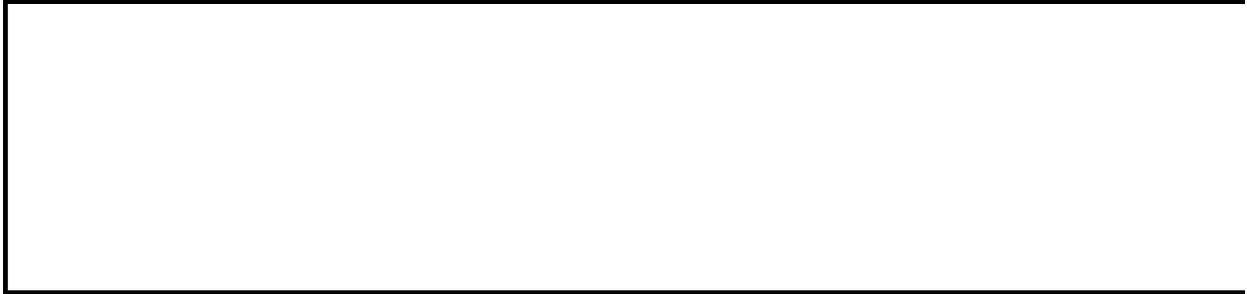
expression.**EnvelopeVisible**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the e-mail message header.

ActivePresentation.**EnvelopeVisible** = msoTrue



[Show All](#)

Exit Property

Returns or sets an [MsoTriState](#) that represents whether the animation effect is an exit effect. Read/write.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The effect is not an exit effect.

msoTriStateMixed

msoTriStateToggle

msoTrue The effect is an exit effect.

expression.**Exit**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays whether the specified animation is an exit animation effect.

```
Sub EffectExit()  
    Dim effMain As Effect  
    Set effMain = ActivePresentation.Slides(1).TimeLine.MainSequence  
    If effMain.Exit = msoTrue Then  
        MsgBox "This is an exit animation effect."  
    Else  
        MsgBox "This is not an exit animation effect."  
    End If  
End Sub
```



[Show All](#)

ExtraColors Property

Returns an [ExtraColors](#) object that represents the [extra colors](#) available in the specified presentation. Read-only.

Example

The following example adds a rectangle to slide one in the active presentation and sets its fill foreground color to the first extra color. If there hasn't been at least one extra color defined for the presentation, this example will fail.

```
With ActivePresentation
  Set rect = .Slides(1).Shapes _
    .AddShape(msoShapeRectangle, 50, 50, 100, 200)
  rect.Fill.ForeColor.RGB = .ExtraColors(1)
End With
```



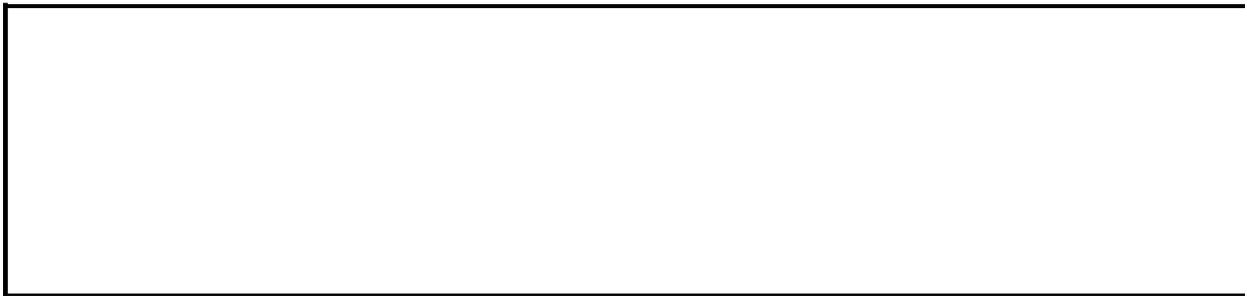
ExtrusionColor Property

Returns a [ColorFormat](#) object that represents the color of the shape's extrusion.
Read-only.

Example

This example adds an oval to myDocument, and then specifies that the oval be extruded to a depth of 50 points and that the extrusion be purple.

```
Set myDocument = ActivePresentation.Slides(1)
Set myShape = myDocument.Shapes _
    .AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    'RGB value for purple
    .ExtrusionColor.RGB = RGB(255, 100, 255)
End With
```



[Show All](#)

ExtrusionColorType Property

Returns or sets a value that indicates whether the extrusion color is based on the extruded shape's fill (the front face of the extrusion) and automatically changes when the shape's fill changes, or whether the extrusion color is independent of the shape's fill. Read/write [MsoExtrusionColorType](#).

MsoExtrusionColorType can be one of these MsoExtrusionColorType constants.

msoExtrusionColorAutomatic Extrusion color is based on shape fill.

msoExtrusionColorCustom Extrusion color is independent of shape fill.

msoExtrusionColorTypeMixed

expression.**ExtrusionColorType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

If shape one on myDocument has an automatic extrusion color, this example gives the extrusion a custom yellow color.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    If .ExtrusionColorType = msoExtrusionColorAutomatic Then
        .ExtrusionColor.RGB = RGB(240, 235, 16)
    End If
End With
```



[Show All](#)

FarEastLineBreakControl Property

Returns or sets the line break control option if you have an Asian language setting specified **True** if the line break control option is selected. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

Example

This example selects the line break option for the text in shape one on the first slide of the active presentation.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame _  
    .TextRange.ParagraphFormat.FarEastLineBreakControl = msoTrue
```



[Show All](#)

FarEastLineBreakLanguage Property

Returns or sets the language used to determine which line break level is used when the line break control option is turned on. Read/write [MsoFarEastLineBreakLanguageID](#).

MsoFarEastLineBreakLanguageID can be one of these MsoFarEastLineBreakLanguageID constants.

MsoFarEastLineBreakLanguageJapanese

MsoFarEastLineBreakLanguageKorean

MsoFarEastLineBreakLanguageSimplifiedChinese

MsoFarEastLineBreakLanguageTraditionalChinese

expression.FarEastLineBreakLanguage

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example sets the line break language to Japanese.

```
ActivePresentation.FarEastLineBreakLanguage = _  
    MsoFarEastLineBreakLanguageJapanese
```



[Show All](#)

FarEastLineBreakLevel Property

Returns or sets the line break based upon Asian character level. Read/write **Long**. Read/write [PpFarEastLineBreakLevel](#).

PpFarEastLineBreakLevel can be one of these PpFarEastLineBreakLevel constants.

ppFarEastLineBreakLevelCustom

ppFarEastLineBreakLevelNormal

ppFarEastLineBreakLevelStrict

expression.**FarEastLineBreakLevel**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets line break control to use level one kinsoku characters.

```
ActivePresentation.FarEastLineBreakLevel = _  
    ppFarEastLineBreakLevelNormal
```



[Show All](#)

FeatureInstall Property

Returns or sets how Microsoft PowerPoint handles calls to methods and properties that require features not yet installed. Read/write [MsoFeatureInstall](#).

MsoFeatureInstall can be one of these MsoFeatureInstall constants.

msoFeatureInstallNone Default. A trappable run-time automation error is generated when uninstalled features are called.

msoFeatureInstallOnDemand A dialog box is displayed prompting the user to install new features.

msoFeatureInstallOnDemandWithUI A progress meter is displayed during installation. The user isn't prompted to install new features.

expression.**FeatureInstall**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

You can use the **msoFeatureInstallOnDemandWithUI** constant to prevent users from believing that the application isn't responding while a feature is being installed. Use the **msoFeatureInstallNone** constant with error trapping routines to exclude end-user feature installation.

Note If you refer to an uninstalled presentation design template in a string, a run-time error is generated. The template is not installed automatically regardless of your **FeatureInstall** property setting. To use the [ApplyTemplate](#) method for a template that is not currently installed, you first must install the additional design templates. To do so, install the Additional Design Templates for PowerPoint by running the Microsoft Office installation program (available through the **Add/Remove Programs** icon in Windows Control Panel).

Example

This example checks the value of the **FeatureInstall** property. If the property is set to **msoFeatureInstallNone**, the code displays a message box that asks the user whether they want to change the property setting. If the user responds "Yes", the property is set to **msoFeatureInstallOnDemand**.

```
With Application
    If .FeatureInstall = msoFeatureInstallNone Then
        Reply = MsgBox("Uninstalled features for this " _
            & "application " & vbCrLf _
            & "may cause a run-time error when called." & vbCrLf
            & vbCrLf _
            & "Would you like to change this setting" & vbCrLf _
            & "to automatically install missing features when ca
            , 52, "Feature Install Setting")
        If Reply = 6 Then
            .FeatureInstall = msoFeatureInstallOnDemand
        End If
    End If
End With
```



[Show All](#)

FileDialog Property

Returns a [FileDialog](#) object that represents a single instance of a file dialog box.

expression.**FileDialog**(*Type*)

expression Required. An expression that returns an [Application](#) object.

Type Required [MsoFileDialogType](#). The type of dialog to return.

MsoFileDialogType can be one of these **MsoFileDialogType** constants.

msoFileDialogFilePicker

msoFileDialogFolderPicker

msoFileDialogOpen

msoFileDialogSaveAs

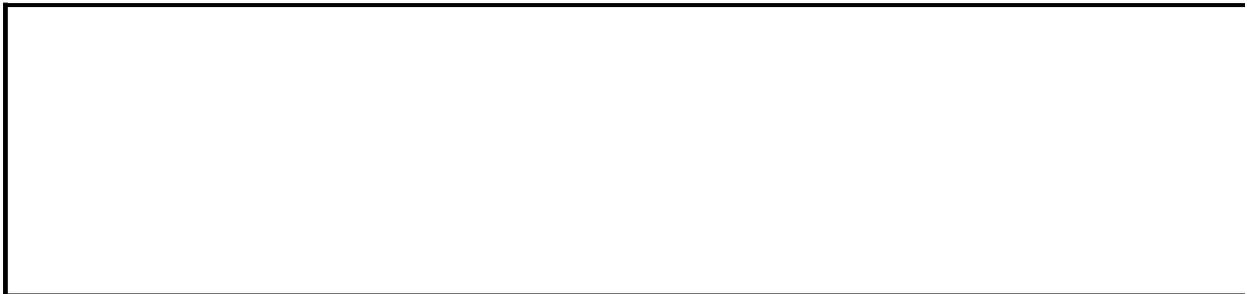
Example

This example displays the **Save As** dialog box.

```
Sub ShowSaveAsDialog()  
    Dim dlgSaveAs As FileDialog  
    Set dlgSaveAs = Application.FileDialog( _  
        Type:=msoFileDialogSaveAs)  
    dlgSaveAs.Show  
End Sub
```

This example displays the **Open** dialog box and allows a user to select multiple files to open.

```
Sub ShowFileDialog()  
    Dim dlgOpen As FileDialog  
    Set dlgOpen = Application.FileDialog( _  
        Type:=msoFileDialogOpen)  
    With dlgOpen  
        .AllowMultiSelect = True  
        .Show  
    End With  
End Sub
```



FileFind Property

Returns an **IFind** object that can be used to locate files. Read-only.

expression.**FileFind**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the names of all files in the My Documents folder that begin with "New."

```
With Application.FileFind
    .Name = "New*.*"
    .SearchPath = "C:\My Documents"
    .Execute
    For I = 1 To .Results.Count
        MsgBox .Results(I)
    Next
End With
```



FileName Property

Returns or sets the path and file name of the Web presentation created when all or part of the active presentation is published. Read/write **String**.

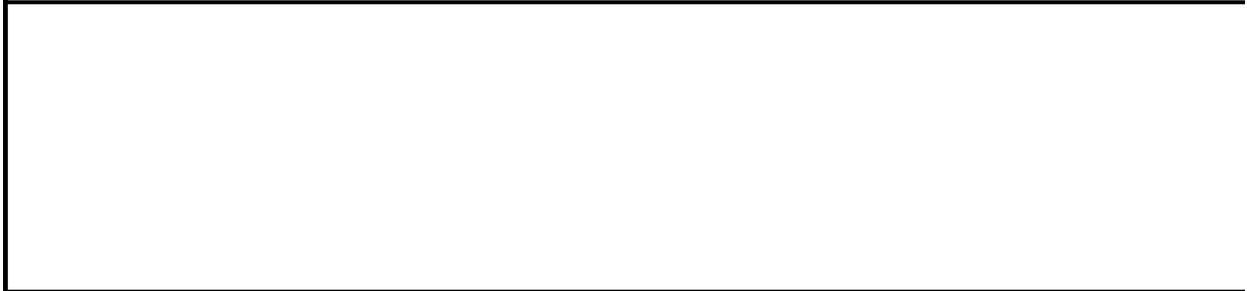
Remarks

The **FileName** property generates an error if a folder in the specified path does not exist.

Example

This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm and saves it in the Test folder.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```



FileSearch Property

Returns a **FileSearch** object that can be used to search for files using either an absolute or relative path. Read-only.

Example

This example displays the names of all files in the My Documents folder that begin with "New."

```
With Application.FileSearch  
    .FileName = "New*.*"  
    .LookIn = "C:\My Documents"  
    .Execute  
    For I = 1 To .FoundFiles.Count  
        MsgBox .FoundFiles(I)  
    Next  
End With
```



Fill Property

Returns a [FillFormat](#) object that contains fill formatting properties for the specified shape. Read-only.

Example

This example adds a rectangle to myDocument and then sets the foreground color, background color, and gradient for the rectangle's fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 90, 90, 90, 50).Fill
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(170, 170, 170)
    .TwoColorGradient msoGradientHorizontal, 1
End With
```



FilterEffect Property

Returns a **FilterEffect** object that represents a filter effect for an animation behavior. Read-only.

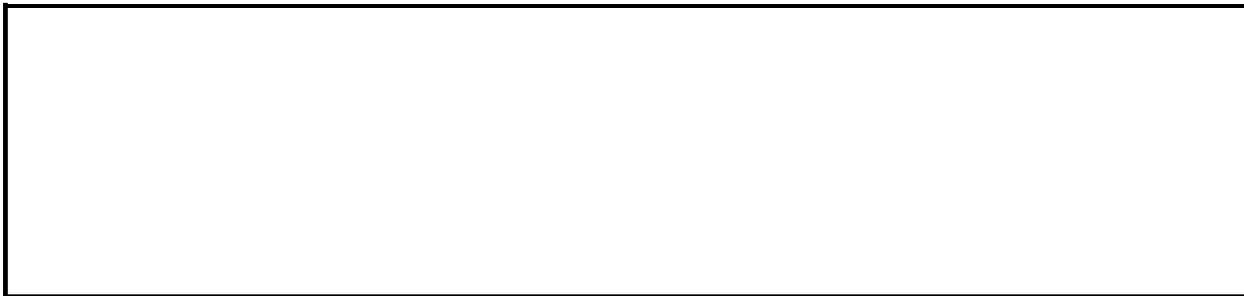
expression.**FilterEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to the first slide of the active presentation and sets a filter effect animation behavior.

```
Sub ChangeFilterEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeFilter)  
  
    With bhvEffect.FilterEffect  
        .Type = msoAnimFilterEffectTypeWipe  
        .Subtype = msoAnimFilterEffectSubtypeUp  
        .Reveal = msoTrue  
    End With  
End Sub
```



FirstChild Property

Returns a **DiagramNode** object representing the first diagram node in a collection of diagram nodes.

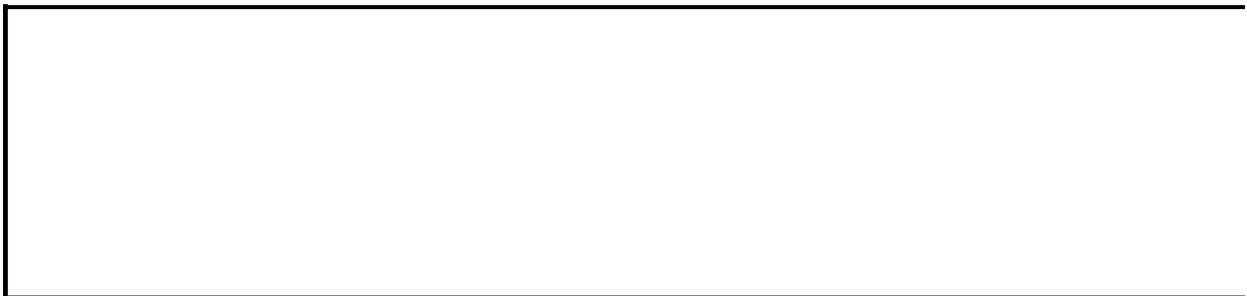
expression.**FirstChild**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates a new diagram and identifies the first child diagram node.

```
Sub FirstChildNodeHello()  
  
Dim dgnNode As DiagramNode  
Dim dgnFirst As DiagramNode  
Dim shpDiagram As Shape  
Dim intNodes As Integer  
  
Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram _  
(Type:=msoDiagramOrgChart, Left:=10, Top:=15, _  
Width:=400, Height:=475)  
Set dg nNode = shpDiagram.DiagramNode.Children.AddNode  
  
For intNodes = 1 To 3  
dg nNode.Children.AddNode  
Next intNodes  
  
Set dgnFirst = dg nNode.Children.FirstChild  
dgnFirst.Shape.TextFrame.TextRange.Text = "Here I am!"  
  
End Sub
```



FirstMargin Property

Returns or sets the first-line indent for the specified outline level, in points.
Read/write **Single**.

Remarks

If a paragraph begins with a bullet, the bullet position is determined by the **FirstMargin** property, and the position of the first text character in the paragraph is determined by the **LeftMargin** property.

Note The **RulerLevels** collection contains five **RulerLevel** objects, each of which corresponds to one of the possible outline levels. The **FirstMargin** property value for the **RulerLevel** object that corresponds to the first outline level has a valid range of (-9.0 to 4095.875). The valid range for the **FirstMargin** property values for the **RulerLevel** objects that correspond to the second through the fifth outline levels are determined as follows:

- The maximum value is always 4095.875.
- The minimum value is the maximum assigned value between the **FirstMargin** property and **LeftMargin** property of the previous level plus 9.

You can use the following equations to determine the minimum value for the **FirstMargin** property. *Index*, the index number of the **RulerLevel** object, indicates the object's corresponding outline level. To determine the minimum **FirstMargin** property values for the **RulerLevel** objects that correspond to the second through the fifth outline levels, substitute 2, 3, 4, or 5 for the *index* placeholder.

$$\text{Minimum}(\text{RulerLevel}(\textit{index}).\text{FirstMargin}) = \text{Maximum}(\text{RulerLevel}(\textit{index} - 1).\text{FirstMargin}, \text{RulerLevel}(\textit{index} - 1).\text{LeftMargin}) + 9$$
$$\text{Minimum}(\text{RulerLevel}(\textit{index}).\text{LeftMargin}) = \text{Maximum}(\text{RulerLevel}(\textit{index} - 1).\text{FirstMargin}, \text{RulerLevel}(\textit{index} - 1).\text{LeftMargin}) + 9$$

Example

This example sets the first-line indent and hanging indent for outline level one in body text on the slide master for the active presentation.

```
With Application.ActivePresentation _  
    .SlideMaster.TextStyles(ppBodyStyle)  
    With .Ruler.Levels(1)  
        .FirstMargin = 9  
        .LeftMargin = 54  
    End With  
End With
```



FirstSlideNumber Property

Returns or sets the slide number for the first slide in the presentation. Read/write **Long**.

Remarks

The slide number is the actual number that will appear in the lower-right corner of the slide when you display slide numbers. This number is determined by the number (order) of the slide within the presentation (the **SlideIndex** property value) and the starting slide number for the presentation (the **FirstSlideNumber** property value). The slide number will always be equal to the starting slide number + the slide index number – 1. The **SlideNumber** property returns the slide number.

Example

This example shows how changing the first slide number in the active presentation affects the slide number of a specific slide.

```
With Application.ActivePresentation
    .PageSetup.FirstSlideNumber = 1 'starts slide numbering at 1
    MsgBox .Slides(2).SlideNumber   'returns 2
    .PageSetup.FirstSlideNumber = 10 'starts slide numbering at 10
    MsgBox .Slides(2).SlideNumber   'returns 11
End With
```



[Show All](#)

FitToPage Property

Determines whether the slides will be scaled to fill the page they're printed on.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. The slides will have the dimensions specified in the **Page Setup** dialog box, whether or not those dimensions match the page they're printed on.

msoTriStateMixed

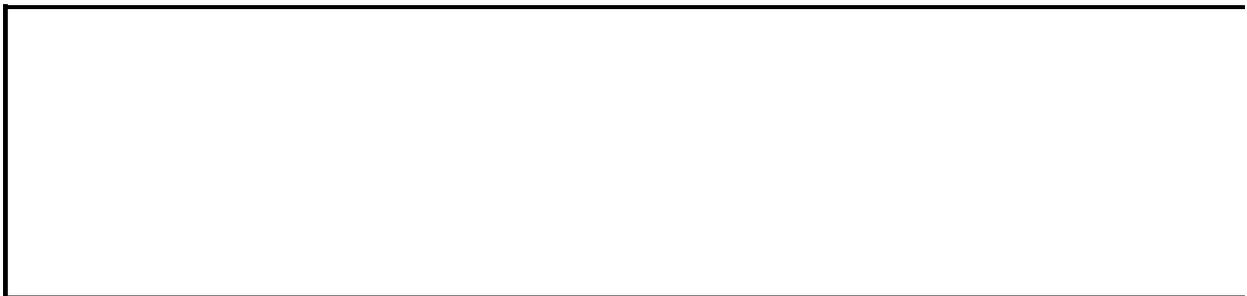
msoTriStateToggle

msoTrue The specified slides will be scaled to fill the page they're printed on, regardless of the values in the **Height** and **Width** boxes in the **Page Setup** dialog box (**File** menu)

Example

This example prints the active presentation and scales each slide to fit the printed page.

```
With ActivePresentation
    .PrintOptions.FitToPage = msoTrue
    .PrintOut
End With
```



FolderSuffix Property

Returns the folder suffix that Microsoft PowerPoint uses when you save or publish a complete or partial presentation as a Web page, use long file names, and choose to save supporting files in a separate folder (that is, if the [UseLongFileNames](#) and [OrganizeInFolder](#) properties are set to **True**). Read-only **String**.

Remarks

Newly created presentations use the suffix returned by the **FolderSuffix** property of the **DefaultWebOptions** object. The value of the **FolderSuffix** property of the **WebOptions** object may differ from that of the **DefaultWebOptions** object if the presentation was previously edited in a different language version of Microsoft PowerPoint. You can use the [UseDefaultFolderSuffix](#) method to change the suffix to that of the language you are currently using in Microsoft Office.

By default, the name of the supporting folder is the name of the Web page plus an underscore (_), a period (.), or a hyphen (-) and the word "files" (appearing in the language of the version of PowerPoint in which the file was saved as a Web page). For example, suppose that you use the Dutch language version of PowerPoint to save a file called "Page1" as a Web page. The default name of the supporting folder would be Page1_bestanden.

The following table lists each language version of Office, and gives its corresponding folder suffix. For the languages that are not listed in the table, the suffix ".files" is used.

Language

Language	Folder suffix
Arabic	.files
Basque	_fitxategiak
Brazilian	_arquivos
Bulgarian	.files
Catalan	_fitxers
Chinese - Simplified	.files
Chinese - Traditional	.files
Croatian	_datoteke
Czech	_soubory
Danish	-filer
Dutch	_bestanden

English	_files
Estonian	_failid
Finnish	_tiedostot
French	_fichiers
German	-Dateien
Greek	.files
Hebrew	.files
Hungarian	_elemei
Italian	-file
Japanese	.files
Korean	.files
Latvian	_fails
Lithuanian	_bylos
Norwegian	-filer
Polish	_pliki
Portuguese	_ficheiros
Romanian	.files
Russian	.files
Serbian (Cyrillic)	.files
Serbian (Latin)	_fajlovi
Slovakian	.files
Slovenian	_datoteke
Spanish	_archivos
Swedish	-filer
Thai	.files
Turkish	_dosyalar
Ukrainian	.files
Vietnamese	.files

Example

This example returns the folder suffix used by presentation one. The suffix is returned in the string variable `strFolderSuffix`.

```
strFolderSuffix = Presentations(1).WebOptions.FolderSuffix
```



[Show All](#)

FollowColors Property

Returns or sets the extent to which the colors in the specified object follow the slide's color scheme. The specified object must be a chart created in either Microsoft Graph or Microsoft Organization Chart. Read/write [PpFollowColors](#).

PpFollowColors can be one of these PpFollowColors constants.

ppFollowColorsNone The chart colors don't follow the slide's color scheme.

ppFollowColorsMixed

ppFollowColorsScheme All the colors in the chart follow the slide's color scheme.

ppFollowColorsTextAndBackground Only the text and background follow the slide's color scheme.

expression.**FollowColors**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example specifies that the text and background of shape two on slide one in the active presentation follow the slide's color scheme. Shape two must be a chart created in either Microsoft Graph or Microsoft Organization Chart.

```
ActivePresentation.Slides(1).Shapes(2).OLEFormat.FollowColors = _  
    ppFollowColorsTextAndBackground
```



[Show All](#)

FollowMasterBackground Property

Determines whether the slide or range of slides follows the slide master background. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The specified slide or range of slides has a custom background.

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide or range of slides follows the slide master background.

Remarks

When you create a new slide, the default value for this property is **True**. If you copy a slide from another presentation, it retains the setting it had in the original presentation. That is, if the slide followed the slide master background in the original presentation, it will automatically follow the slide master background in the new presentation; or, if the slide had a custom background, it will retain that custom background.

Note that the look of the slide's background is determined by the color scheme and background objects as well as by the background itself. If setting the **FollowMasterBackground** property alone doesn't give you the results you want, try setting the [ColorScheme](#) and [DisplayMasterShapes](#) properties as well.

Example

This example copies slide one from presentation two, pastes the slide at the end of presentation one, and matches the slide's background, color scheme, and background objects to the rest of presentation one.

```
Presentations(2).Slides(1).Copy  
With Presentations(1).Slides.Paste  
    .FollowMasterBackground = msoTrue  
    .ColorScheme = Presentations(1).SlideMaster.ColorScheme  
    .DisplayMasterShapes = True  
End With
```



Font Property

Returns a [Font](#) object that represents character formatting. Read-only.

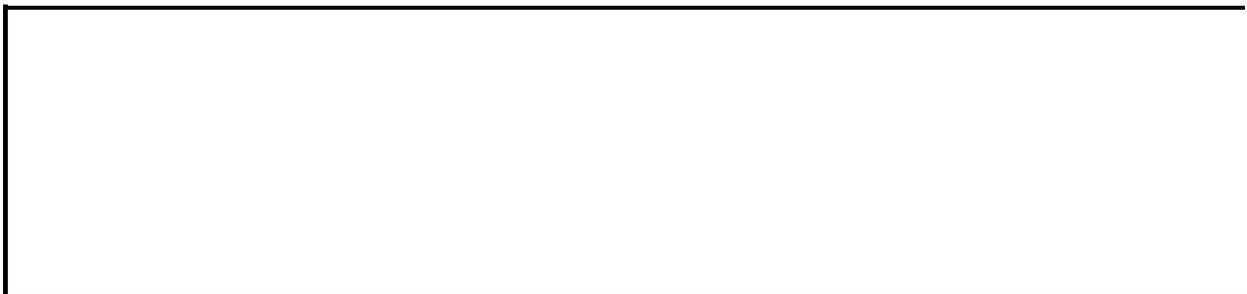
Example

This example sets the formatting for the text in shape one on slide one in the active presentation.

```
With ActivePresentation.Slides(1).Shapes(1)
  With .TextFrame.TextRange.Font
    .Size = 48
    .Name = "Palatino"
    .Bold = True
    .Color.RGB = RGB(255, 127, 255)
  End With
End With
```

This example sets the color and font name for bullets in shape two on slide one.

```
With ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat.Bullet
    .Visible = True
    With .Font
      .Name = "Palatino"
      .Color.RGB = RGB(0, 0, 255)
    End With
  End With
End With
```



[Show All](#)

FontBold Property

Determines whether the font in the specified WordArt is bold. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The font in the specified WordArt is bold.

Example

This example sets the font to bold for shape three on myDocument if the shape is WordArt.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.FontBold = msoTrue
    End If
End With
```



[Show All](#)

FontItalic Property

Determines whether the font in the specified WordArt is italic. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The font in the specified WordArt is italic.

Example

This example sets the font to italic for the shape named "WordArt 4" on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes("WordArt 4").TextEffect.FontItalic = msoTrue
```



FontName Property

Returns or sets the name of the font in the specified WordArt. Read/write **String**.

Example

This example sets the font name to "Courier New" for shape three on myDocument if the shape is WordArt.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.FontName = "Courier New"
    End If
End With
```



[Show All](#)

Fonts Property

 [Fonts property as it applies to the **Presentation** object.](#)

Returns a **Fonts** collection that represents all fonts used in the specified presentation. Read-only.

expression.**Fonts**

expression Required. An expression that returns one of the above objects.

 [Fonts property as it applies to the **DefaultWebOptions** object.](#)

Returns a **WebPageFonts** collection representing the set of available fonts for saving a presentation as a Web page. Read-only.

expression.**Fonts**

expression Required. An expression that returns one of the above objects.

Example

[As it applies to the **Presentation** object.](#)

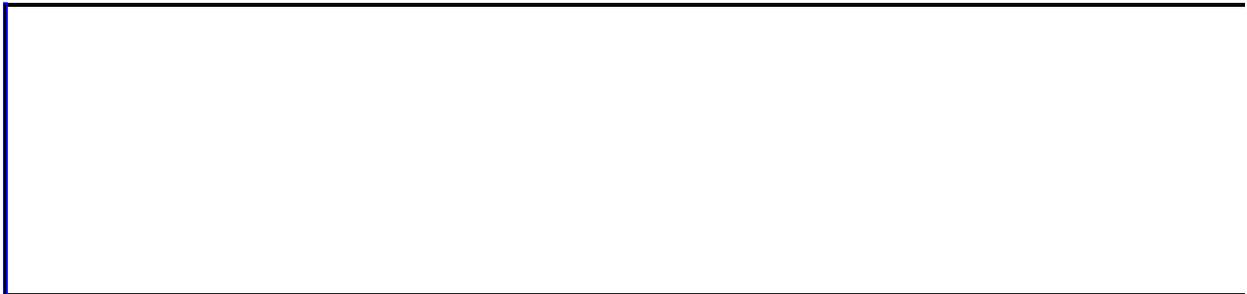
This example replaces the Times New Roman font with the Courier font in the active presentation.

```
Application.ActivePresentation.Fonts _  
    .Replace "Times New Roman", "Courier"
```

[As it applies to the **DefaultWebOptions** object.](#)

This example sets the fixed-width font default Web option, specified by the character set constant **msoCharacterSetEnglishWesternEuropeanOtherLatinScript**, to be Courier New 10 points.

```
With Application.DefaultWebOptions _  
    .Fonts(msoCharacterSetEnglishWesternEuropeanOtherLatinScript)  
        .FixedWidthFont = "Courier New"  
        .FixedWidthFontSize = 10  
End With
```



FontSize Property

Returns or sets the font size for the specified WordArt, in points. Read/write **Single**.

Example

This example sets the font size to 16 points for the shape named "WordArt 4" in myDocument.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes("WordArt 4").TextEffect.FontSize = 16
```



Footer Property

Returns a [HeaderFooter](#) object that represents the footer that appears at the bottom of a slide or in the lower-left corner of a notes page, handout, or outline. Read-only.

Example

This example sets the text for the footer on the slide master in the active presentation and sets the footer, date and time, and slide number to appear on the title slide.

```
With Application.ActivePresentation.SlideMaster.HeadersFooters
    .Footer.Text = "Introduction"
    .DisplayOnTitleSlide = True
End With
```



ForeColor Property

Returns or sets a [ColorFormat](#) object that represents the foreground color for the fill, line, or shadow. Read/write.

Example

This example adds a rectangle to myDocument and then sets the foreground color, background color, and gradient for the rectangle's fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 90, 90, 90, 50).Fill
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(170, 170, 170)
    .TwoColorGradient msoGradientHorizontal, 1
End With
```

This example adds a patterned line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 100, 250, 0).Line
    .Weight = 6
    .ForeColor.RGB = RGB(0, 0, 255)
    .BackColor.RGB = RGB(128, 0, 0)
    .Pattern = msoPatternDarkDownwardDiagonal
End With
```



[Show All](#)

Format Property

Returns or sets the format for the automatically updated date and time. Applies only to **HeaderFooter** objects that represent a date and time (returned from the **HeadersFooters** collection by the [DateAndTime](#) property). Read/write [PpDateTimeFormat](#).

PpDateTimeFormat can be one of these PpDateTimeFormat constants.

ppDateTimeddMMddMMddyyyy

ppDateTimedMMMMyyyy

ppDateTimedMMMyy

ppDateTimeFormatMixed

ppDateTimeHm

ppDateTimehmAMPM

ppDateTimeHmss

ppDateTimehmssAMPM

ppDateTimemdy

ppDateTimeMMddyyHm

ppDateTimeMMddyyhmmAMPM

ppDateTimeMMMMyyyy

ppDateTimeMMMMyy

ppDateTimeMMyy

expression.**Format**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Make sure that the date and time are set to be updated automatically (not displayed as fixed text) by setting the [UseFormat](#) property to **True**.

Example

This example sets the date and time for the slide master of the active presentation to be updated automatically and then it sets the date and time format to show hours, minutes, and seconds.

```
Set myPres = Application.ActivePresentation
With myPres.SlideMaster.HeadersFooters.DateAndTime
    .UseFormat = True
    .Format = ppDateTimeHmss
End With
```



Formula Property

Returns or sets a **String** that represents a formula to use for calculating an animation. Read/write.

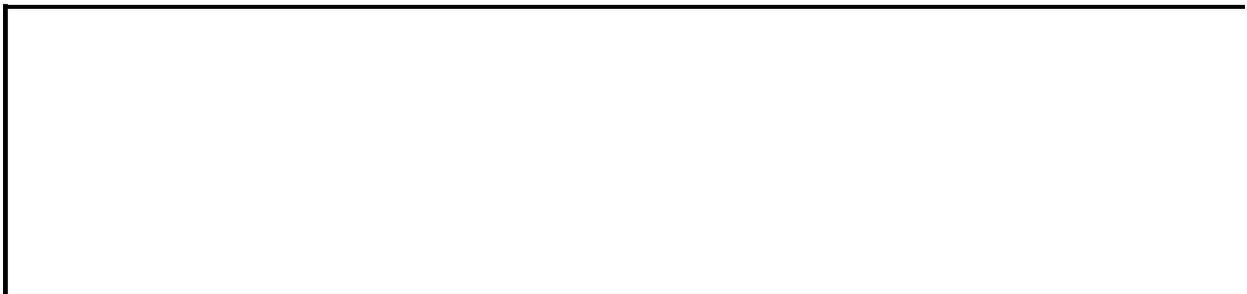
expression.**Formula**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, and adds a three-second fill animation to that shape.

```
Sub AddShapeSetAnimFill()  
  
    Dim effBlinds As Effect  
    Dim shpRectangle As Shape  
    Dim animBlinds As AnimationBehavior  
  
    'Adds rectangle and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effBlinds = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectBlind  
  
    'Sets the duration of the animation  
    effBlinds.Timing.Duration = 3  
  
    'Adds a behavior to the animation  
    Set animBlinds = effBlinds.Behaviors.Add(msoAnimTypeProperty)  
  
    'Sets the animation color effect and the formula to use  
    With animBlinds.PropertyEffect  
        .Property = msoAnimColor  
        .Formula = RGB(Red:=255, Green:=255, Blue:=255)  
    End With  
  
End Sub
```



[Show All](#)

FrameColors Property

Returns or sets the text color for the outline pane and the background color for the outline and slide panes for Web presentations. Read/write [PpFrameColors](#).

PpFrameColors can be one of these PpFrameColors constants.

ppFrameColorsBlackTextOnWhite

ppFrameColorsBrowserColors

ppFrameColorsPresentationSchemeAccentColor

ppFrameColorsPresentationSchemeTextColor

ppFrameColorsWhiteTextOnBlack

expression.FrameColors

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example specifies that when saving or publishing the active presentation as a Web page, the text color for the outline pane is white and the background color for the outline and slide panes is black, and Portable Network Graphics (PNG) are allowed as an image format.

```
With ActivePresentation.WebOptions
    .FrameColors = ppFrameColorsWhiteTextOnBlack
    .AllowPNG = True
End With
```



[Show All](#)

FrameSlides Property

Determines whether a thin frame is placed around the border of the printed slides. Read/write [MsoTriState](#). Applies to printed slides, handouts, and notes pages.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue A thin frame is placed around the border of the printed slides.

Example

This example prints the active presentation with a frame around each slide.

```
With ActivePresentation  
    .PrintOptions.FrameSlides = msoTrue  
    .PrintOut  
End With
```



[Show All](#)

From Property

 [From property as it applies to the **ColorEffect** object.](#)

Sets or returns a **ColorFormat** object that represents the starting RGB color value of an animation behavior.

expression.**From**

expression Required. An expression that returns a [ColorEffect](#) object.

Remarks

Use this property in conjunction with the [To](#) property to transition from one color to another.



[From property as it applies to the **RotationEffect** object.](#)

Sets or returns a **Single** that represents the starting angle in degrees, specified relative to the screen (for example, 90 degrees is completely horizontal).
Read/write.

expression.**From**

expression Required. An expression that returns a [RotationEffect](#) object.

Remarks

Use this property in conjunction with the [To](#) property to transition from one rotation angle to another.

The default value is **Empty** in which case the current position of the object is used.

[From property as it applies to the **PropertyEffect** object.](#)

Sets or returns a **Variant** that represents the starting value of an object's property. Read/write.

expression.**From**

expression Required. An expression that returns a [PropertyEffect](#) object.

Remarks

The **From** property is similar to the [Points](#) property, but using the **From** property is easier for simple tasks.

The default value is **Empty**, in which case the current position of the object is used.

Remarks

Do not confuse this property with the [FromX](#) or [FromY](#) properties of the [ScaleEffect](#) and [MotionEffect](#) objects, which are only used for scaling or motion effects.

Example

 [As it applies to the **ColorEffect** object.](#)

The following example adds a color effect and immediately changes its color.

```
Sub AddAndChangeColorEffect()  
    Dim effBlinds As Effect  
    Dim tlnTiming As TimeLine  
    Dim shpRectangle As Shape  
    Dim animColorEffect As AnimationBehavior  
    Dim clrEffect As ColorEffect  
  
    'Adds rectangle and sets effect and animation  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effBlinds = t.MainSequence.AddEffect(Shape:=shpRectangle, _  
        effectId:=msoAnimEffectBlinds)  
    Set animColorEffect = tlnTiming.MainSequence(1).Behaviors _  
        .Add(Type:=msoAnimTypeColor)  
    Set clrEffect = animColorEffect.ColorEffect  
  
    'Sets the animation effect starting and ending colors  
    clrEffect.From.RGB = RGB(Red:=255, Green:=255, Blue:=0)  
    clrEffect.To.RGB = RGB(Red:=0, Green:=255, Blue:=255)  
End Sub
```

 [As it applies to the **RotationEffect** object.](#)

The following example adds a rotation effect and immediately changes its rotation angle.

```
Sub AddAndChangeRotationEffect()  
    Dim effBlinds As Effect  
    Dim tlnTiming As TimeLine  
    Dim shpRectangle As Shape  
    Dim animRotation As AnimationBehavior  
    Dim rtnEffect As RotationEffect  
  
    'Adds rectangle and sets effect and animation  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _
```

```
        Top:=100, Width:=50, Height:=50)
Set tlnTiming = ActivePresentation.Slides(1).TimeLine
Set effBlinds = tlnTiming.MainSequence.AddEffect(Shape:=shpRecta
effectId:=msoAnimEffectBlinds)
Set animRotation = tlnTiming.MainSequence(1).Behaviors _
    .Add(Type:=msoAnimTypeRotation)
Set rtnEffect = animRotation.RotationEffect

'Sets the rotation effect starting and ending positions
rtnEffect.From = 90
rtnEffect.To = 270
End Sub
```



FromX Property

Sets or returns a **Single** that represents the starting width or horizontal position of a [ScaleEffect](#) or [MotionEffect](#) object, respectively, specified as a percent of the screen width. Read/write.

expression.**FromX**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The default value of this property is **Empty**, in which case the current position of the object is used.

Use this property in conjunction with the [ToX](#) property to resize or jump from one position to another.

Do not confuse this property with the [From](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set or change colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds a motion path and sets the starting and ending horizontal and vertical positions.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animMotion As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets effect and animation properties  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectCusto  
    Set animMotion = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Sets starting and ending horizontal and vertical positions  
    With animMotion.MotionEffect  
        .FromX = 0  
        .FromY = 0  
        .ToX = 50  
        .ToY = 50  
    End With  
  
End Sub
```



FromY Property

Returns or sets a **Single** that represents the starting height or vertical position of a **ScaleEffect** or **MotionEffect** object, respectively, specified as a percentage of the screen width. Read/write.

expression.**FromY**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The default value of this property is **Empty**, in which case the current position of the object is used.

Use this property in conjunction with the [ToY](#) property to resize or jump from one position to another.

Do not confuse this property with the [From](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set or change colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds an animation path and sets the starting and ending horizontal and vertical positions.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animMotion As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets effect and animation properties  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectCusto  
    Set animMotion = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Sets starting and ending horizontal and vertical positions  
    With animMotion.MotionEffect  
        .FromX = 0  
        .FromY = 0  
        .ToX = 50  
        .ToY = 50  
    End With  
  
End Sub
```



FullName Property

Returns the name of the specified add-in or saved presentation, including the path, the current file system separator, and the file name extension. Read-only **String**.

Remarks

This property is equivalent to the [Path](#) property, followed by the current file system separator, followed by the [Name](#) property.

Example

This example displays the path and file name of every available add-in.

```
For Each a In Application.AddIns  
    MsgBox a.FullName  
Next a
```

This example displays the path and file name of the active presentation (assuming that the presentation has been saved).

```
MsgBox Application.ActivePresentation.FullName
```



Gap Property

Returns or sets the horizontal distance (in points) between the end of the callout line and the text bounding box. Read/write **Single**.

Example

This example sets the distance between the callout line and the text bounding box to 3 points for shape one on myDocument. For the example to work, shape one must be a callout.

```
Set myDocument = ActivePresentation.Slides(1)  
myDocument.Shapes(1).Callout.Gap = 3
```



[Show All](#)

GradientColorType Property

Returns the gradient color type for the specified fill. This property is read-only. Use the [OneColorGradient](#), [PresetGradient](#), or [TwoColorGradient](#) method to set the gradient type for the fill. Read-only [MsoGradientColorType](#).

MsoGradientColorType can be one of these MsoGradientColorType constants.

msoGradientColorMixed

msoGradientOneColor

msoGradientPresetColors

msoGradientTwoColors

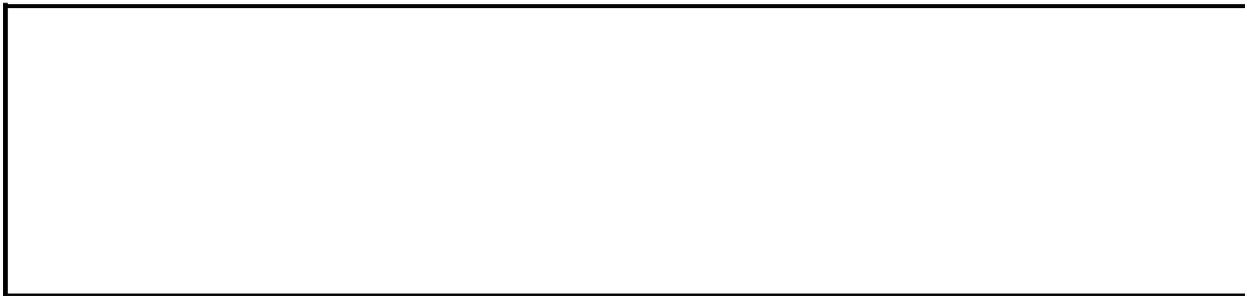
expression.**GradientColorType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example changes the fill for all shapes in myDocument that have a two-color gradient fill to a preset gradient fill.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  With s.Fill
    If .GradientColorType = msoGradientTwoColors Then
      .PresetGradient msoGradientHorizontal, _
        1, msoGradientBrass
    End If
  End With
Next
```



GradientDegree Property

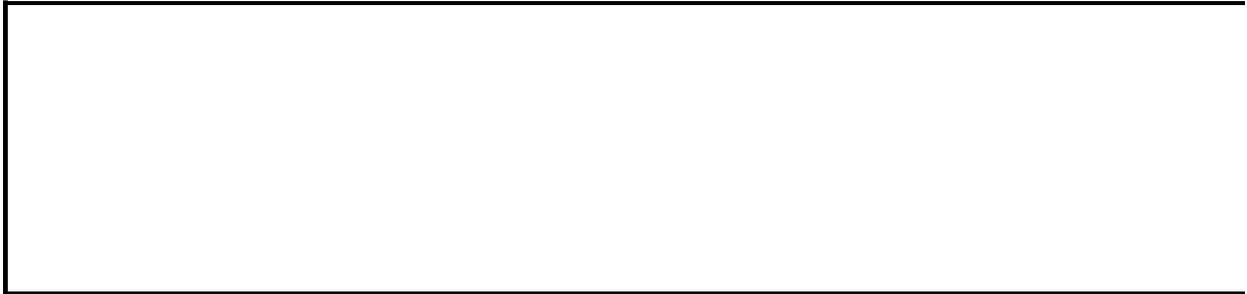
Returns a value that indicates how dark or light a one-color gradient fill is. A value of 0 (zero) means that black is mixed in with the shape's foreground color to form the gradient; a value of 1 means that white is mixed in; and values between 0 and 1 mean that a darker or lighter shade of the foreground color is mixed in. Read-only **Single**.

This property is read-only. Use the [OneColorGradient](#) method to set the gradient degree for the fill.

Example

This example adds a rectangle to myDocument and sets the degree of its fill gradient to match that of the shape named "Rectangle 2." If Rectangle 2 doesn't have a one-color gradient fill, this example fails.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    gradDegree1 = .Item("Rectangle 2").Fill.GradientDegree
    With .AddShape(msoShapeRectangle, 0, 0, 40, 80).Fill
        .ForeColor.RGB = RGB(128, 0, 0)
        .OneColorGradient msoGradientHorizontal, 1, gradDegree1
    End With
End With
```



[Show All](#)

GradientStyle Property

Returns the gradient style for the specified fill. Use the [OneColorGradient](#), [PresetGradient](#), or [TwoColorGradient](#) method to set the gradient style for the fill. Attempting to return this property for a fill that doesn't have a gradient generates an error. Use the [Type](#) property to determine whether the fill has a gradient. Read-only [MsoGradientStyle](#).

MsoGradientStyle can be one of these MsoGradientStyle constants.

msoGradientDiagonalDown

msoGradientDiagonalUp

msoGradientFromCenter

msoGradientFromCorner

msoGradientFromTitle

msoGradientHorizontal

msoGradientMixed

msoGradientVertical

expression.**GradientStyle**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a rectangle to myDocument and sets its fill gradient style to match that of the shape named "rect1." For the example to work, rect1 must have a gradient fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    gradStyle1 = .Item("rect1").Fill.GradientStyle
    With .AddShape(msoShapeRectangle, 0, 0, 40, 80).Fill
        .ForeColor.RGB = RGB(128, 0, 0)
        .OneColorGradient gradStyle1, 1, 1
    End With
End With
```



GradientVariant Property

Returns the gradient variant for the specified fill as an integer value from 1 to 4 for most gradient fills. If the gradient style is **msoGradientFromTitle** or **msoGradientFromCenter**, this property returns either 1 or 2. The values for this property correspond to the gradient variants (numbered from left to right and from top to bottom) on the **Gradient** tab in the **Fill Effects** dialog box. Read-only **Long**.

This property is read-only. Use the [OneColorGradient](#), [PresetGradient](#), or [TwoColorGradient](#) method to set the gradient variant for the fill.

Example

This example adds a rectangle to myDocument and sets its fill gradient variant to match that of the shape named "rect1." For the example to work, rect1 must have a gradient fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    gradVar1 = .Item("rect1").Fill.GradientVariant
    With .AddShape(msoShapeRectangle, 0, 0, 40, 80).Fill
        .ForeColor.RGB = RGB(128, 0, 0)
        .OneColorGradient msoGradientHorizontal, gradVar1, 1
    End With
End With
```



GridDistance Property

Sets or returns a **Single** that represents the distance between grid lines.
Read/write.

expression.**GridDistance**

expression Required. An expression that returns an [Presentation](#) object.

Example

This example displays the gridlines, and then specifies the distance between grid lines and enables the snap to grid setting.

```
Sub SetGridLines()  
    Application.DisplayGridLines = msoTrue  
    With ActivePresentation  
        .GridDistance = 18  
        .SnapToGrid = msoTrue  
    End With  
End Sub
```



GroupItems Property

Returns a [GroupShapes](#) object that represents the individual shapes in the specified group. Use the [Item](#) method of the **GroupShapes** object to return a single shape from the group. Applies to **Shape** or **ShapeRange** objects that represent grouped shapes. Read-only.

Example

This example adds three triangles to myDocument, groups them, sets a color for the entire group, and then changes the color for the second triangle only.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeIsoscelesTriangle, 10, _
        10, 100, 100).Name = "shpOne"
    .AddShape(msoShapeIsoscelesTriangle, 150, _
        10, 100, 100).Name = "shpTwo"
    .AddShape(msoShapeIsoscelesTriangle, 300, _
        10, 100, 100).Name = "shpThree"
    With .Range(Array("shpOne", "shpTwo", "shpThree")).Group
        .Fill.PresetTextured msoTextureBlueTissuePaper
        .GroupItems(2).Fill.PresetTextured msoTextureGreenMarble
    End With
End With
```



HandoutMaster Property

Returns a [Master](#) object that represents the handout master. Read-only.

Example

This example sets the background pattern on the handout master in the active presentation.

```
Application.ActivePresentation.HandoutMaster.Background.Fill _  
    .Patterned msoPatternDarkHorizontal
```



[Show All](#)

HandoutOrder Property

Returns or sets the page layout order in which slides appear on printed handouts that show multiple slides on one page. Read/write [PpPrintHandoutOrder](#).

PpPrintHandoutOrder can be one of these PpPrintHandoutOrder constants.

ppPrintHandoutHorizontalFirst Slides are ordered horizontally, with the first slide in the upper-left corner and the second slide to the right of it. If your language setting specifies a right-to-left language, the first slide is in the upper-right corner with the second slide to the left of it.

ppPrintHandoutVerticalFirst Slides are ordered vertically, with the first slide in the upper-left corner and the second slide below it. If your language setting specifies a right-to-left language, the first slide is in the upper-right corner with the second slide below it.

expression.**HandoutOrder**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets handouts of the active presentation to contain six slides per page, orders the slides horizontally on the handouts, and prints them.

```
With ActivePresentation
    .PrintOptions.OutputType = ppPrintOutputSixSlideHandouts
    .PrintOptions.HandoutOrder = ppPrintHandoutHorizontalFirst
    .PrintOut
End With
```



[Show All](#)

HangingPunctuation Property

Returns or sets the hanging punctuation option if you have an Asian language setting specified. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The hanging punctuation option is selected.

Example

This example selects hanging punctuation for the first paragraph of the active presentation.

```
ActivePresentation.Paragraphs(1).HangingPunctuation = msoTrue
```



HasChildShapeRange Property

True if the selection contains child shapes. Read-only **Boolean**.

expression.**HasChildShapeRange**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example creates a new slide with a drawing canvas, populates the drawing canvas with shapes, and selects the shapes added to the canvas. Then after checking that the shapes selected are child shapes, it fills the child shapes with a pattern.

```
Sub ChildShapes()  
    Dim sldNew As Slide  
    Dim shpCanvas As Shape  
  
    'Create a new slide with a drawing canvas and shapes  
    Set sldNew = Presentations(1).Slides _  
        .Add(Index:=1, Layout:=ppLayoutBlank)  
    Set shpCanvas = sldNew.Shapes.AddCanvas( _  
        Left:=100, Top:=100, Width:=200, Height:=200)  
  
    With shpCanvas.CanvasItems  
        .AddShape msoShapeRectangle, Left:=0, Top:=0, _  
            Width:=100, Height:=100  
        .AddShape msoShapeOval, Left:=0, Top:=50, _  
            Width:=100, Height:=100  
        .AddShape msoShapeDiamond, Left:=0, Top:=100, _  
            Width:=100, Height:=100  
    End With  
  
    'Select all shapes in the canvas  
    shpCanvas.CanvasItems.SelectAll  
  
    'Fill canvas child shapes with a pattern  
    With ActiveWindow.Selection  
        If .HasChildShapeRange = True Then  
            .ChildShapeRange.Fill.Patterned Pattern:=msoPatternDivot  
        Else  
            MsgBox "This is not a range of child shapes."  
        End If  
    End With  
  
End Sub
```



[Show All](#)

HasDiagram Property

MsoTrue if a shape is a diagram. Read-only [MsoTriState](#).

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Shape is not a diagram.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Shape is a diagram.

expression.**HasDiagram**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example searches the current document for diagrams with nodes and if it finds both, creates a black balloon with bold white text.

```
Sub HasDiagramProperties()  
    Dim shpDiagram As Shape  
    Dim shpNode As DiagramNode  
    Dim shpBalloon As Shape  
    Dim sldFirst As Slide  
  
    Set sldFirst = ActivePresentation.Slides(1)  
  
    'Looks through the current document and when it finds a diagram  
    'with one or more diagram nodes, creates a balloon with text  
    For Each shpDiagram In sldFirst.Shapes  
        If shpDiagram.HasDiagram = msoTrue And _  
            shpDiagram.HasDiagramNode = msoTrue Then  
            Set shpBalloon = sldFirst.Shapes.AddShape( _  
                Type:=msoShapeBalloon, Left:=350, _  
                Top:=75, Width:=150, Height:=150)  
            With shpBalloon  
                With .TextFrame  
                    .WordWrap = msoTrue  
                    With .TextRange  
                        .Text = "This is a diagram with nodes."  
                        .Font.Color.RGB = RGB(Red:=255, _  
                            Green:=255, Blue:=255)  
                        .Font.Bold = True  
                        .Font.Name = "Tahoma"  
                        .Font.Size = 15  
                    End With  
                End With  
                .Line.BackColor.RGB = RGB( _  
                    Red:=0, Green:=25, Blue:=25)  
                .Fill.ForeColor.RGB = RGB( _  
                    Red:=0, Green:=25, Blue:=25)  
            End With  
        End If  
    Next shpDiagram  
End Sub
```



[Show All](#)

HasDiagramNode Property

MsoTrue if a shape is a diagram node. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue Doesn't apply to this property.

msoFalse Shape is not a diagram node.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Shape is a diagram node.

expression.**HasDiagramNode**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example searches the current document for diagrams with nodes, and if it finds both, creates a black balloon with bold white text.

```
Sub HasDiagramProperties()  
    Dim shpDiagram As Shape  
    Dim shpNode As DiagramNode  
    Dim shpBalloon As Shape  
    Dim sldFirst As Slide  
  
    Set sldFirst = ActivePresentation.Slides(1)  
  
    'Looks through the current document and when it finds a diagram  
    ' with one or more diagram nodes, creates a balloon with text  
    For Each shpDiagram In sldFirst.Shapes  
        If shpDiagram.HasDiagram = msoTrue And _  
            shpDiagram.HasDiagramNode = msoTrue Then  
            Set shpBalloon = sldFirst.Shapes.AddShape( _  
                Type:=msoShapeBalloon, Left:=350, _  
                Top:=75, Width:=150, Height:=150)  
            With shpBalloon  
                With .TextFrame  
                    .WordWrap = msoTrue  
                    With .TextRange  
                        .Text = "This is a diagram with nodes."  
                        .Font.Color.RGB = RGB(Red:=255, _  
                            Green:=255, Blue:=255)  
                        .Font.Bold = True  
                        .Font.Name = "Tahoma"  
                        .Font.Size = 15  
                    End With  
                End With  
                .Line.BackColor.RGB = RGB( _  
                    Red:=0, Green:=25, Blue:=25)  
                .Fill.ForeColor.RGB = RGB( _  
                    Red:=0, Green:=25, Blue:=25)  
            End With  
        End If  
    Next shpDiagram  
End Sub
```



[Show All](#)

HasRevisionInfo Property

Returns a [PpRevisionInfo](#) constant that represents whether a presentation is a merged author document, a reviewer document with base lines, or a regular Microsoft PowerPoint document. Read-only.

PpRevisionInfo can be one of these **PpRevisionInfo** constants.

ppRevisionInfoBaseline The presentation has a baseline.

ppRevisionInfoMerged The presentation is a merged author presentation.

ppRevisionInfoNone The presentation has no reviewer information.

expression.**HasRevisionInfo**

expression Required. An expression that returns one of the objects in the Applies To list.

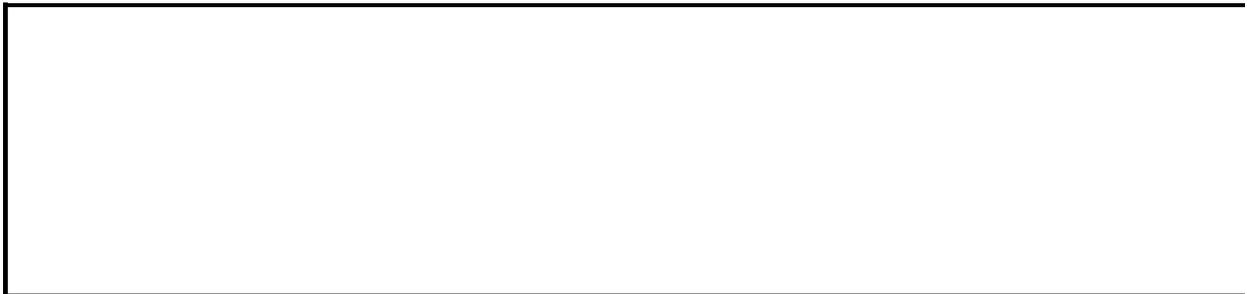
Remarks

Comments are not considered reviewer information.

Example

The following example adds and removes baselines, reporting status to the user along the way.

```
Sub AddABaseline()  
    ActivePresentation.AddBaseline  
    Call ReportRevisionInfo(ActivePresentation)  
    ActivePresentation.RemoveBaseline  
    Call ReportRevisionInfo(ActivePresentation)  
End Sub  
  
Sub ReportRevisionInfo(pre As Presentation)  
    Select Case pre.HasRevisionInfo  
        Case ppRevisionInfoBaseline  
            MsgBox "The presentation has a base line."  
        Case ppRevisionInfoMerged  
            MsgBox "The presentation is a merged author presentation"  
        Case ppRevisionInfoNone  
            MsgBox "The presentation has no reviewer information."  
        Case Else  
            MsgBox "Couldn't determine revision information."  
    End Select  
End Sub
```



[Show All](#)

HasTable Property

Returns whether the specified shape is a table. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is a table.

Example

This example checks the currently selected shape to see if it is a table. If it is, the code sets the width of column one to one inch (72 points).

```
With ActiveWindow.Selection.ShapeRange
    If .HasTable = msoTrue Then
        .Table.Columns(1).Width = 72
    End If
End With
```



[Show All](#)

HasText Property

Returns whether the specified shape has text associated with it. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

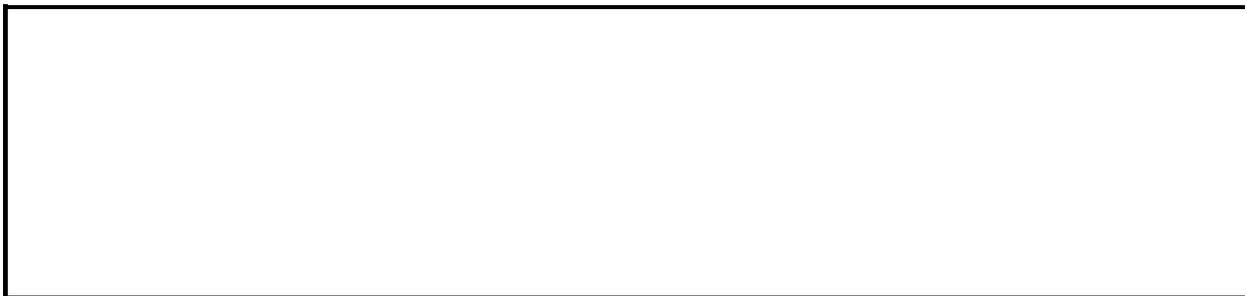
msoTriStateToggle

msoTrue The specified shape has text associated with it.

Example

If shape two on myDocument contains text, this example resizes the shape to fit the text.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(2).TextFrame
    If .HasText Then .AutoSize = ppAutoSizeShapeToFitText
End With
```



[Show All](#)

HasTextFrame Property

Returns whether the specified shape has a text frame. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape has a text frame and can therefore contain text.

Example

This example extracts text from all shapes on the first slide that contain text frames, and then it stores the names of these shapes and the text they contain in an array.

```
Dim shpTextArray() As Variant
Dim numShapes, numAutoShapes, i As Long

Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    numShapes = .Count
    If numShapes > 1 Then
        numTextShapes = 0
        ReDim shpTextArray(1 To 2, 1 To numShapes)
        For i = 1 To numShapes
            If .Item(i).HasTextFrame Then
                numTextShapes = numTextShapes + 1
                shpTextArray(numTextShapes, 1) = .Item(i).Name
                shpTextArray(numTextShapes, 2) = .Item(i) _
                    .TextFrame.TextRange.Text
            End If
        Next
        ReDim Preserve shpTextArray(1 To 2, 1 To numTextShapes)
    End If
End With
```



[Show All](#)

HasTitle Property

Returns whether the collection of objects on the specified slide contains a title placeholder. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

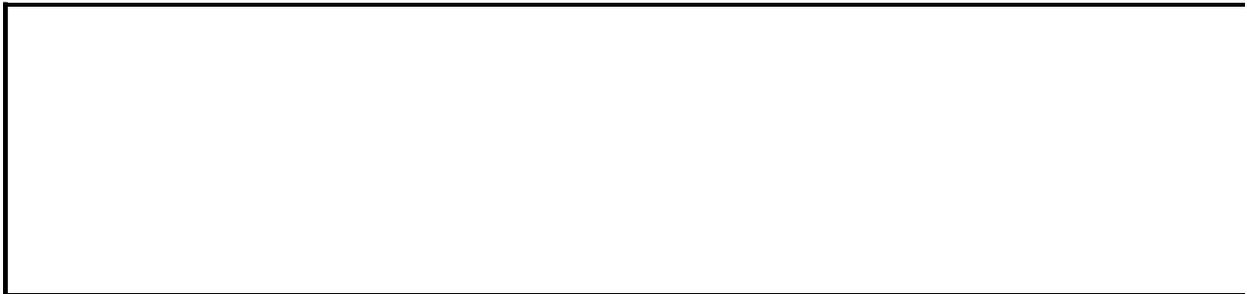
msoTriStateToggle

msoTrue The collection of objects on the specified slide contains a title placeholder.

Example

This example restores the title placeholder to slide one in the active presentation if this placeholder has been deleted. The text of the restored title is "Restored title."

```
With ActivePresentation.Slides(1)
  If .Layout <> ppLayoutBlank Then
    With .Shapes
      If Not .HasTitle Then
        .AddTitle.TextFrame.TextRange _
          .Text = "Restored title"
      End If
    End With
  End If
End With
```



[Show All](#)

HasTitleMaster Property

MsoTrue if the specified presentation has a title master. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified presentation has a title master.

expression.**HasTitleMaster**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a title master to the active presentation if it doesn't already have one.

```
With Application.ActivePresentation  
    If Not .HasTitleMaster Then .AddTitleMaster  
End With
```



Header Property

Returns a [HeaderFooter](#) object that represents the header that appears at the top of a slide or in the upper-left corner of a notes page, handout, or outline. Read-only.

Example

This example sets the header text for the handout master for the active presentation. This text will appear in the upper-left corner of the page when you print your presentation as an outline or a handout.

```
Set myHandHF = Application.ActivePresentation.HandoutMaster _  
    .HeadersFooters  
myHandHF.Header.Text = "Third Quarter Report"
```



HeadersFooters Property

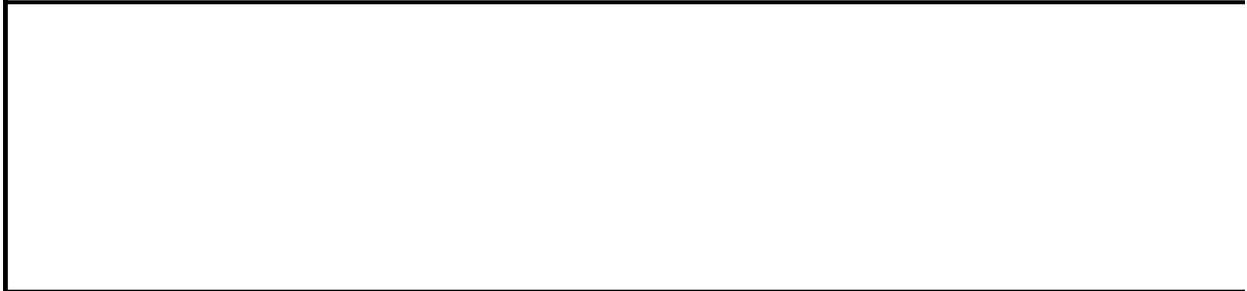
Returns a [HeadersFooters](#) collection that represents the header, footer, date and time, and slide number associated with the slide, slide master, or range of slides. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example sets the footer text and the date and time format for the notes master in the active presentation and sets the date and time to be updated automatically.

```
With ActivePresentation.NotesMaster.HeadersFooters  
    .Footer.Text = "Regional Sales"  
    With .DateAndTime  
        .UseFormat = True  
        .Format = ppDateTimeHmss  
    End With  
End With
```



Height Property

Returns or sets the height of the specified object, in points. Read-only **Single** for the [Master](#) object, read/write **Single** for all other objects.

Remarks

The **Height** property of a **Shape** object returns or sets the height of the forward-facing surface of the specified shape. This measurement doesn't include shadows or 3-D effects.

Example

This example sets the height of document window two to half the height of the application window.

```
Windows(2).Height = Application.Height / 2
```

This example sets the height for row two in the specified table to 100 points (72 points per inch).

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows(2).Height = 100
```



[Show All](#)

Hidden Property

Determines whether the specified slide is hidden during a slide show. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

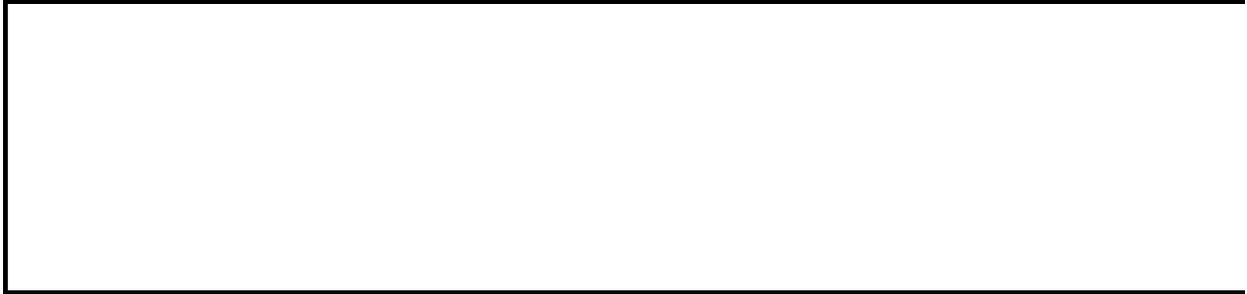
msoTriStateToggle

msoTrue The specified slide is hidden during a slide show.

Example

This example makes slide two in the active presentation a hidden slide.

```
ActivePresentation.Slides(2).SlideShowTransition.Hidden = msoTrue
```



[Show All](#)

HideWhileNotPlaying Property

Determines whether the specified media clip is hidden during a slide show except when it's playing. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified media clip is hidden during a slide show except when it's playing.

Example

This example inserts a movie named "Clock.avi" onto slide one in the active presentation, sets it to play automatically after the slide transition, and specifies that the movie object be hidden during a slide show except when it's playing.

```
With ActivePresentation.Slides(1).Shapes _  
    .AddOLEObject(Left:=10, Top:=10, _  
        Width:=250, Height:=250, _  
        FileName:="c:\winnt\clock.avi")  
    With .AnimationSettings.PlaySettings  
        .PlayOnEntry = True  
        .HideWhileNotPlaying = msoTrue  
    End With  
End With
```



[Show All](#)

HorizontalAnchor Property

Returns or sets the horizontal alignment of text in a text frame. Read/write [MsoHorizontalAnchor](#).

MsoHorizontalAnchor can be one of these MsoHorizontalAnchor constants.

msoAnchorNone

msoHorizontalAnchorMixed

msoAnchorCenter

expression.**HorizontalAnchor**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the alignment of the text in shape one on myDocument to top centered.

```
Set myDocument = ActivePresentation.SlideMaster
With myDocument.Shapes(1)
    .TextFrame.HorizontalAnchor = msoAnchorCenter
    .TextFrame.VerticalAnchor = msoAnchorTop
End With
```



[Show All](#)

HorizontalFlip Property

Returns whether the specified shape is flipped around the horizontal axis. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is flipped around the horizontal axis.

Example

This example restores each shape on myDocument to its original state, if it's been flipped horizontally or vertically.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.HorizontalFlip Then s.Flip msoFlipHorizontal
    If s.VerticalFlip Then s.Flip msoFlipVertical
Next
```



HTMLProject Property

Returns the **HTMLProject** object, which is a Web presentation (HTML format) accessed through the Microsoft Script Editor. Read-only.

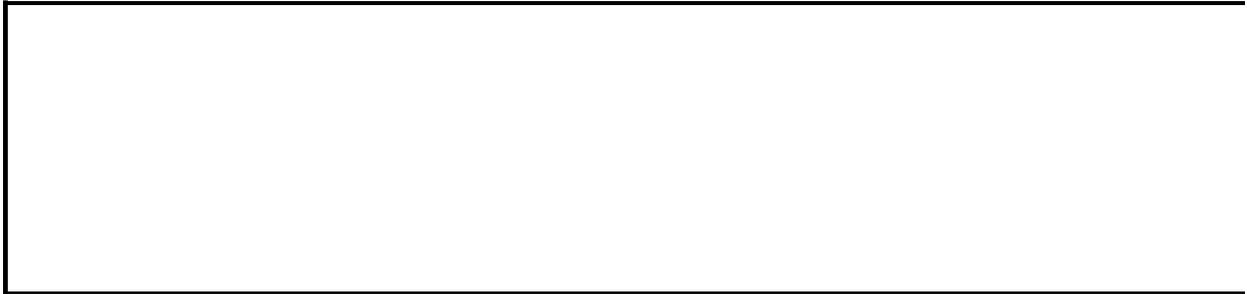
Remarks

The **HTMLProject** object can be interpreted as the top-level project branch in the Project Explorer window of the Script Editor, for a loaded presentation. It contains the **HTMLProjectItems** collection. Members of the **HTMLProjectItems** collection represent a slide, master, or the handout for the Web presentation.

Example

This example checks the name of each member in the **HTMLProjectItems** collection for the loaded **HTMLProject**. If the name is Slide2, it then opens the generated HTML for that slide in the Microsoft Script Editor.

```
Dim i As Integer
With ActivePresentation.HTMLProject
    For i = 1 To .HTMLProjectItems.Count
        If .HTMLProjectItems(i).Name = "Slide2" Then
            .HTMLProjectItems(i).Open
        End If
    Next
End With
```



[Show All](#)

HTMLVersion Property

Returns or sets the version of HTML for a published presentation. Read/write [PpHTMLVersion](#).

PpHTMLVersion can be one of these PpHTMLVersion constants.

ppHTMLAutodetect

ppHTMLDual

ppHTMLv3

ppHTMLv4 Default.

expression.**HTMLVersion**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example publishes slides three through five of the active presentation in HTML version 3.0. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .HTMLVersion = ppHTMLv3
    .Publish
End With
```



Hyperlink Property

Returns a [Hyperlink](#) object that represents the hyperlink for the specified shape. For the hyperlink to be active during a slide show, the [Action](#) property must be set to **ppActionHyperlink**. Read-only.

Example

This example sets shape one on slide one in the active presentation to jump to the Microsoft Web site when the shape is clicked during a slide show.

```
With ActivePresentation.Slides(1).Shapes(1) _  
    .ActionSettings(ppMouseClick)  
    .Action = ppActionHyperlink  
    .Hyperlink.Address = "http://www.microsoft.com/"  
End With
```



Hyperlinks Property

Returns a [Hyperlinks](#) collection that represents all the hyperlinks on the specified slide. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example allows the user to update an outdated internet address for all hyperlinks in the active presentation.

```
oldAddr = InputBox("Old internet address")
newAddr = InputBox("New internet address")
For Each s In ActivePresentation.Slides
    For Each h In s.Hyperlinks
        If LCase(h.Address) = oldAddr Then h.Address = newAddr
    Next
Next
```



Id Property

Returns a **Long** that identifies the shape or range of shapes. Read-only.

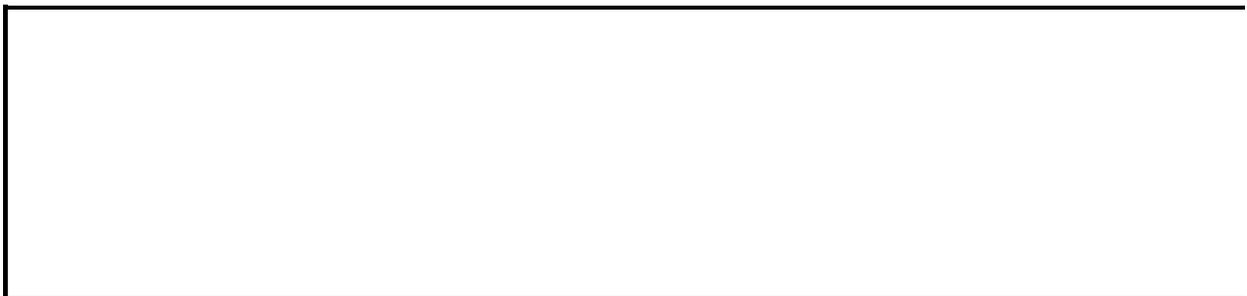
expression.**Id**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a new shape to the active presentation, then fills the shape according to the value of the **ID** property.

```
Sub ShapeID()  
  With ActivePresentation.Slides(1).Shapes.AddShape _  
    (Type:=msoShape5pointStar, Left:=100, _  
     Top:=100, Width:=100, Height:=100)  
  
    Select Case .Id  
      Case 0 To 500  
        .Fill.ForeColor.RGB = RGB(Red:=255, Green:=0, Blue:=  
      Case 500 To 1000  
        .Fill.ForeColor.RGB = RGB(Red:=255, Green:=255, Blue  
      Case 1000 To 1500  
        .Fill.ForeColor.RGB = RGB(Red:=255, Green:=0, Blue:=  
      Case 1500 To 2000  
        .Fill.ForeColor.RGB = RGB(Red:=0, Green:=255, Blue:=  
      Case 2000 To 2500  
        .Fill.ForeColor.RGB = RGB(Red:=0, Green:=255, Blue:=  
      Case Else  
        .Fill.ForeColor.RGB = RGB(Red:=0, Green:=0, Blue:=25  
    End Select  
  End With  
End Sub
```



[Show All](#)

IncludeNavigation Property

Determines whether the link bar for Web presentations is visible. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The navigation bar for Web presentations is not visible, which enlarges the slide.

msoTriStateMixed

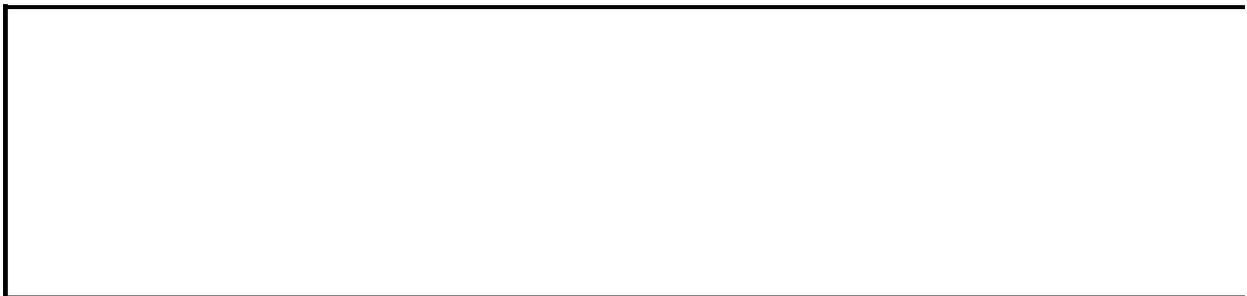
msoTriStateToggle

msoTrue Default. The navigation bar for Web presentations is visible at the bottom of the page.

Example

This example specifies that the link bar is not to be included in the specified Web presentation. It then previews the presentation in the active Web browser.

```
With Presentations(2)  
    .WebOptions.IncludeNavigation = msoFalse  
    .WebPagePreview  
End With
```



IndentLevel Property

Returns or sets the the indent level for the specified text as an integer from 1 to 5, where 1 indicates a first-level paragraph with no indentation. Read/write **Long**.

Example

This example indents the second paragraph in shape two on slide two in the active presentation.

```
Application.ActivePresentation.Slides(2).Shapes(2).TextFrame _  
    .TextRange.Paragraphs(2).IndentLevel1 = 2
```



Index Property

Returns a **Long** that represents the index number for an animation effect or design. Read-only.

expression.**Index**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example displays the name and index number for all effects in the main animation sequence of the first slide.

```
Sub EffectInfo()  
    Dim effIndex As Effect  
    Dim seqMain As Sequence  
    Set seqMain = ActivePresentation.Slides(1).TimeLine.MainSequence  
    For Each effIndex In seqMain  
        With effIndex  
            MsgBox "Effect Name: " & .DisplayName & vbLf & _  
                "Effect Index: " & .Index  
        End With  
    Next  
End Sub
```



[Show All](#)

InsetPen Property

MsoTrue to draw lines on the inside of a specified shape. Read/write [MsoTriState](#).

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. An inset pen is not enabled.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue An inset pen is enabled.

expression.**InsetPen**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

An error occurs if this property attempts to set inset pen drawing on any Microsoft Office AutoShape which does not support inset pen drawing.

Example

The following line of code enables an inset pen for a shape. This example assumes that the first slide of the active presentation contains a shape and the shape supports inset pen drawing.

```
Sub DrawLinesInsideShape  
    ActivePresentation.Slides(1).Shapes(1).Line.InsetPen = msoTrue  
End Sub
```



InteractiveSequences Property

Returns a [Sequences](#) object that represents animations that are triggered by clicking on a specified shape.

expression.**InteractiveSequences**

expression Required. An expression that returns a [TimeLine](#) object.

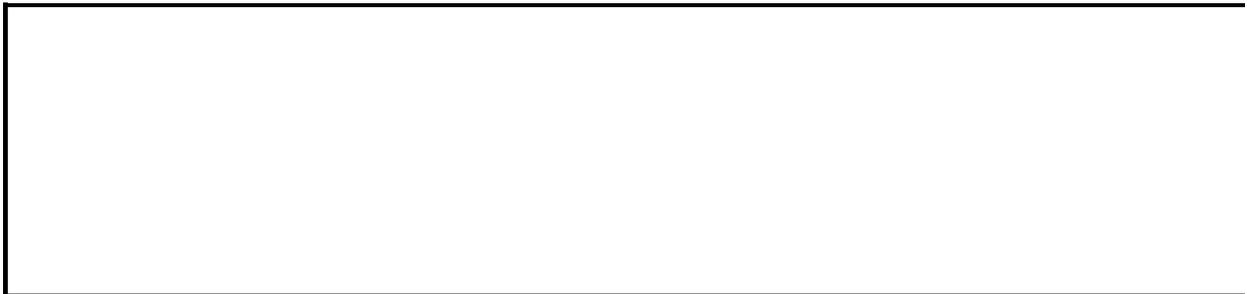
Remarks

The default value of the **InteractiveSequences** property is an empty [Sequences](#) collection.

Example

The following example adds an interactive sequence to the first slide and sets the text effect properties for the new animation sequence.

```
Sub NewInteractiveSequence()  
    Dim seqInteractive As Sequence  
    Dim shpText As Shape  
    Dim effText As Effect  
  
    Set seqInteractive = ActivePresentation.Slides(1).TimeLine _  
        .InteractiveSequences.Add(1)  
    Set shpText = ActivePresentation.Slides(1).Shapes(1)  
    Set effText = ActivePresentation.Slides(1).TimeLine _  
        .MainSequence.AddEffect(Shape:=shpText, _  
            EffectId:=msoAnimEffectChangeFont, _  
            Trigger:=msoAnimTriggerOnPageClick)  
  
    effText.EffectParameters.FontName = "Broadway"  
  
    seqInteractive.ConvertToTextUnitEffect Effect:=effText, _  
        UnitEffect:=msoAnimTextUnitEffectByWord  
  
End Sub
```



[Show All](#)

IsFullScreen Property

Returns whether the specified slide show window occupies the full screen. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide show window occupies the full screen.

Example

This example reduces the height of a full-screen slide show window just enough so that you can see the taskbar.

```
With Application.SlideShowWindows(1)
    If .IsFullScreen Then
        .Height = .Height - 20
    End If
End With
```



[Show All](#)

IsNamedShow Property

Determines whether a custom (named) slide show is displayed in the specified slide show view. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue A custom (named) slide show is displayed in the specified slide show view.

Example

If the slide show running in slide show window one is a custom slide show, this example displays its name.

```
With SlideShowWindows(1).View
  If .IsNamedShow Then
    MsgBox "Now showing in slide show window 1: " _
      & .SlideShowName
  End If
End With
```



[Show All](#)

Italic Property

Determines whether the character format is italic. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The character format is not italic.

msoTriStateMixed The specified text range contains both italic and nonitalic characters.

msoTriStateToggle

msoTrue The character format is italic.

Example

This example sets the title text on slide one and makes the title blue and italic.

```
With Application.ActivePresentation.Slides(1) _  
    .Shapes.Title.TextFrame.TextRange  
    .Text = "Volcano Coffee"  
    With .Font  
        .Italic = msoTrue  
        .Name = "palatino"  
        .Color.RGB = RGB(0, 0, 255)  
    End With  
End With
```



Item Property

Returns or sets the adjustment value specified by the ***Index*** argument. For linear adjustments, an adjustment value of 0.0 generally corresponds to the left or top edge of the shape, and a value of 1.0 generally corresponds to the right or bottom edge of the shape. However, adjustments can pass beyond shape boundaries for some shapes. For radial adjustments, an adjustment value of 1.0 corresponds to the width of the shape. For angular adjustments, the adjustment value is specified in degrees. The **Item** property applies only to shapes that have adjustments. Read/write **Single**.

expression.**Item**(***Index***)

expression Required. An expression that returns an **Adjustments** object.

Index Required **Long**. The index number of the adjustment.

Remarks

AutoShapes, connectors, and WordArt objects can have up to eight adjustments.

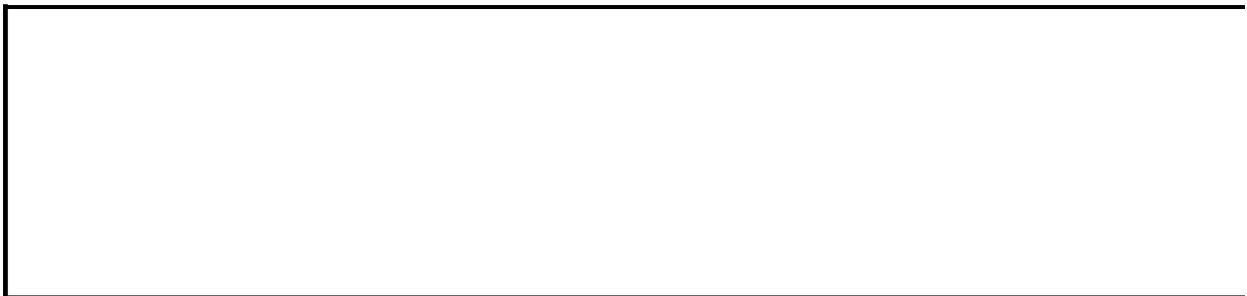
Example

This example adds two crosses to myDocument and then sets the value for adjustment one (the only one on this type of AutoShape) on each cross.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeCross, 10, 10, 100, 100) _
        .Adjustments.Item(1) = 0.4
    .AddShape(msoShapeCross, 150, 10, 100, 100) _
        .Adjustments.Item(1) = 0.2
End With
```

This example has the same result as the previous example even though it doesn't explicitly use the **Item** property.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddShape(msoShapeCross, 10, 10, 100, 100) _
        .Adjustments(1) = 0.4
    .AddShape(msoShapeCross, 150, 10, 100, 100) _
        .Adjustments(1) = 0.2
End With
```



[Show All](#)

KernedPairs Property

Determines whether the character pairs in the specified WordArt are kerned.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Character pairs in the specified WordArt are kerned.

Example

This example turns on character pair kerning for shape three on myDocument if the shape is WordArt.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.KernedPairs = msoTrue
    End If
End With
```



[Show All](#)

LanguageID Property

Returns or sets the language for the specified text range. Used for tagging portions of text written in a different language than the [DefaultLanguageID](#) property specifies. This allows Microsoft PowerPoint to check spelling and grammar according to the language for each text range. This property is not related to the application interface language. Read/write [MsoLanguageID](#).

MsoLanguageID can be one of these MsoLanguageID constants.

msoLanguageIDAfrikaans

msoLanguageIDAlbanian

msoLanguageIDAmharic

msoLanguageIDArabic

msoLanguageIDArabicAlgeria

msoLanguageIDArabicBahrain

msoLanguageIDArabicEgypt

msoLanguageIDArabicIraq

msoLanguageIDArabicJordan

msoLanguageIDArabicKuwait

msoLanguageIDArabicLebanon

msoLanguageIDArabicLibya

msoLanguageIDArabicMorocco

msoLanguageIDArabicOman

msoLanguageIDArabicQatar

msoLanguageIDArabicSyria

msoLanguageIDArabicTunisia

msoLanguageIDArabicUAE

msoLanguageIDArabicYemen

msoLanguageIDArmenian

msoLanguageIDAssamese

msoLanguageIDAzeriCyrillic

msoLanguageIDAzeriLatin

msoLanguageIDBasque
msoLanguageIDBelgianDutch
msoLanguageIDBelgianFrench
msoLanguageIDBengali
msoLanguageIDBrazilianPortuguese
msoLanguageIDBulgarian
msoLanguageIDBurmese
msoLanguageIDByelorussian
msoLanguageIDCatalan
msoLanguageIDCherokee
msoLanguageIDChineseHongKong
msoLanguageIDChineseMacao
msoLanguageIDChineseSingapore
msoLanguageIDCroatian
msoLanguageIDCzech
msoLanguageIDDanish
msoLanguageIDDutch
msoLanguageIDEnglishAUS
msoLanguageIDEnglishBelize
msoLanguageIDEnglishCanadian
msoLanguageIDEnglishCaribbean
msoLanguageIDEnglishIreland
msoLanguageIDEnglishJamaica
msoLanguageIDEnglishNewZealand
msoLanguageIDEnglishPhilippines
msoLanguageIDEnglishSouthAfrica
msoLanguageIDEnglishTrinidad
msoLanguageIDEnglishUK
msoLanguageIDEnglishUS
msoLanguageIDEnglishZimbabwe
msoLanguageIDEstonian
msoLanguageIDFaeroese
msoLanguageIDFarsi

msoLanguageIDFinnish
msoLanguageIDFrench
msoLanguageIDFrenchCameroon
msoLanguageIDFrenchCanadian
msoLanguageIDFrenchCotedIvoire
msoLanguageIDFrenchLuxembourg
msoLanguageIDFrenchMali
msoLanguageIDFrenchMonaco
msoLanguageIDFrenchReunion
msoLanguageIDFrenchSenegal
msoLanguageIDFrenchWestIndies
msoLanguageIDFrenchZaire
msoLanguageIDFrisianNetherlands
msoLanguageIDGaelicIreland
msoLanguageIDGaelicScotland
msoLanguageIDGalician
msoLanguageIDGeorgian
msoLanguageIDGerman
msoLanguageIDGermanAustria
msoLanguageIDGermanLiechtenstein
msoLanguageIDGermanLuxembourg
msoLanguageIDGreek
msoLanguageIDGujarati
msoLanguageIDHebrew
msoLanguageIDHindi
msoLanguageIDHungarian
msoLanguageIDIcelandic
msoLanguageIDIndonesian
msoLanguageIDInuktitut
msoLanguageIDItalian
msoLanguageIDJapanese
msoLanguageIDKannada
msoLanguageIDKashmiri

msoLanguageIDKazakh
msoLanguageIDKhmer
msoLanguageIDKirghiz
msoLanguageIDKonkani
msoLanguageIDKorean
msoLanguageIDLao
msoLanguageIDLatvian
msoLanguageIDLithuanian
msoLanguageIDMacedonian
msoLanguageIDMalayalam
msoLanguageIDMalayBruneiDarussalam
msoLanguageIDMalaysian
msoLanguageIDMaltese
msoLanguageIDManipuri
msoLanguageIDMarathi
msoLanguageIDMexicanSpanish
msoLanguageIDMixed
msoLanguageIDMongolian
msoLanguageIDNepali
msoLanguageIDNone
msoLanguageIDNoProofing
msoLanguageIDNorwegianBokmol
msoLanguageIDNorwegianNynorsk
msoLanguageIDOriya
msoLanguageIDPolish
msoLanguageIDPunjabi
msoLanguageIDRhaetoRomanic
msoLanguageIDRomanian
msoLanguageIDRomanianMoldova
msoLanguageIDRussian
msoLanguageIDRussianMoldova
msoLanguageIDSamiLappish
msoLanguageIDSanskrit

msoLanguageIDSerbianCyrillic
msoLanguageIDSerbianLatin
msoLanguageIDSesotho
msoLanguageIDSimplifiedChinese
msoLanguageIDSindhi
msoLanguageIDSlovak
msoLanguageIDSlovenian
msoLanguageIDSorbian
msoLanguageIDSpanish
msoLanguageIDSpanishArgentina
msoLanguageIDSpanishBolivia
msoLanguageIDSpanishChile
msoLanguageIDSpanishColombia
msoLanguageIDSpanishCostaRica
msoLanguageIDSpanishDominicanRepublic
msoLanguageIDSpanishEcuador
msoLanguageIDSpanishElSalvador
msoLanguageIDSpanishGuatemala
msoLanguageIDSpanishHonduras
msoLanguageIDSpanishModernSort
msoLanguageIDSpanishNicaragua
msoLanguageIDSpanishPanama
msoLanguageIDSpanishParaguay
msoLanguageIDSpanishPeru
msoLanguageIDSpanishPuertoRico
msoLanguageIDSpanishUruguay
msoLanguageIDSpanishVenezuela
msoLanguageIDSutu
msoLanguageIDSwahili
msoLanguageIDSwedish
msoLanguageIDSwedishFinland
msoLanguageIDSwissFrench
msoLanguageIDSwissGerman

msoLanguageIDSwissItalian
msoLanguageIDTajik
msoLanguageIDTamil
msoLanguageIDTatar
msoLanguageIDTelugu
msoLanguageIDThai
msoLanguageIDTibetan
msoLanguageIDTraditionalChinese
msoLanguageIDTsonga
msoLanguageIDTswana
msoLanguageIDTurkish
msoLanguageIDTurkmen
msoLanguageIDUkrainian
msoLanguageIDUrdu
msoLanguageIDUzbekCyrillic
msoLanguageIDUzbekLatin
msoLanguageIDVenda
msoLanguageIDVietnamese
msoLanguageIDWelsh
msoLanguageIDXhosa
msoLanguageIDZulu
msoLanguageIDPortuguese

expression.**LanguageID**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the language for the specified text in shape one to US English.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame.TextRange _  
    .LanguageID = msoLanguageIDEnglishUS
```



LanguageSettings Property

Returns a **LanguageSettings** object which contains information about the language settings in Microsoft PowerPoint. Read-only.



LastChild Property

Returns a **DiagramNode** object representing the last diagram node in a collection of diagram nodes.

expression.**LastChild**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates a new diagram and identifies the last child diagram node.

```
Sub LastChildNodeHello()  
  
    Dim shpDiagram As Shape  
    Dim dgnNode As DiagramNode  
    Dim dgnLast As DiagramNode  
    Dim intNodes As Integer  
  
    'Add org chart to first slide and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramOrgChart, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Add three additional nodes  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Enter text into last child node  
    Set dgnLast = dgnNode.Children.LastChild  
    dgnLast.Shape.TextFrame.TextRange.Text = "Here I am!"  
  
End Sub
```



LastSlideViewed Property

Returns a [Slide](#) object that represents the slide viewed immediately before the current slide in the specified slide show view.

Example

This example takes you to the slide viewed immediately before the current slide in slide show window one.

```
With SlideShowWindows(1).View  
    .GotoSlide (.LastSlideViewed.SlideIndex)  
End With
```



[Show All](#)

Layout Property

 [Layout property as it applies to the **DiagramNode** object.](#)

Returns or sets an [MsoOrgChartLayoutType](#) constant that represents the layout type for the diagram nodes in an organization chart. Read/write.

MsoOrgChartLayoutType can be one of these MsoOrgChartLayoutType constants.

msoOrgChartLayoutAssistant

msoOrgChartLayoutBothHanging

msoOrgChartLayoutLeftHanging

msoOrgChartLayoutMixed

msoOrgChartLayoutRightHanging

msoOrgChartLayoutStandard

expression.Layout

expression Required. An expression that returns a [DiagramNode](#) object.

Remarks

This property generates an error unless the diagram's [Type](#) property is `msoDiagramTypeOrgChart`.



[Layout property as it applies to the **Slide** and **SlideRange** objects.](#)

Returns or sets a [PpSlideLayout](#) constant that represents the slide layout.
Read/write.

PpSlideLayout can be one of these PpSlideLayout constants.

ppLayoutBlank

ppLayoutChart

ppLayoutChartAndText

ppLayoutClipartAndText

ppLayoutClipArtAndVerticalText

ppLayoutFourObjects

ppLayoutLargeObject

ppLayoutMediaClipAndText

ppLayoutMixed

ppLayoutObject

ppLayoutObjectAndText

ppLayoutObjectOverText

ppLayoutOrgchart

ppLayoutTable

ppLayoutText

ppLayoutTextAndChart

ppLayoutTextAndClipart

ppLayoutTextAndMediaClip

ppLayoutTextAndObject

ppLayoutTextAndTwoObjects

ppLayoutTextOverObject

ppLayoutTitle

ppLayoutTitleOnly
ppLayoutTwoColumnText
ppLayoutTwoObjectsAndText
ppLayoutTwoObjectsOverText
ppLayoutVerticalText
ppLayoutVerticalTitleAndText
ppLayoutVerticalTitleAndTextOverChart

expression.Layout

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **DiagramNode** object.](#)

The following example changes the layout of a newly-created diagram.

```
Sub ChangeDiagramLayout()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramOrgChart, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    dgnNode.Layout = msoOrgChartLayoutRightHanging  
  
End Sub
```

 [As it applies to the **Slide** object.](#)

This example changes the layout of slide one in the active presentation to include a title and subtitle if it initially has only a title.

```
With ActivePresentation.Slides(1)  
    If .Layout = ppLayoutTitleOnly Then  
        .Layout = ppLayoutTitle  
    End If  
End With
```



[Show All](#)

LayoutDirection Property

Returns or sets the layout direction for the user interface. Can be one of the following. The default value depends on the language support you have selected or installed. Read/write [PpDirection](#).

PpDirection can be one of these PpDirection constants.

ppDirectionLeftToRight

ppDirectionMixed

ppDirectionRightToLeft

expression.**LayoutDirection**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the layout direction to right-to-left.

```
Application.ActivePresentation.LayoutDirection = _  
    ppDirectionRightToLeft
```



[Show All](#)

Left Property

 [Left property as it applies to the **Application**, **DocumentWindow**, **Shape**, **ShapeRange**, and **SlideShowWindow** objects.](#)

Application, **DocumentWindow** and **SlideShowWindow** objects: Returns or sets a **Single** that represents the distance in points from the left edge of the document, application, and slide show windows to the left edge of the application window's client area. Setting this property to a very large positive or negative value may position the window completely off the desktop. Read/write.

Shape object: Returns or sets a **Single** that represents the distance in points from the left edge of the shape's bounding box to the left edge of the slide. Read/write.

ShapeRange object: Returns or sets a **Single** that represents the distance in points from the left edge of the leftmost shape in the shape range to the left edge of the slide. Read/write.

expression.**Left**

expression Required. An expression that returns one of the above objects.

 [Left property as it applies to the **Comment** object.](#)

Returns a **Single** that represents the distance in points from the left edge of the comment to the left edge of the slide. Read-only.

expression.**Left**

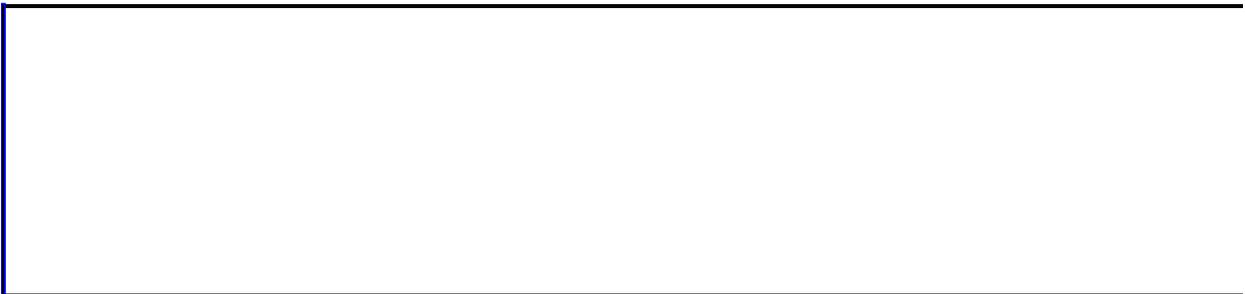
expression Required. An expression that returns a **Comment** object.

Example

 [As it applies to the **DocumentWindows** object.](#)

This example arranges windows one and two horizontally; in other words, each window occupies half the available vertical space and all the available horizontal space in the application window's client area. For this example to work, there must be only two document windows open.

```
Windows.Arrange ppArrangeTiled
sngHeight = Windows(1).Height           ' available height
sngWidth = Windows(1).Width + Windows(2).Width ' available width
With Windows(1)
    .Width = sngWidth
    .Height = sngHeight / 2
    .Left = 0
End With
With Windows(2)
    .Width = sngWidth
    .Height = sngHeight / 2
    .Top = sngHeight / 2
    .Left = 0
End With
```



LeftMargin Property

Returns or sets the left indent for the specified outline level, in points.

Read/write **Single**.

Remarks

If a paragraph begins with a bullet, the bullet position is determined by the **FirstMargin** property, and the position of the first text character in the paragraph is determined by the **LeftMargin** property.

Note The **RulerLevels** collection contains five **RulerLevel** objects, each of which corresponds to one of the possible outline levels. The **LeftMargin** property value for the **RulerLevel** object that corresponds to the first outline level has a valid range of (-9.0 to 4095.875). The valid range for the **LeftMargin** property values for the **RulerLevel** objects that correspond to the second through the fifth outline levels are determined as follows:

- The maximum value is always 4095.875.
- The minimum value is the maximum assigned value between the **FirstMargin** property and **LeftMargin** property of the previous level plus 9.

You can use the following equations to determine the minimum value for the **LeftMargin** property. *Index*, the index number of the **RulerLevel** object, indicates the object's corresponding outline level. To determine the minimum **LeftMargin** property values for the **RulerLevel** objects that correspond to the second through the fifth outline levels, substitute 2, 3, 4, or 5 for the *index* placeholder.

$$\text{Minimum}(\text{RulerLevel}(\textit{index}).\text{FirstMargin}) = \text{Maximum}(\text{RulerLevel}(\textit{index} - 1).\text{FirstMargin}, \text{RulerLevel}(\textit{index} - 1).\text{LeftMargin}) + 9$$
$$\text{Minimum}(\text{RulerLevel}(\textit{index}).\text{LeftMargin}) = \text{Maximum}(\text{RulerLevel}(\textit{index} - 1).\text{FirstMargin}, \text{RulerLevel}(\textit{index} - 1).\text{LeftMargin}) + 9$$

Example

This example sets the first-line indent and hanging indent for outline level one in body text on the slide master for the active presentation.

```
With Application.ActivePresentation _  
    .SlideMaster.TextStyles(ppBodyStyle)  
    With .Ruler.Levels(1)  
        .FirstMargin = 9  
        .LeftMargin = 54  
    End With  
End With
```



[Show All](#)

Length Property

 [Length property as it applies to the **CalloutFormat** object.](#)

When the [AutoLength](#) property of the specified callout is set to **False**, the **Length** property returns the length (in points) of the first segment of the callout line (the segment attached to the text callout box). Applies only to callouts whose lines consist of more than one segment (types **msoCalloutThree** and **msoCalloutFour**). Read-only **Float**. Use the [CustomLength](#) method to set the value of this property for the **CalloutFormat** object.

expression.**Length**

expression Required. An expression that returns one of the above objects.

 [Length property as it applies to the **TextRange** object.](#)

Returns the length of the specified text range, in characters. Read-only **Long**.

expression.**Length**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **CalloutFormat** object.](#)

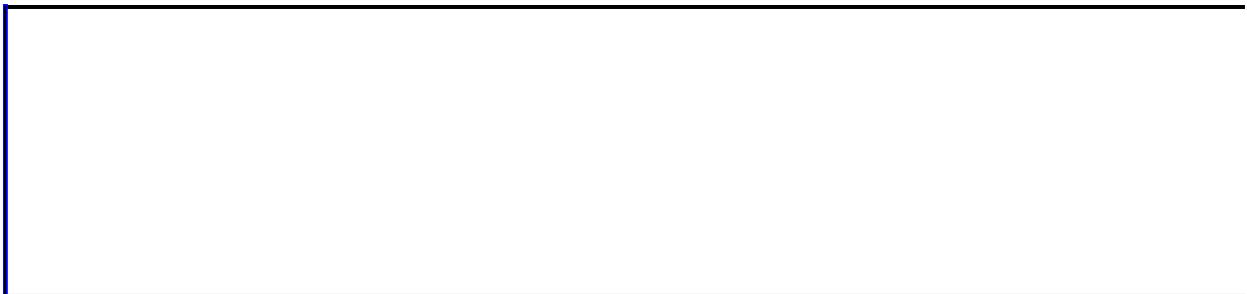
If the first line segment in the callout named "co1" has a fixed length, this example specifies that the length of the first line segment in the callout named "co2" will also be fixed at that length. For the example to work, both callouts must have multiple-segment lines.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
  With .Item("co1").Callout
    If Not .AutoLength Then len1 = .Length
  End With
  If len1 Then .Item("co2").Callout.CustomLength len1
End With
```

 [As it applies to the **TextRange** object.](#)

This example sets the title font size to 48 points if the title of slide two contains more than five characters, or it sets the font size to 72 points if the title has five or fewer characters.

```
Set myDocument = ActivePresentation.Slides(2)
With myDocument.Shapes(1).TextFrame.TextRange
  If .Length > 5 Then
    .Font.Size = 48
  Else
    .Font.Size = 72
  End If
End With
```



[Show All](#)

Levels Property

 [Levels property as it applies to the **Ruler** object.](#)

Returns a [RulerLevels](#) object that represents outline indent formatting. Read-only.

expression.**Levels**

expression Required. An expression that returns one of the above objects.

 [Levels property as it applies to the **TextStyle** object.](#)

Returns a [TextStyleLevels](#) object that represents outline text formatting. Read-only.

expression.**Levels**

expression Required. An expression that returns one of the above objects.

Example

This example sets the first-line indent and hanging indent for outline level one in body text on the slide master for the active presentation, and then it sets the font name and font size for text at that level.

```
With Application.ActivePresentation _  
    .SlideMaster.TextStyles(ppBodyStyle)  
    With .Ruler.Levels(1) ' sets indents for level 1  
        .FirstMargin = 9  
        .LeftMargin = 54  
    End With  
    With .Levels(1).Font ' sets text formatting for level 1  
        .Name = "arial"  
        .Size = 36  
    End With  
End With
```



Line Property

Returns a [LineFormat](#) object that contains line formatting properties for the specified shape. (For a line, the **LineFormat** object represents the line itself; for a shape with a border, the **LineFormat** object represents the border.) Read-only.

Example

This example adds a blue dashed line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(50, 0, 128)
End With
```

This example adds a cross to the first slide and then sets its border to be 8 points thick and red.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeCross, 10, 10, 50, 70).Line
    .Weight = 8
    .ForeColor.RGB = RGB(255, 0, 0)
End With
```



[Show All](#)

LineRuleAfter Property

Determines whether line spacing after the last line in each paragraph is set to a specific number of points or lines. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Line spacing after the last line in each paragraph is set to a specific number of points.

msoTriStateMixed

msoTriStateToggle

msoTrue Line spacing after the last line in each paragraph is set to a specific number of lines.

Example

This example displays a message box that shows the setting for space after paragraphs for the text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat
    sa = .SpaceAfter
    If .LineRuleAfter Then
      saUnits = " lines"
    Else
      saUnits = " points"
    End If
  End With
End With
MsgBox "Current spacing after paragraphs: " & sa & saUnits
```



[Show All](#)

LineRuleBefore Property

Determines whether line spacing before the first line in each paragraph is set to a specific number of points or lines. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Line spacing before the first line in each paragraph is set to a specific number of points.

msoTriStateMixed

msoTriStateToggle

msoTrue Line spacing before the first line in each paragraph is set to a specific number of lines.

Example

This example displays a message box that shows the setting for space before paragraphs for text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat
    sb = .SpaceBefore
    If .LineRuleBefore Then
      sbUnits = " lines"
    Else
      sbUnits = " points"
    End If
  End With
End With
MsgBox "Current spacing before paragraphs: " & sb & sbUnits
```



[Show All](#)

LineRuleWithin Property

Determines whether line spacing between base lines is set to a specific number of points or lines. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Line spacing between base lines is set to a specific number of points.

msoTriStateMixed

msoTriStateToggle

msoTrue Line spacing between base lines is set to a specific number of lines.

Example

This example displays a message box that shows the setting for line spacing for text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat
    ls = .SpaceWithin
    If .LineRuleWithin Then
      lsUnits = " lines"
    Else
      lsUnits = " points"
    End If
  End With
End With
MsgBox "Current line spacing: " & ls & lsUnits
```



LinkFormat Property

Returns a [LinkFormat](#) object that contains the properties that are unique to linked OLE objects. Read-only.

Example

This example updates the links between any OLE objects on slide one in the active presentation and their source files.

```
For Each sh In ActivePresentation.Slides(1).Shapes
  If sh.Type = msoLinkedOLEObject Then
    With sh.LinkFormat
      .Update
    End With
  End If
Next
```



[Show All](#)

Loaded Property

Determines whether the specified add-in is loaded. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified add-in is loaded. In the **Add-Ins** dialog box (**Tools** menu), the check boxes next to loaded add-ins are selected.

Example

This example adds MyTools.ppa to the list in the **Add-Ins** dialog box (**Tools** menu) and then loads it.

```
Addins.Add("c:\my documents\mytools.ppa").Loaded = msoTrue
```

This example unloads the add-in named "MyTools."

```
Application.Addins("mytools").Loaded = msoFalse
```



[Show All](#)

LockAspectRatio Property

Determines whether the specified shape retains its original proportions when you resize it. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse You can change the height and width of the shape independently of one another when you resize it.

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape retains its original proportions when you resize it.

Example

This example adds a cube to myDocument. The cube can be moved and resized, but not reproportioned.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddShape(msoShapeCube, 50, 50, 100, 200) _
    .LockAspectRatio = msoTrue
```



[Show All](#)

LoopSoundUntilNext Property

Read/write [MsoTriState](#). Specifies whether the sound that's been set for the specified slide transition loops until the next sound starts.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The sound that's been set for the specified slide transition loops until the next sound starts.

Example

This example specifies that the file `Dudududu.wav` will start to play at the transition to slide two in the active presentation and will continue to play until the next sound starts.

```
With ActivePresentation.Slides(2).SlideShowTransition
    .SoundEffect.ImportFromFile "c:\sndsys\dudududu.wav"
    .LoopSoundUntilNext = msoTrue
End With
```



[Show All](#)

LoopUntilStopped Property

 [As it applies to the **PlaySettings** object.](#)

Determines whether the specified movie or sound loops continuously until either the next movie or sound starts, the user clicks the slide, or a slide transition occurs. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified movie or sound loops continuously until either the next movie or sound starts, the user clicks the slide, or a slide transition occurs.

expression.**LoopUntilStopped**

expression Required. An expression that returns one of the objects in the Applies To list.

 [As it applies to the **SlideShowSettings** object.](#)

Determines whether specified slide show loops continuously until the user presses ESC. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide show loops continuously until the user presses ESC.

expression.**LoopUntilStopped**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

[As it applies to the **PlaySettings** object.](#)

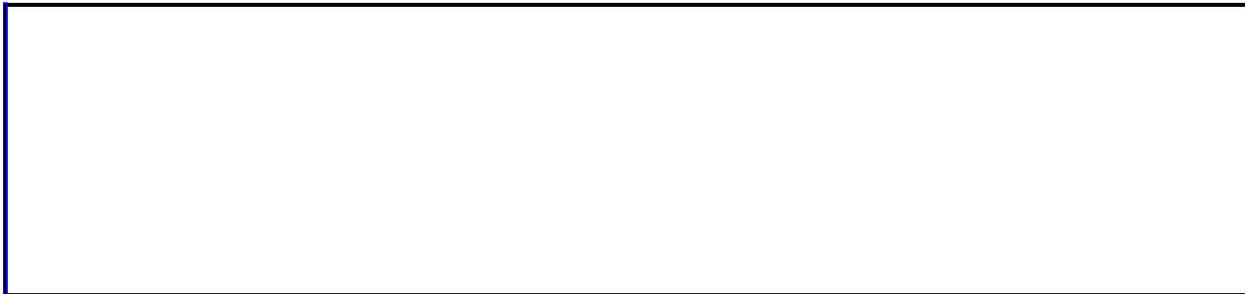
This example specifies that shape three on slide one in the active presentation will start to play in the animation order and will continue to play until the next media clip starts. Shape three must be a sound or movie object.

```
ActivePresentation.Slides(1).Shapes(3) _  
    .AnimationSettings.PlaySettings.LoopUntilStopped = msoTrue
```

[As it applies to the **SlideShowSettings** object.](#)

This example starts a slide show of the active presentation that will automatically advance the slides (using the stored timings) and will loop continuously through the presentation until the user presses ESC.

```
With ActivePresentation.SlideShowSettings  
    .AdvanceMode = ppSlideShowUseSlideTimings  
    .LoopUntilStopped = msoTrue  
    .Run  
End With
```



MainSequence Property

Returns a [Sequence](#) object that represents the collection of [Effect](#) objects in the main animation sequence of a slide.

expression.**MainSequence**

expression Required. An expression that returns one of the objects in the Applies To list.

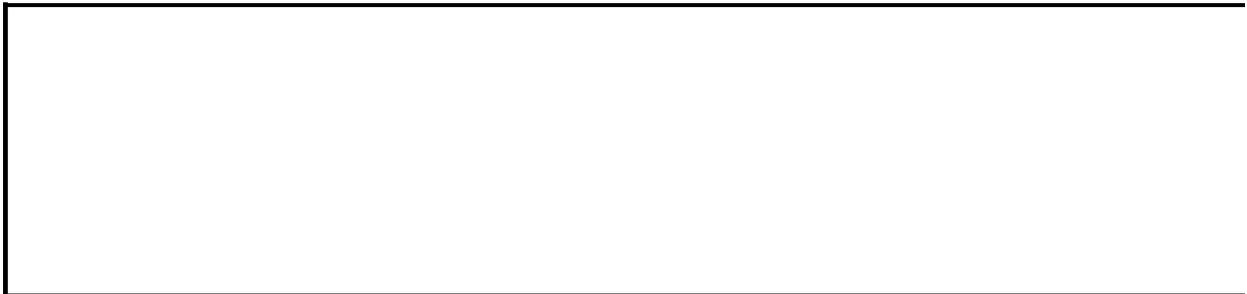
Remarks

The default value of the **MainSequence** property is an empty **Sequence** collection. Any attempt to return a value from this property without adding one or more **Effect** objects to the main animation sequence will result in a run-time error.

Example

The following example adds a boomerang animation to a new shape on a new slide added to the active presentation.

```
Sub NewSequence()  
    Dim sldNew As Slide  
    Dim shpnew As Shape  
  
    Set sldNew = ActivePresentation.Slides.Add(Index:=1, Layout:=ppL  
    Set shpnew = sldNew.Shapes.AddShape(Type:=msoShape5pointStar, _  
        Left:=25, Top:=25, Width:=100, Height:=100)  
  
    With sldNew.TimeLine.MainSequence.AddEffect(Shape:=shpnew, _  
        EffectId:=msoAnimEffectBoomerang)  
        .Timing.Speed = 0.5  
        .Timing.Accelerate = 0.2  
    End With  
End Sub
```



MarginBottom Property

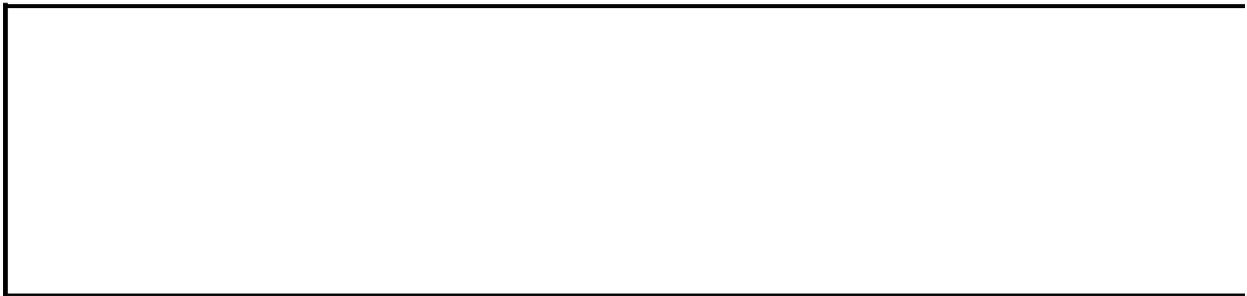
Returns or sets the distance (in points) between the bottom of the text frame and the bottom of the inscribed rectangle of the shape that contains the text.

Read/write **Single**.

Example

This example adds a rectangle to myDocument, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    0, 0, 250, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 10
    .MarginRight = 0
    .MarginTop = 20
End With
```



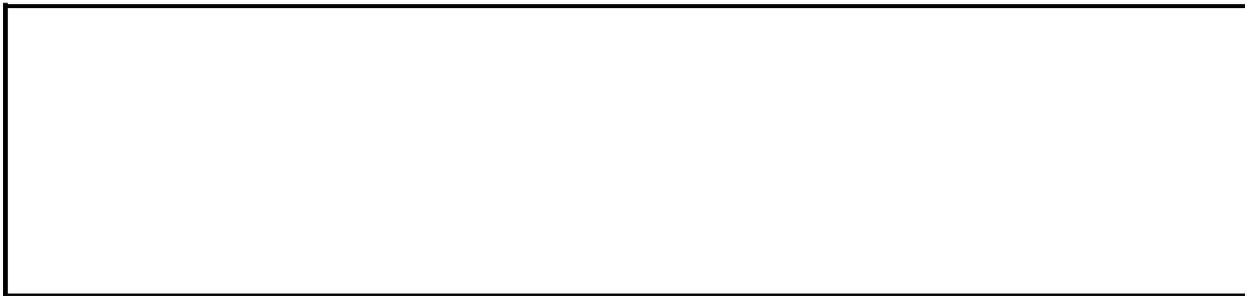
MarginLeft Property

Returns or sets the distance (in points) between the left edge of the text frame and the left edge of the inscribed rectangle of the shape that contains the text. Read/write **Single**.

Example

This example adds a rectangle to myDocument, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    0, 0, 250, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 10
    .MarginRight = 0
    .MarginTop = 20
End With
```



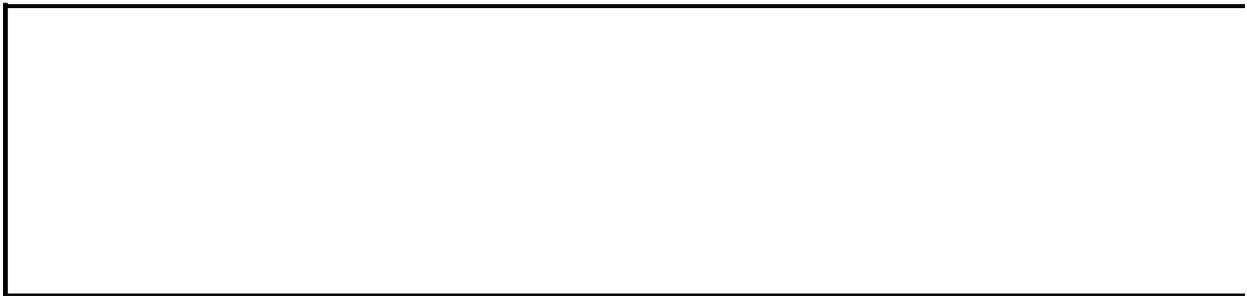
MarginRight Property

Returns or sets the distance (in points) between the right edge of the text frame and the right edge of the inscribed rectangle of the shape that contains the text. Read/write **Single**.

Example

This example adds a rectangle to myDocument, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    0, 0, 250, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 10
    .MarginRight = 5
    .MarginTop = 20
End With
```



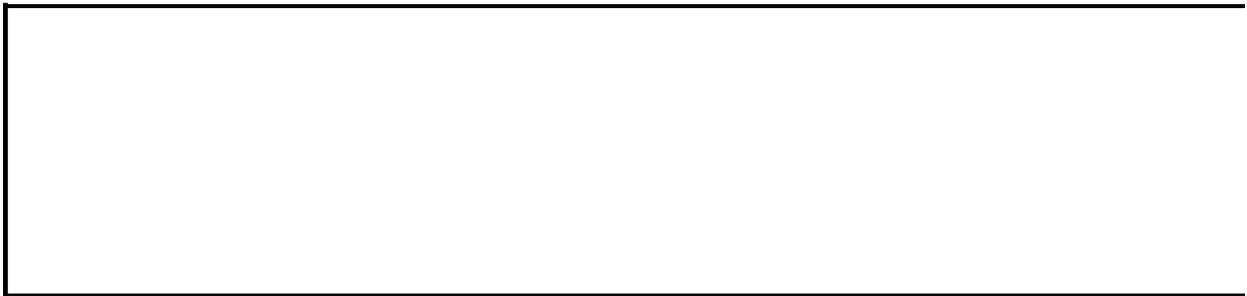
MarginTop Property

Returns or sets the distance (in points) between the top of the text frame and the top of the inscribed rectangle of the shape that contains the text. Read/write **Single**.

Example

This example adds a rectangle to myDocument, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    0, 0, 250, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 10
    .MarginRight = 0
    .MarginTop = 20
End With
```



Master Property

Returns a [Master](#) object that represents the slide master. Read-only.

Example

This example sets the background fill for the slide master for slide one in the active presentation.

```
ActivePresentation.Slides(1).Master.Background.Fill _  
    .PresetGradient msoGradientDiagonalUp, 1, msoGradientDaybreak
```



[Show All](#)

MediaType Property

Returns the OLE media type. Read-only [PpMediaType](#).

PpMediaType can be one of these PpMediaType constants.

ppMediaTypeMixed

ppMediaTypeMovie

ppMediaTypeOther

ppMediaTypeSound

expression.**MediaType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets all native sound objects on slide one in the active presentation to loop until manually stopped during a slide show.

```
For Each so In ActivePresentation.Slides(1).Shapes
    If so.Type = msoMedia Then
        If so.MediaType = ppMediaTypeSound Then
            so.AnimationSettings.PlaySettings _
                .LoopUntilStopped = True
        End If
    End If
Next
```



MotionEvent Property

Returns a [MotionEffect](#) object that represents the properties of a motion animation.

expression.**MotionEvent**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a new motion behavior to the first slide's main sequence that moves the specified animation sequence from one side of the page to the shape's original position.

```
Sub NewMotion()  
    With ActivePresentation.Slides(1).TimeLine.MainSequence(1) _  
        .Behaviors.Add(msoAnimTypeMotion).MotionEffect  
            .FromX = 100  
            .FromY = 100  
            .ToX = 0  
            .ToY = 0  
    End With  
End Sub
```



[Show All](#)

Name Property

 [Name property as it applies to the **ColorFormat**, **Design**, **Font**, **Master**, **Shape**, **ShapeRange**, **Slide**, **SlideRange**, and **SoundEffect** objects.](#)

ColorFormat, **Design**, **Font**, and **Master** objects: Returns or sets the name of the specified object. Read/write **String**.

Shape or **ShapeRange** objects. When a shape is created, Microsoft PowerPoint automatically assigns it a name in the form *ShapeType Number*, where *ShapeType* identifies the type of shape or AutoShape, and *Number* is an integer that's unique within the collection of shapes on the slide. For example, the automatically generated names of the shapes on a slide could be Placeholder 1, Oval 2, and Rectangle 3. To avoid conflict with automatically assigned names, don't use the form *ShapeType Number* for user-defined names, where *ShapeType* is a value that is used for automatically generated names, and *Number* is any positive integer. A shape range must contain exactly one shape. Read/write **String**.

Slide or **SlideRange** objects: When a slide is inserted into a presentation, PowerPoint automatically assigns it a name in the form *Sliden*, where *n* is an integer that represents the order in which the slide was created in the presentation. For example, the first slide inserted into a presentation is automatically named Slide1. If you copy a slide from one presentation to another, the slide loses the name it had in the first presentation and is automatically assigned a new name in the second presentation. A slide range must contain exactly one slide. Read/write **String**.

SoundEffect object: The set of valid names for a presentation appears in the **Sound** box in the **Slide Transition** task pane (**Slide Show** menu). Read/write **String**.

expression.**Name**

expression Required. An expression that returns one of the above objects.

 Name property as it applies to the [AddIn](#), [Application](#), [NamedSlideShow](#), and [Presentation](#) objects.

AddIn object: The name (title) of the add-in for file types that are registered. Read-only **String**.

Application object: Returns the string "Microsoft PowerPoint." Read-only **String**.

NamedSlideShow object: You cannot use this property to set the name for a custom slide show. Use the [Add](#) method to redefine a custom slide show under a new name. Read-only **String**.

Presentation object: The name of the presentation includes the file name extension (for file types that are registered) but doesn't include its path. You cannot use this property to set the name. Use the [SaveAs](#) method to save the presentation under a different name if you need to change the name. Read-only **String**.

expression.**Name**

expression Required. An expression that returns one of the above objects.

Remarks

You can use the object's name in conjunction with the **Item** method to return a reference to the object if the **Item** method for the collection that contains the object takes a **Variant** argument. For example, if the value of the **Name** property for a shape is Rectangle 2, then `.Shapes("Rectangle 2")` will return a reference to that shape.

Example

 [As it applies to the **Shape** object.](#)

This example sets the name of object two on slide one in the active presentation to "big triangle."

```
ActivePresentation.Slides(1).Shapes(2).Name = "big triangle"
```

This example sets the fill color for the shape named "big triangle" on slide one in the active presentation.

```
ActivePresentation.Slides(1) _  
    .Shapes("big triangle").Fill.ForeColor.RGB = RGB(0, 0, 255)
```



NameAscii Property

Returns or sets the font used for ASCII characters (characters with character set numbers within the range of 0 to 127). Read/write **String**.

Remarks

The default value of this property is Times New Roman. Use the [Replace](#) method to change the font that's applied to all text and that appears in the **Font** box on the **Formatting** toolbar.

Example

This example sets the font used for ASCII characters in the title of the first slide to Century.

```
Application.ActivePresentation.Slides(1).Shapes.Title _  
    .TextFrame.TextRange.Font.NameAscii = "Century"
```



NameComplexScript Property

Returns or sets the complex script font name. Used for mixed language text.
Read/write **String**.

Remarks

When you have a right-to-left language setting specified, this property is equivalent to the **Complex scripts font** list in the **Font** dialog box (**Format** menu). When you have an Asian language setting specified, this property is equivalent to the **Asian text font** list in the **Font** dialog box (**Format** menu).

Example

This example sets the complex script font to Times New Roman.

```
ActivePresentation.Slides(1).Shapes.Title.TextFrame _  
    .TextRange.Font.NameComplexScript = "Times New Roman"
```



NamedSlideShows Property

Returns a [NamedSlideShows](#) collection that represents all the named slide shows (custom slide shows) in the specified presentation. Each named slide show, or custom slide show, is a user-selected subset of the specified presentation. Read-only.

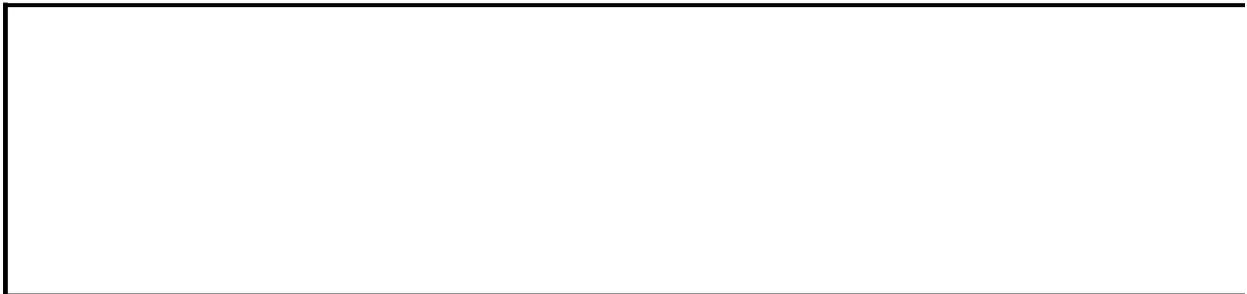
Remarks

Use the [Add](#) method of the **NamedSlideShows** object to create a named slide show.

Example

This example adds to the active presentation a named slide show "Quick Show" that contains slides 2, 7, and 9. The example then runs this slide show.

```
Dim qSlides(1 To 3) As Long
With ActivePresentation
    With .Slides
        qSlides(1) = .Item(2).SlideID
        qSlides(2) = .Item(7).SlideID
        qSlides(3) = .Item(9).SlideID
    End With
    With .SlideShowSettings
        .RangeType = ppShowNamedSlideShow
        .NamedSlideShows.Add "Quick Show", qSlides
        .SlideShowName = "Quick Show"
        .Run
    End With
End With
```



NameFarEast Property

Returns or sets the Asian font name. Read/write **String**.

Remarks

Use the [Replace](#) method to change the font that's applied to all text and that appears in the **Font** box on the **Formatting** toolbar.

Example

This example displays the name of the Asian font applied to the selection.

```
MsgBox ActiveWindow.Selection.ShapeRange _  
    .TextFrame.TextRange.Font.NameFarEast
```



NameOther Property

Returns or sets the font used for characters whose character set numbers are greater than 127. Read/write **String**.

Remarks

In the U.S. English version of Microsoft PowerPoint, this property is read-only and the default value is Times New Roman. Use the [Replace](#) method to change a font in a presentation. The **NameOther** property setting is the same as the **NameASCII** property setting except when the **NameASCII** property is set to "Use FE Font."

Example

This example sets the font used for characters whose character set numbers are greater than 127, for the first member of the **Fonts** collection.

```
ActivePresentation.Fonts(1).NameOther = "Tahoma"
```



NewPresentation Property

Returns a **NewFile** object that represents a presentation listed on the **New Presentation** task pane. Read-only.

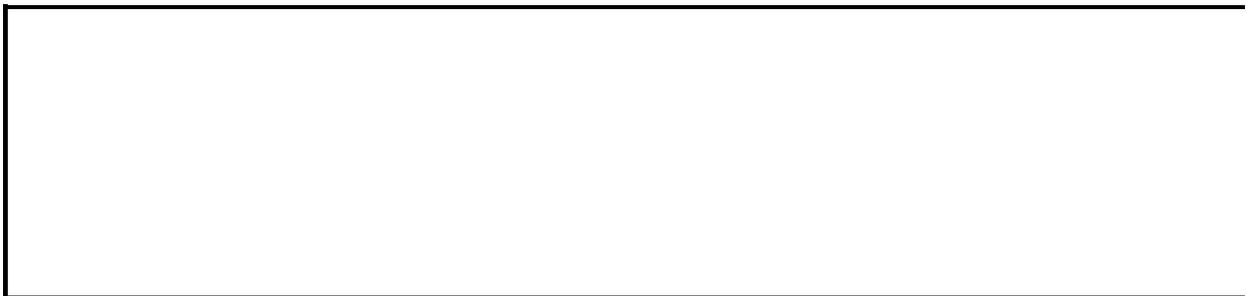
expression.**NewPresentation**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example lists a presentation on the New Presentation task pane at the bottom of the last section in the pane.

```
Sub CreateNewPresentationListItem()  
    Application.NewPresentation.Add FileName:="C:\Presentation.ppt"  
    Application.CommandBars("Task Pane").Visible = True  
End Sub
```



[Show All](#)

Nodes Property

 [Nodes property as it applies to the **Diagram** object.](#)

Returns a [DiagramNodes](#) object that contains a flat list of all of the nodes in the specified diagram.

expression.**Nodes**

expression Required. An expression that returns a [Diagram](#) object.

 [Nodes property as it applies to the **Shape** and **ShapeRange** objects.](#)

Returns a [ShapeNodes](#) collection that represents the geometric description of the specified shape. Applies to **Shape** or **ShapeRange** objects that represent freeform drawings.

expression.**Nodes**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **Diagram** object.](#)

The following example returns the number of nodes in a newly-created diagram.

```
Sub ConvertPyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Create pyramid diagram and add first node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramPyramid, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Add three child nodes to the first node  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically layout diagram and convert to radial diagram  
    With dgnNode.Diagram  
        .AutoLayout = msoTrue  
        .Convert Type:=msoDiagramRadial  
    End With  
  
    'Display the number of nodes in the diagram  
    MsgBox dgnNode.Diagram.Nodes.Count  
  
End Sub
```

 [As it applies to the **Shape** object.](#)

This example adds a smooth node with a curved segment after node four in shape three on myDocument. Shape three must be a freeform drawing with at least four nodes.

```
Set myDocument = ActivePresentation.Slides(1)  
With myDocument.Shapes(3).Nodes  
    .Insert Index:=4, SegmentType:=msoSegmentCurve, _
```

EditingType:=msoEditingSmooth, X1:=210, Y1:=100
End With



NoLineBreakAfter Property

Returns or sets the characters that cannot end a line. Read/write **String**.

Example

This example sets "\$", "(", "[", "\", and "{" as characters that cannot end a line.

With ActivePresentation

```
.FarEastLineBreakLevel = ppFarEastLineBreakLevelCustom
```

```
.NoLineBreakAfter = "$([\\"
```

End With



NoLineBreakBefore Property

Returns or sets the characters that cannot begin a line. Read/write **String**.

Example

This example sets "!", ")", and "]" as characters that cannot begin a line.

```
With ActivePresentation
    .FarEastLineBreakLevel = ppFarEastLineBreakLevelCustom
    .NoLineBreakBefore = "!)]"
End With
```



[Show All](#)

NormalizedHeight Property

Determines whether the characters (both uppercase and lowercase) in the specified WordArt are the same height. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

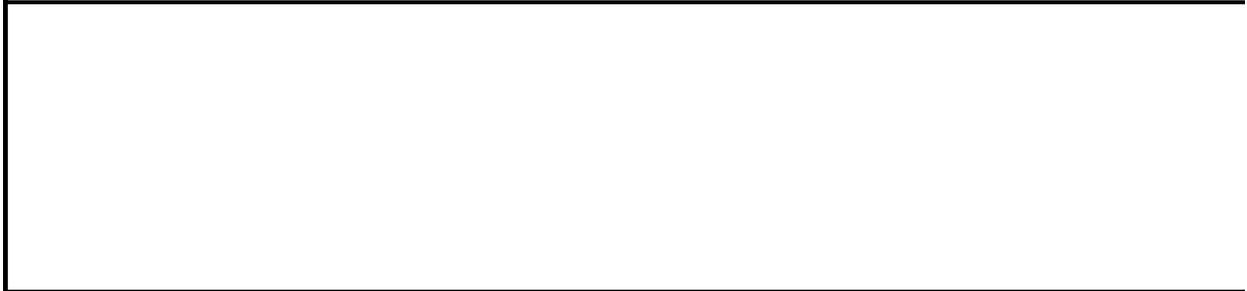
msoTriStateToggle

msoTrue All characters (both uppercase and lowercase) in the specified WordArt are the same height.

Example

This example adds WordArt that contains the text "Test Effect" to myDocument and gives the new WordArt the name "texteff1." The code then makes all characters in the shape named "texteff1" the same height.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddTextEffect(PresetTextEffect:=msoTextEffect1, _
    Text:="Test Effect", FontName:="Courier New", _
    FontSize:=44, FontBold:=True, _
    FontItalic:=False, Left:=10, Top:=10)_
    .Name = "texteff1"
myDocument.Shapes("texteff1").TextEffect.NormalizedHeight = msoTrue
```



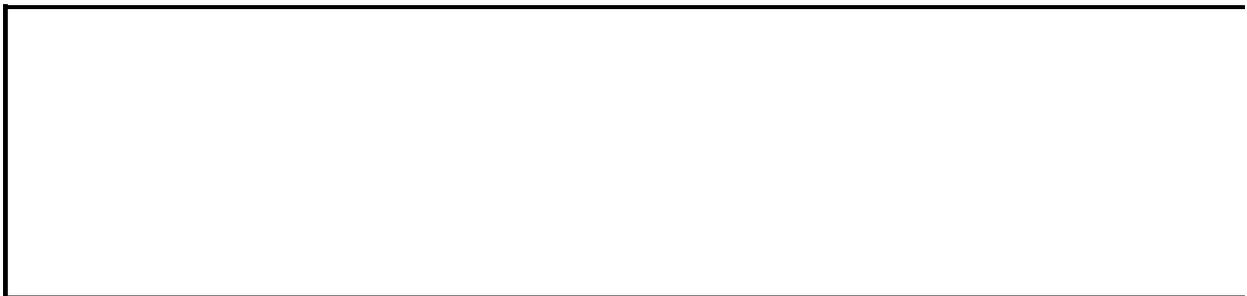
NotesMaster Property

Returns a [Master](#) object that represents the notes master. Read-only.

Example

This example sets the header and footer text for the notes master for the active presentation.

```
With Application.ActivePresentation.NotesMaster .HeadersFooters
    .Header.Text = "Employee Guidelines"
    .Footer.Text = "Volcano Coffee"
End With
```



[Show All](#)

NotesOrientation Property

Returns or sets the on-screen and printed orientation of notes pages, handouts, and outlines for the specified presentation. Read/write [MsoOrientation](#).

MsoOrientation can be one of these MsoOrientation constants.

msoOrientationHorizontal

msoOrientationMixed

msoOrientationVertical

expression.NotesOrientation

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the orientation of all notes pages, handouts, and outlines in the active presentation to horizontal (landscape).

```
Application.ActivePresentation.PageSetup.NotesOrientation = _  
    msoOrientationHorizontal
```



NotesPage Property

Returns a [SlideRange](#) object that represents the notes pages for the specified slide or range of slides. Read-only.

Note The following properties and methods will fail if applied to a **SlideRange** object that represents a notes page: **Copy** method, **Cut** method, **Delete** method, **Duplicate** method, **HeadersFooters** property, **Hyperlinks** property, **Layout** property, **PrintSteps** property, **SlideShowTransition** property.

Remarks

The **NotesPage** property returns the notes page for either a single slide or a range of slides and allows you to make changes only to those notes pages. If you want to make changes that affect all notes pages, use the [NotesMaster](#) property to return the **Slide** object that represents the notes master.

Example

This example sets the background fill for the notes page for slide one in the active presentation.

```
With ActivePresentation.Slides(1).NotesPage
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient _
        msoGradientHorizontal, 1, msoGradientLateSunset
End With
```



Number Property

Returns the bullet number of a paragraph when the [Type](#) property of the **BulletFormat** object is set to **ppBulletNumbered**. Read-only **Long**.

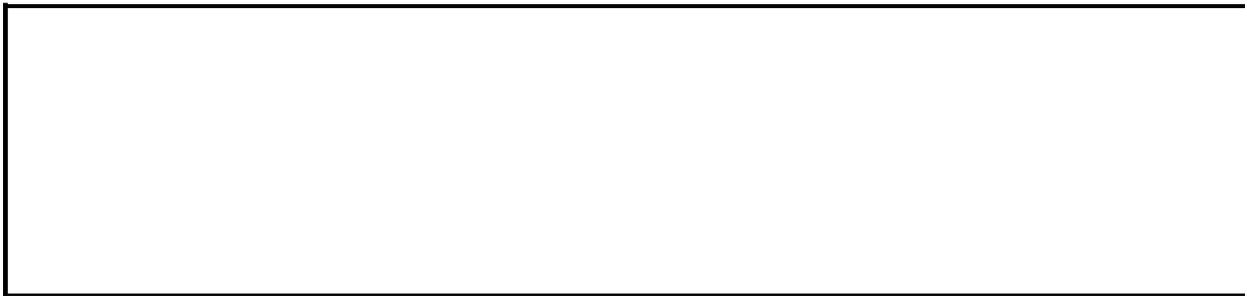
Remarks

If this property is queried for multiple paragraphs with different numbers, then the value **ppBulletMixed** is returned. If this property is queried for a paragraph with a type other than **ppBulletNumbered**, then a run-time error occurs.

Example

This example returns the bullet number of paragraph one in the selected text range to a variable named myParnum.

```
With ActiveWindow.Selection
  If .Type = ppSelectionTextRange Then
    With .TextRange.Paragraphs(1).ParagraphFormat.Bullet
      If .Type = ppBulletNumbered Then
        myParnum = .Number
      End If
    End With
  End If
End With
```



NumberOfCopies Property

Returns or sets the number of copies of a presentation to be printed. The default value is 1. Read/write **Long**.

Remarks

Specifying a value for the *Copies* argument of the [PrintOut](#) method sets the value of this property.

Example

This example prints three collated copies of the active presentation.

```
With ActivePresentation.PrintOptions
    .NumberOfCopies = 3
    .Collate = True
    .Parent.PrintOut
End With
```



Object Property

Returns the object that represents the specified OLE object's top-level interface. This property allows you to access the properties and methods of the application in which an OLE object was created. Read-only.

Remarks

Use the **TypeName** function to determine the type of object this property returns for a specific OLE object.

Example

This example displays the type of object contained in shape one on slide one in the active presentation. Shape one must contain an OLE object.

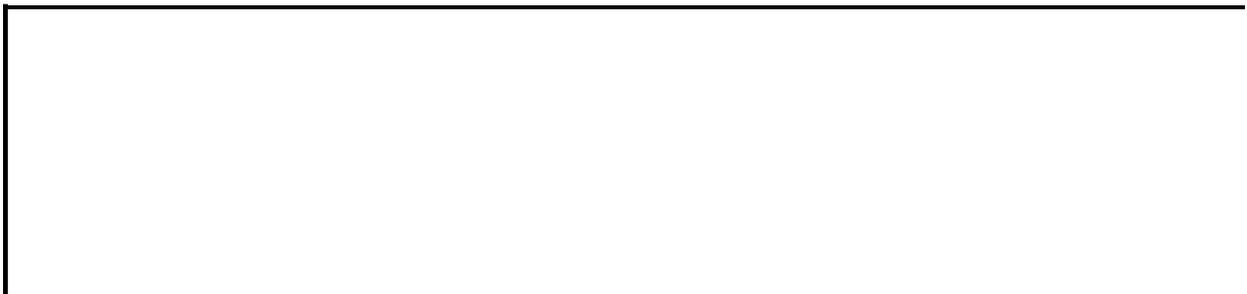
```
MsgBox TypeName(ActivePresentation.Slides(1) _  
    .Shapes(1).OLEFormat.Object)
```

This example displays the name of the application in which each embedded OLE object on slide one in the active presentation was created.

```
For Each s In ActivePresentation.Slides(1).Shapes  
    If s.Type = msoEmbeddedOLEObject Then  
        MsgBox s.OLEFormat.Object.Application.Name  
    End If  
Next
```

This example adds text to cell A1 on worksheet one in the Microsoft Excel workbook contained in shape three on slide one in the active presentation.

```
With ActivePresentation.Slides(1).Shapes(3)  
    .OLEFormat.Object.Worksheets(1).Range("A1").Value = "New text"  
End With
```



[Show All](#)

ObjectVerbs Property

Returns a [ObjectVerbs](#) collection that contains all the OLE [verbs](#) for the specified OLE object. Read-only.

Example

This example displays all the available verbs for the OLE object contained in shape one on slide two in the active presentation. For this example to work, shape one must be a shape that represents an OLE object.

```
With ActivePresentation.Slides(2).Shapes(1).OLEFormat
    For Each v In .ObjectVerbs
        MsgBox v
    Next
End With
```

This example specifies that the OLE object represented by shape one on slide two in the active presentation will open when it's clicked during a slide show if "Open" is one of the OLE verbs for that object. For this example to work, shape one must be a shape that represents an OLE object.

```
With ActivePresentation.Slides(2).Shapes(1)
    For Each sVerb In .OLEFormat.ObjectVerbs
        If sVerb = "Open" Then
            With .ActionSettings(ppMouseClick)
                .Action = ppActionOLEVerb
                .ActionVerb = sVerb
            End With
        Exit For
    End If
Next
End With
```



[Show All](#)

Obscured Property

Determines whether the shadow of the specified shape appears filled in and is obscured by the shape. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The shadow has no fill and the outline of the shadow is visible through the shape if the shape has no fill.

msoTriStateMixed

msoTriStateToggle

msoTrue The shadow of the specified shape appears filled in and is obscured by the shape, even if the shape has no fill.

Example

This example sets the horizontal and vertical offsets of the shadow for shape three on myDocument. The shadow is offset 5 points to the right of the shape and 3 points above it. If the shape doesn't already have a shadow, this example adds one to it. The shadow will be filled in and obscured by the shape, even if the shape has no fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
    .Obscured = msoTrue
End With
```



OffsetX Property

Returns or sets the horizontal offset of the shadow from the specified shape, in points. A positive value offsets the shadow to the right of the shape; a negative value offsets it to the left. Read/write **Single**.

Remarks

If you want to nudge a shadow horizontally or vertically from its current position without having to specify an absolute position, use the [IncrementOffsetX](#) method or the [IncrementOffsetY](#) method.

Example

This example sets the horizontal and vertical offsets of the shadow for shape three on myDocument. The shadow is offset 5 points to the right of the shape and 3 points above it. If the shape doesn't already have a shadow, this example adds one to it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
End With
```



OffsetY Property

Returns or sets the vertical offset of the shadow from the specified shape, in points. A positive value offsets the shadow to the right of the shape; a negative value offsets it to the left. Read/write **Single**.

Remarks

If you want to nudge a shadow horizontally or vertically from its current position without having to specify an absolute position, use the [IncrementOffsetX](#) method or the [IncrementOffsetY](#) method.

Example

This example sets the horizontal and vertical offsets of the shadow for shape three on myDocument. The shadow is offset 5 points to the right of the shape and 3 points above it. If the shape doesn't already have a shadow, this example adds one to it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
End With
```



OLEFormat Property

Returns an [OLEFormat](#) object that contains OLE formatting properties for the specified shape. Applies to **Shape** or **ShapeRange** objects that represent OLE objects. Read-only

Example

This example loops through all the objects on all the slides in the active presentation and sets all linked Microsoft Word documents to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Word.Document" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
```



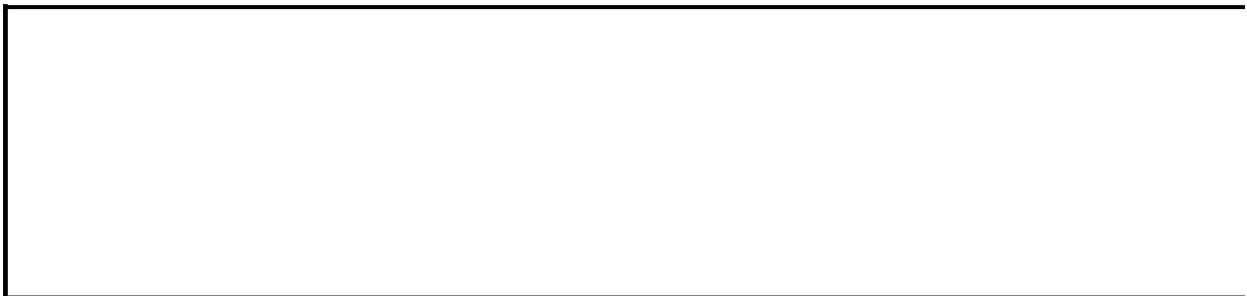
OperatingSystem Property

Returns the name of the operating system. Read-only **String**.

Example

This example tests the **OperatingSystem** property to see whether PowerPoint is running with a 32-bit version of Microsoft Windows.

```
os = Application.OperatingSystem  
If InStr(os, "Windows (32-bit)") <> 0 Then  
    MsgBox "Running a 32-bit version of Microsoft Windows"  
End If
```



Options Property

Returns an [Options](#) object that represents application options in Microsoft PowerPoint.

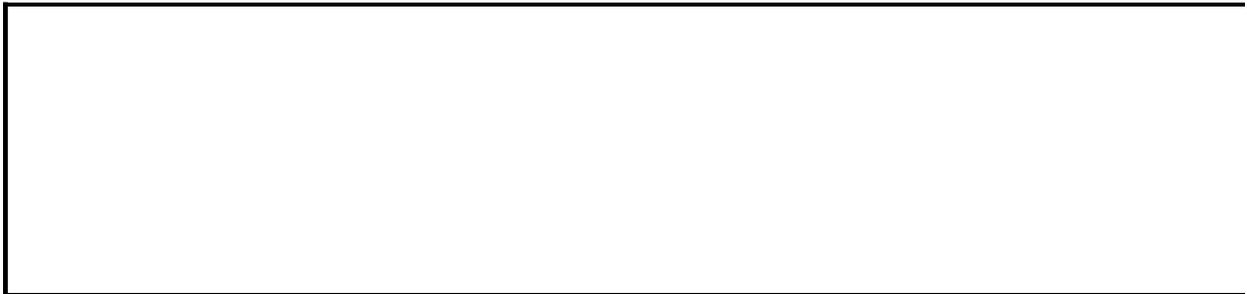
expression.**Options**

expression Required. An expression that returns an [Application](#) object.

Example

Use the **Options** property to return the **Options** object. The following example sets three application options for PowerPoint.

```
Sub TogglePasteOptionsButton()  
    With Application.Options  
        If .DisplayPasteOptions = False Then  
            .DisplayPasteOptions = True  
        End If  
    End With  
End Sub
```



[Show All](#)

OrganizeInFolder Property

Determines whether all supporting files, such as background textures and graphics, are organized in a separate folder when you save or publish the specified presentation as a Web page. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Supporting files are saved in the same folder as the Web page.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. All supporting files, such as background textures and graphics, are organized in a separate folder when you save or publish the specified presentation as a Web page.

Remarks

The new folder is created within the folder where you have saved the Web page, and is named after the presentation. If the [UseLongFileNames](#) property is set to **True**, a suffix is added to the folder name.

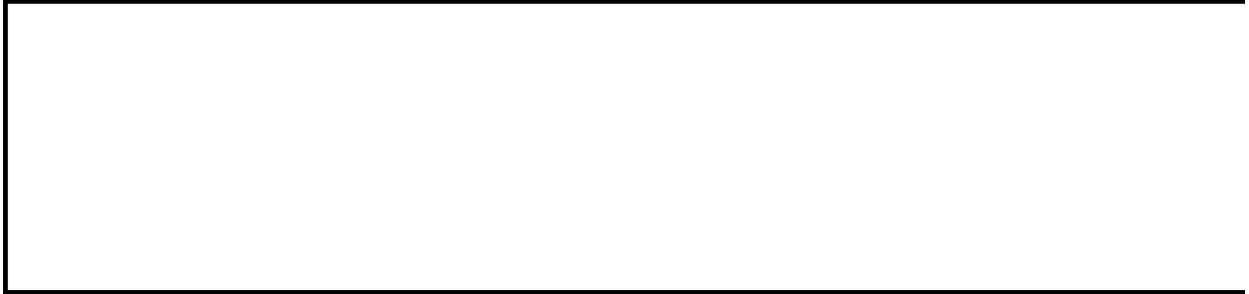
If you save a presentation that was previously saved with the **OrganizeInFolder** property set to a different value, Microsoft PowerPoint automatically moves the supporting files into or out of the folder as appropriate.

If you don't use long file names (that is, if the **UseLongFileNames** property is set to **False**), PowerPoint automatically saves any supporting files in a separate folder. The files cannot be saved in the same folder as the Web page.

Example

This example specifies that all supporting files are saved in the same folder when presentation two is saved or published as a Web page.

```
Presentations(2).WebOptions.OrganizeInFolder = msoFalse
```



[Show All](#)

Orientation Property

Returns or sets text orientation. Read/write [MsoTextOrientation](#). Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.

MsoTextOrientation can be one of these MsoTextOrientation constants.

msoTextOrientationDownward

msoTextOrientationHorizontal

msoTextOrientationHorizontalRotatedFarEast

msoTextOrientationMixed

msoTextOrientationUpward

msoTextOrientationVertical

msoTextOrientationVerticalFarEast

expression.**Orientation**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example orients the text horizontally within shape three on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(3).TextFrame _
    .Orientation = msoTextOrientationHorizontal
```



[Show All](#)

OutputType Property

Returns or sets a value that indicates which component (slides, handouts, notes pages, or an outline) of the presentation is to be printed. Read/write [PpPrintOutputType](#).

PpPrintOutputType can be one of these PpPrintOutputType constants.

ppPrintOutputBuildSlides

ppPrintOutputFourSlideHandouts

ppPrintOutputNineSlideHandouts

ppPrintOutputNotesPages

ppPrintOutputOneSlideHandouts

ppPrintOutputOutline

ppPrintOutputSixSlideHandouts

ppPrintOutputSlides

ppPrintOutputThreeSlideHandouts

ppPrintOutputTwoSlideHandouts

expression.**OutputType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example prints handouts of the active presentation with six slides to a page.

```
With ActivePresentation
    .PrintOptions.OutputType = ppPrintOutputSixSlideHandouts
    .PrintOut
End With
```



PageSetup Property

Returns a [PageSetup](#) object whose properties control slide setup attributes for the specified presentation. Read-only.

Example

The following example sets the slide size and slide orientation for the presentation named "Pres1."

```
With Presentations("pres1").PageSetup
    .SlideSize = ppSlideSize35MM
    .SlideOrientation = msoOrientationHorizontal
End With
```



[Show All](#)

Panes Property

Returns a [Panels](#) collection that represents the [panes](#) in the document window.
Read-only.

Example

This example tests for the number of panes in the active window. If the value is one, indicating any view other than normal view, normal view is activated.

```
If ActiveWindow.Panes.Count = 1 Then  
    ActiveWindow.ViewType = ppViewNormal  
End If
```



Paragraph Property

Returns or sets a **Long** that represents the paragraph in a text range to which to apply animation effects. Read/write.

expression.**Paragraph**

expression Required. An expression that returns one of the objects in the Applies To list.



ParagraphFormat Property

Returns a [ParagraphFormat](#) object that represents paragraph formatting for the specified text. Read-only.

Example

This example sets the line spacing before, within, and after each paragraph in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(2).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat
    .LineRuleWithin = msoTrue
    .SpaceWithin = 1.4
    .LineRuleBefore = msoTrue
    .SpaceBefore = 0.25
    .LineRuleAfter = msoTrue
    .SpaceAfter = 0.75
  End With
End With
```



Parent Property

Returns the parent object for the specified object.

expression.**Parent**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds an oval containing text to slide one in the active presentation and rotates the oval and the text 45 degrees. The parent object for the text frame is the **Shape** object that contains the text.

```
Set myShapes = ActivePresentation.Slides(1).Shapes
With myShapes.AddShape(Type:=msoShapeOval, Left:=50, _
    Top:=50, Width:=300, Height:=150).TextFrame
    .TextRange.Text = "Test text"
    .Parent.Rotation = 45
End With
```



ParentGroup Property

Returns a **Shape** object that represents the common parent shape of a child shape or a range of child shapes.

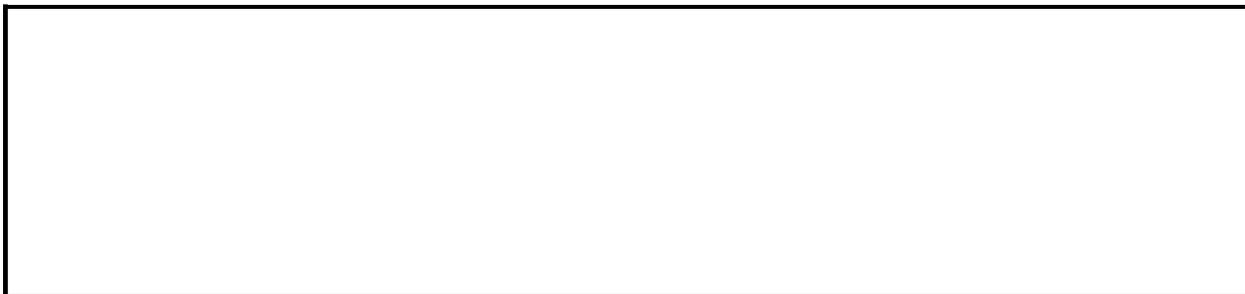
expression.**ParentGroup**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example creates two shapes on the first slide in the active presentation and groups those shapes; then using one shape in the group, accesses the parent group and fills all shapes in the parent group with the same fill color. This example assumes that the first slide of the active presentation does not currently contain any shapes. If it does, you will receive an error.

```
Sub ParentGroup()  
    Dim sldNewSlide As Slide  
    Dim shpParentGroup As Shape  
  
    'Add two shapes to active document and group  
    Set sldNewSlide = ActivePresentation.Slides _  
        .Add(Index:=1, Layout:=ppLayoutBlank)  
  
    With sldNewSlide.Shapes  
        .AddShape Type:=msoShapeBalloon, Left:=72, _  
            Top:=72, Width:=100, Height:=100  
        .AddShape Type:=msoShapeOval, Left:=110, _  
            Top:=120, Width:=100, Height:=100  
        .Range(Array(1, 2)).Group  
    End With  
  
    Set shpParentGroup = ActivePresentation.Slides(1).Shapes(1) _  
        .GroupItems(1).ParentGroup  
    shpParentGroup.Fill.ForeColor.RGB = RGB _  
        (Red:=151, Green:=51, Blue:=250)  
  
End Sub
```



Password Property

Returns or sets a **String** that represents a password that must be supplied to open the specified presentation. Read/write.

expression.**Password**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example opens Earnings.ppt, sets a password for it, and then closes the presentation.

```
Sub SetPassword()  
    With Presentations.Open(FileName:="C:\My Documents\Earnings.ppt"  
        .Password = complexstrPWD 'global variable  
        .Save  
        .Close  
    End With  
End Sub
```



PasswordEncryptionAlgorithm Property

Returns a **String** indicating the algorithm Microsoft PowerPoint uses for encrypting documents with passwords. Read-only.

expression.**PasswordEncryptionAlgorithm**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [SetPasswordEncryptionOptions](#) method to specify the algorithm PowerPoint uses for encrypting documents with passwords.

Example

This example sets the password encryption options if the password encryption algorithm in use is not RC4.

```
Sub PasswordSettings()  
  With ActivePresentation  
    If .PasswordEncryptionAlgorithm <> "RC4" Then  
      .SetPasswordEncryptionOptions _  
        PasswordEncryptionProvider:="Microsoft RSA SChannel  
        PasswordEncryptionAlgorithm:="RC4", _  
        PasswordEncryptionKeyLength:=56, _  
        PasswordEncryptionFileProperties:=True  
    End If  
  End With  
End Sub
```



[Show All](#)

PasswordEncryptionFileProperties Property

Returns **MsoTrue** if Microsoft PowerPoint encrypts file properties for password-protected documents. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**PasswordEncryptionFileProperties**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [SetPasswordEncryptionOptions](#) method to specify the algorithm PowerPoint uses for encrypting documents with passwords.

Example

This example sets the password encryption options if the file properties are not encrypted for password-protected documents.

```
Sub PasswordSettings()  
  With ActivePresentation  
    If .PasswordEncryptionFileProperties = msoFalse Then  
      .SetPasswordEncryptionOptions _  
        PasswordEncryptionProvider:="Microsoft RSA SChannel  
        PasswordEncryptionAlgorithm:="RC4", _  
        PasswordEncryptionKeyLength:=56, _  
        PasswordEncryptionFileProperties:=True  
    End If  
  End With  
End Sub
```



PasswordEncryptionKeyLength Property

Returns a **Long** indicating the key length of the algorithm Microsoft PowerPoint uses when encrypting documents with passwords. Read-only.

expression.**PasswordEncryptionKeyLength**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [SetPasswordEncryptionOptions](#) method to specify the algorithm PowerPoint uses for encrypting documents with passwords.

Example

This example sets the password encryption options if the password encryption key length is less than 40.

```
Sub PasswordSettings()  
  With ActivePresentation  
    If .PasswordEncryptionKeyLength < 40 Then  
      .SetPasswordEncryptionOptions  
    —  
      PasswordEncryptionProvider:="Microsoft RSA SChannel  
      PasswordEncryptionAlgorithm:="RC4", _  
      PasswordEncryptionKeyLength:=56, _  
      PasswordEncryptionFileProperties:=True  
    End If  
  End With  
End Sub
```



PasswordEncryptionProvider Property

Returns a **String** specifying the name of the algorithm encryption provider that Microsoft PowerPoint uses when encrypting documents with passwords. Read-only.

expression.**PasswordEncryptionProvider**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [SetPasswordEncryptionOptions](#) method to specify the algorithm PowerPoint uses for encrypting documents with passwords.

Example

This example sets the password encryption options if the password encryption algorithm in use is not the Microsoft RSA SChannel Cryptographic Provider.

```
Sub PasswordSettings()  
  With ActivePresentation  
    If .PasswordEncryptionProvider <> "Microsoft RSA SChannel Cr  
      .SetPasswordEncryptionOptions _  
        PasswordEncryptionProvider:="Microsoft RSA SChannel  
        PasswordEncryptionAlgorithm:="RC4", _  
        PasswordEncryptionKeyLength:=56, _  
        PasswordEncryptionFileProperties:=True  
    End If  
  End With  
End Sub
```



Path Property

Returns a **String** that represents the path to the specified [AddIn](#), [Application](#), or [Presentation](#) object or the path followed by a [MotionEffect](#) object. Read-only.

expression.**Path**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

If you use this property to return a path for a presentation that has not been saved, it returns an empty string. Using this property to return the path for an add-in that has not been loaded causes an error.

The path doesn't include the final backslash (\) or the name of the specified object. Use the [Name](#) property of the **Presentation** object to return the file name without the path, and use the [FullName](#) property to return the file name and the path together.

The **String** returned for a **MotionEffect** object is a specific path that the motion effect follows between From and To using the same syntax as the VML path description.

Example

This example saves the active presentation in the same folder as PowerPoint.

```
With Application
    fName = .Path & "\test presentation"
    ActivePresentation.SaveAs fName
End With
```



[Show All](#)

Pattern Property

 [Pattern property as it applies to the **LineFormat** object.](#)

Sets or returns a value that represents the pattern applied to the specified line.
Read/write [MsoPatternType](#).

MsoPatternType can be one of these MsoPatternType constants.

msoPattern10Percent

msoPattern20Percent

msoPattern25Percent

msoPattern30Percent

msoPattern40Percent

msoPattern50Percent

msoPattern5Percent

msoPattern60Percent

msoPattern70Percent

msoPattern75Percent

msoPattern80Percent

msoPattern90Percent

msoPatternDarkDownwardDiagonal

msoPatternDarkHorizontal

msoPatternDarkUpwardDiagonal

msoPatternDashedDownwardDiagonal

msoPatternDashedHorizontal

msoPatternDashedUpwardDiagonal

msoPatternDashedVertical

msoPatternDiagonalBrick

msoPatternDivot

msoPatternDottedDiamond

msoPatternDottedGrid

msoPatternHorizontalBrick

msoPatternLargeCheckerBoard
msoPatternLargeConfetti
msoPatternLargeGrid
msoPatternLightDownwardDiagonal
msoPatternLightHorizontal
msoPatternLightUpwardDiagonal
msoPatternLightVertical
msoPatternMixed
msoPatternNarrowHorizontal
msoPatternNarrowVertical
msoPatternOutlinedDiamond
msoPatternPlaid
msoPatternShingle
msoPatternSmallCheckerBoard
msoPatternSmallConfetti
msoPatternSmallGrid
msoPatternSolidDiamond
msoPatternSphere
msoPatternTrellis
msoPatternWave
msoPatternWeave
msoPatternWideDownwardDiagonal
msoPatternWideUpwardDiagonal
msoPatternZigZag
msoPatternDarkVertical

expression.**Pattern**

expression Required. An expression that returns one of the above objects.

[Pattern property as it applies to the **FillFormat** object.](#)

Sets or returns a value that represents the pattern applied to the specified fill. Use the [BackColor](#) and [ForeColor](#) properties to set the colors used in the pattern. Read-only [MsoPatternType](#).

MsoPatternType can be one of these MsoPatternType constants.

msoPattern10Percent

msoPattern20Percent

msoPattern25Percent

msoPattern30Percent

msoPattern40Percent

msoPattern50Percent

msoPattern5Percent

msoPattern60Percent

msoPattern70Percent

msoPattern75Percent

msoPattern80Percent

msoPattern90Percent

msoPatternDarkDownwardDiagonal

msoPatternDarkHorizontal

msoPatternDarkUpwardDiagonal

msoPatternDashedDownwardDiagonal

msoPatternDashedHorizontal

msoPatternDashedUpwardDiagonal

msoPatternDashedVertical

msoPatternDiagonalBrick

msoPatternDivot

msoPatternDottedDiamond

msoPatternDottedGrid

msoPatternHorizontalBrick

msoPatternLargeCheckerBoard

msoPatternLargeConfetti

msoPatternLargeGrid

msoPatternLightDownwardDiagonal

msoPatternLightHorizontal

msoPatternLightUpwardDiagonal

msoPatternLightVertical

msoPatternMixed

msoPatternNarrowHorizontal
msoPatternNarrowVertical
msoPatternOutlinedDiamond
msoPatternPlaid
msoPatternShingle
msoPatternSmallCheckerBoard
msoPatternSmallConfetti
msoPatternSmallGrid
msoPatternSolidDiamond
msoPatternSphere
msoPatternTrellis
msoPatternWave
msoPatternWeave
msoPatternWideDownwardDiagonal
msoPatternWideUpwardDiagonal
msoPatternZigZag
msoPatternDarkVertical

expression.**Pattern**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **LineFormat** object.](#)

This example adds a patterned line to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 100, 250, 0).Line
    .Weight = 6
    .ForeColor.RGB = RGB(0, 0, 255)
    .BackColor.RGB = RGB(128, 0, 0)
    .Pattern = msoPatternDarkDownwardDiagonal
End With
```

 [As it applies to the **FillFormat** object.](#)

This example adds a rectangle to myDocument and sets its fill pattern to match that of the shape named "rect1." The new rectangle has the same pattern as rect1, but not necessarily the same colors. The colors used in the pattern are set with the **BackColor** and **ForeColor** properties.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    pattern1 = .Item("rect1").Fill.Pattern
    With .AddShape(msoShapeRectangle, 100, 100, 120, 80).Fill
        .ForeColor.RGB = RGB(128, 0, 0)
        .BackColor.RGB = RGB(0, 0, 255)
        .Patterned pattern1
    End With
End With
```



[Show All](#)

PauseAnimation Property

Determines whether the slide show pauses until the specified media clip is finished playing. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The slide show continues while the media clip plays in the background.

msoTriStateMixed

msoTriStateToggle

msoTrue The slide show pauses until the specified media clip is finished playing.

Remarks

For the **PauseAnimation** property setting to take effect, the [PlayOnEntry](#) property of the specified shape must be set to **msoTrue**.

Example

This example specifies that shape three on slide one in the active presentation will be played automatically when it's animated and that the slide show won't continue while the movie is playing in the background. Shape three must be a sound or movie object.

```
Set OLEobj = ActivePresentation.Slides(1).Shapes(3)
With OLEobj.AnimationSettings.PlaySettings
    .PlayOnEntry = msoTrue
    .PauseAnimation = msoTrue
End With
```



Permission Property

Returns a [Permission](#) object that can be used to restrict permissions to the active presentation and to return or set specific permissions settings . Read-only.

Note Use of the **Permission** object raises an error if the Windows Rights Management client is not installed.

expression.**Permission**

expression Required. An expression that returns a **Presentation** object.

Remarks

Use the **Permission** object to restrict permissions to the active document and to return or set specific permissions settings.

Use the **Enabled** property to determine whether permissions are restricted on the active document. Use the **Count** property to return the number of users with permissions, and the **RemoveAll** method to reset all existing permissions.

The **DocumentAuthor**, **EnableTrustedBrowser**, **RequestPermissionURL**, and **StoreLicenses** properties provide additional information about permission settings.

The **Permission** object gives access to a collection of **UserPermission** objects. Use the **UserPermission** object to associate specific sets of rights with individual users. While some permissions granted through the user interface (such as **msoPermissionPrint**) apply to all users, you can use the **UserPermission** object to assign them on a per-user basis with per-user expiration dates.

Information Rights Management, included in Microsoft Office 2003, supports the use of administrative permission policies which list users and groups and their document permissions. Use the **ApplyPolicy** method to apply a permission policy, and the **PermissionFromPolicy**, **PolicyName**, and **PolicyDescription** properties to return policy information.

The **Permission** object model is available whether permissions are restricted on the active document or not. The **Permission** property of the **Presentation** object does not return **Nothing** when the active document does not have restricted permissions. Use the **Enabled** property to determine whether a document has restricted permissions.

Example

The following example creates a new presentation and assigns the user with e-mail address "someone@example.com" read permission on the new presentation. The example will display the permissions of the owner and the new user.

```
Sub AddUserPermissions()  
    Dim myPres As PowerPoint.Presentation  
    Dim myPer As Office.Permission  
    Dim NewOwnerPer As Office.UserPermission  
    Set myPres = Application.Presentations.Add(msoTrue)  
    Set myPer = myPres.Permission  
    myPer.Enabled = True  
    Set NewOwnerPer = myPer.Add("someone@example.com", msoPermissionRea  
    MsgBox myPer(1).UserId + " " + Str(myPer(1).Permission)  
    MsgBox myPer(2).UserId + " " + Str(myPer(2).Permission)  
End Sub
```



[Show All](#)

Perspective Property

Determines whether the extrusion appears in perspective. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The extrusion is a parallel, or orthographic, projection— that is, if the walls don't narrow toward a vanishing point.

msoTriStateMixed

msoTriStateToggle

msoTrue The extrusion appears in perspective— that is, if the walls of the extrusion narrow toward a vanishing point.

Example

This example sets the extrusion depth for shape one on myDocument to 100 points and specifies that the extrusion be parallel, or orthographic.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .Depth = 100
    .Perspective = msoFalse
End With
```



PictureFormat Property

Returns a [PictureFormat](#) object that contains picture formatting properties for the specified shape. Applies to **Shape** or **ShapeRange** objects that represent pictures or OLE objects. Read-only.

Example

This example sets the brightness and contrast for shape one on myDocument. Shape one must be a picture or an OLE object.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).PictureFormat
    .Brightness = 0.3
    .Contrast = .75
End With
```



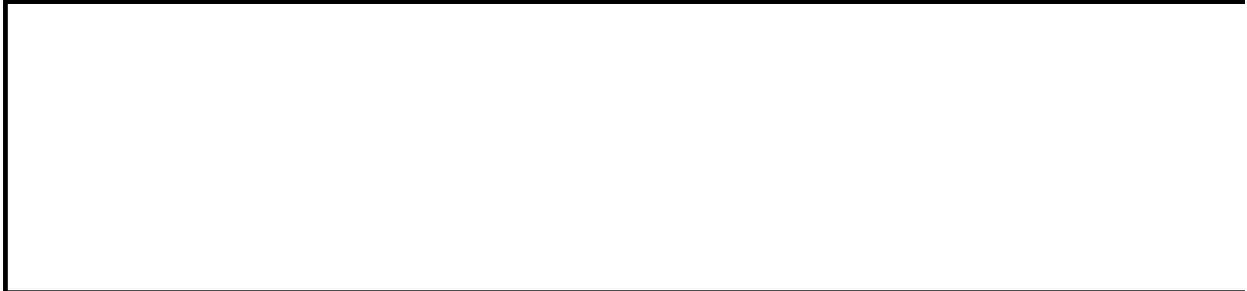
PlaceholderFormat Property

Returns a [PlaceholderFormat](#) object that contains the properties that are unique to placeholders. Read-only.

Example

This example adds text to placeholder one on slide one in the active presentation if that placeholder is a horizontal title placeholder.

```
With ActivePresentation.Slides(1).Shapes.Placeholders
  If .Count > 0 Then
    With .Item(1)
      Select Case .PlaceholderFormat.Type
        Case ppPlaceholderTitle
          .TextFrame.TextRange = "Title Text"
        Case ppPlaceholderCenterTitle
          .TextFrame.TextRange = "Centered Title Text"
        Case Else
          MsgBox "There's no horizontal" & _
            "title on this slide"
      End Select
    End With
  End If
End With
```



Placeholders Property

Returns a [Placeholders](#) collection that represents the collection of all the placeholders on a slide. Each placeholder in the collection can contain text, a chart, a table, an organizational chart, or another object. Read-only.

Example

This example adds a slide to the active presentation and then adds text to both the title (which is the first placeholder on the slide) and the subtitle.

```
Set myDocument = ActivePresentation.Slides(1)
With ActivePresentation.Slides _
    .Add(1, ppLayoutTitle).Shapes.Placeholders
        .Item(1).TextFrame.TextRange.Text = "This is the title text"
        .Item(2).TextFrame.TextRange.Text = "This is subtitle text"
End With
```



[Show All](#)

PlayOnEntry Property

Determines whether the specified movie or sound is played automatically when it's animated. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified movie or sound is played automatically when it's animated.

Remarks

Setting this property to **msoTrue** sets the [Animate](#) property of the **AnimationSettings** object to **msoTrue**. Setting the **Animate** property to **msoFalse** automatically sets the **PlayOnEntry** property to **msoFalse**.

Use the [ActionVerb](#) property to set the verb that will be invoked when the media clip is animated.

Example

This example specifies that shape three on slide one in the active presentation will be played automatically when it's animated. Shape three must be a sound or movie object.

```
Set OLEobj = ActivePresentation.Slides(1).Shapes(3)  
OLEobj.AnimationSettings.PlaySettings.PlayOnEntry = msoTrue
```



PlaySettings Property

Returns a [PlaySettings](#) object that contains information about how the specified media clip plays during a slide show.

expression.**PlaySettings**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example inserts a movie named Clock.avi onto slide one in the active presentation, sets it to play automatically after the slide transition, and specifies that the movie object be hidden during a slide show except when it's playing.

```
With ActivePresentation.Slides(1).Shapes.AddOLEObject(Left:=10, _  
    Top:=10, Width:=250, Height:=250, _  
    FileName:="c:\winnt\Clock.avi")  
    With .AnimationSettings.PlaySettings  
        .PlayOnEntry = True  
        .HideWhileNotPlaying = True  
    End With  
End With
```



[Show All](#)

PointerColor Property

[PointerColor Property as it applies to the **SlideShowSettings** object.](#)

Returns the pointer color for the specified presentation as a **ColorFormat** object. This color is saved with the presentation and is the default pen color each time you show the presentation. Read-only.

[PointerColor Property as it applies to the **SlideShowView** object.](#)

Returns a **ColorFormat** object that represents the pointer color for the specified presentation during one slide show. As soon as the slide show is finished, the color reverts to the default color for the presentation. Read-only.

Remarks

To change the pointer to a pen, set the [PointerType](#) property to **ppSlideShowPointerPen**.

Example

 [As it applies to the **SlideShowSettings** object.](#)

This example sets the default pen color for the active presentation to blue, starts a slide show, changes the pointer to a pen, and then sets the pen color to red for this slide show only.

```
With ActivePresentation.SlideShowSettings
    .PointerColor.RGB = RGB(0, 0, 255)           'blue
With .Run.View
    .PointerColor.RGB = RGB(255, 0, 0)         'red
    .PointerType = ppSlideShowPointerPen
End With
End With
```



[Show All](#)

PointerType Property

Returns or sets the type of pointer used in the slide show. Read/write [PpSlideShowPointerType](#).

PpSlideShowPointerType can be one of these PpSlideShowPointerType constants.

ppSlideShowPointerAlwaysHidden

ppSlideShowPointerArrow

ppSlideShowPointerAutoArrow

ppSlideShowPointerNone

ppSlideShowPointerPen

expression.**PointerType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example runs a slide show of the active presentation, changes the pointer to a pen, and sets the pen color for this slide show to red.

```
Set currView = ActivePresentation.SlideShowSettings.Run.View  
With currView  
    .PointerColor.RGB = RGB(255, 0, 0)  
    .PointerType = ppSlideShowPointerPen  
End With
```



[Show All](#)

Points Property

 [Points property as it applies to the **PropertyEffect** object.](#)

Returns an **AnimationPoints** object that represents a point in an animation. Use the **From** and **To** properties to set the value of this property.

expression.**Points**

expression Required. An expression that returns a **PropertyEffect** object.

 [Points property as it applies to the **ShapeNode** object.](#)

Returns a **Variant** that represents the position of the specified node as a [coordinate pair](#). Each coordinate is expressed in points. Use the **SetPosition** method to set the value of this property. Read-only.

expression.**Points**

expression Required. An expression that returns a **ShapeNode** object.

Example

 [As it applies to the **ShapeNode** object.](#)

This example moves node two in shape three in the active presentation to the right 200 points and down 300 points. Shape three must be a freeform drawing.

```
With ActivePresentation.Slides(1).Shapes(3).Nodes
  pointsArray = .Item(2).Points
  currXvalue = pointsArray(1, 1)
  currYvalue = pointsArray(1, 2)
  .SetPosition Index:=2, X1:=currXvalue + 200, Y1:=currYvalue + 300
End With
```



Position Property

Returns or sets the position of the specified tab stop, in points. Read/write **Single**.

Example

This example deletes all tab stops greater than 1 inch (72 points) for the text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame _  
    .Ruler.TabStops  
    For i = .Count To 1 Step -1  
        With .Item(i)  
            If .Position > 72 Then .Clear  
        End With  
    Next  
End With
```



Presentation Property

Returns a [Presentation](#) object that represents the presentation in which the specified document window or slide show window was created. Read-only.

Remarks

If the slide that's currently displayed in document window one is from an embedded presentation, `Windows(1).View.Slide.Parent` returns the embedded presentation, and `Windows(1).Presentation` returns the presentation in which document window one was created.

If the slide that's currently displayed in slide show window one is from an embedded presentation, `SlideShowWindows(1).View.Slide.Parent` returns the embedded presentation, and `SlideShowWindows(1).Presentation` returns the presentation in which the slide show was started.

Example

This example continues the slide numbering for the presentation in window one into the slide numbering for the presentation in window two.

```
firstPresSlides = Windows(1).Presentation.Slides.Count  
Windows(2).Presentation.PageSetup _  
    .FirstSlideNumber = firstPresSlides + 1
```



PresentationElapsedTime Property

Returns the number of seconds that have elapsed since the beginning of the specified slide show. Read-only **Long**.

Example

This example goes to slide seven in slide show window one if more than five minutes have elapsed since the beginning of the slide show.

```
With SlideShowWindows(1).View
  If .PresentationElapsedTime > 300 Then
    .GotoSlide 7
  End If
End With
```



Presentations Property

Returns a [Presentations](#) collection that represents all open presentations. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example opens the presentation named "Long Version.ppt."

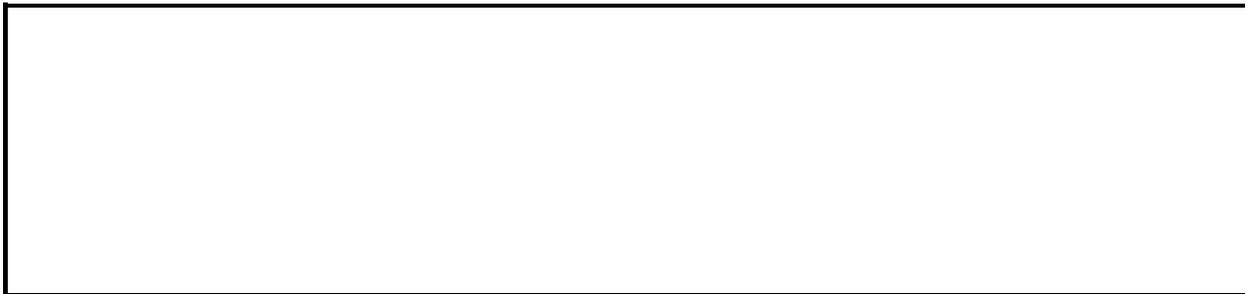
```
Application.Presentations.Open _  
    FileName:="c:\My Documents\Long version.ppt"
```

This example saves presentation one as "Year-End Report.ppt."

```
Application.Presentations(1).SaveAs "Year-End Report"
```

This example closes the Year-end report presentation.

```
Application.Presentations("Year-End Report.ppt").Close
```



[Show All](#)

Preserved Property

Sets or returns an [MsoTriState](#) constant that represents whether a design master is preserved from changes. Read/write.

MsoTriState can be one of these MsoTriState constants.

msoCTrue Doesn't apply to this property.

msoFalse The design master is not preserved and can be edited.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The design master is preserved and cannot be edited.

expression.**Preserved**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following line of code locks and preserves the first design master.

```
Sub PreserveMaster  
    ActivePresentation.Designs(1).Preserved = msoTrue  
End Sub
```



[Show All](#)

PresetExtrusionDirection Property

Returns the direction that the extrusion's sweep path takes away from the extruded shape (the front face of the extrusion). Read-only

[MsoPresetExtrusionDirection](#).

MsoPresetExtrusionDirection can be one of these MsoPresetExtrusionDirection constants.

msoExtrusionBottom

msoExtrusionBottomLeft

msoExtrusionBottomRight

msoExtrusionLeft

msoExtrusionNone

msoExtrusionRight

msoExtrusionTop

msoExtrusionTopLeft

msoExtrusionTopRight

msoPresetExtrusionDirectionMixed

expression.**PresetExtrusionDirection**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This property is read-only. To set the value of this property, use the [SetExtrusionDirection](#) method.

Example

This example changes each extrusion on myDocument that extends toward the upper-left corner of the extrusion's front face to an extrusion that extends toward the lower-right corner of the front face.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  With s.ThreeD
    If .PresetExtrusionDirection = msoExtrusionTopLeft Then
      .SetExtrusionDirection msoExtrusionBottomRight
    End If
  End With
Next
```



[Show All](#)

PresetGradientType Property

Returns the preset gradient type for the specified fill. Read-only [MsoPresetGradientType](#). Use the [PresetGradient](#) method to set the preset gradient type for the fill.

MsoPresetGradientType can be one of these MsoPresetGradientType constants.

msoGradientBrass

msoGradientCalmWater

msoGradientChrome

msoGradientChromeII

msoGradientDaybreak

msoGradientDesert

msoGradientEarlySunset

msoGradientFire

msoGradientFog

msoGradientGold

msoGradientGoldII

msoGradientHorizon

msoGradientLateSunset

msoGradientMahogany

msoGradientMoss

msoGradientNightfall

msoGradientOcean

msoGradientParchment

msoGradientPeacock

msoGradientRainbow

msoGradientRainbowII

msoGradientSapphire

msoGradientSilver

msoGradientWheat

msoPresetGradientMixed

expression.**PresetGradientType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example changes the fill for all shapes in myDocument with the Moss preset gradient fill to the Fog preset gradient fill.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  With s.Fill
    If .PresetGradientType = msoGradientMoss Then
      .PresetGradient = msoGradientFog
    End If
  End With
Next
```



[Show All](#)

PresetLightingDirection Property

Returns or sets the position of the light source relative to the extrusion.

Read/write [MsoPresetLightingDirection](#).

MsoPresetLightingDirection can be one of these MsoPresetLightingDirection constants.

msoLightingBottom

msoLightingBottomLeft

msoLightingBottomRight

msoLightingLeft

msoLightingNone

msoLightingRight

msoLightingTop

msoLightingTopLeft

msoLightingTopRight

msoPresetLightingDirectionMixed

expression.**PresetLightingDirection**

expression Required. An expression that returns one of the objects in the Applies To list.

Note You won't see the lighting effects you set if the extrusion has a wire frame surface.

Example

This example specifies that the extrusion for shape one on myDocument extend toward the top of the shape and that the lighting for the extrusion come from the left.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```



[Show All](#)

PresetLightingSoftness Property

Returns or sets the intensity of the extrusion lighting. Read/write [MsoPresetLightingSoftness](#).

MsoPresetLightingSoftness can be one of these MsoPresetLightingSoftness constants.

msoLightingBright

msoLightingDim

msoLightingNormal

msoPresetLightingSoftnessMixed

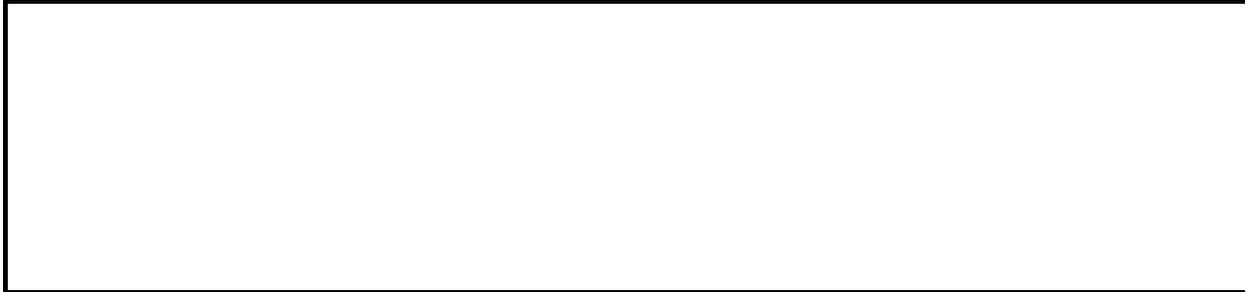
expression.**PresetLightingSoftness**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example specifies that the extrusion for shape one on myDocument be lit brightly from the left.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .PresetLightingSoftness = msoLightingBright
    .PresetLightingDirection = msoLightingLeft
End With
```



[Show All](#)

PresetMaterial Property

Returns or sets the extrusion surface material. Read/write [MsoPresetMaterial](#).

MsoPresetMaterial can be one of these MsoPresetMaterial constants.

msoMaterialMatte

msoMaterialMetal

msoMaterialPlastic

msoMaterialWireFrame

msoPresetMaterialMixed

expression.**PresetMaterial**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example specifies that the extrusion surface for shape one in myDocument be wire frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .PresetMaterial = msoMaterialWireFrame
End With
```



[Show All](#)

PresetShape Property

Returns or sets the shape of the specified WordArt. Read/write [MsoPresetTextEffectShape](#).

MsoPresetTextEffectShape can be one of these MsoPresetTextEffectShape constants.

msoTextEffectShapeArchDownCurve

msoTextEffectShapeArchDownPour

msoTextEffectShapeArchUpCurve

msoTextEffectShapeArchUpPour

msoTextEffectShapeButtonCurve

msoTextEffectShapeButtonPour

msoTextEffectShapeCanDown

msoTextEffectShapeCanUp

msoTextEffectShapeCascadeDown

msoTextEffectShapeCascadeUp

msoTextEffectShapeChevronDown

msoTextEffectShapeChevronUp

msoTextEffectShapeCircleCurve

msoTextEffectShapeCirclePour

msoTextEffectShapeCurveDown

msoTextEffectShapeCurveUp

msoTextEffectShapeDeflate

msoTextEffectShapeDeflateBottom

msoTextEffectShapeDeflateInflate

msoTextEffectShapeDeflateInflateDeflate

msoTextEffectShapeDeflateTop

msoTextEffectShapeDoubleWave2

msoTextEffectShapeFadeDown

msoTextEffectShapeFadeLeft

msoTextEffectShapeFadeRight

msoTextEffectShapeFadeUp
msoTextEffectShapeInflate
msoTextEffectShapeInflateBottom
msoTextEffectShapeInflateTop
msoTextEffectShapeMixed
msoTextEffectShapePlainText
msoTextEffectShapeRingInside
msoTextEffectShapeRingOutside
msoTextEffectShapeSlantDown
msoTextEffectShapeSlantUp
msoTextEffectShapeStop
msoTextEffectShapeTriangleDown
msoTextEffectShapeTriangleUp
msoTextEffectShapeWave1
msoTextEffectShapeWave2
msoTextEffectShapeDoubleWave1

expression.**PresetShape**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Setting the [PresetTextEffect](#) property automatically sets the **PresetShape** property.

Example

This example sets the shape of all WordArt on myDocument to a chevron whose center points down.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.Type = msoTextEffect Then
        s.TextEffect.PresetShape = msoTextEffectShapeChevronDown
    End If
Next
```



[Show All](#)

PresetTextEffect Property

Returns or sets the style of the specified WordArt. The values for this property correspond to the formats in the **WordArt Gallery** dialog box (numbered from left to right, top to bottom). Read/write [MsoPresetTextEffect](#).

MsoPresetTextEffect can be one of these MsoPresetTextEffect constants.

msoTextEffect1

msoTextEffect2

msoTextEffect3

msoTextEffect4

msoTextEffect5

msoTextEffect6

msoTextEffect7

msoTextEffect8

msoTextEffect9

msoTextEffect10

msoTextEffect11

msoTextEffect12

msoTextEffect13

msoTextEffect14

msoTextEffect15

msoTextEffect16

msoTextEffect17

msoTextEffect18

msoTextEffect19

msoTextEffect20

msoTextEffect21

msoTextEffect22

msoTextEffect23

msoTextEffect24

msoTextEffect25

msoTextEffect26

msoTextEffect27

msoTextEffect28

msoTextEffect29

msoTextEffect30

msoTextEffectMixed

expression.**PresetTextEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Setting the **PresetTextEffect** property automatically sets many other formatting properties of the specified shape.

Example

This example sets the style for all WordArt on myDocument to the first style listed in the **WordArt Gallery** dialog box.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.Type = msoTextEffect Then
        s.TextEffect.PresetTextEffect = msoTextEffect1
    End If
Next
```



[Show All](#)

PresetTexture Property

Returns the preset texture for the specified fill. Read-only [MsoPresetTexture](#).

MsoPresetTexture can be one of these MsoPresetTexture constants.

msoPresetTextureMixed
msoTextureBlueTissuePaper
msoTextureBouquet
msoTextureBrownMarble
msoTextureCanvas
msoTextureCork
msoTextureDenim
msoTextureFishFossil
msoTextureGranite
msoTextureGreenMarble
msoTextureMediumWood
msoTextureNewsprint
msoTextureOak
msoTexturePaperBag
msoTexturePapyrus
msoTextureParchment
msoTexturePinkTissuePaper
msoTexturePurpleMesh
msoTextureRecycledPaper
msoTextureSand
msoTextureStationery
msoTextureWalnut
msoTextureWaterDroplets
msoTextureWhiteMarble
msoTextureWovenMat

expression.PresetTexture

expression Required. An expression that returns one of the objects in the Applies To list.

Use the [PresetTextured](#) method to set the preset texture for the fill.

Example

This example adds a rectangle to the myDocument and sets its preset texture to match that of shape two. For the example to work, shape two must have a preset textured fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    presetTexture2 = .Item(2).Fill.PresetTexture
    .AddShape(msoShapeRectangle, 100, 0, 40, 80).Fill _
        .PresetTextured presetTexture2
End With
```



[Show All](#)

PresetThreeDFormat Property

Returns the preset extrusion format. Each preset extrusion format contains a set of preset values for the various properties of the extrusion. The values for this property correspond to the options (numbered from left to right, top to bottom) displayed when you click the **3-D** button on the **Drawing** toolbar. Read-only [MsoPresetThreeDFormat](#).

MsoPresetThreeDFormat can be one of these MsoPresetThreeDFormat constants.

msoPresetThreeDFormatMixed The extrusion has a custom format rather than a preset format.

msoThreeD1

msoThreeD2

msoThreeD3

msoThreeD4

msoThreeD5

msoThreeD6

msoThreeD7

msoThreeD8

msoThreeD9

msoThreeD10

msoThreeD11

msoThreeD12

msoThreeD13

msoThreeD14

msoThreeD15

msoThreeD16

msoThreeD17

msoThreeD18

msoThreeD19

msoThreeD20

expression.**PresetThreeDFormat**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This property is read-only. To set the preset extrusion format, use the [SetThreeDFormat](#) method.

Example

This example sets the extrusion format for shape one on myDocument to 3D Style 12 if the shape initially has a custom extrusion format.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    If .PresetThreeDFormat = msoPresetThreeDFormatMixed Then
        .SetThreeDFormat msoThreeD12
    End If
End With
```



[Show All](#)

PrintColorType Property

Returns or sets the way the specified document will be printed: in black and white, in pure black and white (also referred to as high contrast), or in color. The default value is set by the printer. Read/write [PpPrintColorType](#).

PpPrintColorType can be one of these PpPrintColorType constants.

ppPrintBlackAndWhite

ppPrintColor

ppPrintPureBlackAndWhite

expression.**PrintColorType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example prints the slides in the active presentation in color.

```
With Application.ActivePresentation  
    .PrintOptions.PrintColorType = ppPrintColor  
    .PrintOut  
End With
```



[Show All](#)

PrintComments Property

Sets or returns whether comments will be printed. Read/write [MsoTriState](#).

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. Comments will not be printed.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Comments will be printed.

expression.**PrintComments**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example instructs Microsoft PowerPoint to print comments.

```
Sub PrintPresentationComments
    ActivePresentation.PrintOptions.PrintComments = msoTrue
End Sub
```



[Show All](#)

PrintFontsAsGraphics Property

Determines whether TrueType fonts are printed as graphics. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue TrueType fonts are printed as graphics.

Example

This example specifies that TrueType fonts in the active presentation be printed as graphics.

```
ActivePresentation.PrintOptions.PrintFontsAsGraphics = msoTrue
```



[Show All](#)

PrintHiddenSlides Property

Determines whether hidden slides in the specified presentation will be printed.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

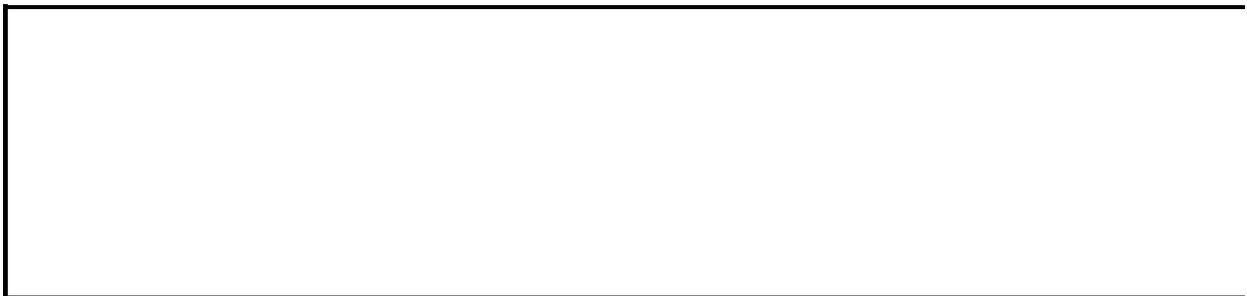
msoTriStateToggle

msoTrue The hidden slides in the specified presentation will be printed.

Example

This example prints all slides, whether visible or hidden, in the active presentation.

```
With ActivePresentation
    .PrintOptions.PrintHiddenSlides = msoTrue
    .PrintOut
End With
```



[Show All](#)

PrintInBackground Property

Determines whether the specified presentation is printed in the background.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Default. The specified presentation is printed in the background, which means that you can continue to work while it's being printed.

Example

This example prints the active presentation in the background.

```
With ActivePresentation  
    .PrintOptions.PrintInBackground = msoTrue  
    .PrintOut  
End With
```



PrintOptions Property

Returns a [PrintOptions](#) object that represents print options that are saved with the specified presentation. Read-only.

Example

This example causes hidden slides in the active presentation to be printed, and it scales the printed slides to fit the paper size.

```
With Application.ActivePresentation
  With .PrintOptions
    .PrintHiddenSlides = True
    .FitToPage = True
  End With
  .PrintOut
End With
```



PrintSteps Property

Returns the number of slides you'd need to print to simulate the builds on the specified slide, slide master, or range of slides. Read-only **Long**.

Example

This example sets a variable to the number of slides you'd need to print to simulate the builds on slide one in the active presentation and then displays the value of the variable.

```
steps1 = ActivePresentation.Slides(1).PrintSteps  
MsgBox steps1
```



ProductCode Property

Returns the Microsoft PowerPoint globally unique identifier (GUID). You might use the GUID, for example, when making program calls to an Application Programming Interface (API). Read-only **String**.

Example

This example returns the PowerPoint GUID to the variable pptGUID.

```
Dim pptGUID As String  
pptGUID = Application.ProductCode
```



[Show All](#)

ProgID Property

Returns the [programmatic identifier \(ProgID\)](#) for the specified OLE object.
Read-only **String**.

Example

This example loops through all the objects on all the slides in the active presentation and sets all linked Microsoft Excel worksheets to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Excel.Sheet" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
```



[Show All](#)

Property Property

Sets or returns an [MsoAnimProperty](#) constant that represents an animation property. Read/write.

MsoAnimProperty can be one of these **MsoAnimProperty** constants.

msoAnimColor

msoAnimHeight

msoAnimNone Default.

msoAnimOpacity

msoAnimRotation

msoAnimShape3DExtrudeForward

msoAnimShape3DExtrusionColor

msoAnimShape3DXRotationAngle

msoAnimShape3DYRotationAngle

msoAnimShapefBackColor

msoAnimShapefColor

msoAnimShapefGradientPreset

msoAnimShapefGradientType

msoAnimShapeFlipH

msoAnimShapeFlipV

msoAnimShapefOn

msoAnimShapefOpacity

msoAnimShapefType

msoAnimShapelColor

msoAnimShapelDashes

msoAnimShapelEndArrowHead

msoAnimShapelEndArrowLength

msoAnimShapelEndArrowWidth

msoAnimShapelOn

msoAnimShapelStartArrowHead

msoAnimShapelStartArrowLength

msoAnimShapelStartArrowWidth
msoAnimShapelStyle
msoAnimShapelWidth
msoAnimShapepBrightness
msoAnimShapepContrast
msoAnimShapepCropFromBottom
msoAnimShapepCropFromLeft
msoAnimShapepCropFromRight
msoAnimShapepCropFromTop
msoAnimShapepFilename
msoAnimShapepGamma
msoAnimShapepGrayscale
msoAnimShapepRotation
msoAnimShapesColor
msoAnimShapesEmbossed
msoAnimShapesOffsetX
msoAnimShapesOffsetY
msoAnimShapesOn
msoAnimShapesOpacity
msoAnimShapesType
msoAnimShapeType
msoAnimShapewfontBold
msoAnimShapewfontItalic
msoAnimShapewfontName
msoAnimShapewfontShadow
msoAnimShapewfontSize
msoAnimShapewfontSmallCaps
msoAnimShapewfontStrikeThrough
msoAnimShapewfontUnderline
msoAnimShapewSpacing
msoAnimShapewVertical
msoAnimTextBulletCharacter
msoAnimTextBulletColor

msoAnimTextBulletFontName
msoAnimTextBulletNumber
msoAnimTextBulletPicture
msoAnimTextBulletRelativeSize
msoAnimTextBulletStyle
msoAnimTextBulletType
msoAnimTextFontBold
msoAnimTextFontColor
msoAnimTextFontEmboss
msoAnimTextFontItalic
msoAnimTextFontName
msoAnimTextFontShadow
msoAnimTextFontSize
msoAnimTextFontStrikeThrough
msoAnimTextFontSubscript
msoAnimTextFontSuperscript
msoAnimTextFontUnderline
msoAnimWidth
msoAnimX
msoAnimY

expression.**Property**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, adds a three-second fill animation to that shape, and sets the fill animation to color.

```
Sub AddShapeSetAnimFill()  
  
    Dim effBlinds As Effect  
    Dim shpRectangle As Shape  
    Dim animProperty As AnimationBehavior  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effBlinds = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectBlind  
  
    effBlinds.Timing.Duration = 3  
  
    Set animProperty = effBlinds.Behaviors.Add(msoAnimTypeProperty)  
  
    With animProperty.PropertyEffect  
        .Property = msoAnimColor  
        .From = RGB(Red:=0, Green:=0, Blue:=255)  
        .To = RGB(Red:=255, Green:=0, Blue:=0)  
    End With  
  
End Sub
```



PropertyEffect Property

Returns a [PropertyEffect](#) object for a given animation behavior.

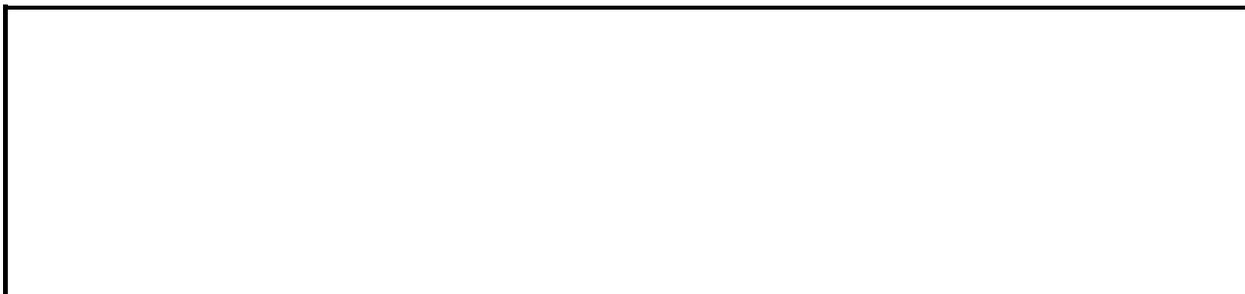
expression.**PropertyEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape with an effect to the active presentation and sets the animation effect properties for the shape to change colors.

```
Sub AddShapeSetAnimFill()  
  
    Dim effBlinds As Effect  
    Dim shpRectangle As Shape  
    Dim animBlinds As AnimationBehavior  
  
    'Adds rectangle and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effBlinds = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectBlind  
  
    'Sets the duration of the animation  
    effBlinds.Timing.Duration = 3  
  
    'Adds a behavior to the animation  
    Set animBlinds = effBlinds.Behaviors.Add(msoAnimTypeProperty)  
  
    'Sets the animation color effect and the formula to use  
    With animBlinds.PropertyEffect  
        .Property = msoAnimColor  
        .From = RGB(Red:=0, Green:=0, Blue:=255)  
        .To = RGB(Red:=255, Green:=0, Blue:=0)  
    End With  
  
End Sub
```



PublishObjects Property

Returns a [PublishObjects](#) collection representing the set of complete or partial loaded presentations that are available to publish to HTML. Read-only.

Example

This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects.Item(1)  
    .FileName = "C:\Test\Mallard.htm"  
    .SourceType = ppPublishSlideRange  
    .RangeStart = 3  
    .RangeEnd = 5  
    .Publish  
End With
```



RangeEnd Property

Returns or sets the number of the last slide in a range of slides you are publishing as a Web presentation. Read/write **Integer**.

Example

This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```



Ranges Property

Returns the [PrintRanges](#) object, which represents the ranges of slides in the presentation to be printed. Read-only.

Remarks

If you don't want to print an entire presentation, you must use the [Add](#) method to create a [PrintRange](#) object for each consecutive run of slides you want to print. For example, if you want to print slide 1, slides 3 through 5, and slides 8 and 9 in a specified presentation, you must create three **PrintRange** objects: one that represents slide 1; one that represents slides 3 through 5; and one that represents slides 8 and 9. For more information, see the example for this property.

The [RangeType](#) property must be set to **ppPrintSlideRange** for the ranges in the **PrintRanges** collection to be applied.

To clear all the existing print ranges from the **PrintRanges** collection, use the [ClearAll](#) method.

Specifying a value for the *To* and *From* arguments of the [PrintOut](#) method sets the contents of the [PrintRanges](#) object.

Example

This example prints slide 1, slides 3 through 5, and slides 8 and 9 in the active presentation.

```
With ActivePresentation
  With .PrintOptions
    .RangeType = ppPrintSlideRange
    With .Ranges
      .Add 1, 1
      .Add 3, 5
      .Add 8, 9
    End With
  End With
  .PrintOut
End With
```



RangeStart Property

Returns or sets the number of the first slide in a range of slides you are publishing as a Web presentation. Read/write **Integer**.

Example

This example publishes slides three through five of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```



[Show All](#)

RangeType Property

 [RangeType property as it applies to the **PrintOptions** object.](#)

Returns or sets the type of print range for the presentation. Read/write [PpPrintRangeType](#).

PpPrintRangeType can be one of these PpPrintRangeType constants.

ppPrintAll

ppPrintCurrent

ppPrintNamedSlideShow

ppPrintSelection

ppPrintSlideRange

expression.**RangeType**

expression Required. An expression that returns one of the above objects.

 [RangeType property as it applies to the **SlideShowSettings** object.](#)

Returns or sets the type of slide show to run. Read/write [PpSlideShowRangeType](#).

PpSlideShowRangeType can be one of these PpSlideShowRangeType constants.

ppShowAll

ppShowNamedSlideShow

ppShowSlideRange

expression.**RangeType**

expression Required. An expression that returns one of the above objects.

Remarks

To print the slides ranges you've defined in the [PrintRanges](#) collection, you must first set the **RangeType** property to **ppPrintSlideRange**. Setting **RangeType** to anything other than **ppPrintSlideRange** means that the ranges you've defined in the **PrintRanges** collection won't be applied. However, this doesn't affect the contents of the **PrintRanges** collection in any way. That is, if you define some print ranges, set the **RangeType** property to a value other than **ppPrintSlideRange**, and then later set **RangeType** back to **ppPrintSlideRange**, the print ranges you defined before will remain unchanged.

Specifying a value for the *To* and *From* arguments of the [PrintOut](#) method sets the value of this property.

Example

[As it applies to the **PrintOptions** object.](#)

This example prints the current slide the active presentation.

```
With ActivePresentation
    .PrintOptions.RangeType = ppPrintCurrent
    .PrintOut
End With
```

[As it applies to the **SlideShowSettings** object.](#)

This example runs the named slide show "Quick Show."

```
With ActivePresentation.SlideShowSettings
    .RangeType = ppShowNamedSlideShow
    .SlideShowName = "Quick Show"
    .Run
End With
```



[Show All](#)

ReadOnly Property

Returns whether the specified presentation is read-only. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified presentation is read-only.

Example

If the active presentation is read-only, this example saves it as Newfile.ppt.

```
With Application.ActivePresentation  
    If .ReadOnly Then .SaveAs FileName:="newfile"  
End With
```



[Show All](#)

Registered Property

Returns whether the specified add-in is registered in the Windows registry.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

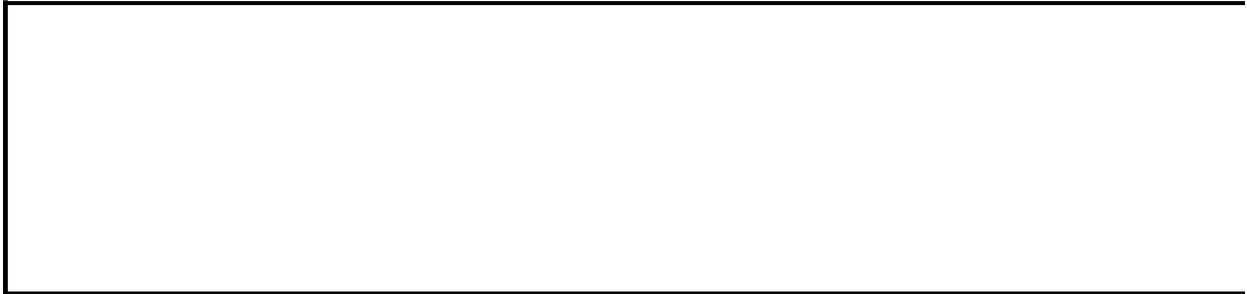
msoTriStateToggle

msoTrue The specified add-in is registered in the Windows registry.

Example

This example registers the add-in named "MyTools" in the Windows registry.

```
Application.Addins("MyTools").Registered = msoTrue
```



[Show All](#)

Relative Property

MsoTrue to set the motion position relative to the position of the shape. This property is only used in conjunction with motion paths. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. The motion path is absolute.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The motion path is relative.

expression.**Relative**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape, adds an animated motion path to the shape, and reports on its motion path relativity.

```
Sub AddShapeSetAnimPath()  
  
    Dim effDiamond As Effect  
    Dim shpCube As Shape  
  
    Set shpCube = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeCube, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpCube, effectId:=msoAnimEffectPathDiamon  
  
    effDiamond.Timing.Duration = 3  
    MsgBox "Is motion path relative or absolute: " & _  
        effDiamond.EffectParameters.Relative & vbCrLf & _  
        "0 = Relative, -1 = Absolute"  
  
End Sub
```



RelativeSize Property

Returns or sets the bullet size relative to the size of the first text character in the paragraph. Can be a floating-point value from 0.25 through 4, indicating that the bullet size can be from 25 percent through 400 percent of the text-character size. Read/write **Single**.

Example

This example sets the formatting for the bullet in shape two on slide one in the active presentation. The size of the bullet is 125 percent of the size of the first text character in the paragraph.

```
With ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat.Bullet
    .Visible = True
    .RelativeSize = 1.25
    .Character = 169
    With .Font
      .Name = "Symbol"
      .Color.RGB = RGB(255, 0, 0)
    End With
  End With
End With
```



[Show All](#)

RelyOnVML Property

Determines whether image files are generated from drawing objects when you save or publish a complete or partial presentation as a Web page. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. Images are not generated from drawing objects when you save or publish a complete or partial presentation as a Web page.

msoTriStateMixed

msoTriStateToggle

msoTrue Image files are generated from drawing objects when you save or publish a complete or partial presentation as a Web page.

Remarks

You can reduce file sizes by not generating images for drawing objects, if your Web browser supports Vector Markup Language (VML). For example, Microsoft Internet Explorer 5 and higher support this feature, and you should set the **RelyOnVML** property to **msoTrue** if you are targeting this browser. For browsers that do not support VML, the image will not appear when you view a Web page saved with this property set to **msoTrue**.

For example, you should not generate images if your Web page uses image files that you have generated earlier, and if the location where you save the presentation is different from the final location of the page on the Web server.

Example

This example specifies that image files are generated when saving or publishing the active presentation to a Web page.

```
ActivePresentation.WebOptions.RelyOnVML = msoFalse
```



[Show All](#)

RemovePersonalInformation Property

MsoTrue for Microsoft PowerPoint to remove all user information from comments, revisions, and the **Properties** dialog box upon saving a presentation. Read/write [MsoTriState](#).

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Comments, revisions, and personal information remain in the presentation.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Removes comments, revisions, and personal information when saving presentation.

expression.**RemovePersonalInformation**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Before you give others a copy of the document, it is a good idea to review personal and hidden information, and decide whether it is appropriate to include. You may want to remove some information from the document and from the document file properties before you share the document with others.

Where is personal or hidden information stored?

File properties These properties include Author, Manager, Company, and Last Saved By.

Other hidden information For example, hidden, revised text, comments, or field code can remain in a document even though you don't see it or expect it to be in the final version. If you entered personal information such as your name or e-mail address when you registered your software, some Microsoft Office documents store that information as part of the document.

Information contained in custom fields that you add to the document, such as an 'author' or 'owner' field, is not automatically removed. You must edit or remove the custom field to remove that information.

Example

This example sets the active presentation to remove personal information the next time the user saves it.

```
Sub RemovePersonalInfo()  
    ActivePresentation.RemovePersonalInformation = msoTrue  
End Sub
```



RepeatCount Property

Sets or returns an **Long** that represents the number of times to repeat an animation. Read/write.

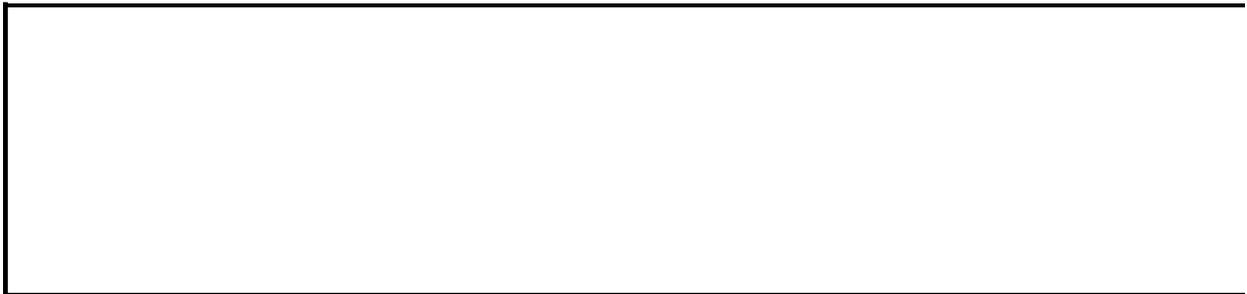
expression.**RepeatCount**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example creates a shape and adds an animation to it, then repeats the animation twice.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    With effDiamond.Timing  
        .Duration = 5 ' Length of effect.  
        .RepeatCount = 2 ' How many times to repeat.  
    End With  
  
End Sub
```



RepeatDuration Property

Sets or returns a **Single** that represents, in seconds, how long repeated animations should last. Read/write.

expression.**RepeatDuration**

expression Required. An expression that returns one of the objects in the Applies To list.

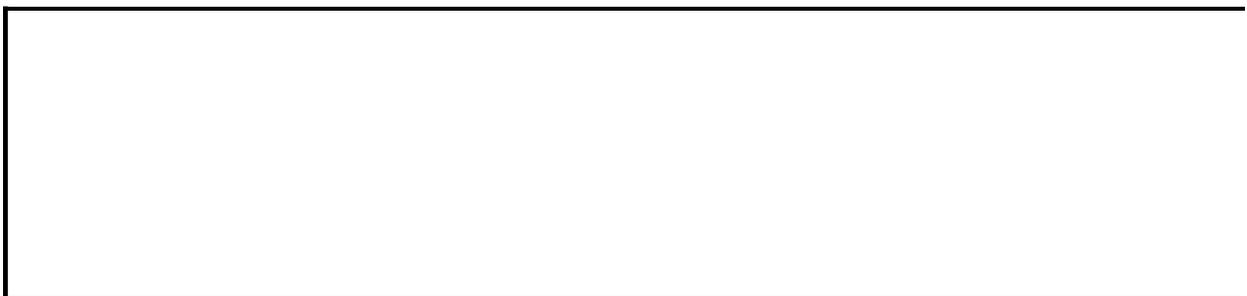
Remarks

An animation will stop at the end of its time sequence or the value of the **RepeatDuration** property, whichever is shorter.

Example

This examples adds a shape and an animation to it, then repeats the animation ten times. However, after five seconds, the animation will be cut off, even though the animation is dimensioned for a 20-second timeline (if the **Duration** property is not specified, an animation defaults to two seconds).

```
Sub AddShapeSetTiming()  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds new shape and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    'Sets repeat duration and number of times to repeat animation  
    With effDiamond.Timing  
        .RepeatDuration = 5  
        .RepeatCount = 10  
    End With  
  
End Sub
```



[Show All](#)

ResizeGraphics Property

Determines whether slides and any graphics on them are sized to fit the Web page display area of the target Web browser. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Slides and graphics remain the size they are in the source presentation, regardless of the Web browser display area.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Slides and any graphics on them are sized to fit the Web page display area of the target Web browser.

Example

This example sets graphics in the specified Web presentation to be resized for the target Web browser. It then publishes the complete presentation, with speaker's notes, to a file named "Mallard.htm."

```
With Presentations(2)
    .WebOptions.ResizeGraphics = msoTrue
    With .PublishObjects(1)
        .FileName = "C:\Mallard.htm"
        .SourceType = ppPublishAll
        .SpeakerNotes = True
        .Publish
    End With
End With
```



[Show All](#)

Restart Property

Sets or returns an [MsoAnimEffectRestart](#) constant that represents whether the animation effect restarts after the effect has started once. Read/write.

MsoAnimEffectRestart can be one of these MsoAnimEffectRestart constants.

msoAnimEffectRestartAlways

msoAnimEffectRestartNever Default.

msoAnimEffectRestartWhenOff

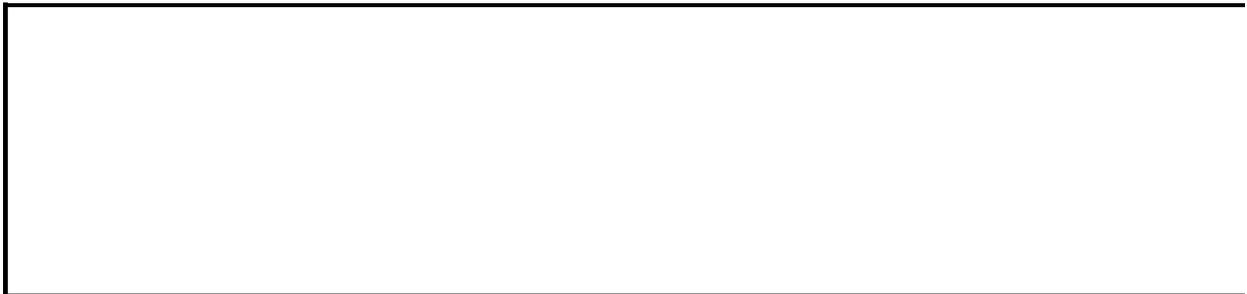
expression.**Restart**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape and an animation to it, then sets the animation's restart behavior.

```
Sub AddShapeSetTiming()  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets animation  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, Top:=100, _  
        Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    With effDiamond.Timing  
        .Duration = 3  
        .RepeatDuration = 5  
        .RepeatCount = 3  
        .Restart = msoAnimEffectRestartAlways  
    End With  
  
End Sub
```



[Show All](#)

Reveal Property

Sets or returns a [MsoTriState](#) constant that determines how the embedded objects will be revealed. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Comments, revisions, and personal information remain in the presentation.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Removes comments, revisions, and personal information when saving the presentation.

expression.**Reveal**

expression Required. An expression that returns one of the objects in the Applies To list.

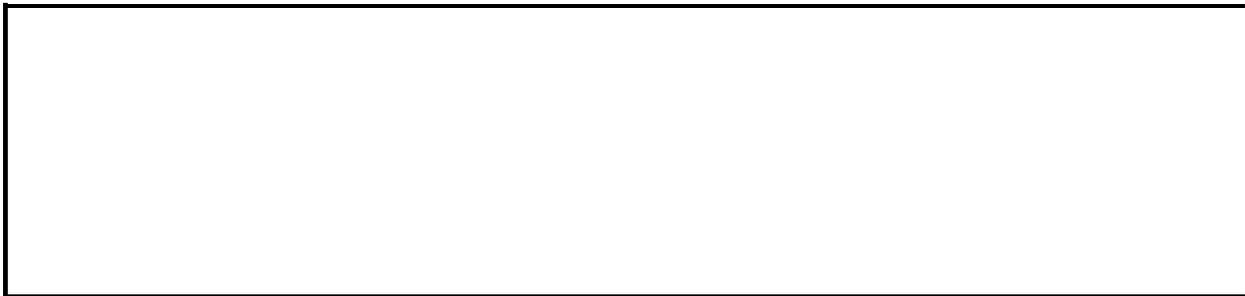
Remarks

Setting a value of **msoTrue** for the **Reveal** property when the filter effect type is **msoAnimFilterEffectTypeWipe** will make the shape appear. Setting a value of **msoFalse** will make the object disappear. In other words, if your filter is set to wipe and Reveal is true, you will get a wipe in effect and when Reveal is false, you will get a wipe out effect.

Example

The following example adds a shape to the first slide of the active presentation and sets a filter effect animation behavior.

```
Sub ChangeFilterEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeFilter)  
  
    With bhvEffect.FilterEffect  
        .Type = msoAnimFilterEffectTypeWipe  
        .Subtype = msoAnimFilterEffectSubtypeUp  
        .Reveal = msoTrue  
    End With  
End Sub
```



[Show All](#)

Reverse Property

Sets or returns an [MsoTriState](#) constant that represents a diagram's reverse state.
Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse The diagram is not reversed.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The diagram is reversed.

expression.**Reverse**

expression Required. An expression that returns one of the objects in the Applies To list.

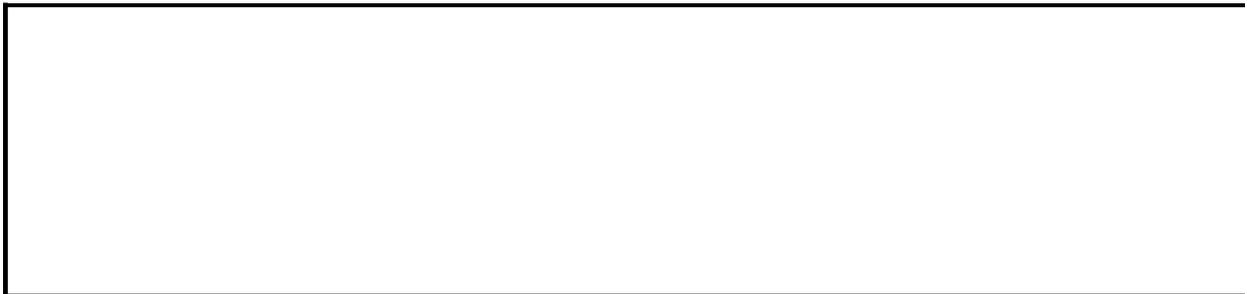
Remarks

This method generates an error if the value of the target diagram's [Type](#) property is an organization chart (**msoDiagramTypeOrgChart**).

Example

The following example creates a pyramid diagram, and reverses its coloring.

```
Sub ReversePyramidDiagram()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
  
    'Adds a pyramid diagram and first child node  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram  
        (Type:=msoDiagramPyramid, Left:=10, Top:=15, _  
        Width:=400, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three additional nodes to diagram  
    For intNodes = 1 To 3  
        dgnNode.AddNode  
    Next intNodes  
  
    'Automatically places nodes, and reverses node order  
    With dgnNode.Diagram  
        .AutoLayout = msoTrue  
        .Reverse = msoTrue  
    End With  
  
End Sub
```



[Show All](#)

RewindAtEnd Property

Sets or returns an [MsoTriState](#) constant that represents whether an object returns to its beginning position after an animation has ended. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. The object does not return to its beginning position after an animation has ended.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The object returns to its beginning position after an animation has ended.

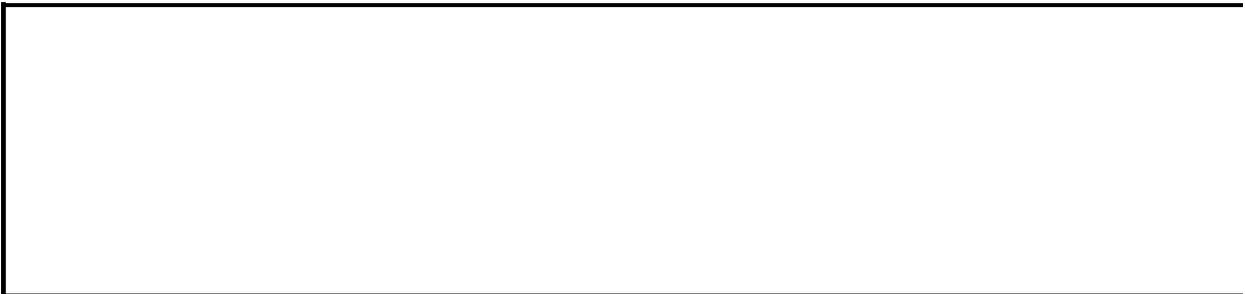
expression.**RewindAtEnd**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape and an animation to the shape, then instructs the shape to return to its beginning position after the animation has ended.

```
Sub AddShapeSetTiming()  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    'Sets duration of animation and returns shape to its original po  
    With effDiamond.Timing  
        .Duration = 3  
        .RewindAtEnd = msoTrue  
    End With  
  
End Sub
```



[Show All](#)

RewindMovie Property

Determines whether the first frame of the specified movie is automatically redisplayed as soon as the movie has finished playing. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The first frame of the specified movie is automatically redisplayed as soon as the movie has finished playing.

Example

This example specifies that the first frame of the movie represented by shape three on slide one in the active presentation will be automatically redisplayed when the movie has finished playing. Shape three must be a movie object.

```
Set OLEobj = ActivePresentation.Slides(1).Shapes(3)  
OLEobj.AnimationSettings.PlaySettings.RewindMovie = msoTrue
```



[Show All](#)

RGB Property

 [RGB property as it applies to the **ColorFormat** object.](#)

Returns or sets the [red-green-blue \(RGB\) value](#) of the specified color. Read/write **Long**.

 [RGB property as it applies to the **RGBColor** object.](#)

Returns or sets the red-green-blue (RGB) value of a specified color-scheme color or extra color when used with a read/write [PpColorSchemeIndex](#) constant. The **Colors** method is used to return the **RGBColor** object.

PpColorSchemeIndex can be one of these PpColorSchemeIndex constants.

ppAccent1

ppAccent2

ppAccent3

ppBackground

ppFill

ppForeground

ppShadow

ppTitle

Example

 [As it applies to the **ColorFormat** object.](#)

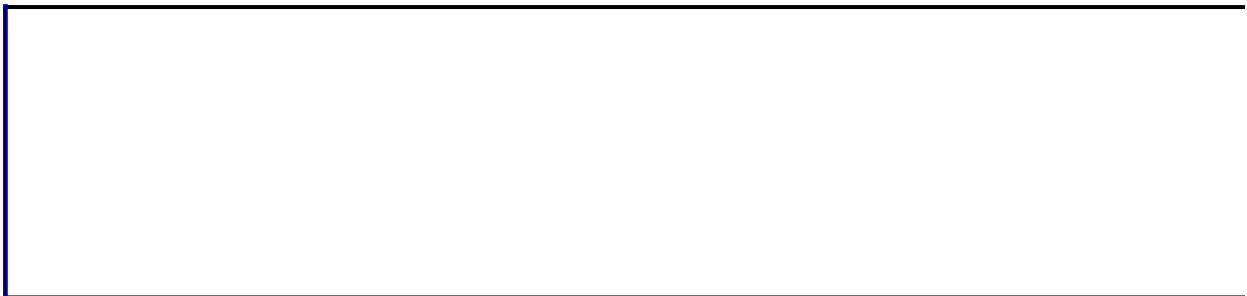
This example sets the background color for color scheme three in the active presentation and then applies the color scheme to all slides in the presentation that are based on the slide master.

```
With ActivePresentation
    Set cs1 = .ColorSchemes(3)
    cs1.Colors(ppBackground).RGB = RGB(128, 128, 0)
    .SlideMaster.ColorScheme = cs1
End With
```

 [As it applies to the **RGBColor** object.](#)

This example displays the value of the red, green, and blue components of the fill forecolor for shape one on slide one in the active document.

```
Set myDocument = ActivePresentation.Slides(1)
c = myDocument.Shapes(1).Fill.ForeColor.RGB
redComponent = c Mod 256
greenComponent = c \ 256 Mod 256
blueComponent = c \ 65536 Mod 256
MsgBox "RGB components: " & redComponent & _
    ", " & greenComponent & ", " & blueComponent
```



Root Property

Returns a [DiagramNode](#) object that represents the root diagram node to which the source diagram node belongs.

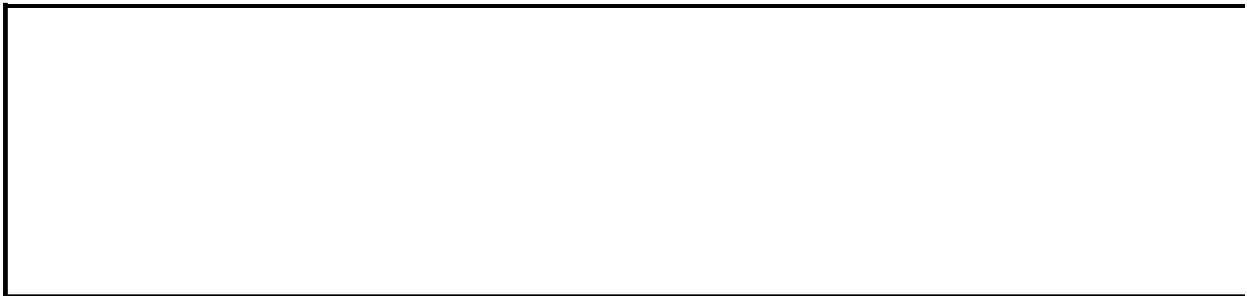
expression.**Root**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example creates an organization chart and adds child nodes to the root diagram node.

```
Sub AddChildNodesToRoot()  
  
    Dim dgnNode As DiagramNode  
    Dim shpOrgChart As Shape  
    Dim intNodes As Integer  
  
    'Adds organization chart and first node  
    Set shpOrgChart = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramOrgChart, Left:=10, _  
            Top:=15, Width:=400, Height:=475)  
    shpOrgChart.DiagramNode.Children.AddNode  
  
    Set dgnNode = shpOrgChart.DiagramNode.Root  
  
    'Adds three child nodes to root node  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
End Sub
```



[Show All](#)

RotatedChars Property

Determines whether characters in the specified WordArt are rotated 90 degrees relative to the WordArt's bounding shape. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Characters in the specified WordArt retain their original orientation relative to the bounding shape.

msoTriStateMixed

msoTriStateToggle

msoTrue Characters in the specified WordArt are rotated 90 degrees relative to the WordArt's bounding shape.

Remarks

If the WordArt has horizontal text, setting the **RotatedChars** property to **msoTrue** rotates the characters 90 degrees counterclockwise. If the WordArt has vertical text, setting the **RotatedChars** property to **msoFalse** rotates the characters 90 degrees clockwise. Use the **ToggleVerticalText** method to switch between horizontal and vertical text flow.

The [Flip](#) method and [Rotation](#) property of the [Shape](#) object and the **RotatedChars** property and [ToggleVerticalText](#) method of the **TextEffectFormat** object all affect the character orientation and direction of text flow in a **Shape** object that represents WordArt. You may have to experiment to find out how to combine the effects of these properties and methods to get the result you want.

Example

This example adds WordArt that contains the text "Test" to myDocument and rotates the characters 90 degrees counterclockwise.

```
Set myDocument = ActivePresentation.Slides(1)
Set newWordArt = myDocument.Shapes.AddTextEffect _
    (PresetTextEffect:=msoTextEffect1, Text:="Test", _
    FontName:="Arial Black", FontSize:=36, _
    FontBold:=msoFalse, FontItalic:=msoFalse, Left:=10, Top:=10)
newWordArt.TextEffect.RotatedChars = msoTrue
```



[Show All](#)

Rotation Property

Returns or sets the number of degrees the specified shape is rotated around the [z-axis](#). A positive value indicates clockwise rotation; a negative value indicates counterclockwise rotation. Read/write **Single**

Remarks

To set the rotation of a three-dimensional shape around the x-axis or the y-axis, use the [RotationX](#) property or the [RotationY](#) property of the [ThreeDFormat](#) object.

Example

This example matches the rotation of all shapes on myDocument to the rotation of shape one.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    sh1Rotation = .Item(1).Rotation
    For o = 1 To .Count
        .Item(o).Rotation = sh1Rotation
    Next
End With
```



RotationEffect Property

Returns a [RotationEffect](#) object for an animation behavior.

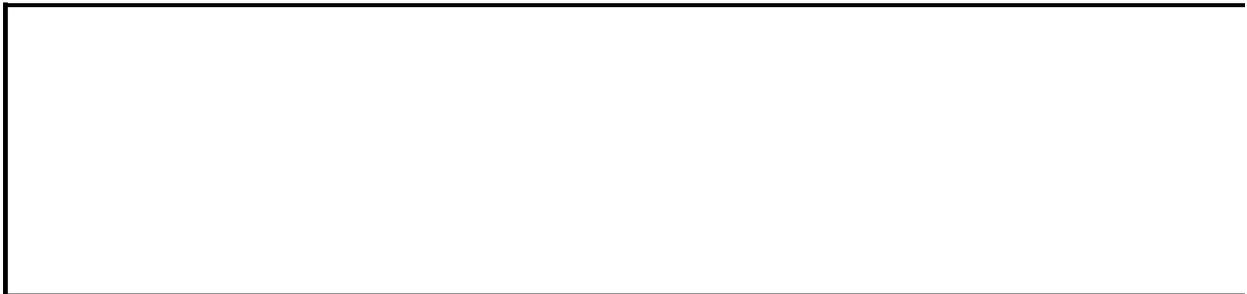
expression.**RotationEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a new shape to the first slide and sets the rotation animation behavior.

```
Sub AddRotation()  
  
    Dim shpNew As Shape  
    Dim effNew As Effect  
    Dim aniNew As AnimationBehavior  
  
    Set shpNew = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShape5pointStar, Left:=0, _  
        Top:=0, Width:=100, Height:=100)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpNew, effectId:=msoAnimEffectCustom)  
    Set aniNew = effNew.Behaviors.Add(msoAnimTypeRotation)  
  
    With aniNew.RotationEffect  
        'Rotate 270 degrees from current position  
        .By = 270  
    End With  
  
End Sub
```



[Show All](#)

RotationX Property

Returns or sets the rotation of the extruded shape around the [x-axis](#), in degrees. Can be a value from – 90 through 90. A positive value indicates upward rotation; a negative value indicates downward rotation. Read/write **Single**.

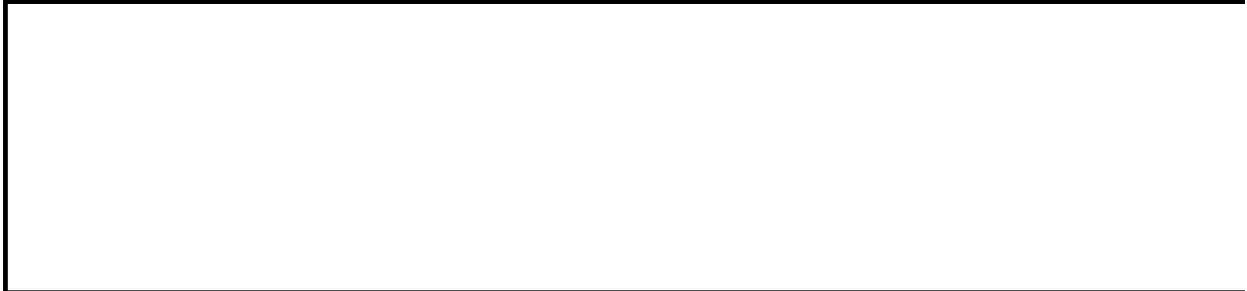
Remarks

To set the rotation of the extruded shape around the y-axis, use the [RotationY](#) property of the **ThreeDFormat** object. To set the rotation of the extruded shape around the z-axis, use the [Rotation](#) property of the [Shape](#) object. To change the direction of the extrusion's sweep path without rotating the front face of the extrusion, use the [SetExtrusionDirection](#) method.

Example

This example adds three identical extruded ovals to myDocument and sets their rotation around the x-axis to – 30, 0, and 30 degrees, respectively.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
  With .AddShape(msoShapeOval, 30, 60, 50, 25).ThreeD
    .Visible = True
    .RotationX = -30
  End With
  With .AddShape(msoShapeOval, 90, 60, 50, 25).ThreeD
    .Visible = True
    .RotationX = 0
  End With
  With .AddShape(msoShapeOval, 150, 60, 50, 25).ThreeD
    .Visible = True
    .RotationX = 30
  End With
End With
```



[Show All](#)

RotationY Property

Returns or sets the rotation of the extruded shape around the [y-axis](#), in degrees. Can be a value from – 90 through 90. A positive value indicates rotation to the left; a negative value indicates rotation to the right. Read/write **Single**.

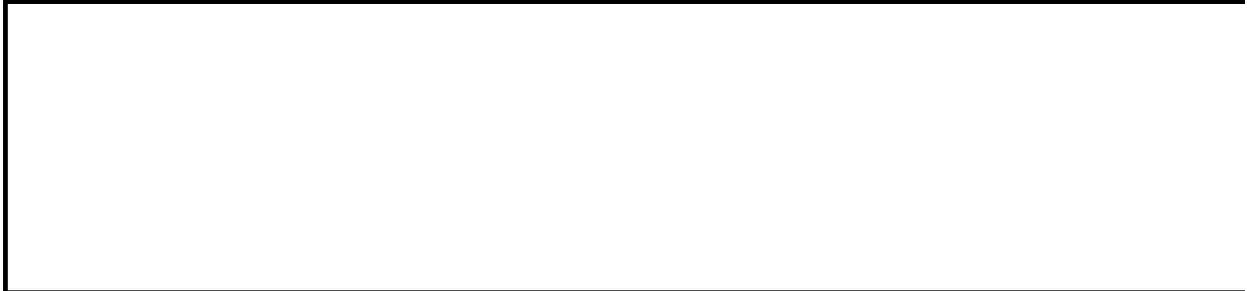
Remarks

To set the rotation of the extruded shape around the x-axis, use the [RotationX](#) property of the **ThreeDFormat** object. To set the rotation of the extruded shape around the z-axis, use the [Rotation](#) property of the [Shape](#) object. To change the direction of the extrusion's sweep path without rotating the front face of the extrusion, use the [SetExtrusionDirection](#) method.

Example

This example adds three identical extruded ovals to myDocument and sets their rotation around the y-axis to – 30, 0, and 30 degrees, respectively.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
  With .AddShape(msoShapeOval, 30, 30, 50, 25).ThreeD
    .Visible = True
    .RotationY = -30
  End With
  With .AddShape(msoShapeOval, 30, 70, 50, 25).ThreeD
    .Visible = True
    .RotationY = 0
  End With
  With .AddShape(msoShapeOval, 30, 110, 50, 25).ThreeD
    .Visible = True
    .RotationY = 30
  End With
End With
```



Rows Property

Returns a [Rows](#) collection that represents all the rows in a table. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example deletes the third row from the table in shape five of slide two in the active presentation.

```
ActivePresentation.Slides(2).Shapes(5).Table.Rows(3).Delete
```

This example applies a dashed line style to the bottom border of the second row of table cells.

```
ActiveWindow.Selection.ShapeRange.Table.Rows(2) _  
    .Cells.Borders(ppBorderBottom).DashStyle = msoLineDash
```



Ruler Property

Returns a [Ruler](#) object that represents the ruler for the specified text. Read-only.

Example

This example sets a left-aligned tab stop at 2 inches (144 points) for the text in shape two on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes(2).TextFrame.Ruler.TabStops _
    .Add ppTabStopLeft, 144
```



Run Property

Returns or sets the name of the presentation or macro to be run when the specified shape is clicked or the mouse pointer passes over the shape during a slide show. The [Action](#) property must be set to **ppActionRunMacro** or **ppActionRunProgram** for this property to affect the slide show action. Read/write **String**.

Remarks

If the value of the **Action** property is **ppActionRunMacro**, the specified string value should be the name of a global macro that's currently loaded. If the value of the **Action** property is **ppActionRunProgram**, the specified string value should be the full path and file name of a program.

You can set the **Run** property to a macro that takes no arguments or a macro that takes a single **Shape** or **Object** argument. The shape that was clicked during the slide show will be passed as this argument.

Example

This example specifies that the CalculateTotal macro be run whenever the mouse pointer passes over the shape during a slide show.

```
With ActivePresentation.Slides(1) _  
    .Shapes(3).ActionSettings(ppMouseOver)  
        .Action = ppActionRunMacro  
        .Run = "CalculateTotal"  
        .AnimateAction = True  
End With
```



[Show All](#)

Saved Property

Determines whether changes have been made to a presentation since it was last saved. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue No changes have been made to a presentation since it was last saved.

Remarks

If the **Saved** property of a modified presentation is set to **msoTrue**, the user won't be prompted to save changes when closing the presentation, and all changes made to it since it was last saved will be lost.

Example

This example saves the active presentation if it's been changed since the last time it was saved.

```
With Application.ActivePresentation  
    If Not .Saved And .Path <> "" Then .Save  
End With
```



[Show All](#)

SaveNewWebPagesAsWebArchives Property

MsoTrue for Microsoft PowerPoint to save new Web pages as Web archives.
Read/write [MsoTriState](#).

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Saves new Web pages as individual Web pages.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Saves new Web pages as Web archives.

expression.**SaveNewWebPagesAsWebArchives**

expression Required. An expression that returns one of the objects in the Applies To list.

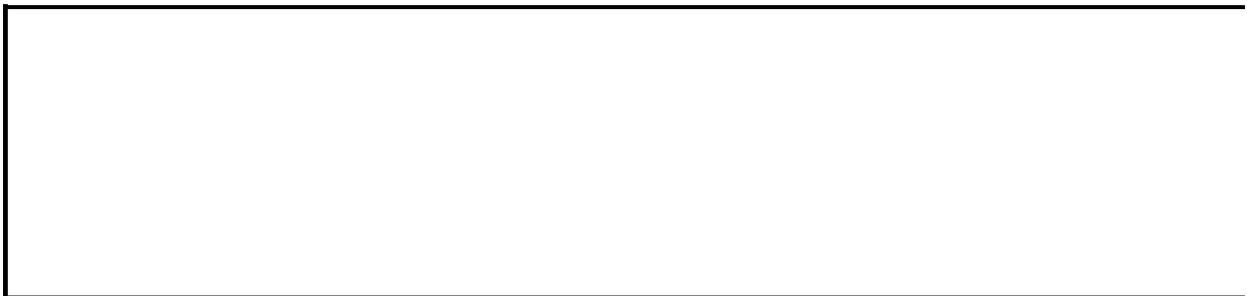
Remarks

Setting the **SaveNewWebPagesAsWebArchives** property won't change the format of any currently open Web pages. You must individually save open Web pages and explicitly set the Web page format using the [SaveAs](#) method.

Example

This example enables the **SaveNewWebPagesAsWebArchives** property, so that when new Web pages are saved, they are saved as Web archives.

```
Sub SetWebOption()  
    Application.DefaultWebOptions _  
        .SaveNewWebPagesAsWebArchives = True  
End Sub
```



ScaleEffect Property

Returns a [ScaleEffect](#) object for a given animation behavior.

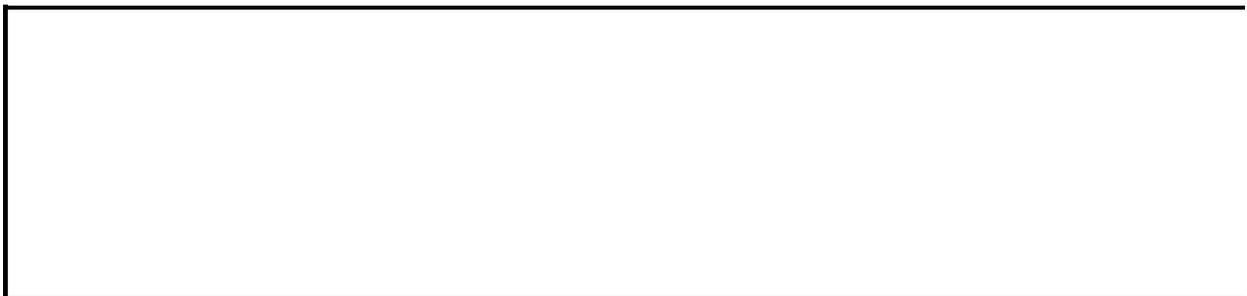
expression.**ScaleEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example scales the first shape on the first slide starting at zero and increasing in size until it reaches 100 percent of its original size.

```
Sub ChangeScale()  
  
    Dim shpFirst As Shape  
    Dim effNew As Effect  
    Dim aniScale As AnimationBehavior  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpFirst, effectId:=msoAnimEffectCustom)  
    Set aniScale = effNew.Behaviors.Add(msoAnimTypeScale)  
  
    With aniScale.ScaleEffect  
  
        'Starting size  
        .FromX = 0  
        .FromY = 0  
  
        'Size after scale effect  
        .ToX = 100  
        .ToY = 100  
    End With  
End Sub
```



[Show All](#)

SchemeColor Property

Returns or sets the color in the applied color scheme that's associated with the specified object. Read/write [PpColorSchemeIndex](#).

PpColorSchemeIndex can be one of these PpColorSchemeIndex constants.

ppAccent1

ppAccent2

ppAccent3

ppBackground

ppFill

ppForeground

ppNotSchemeColor

ppSchemeColorMixed

ppShadow

ppTitle

expression.**SchemeColor**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example switches the background color on slide one in the active presentation between an explicit red-green-blue value and the color-scheme background color.

```
With ActivePresentation.Slides(1)
    .FollowMasterBackground = False
    With .Background.Fill.ForeColor
        If .Type = msoColorTypeScheme Then
            .RGB = RGB(0, 128, 128)
        Else
            .SchemeColor = ppBackground
        End If
    End With
End With
```



[Show All](#)

ScreenSize Property

Returns or sets the ideal minimum screen size (width by height, in pixels) that you should use when viewing the saved presentation in a Web browser.

Read/write [MsoScreenSize](#).

MsoScreenSize can be one of these MsoScreenSize constants.

msoScreenSize1024x768

msoScreenSize1152x882

msoScreenSize1152x900

msoScreenSize1280x1024

msoScreenSize1600x1200

msoScreenSize1800x1440

msoScreenSize1920x1200

msoScreenSize544x376

msoScreenSize640x480

msoScreenSize720x512

msoScreenSize800x600 Default.

expression.**ScreenSize**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the target screen size to 640x480 pixels.

```
Presentations(2).WebOptions.ScreenSize = _  
    msoScreenSize640x480
```



ScreenTip Property

Returns or sets the ScreenTip text of a hyperlink. Read/write **String**.

Remarks

ScreenTip text appears, for example, when you save a presentation to HTML, view it in a Web browser, and rest the mouse pointer over a hyperlink. Some browsers may not support ScreenTips.

Example

This example sets the ScreenTip text for the first hyperlink.

```
ActivePresentation.Slides(1).Hyperlinks(1) _  
    .ScreenTip = "Go to the Microsoft home page"
```



Script Property

Returns a **Script** object that represents a block of script code on a Microsoft PowerPoint slide. In PowerPoint, script is associated with an anchor shape. If no script is associated with the specified shape, then nothing is returned. Read-only.

Remarks

Script code inserted on a slide can only be run in Web presentations.

By default, script anchor shapes are not visible. To make them visible, use the PowerPoint user interface. You cannot make script anchor shapes visible with Visual Basic code.

It is possible to use the **Script** property on a range of shapes (`ShapeRange.Script`) instead of specifying a single anchor shape. However, if the range contains more than one shape, your code will not work and will return a message that indicates the **Script** property cannot be accessed.

Example

This example sets the scripting language for the script anchor (shape eight on slide one) to Microsoft Visual Basic Scripting Edition (VBScript).

```
With ActivePresentation.Slides(1).Shapes(8)  
    .Script.Language = msoScriptLanguageVisualBasic  
End With
```



Scripts Property

Returns a **Scripts** collection that represents all **Script** objects (blocks of script code) in a presentation. Read-only.

Remarks

Script code, represented by a **Script** object, can be run only in a Web presentation.

[Show All](#)

SegmentType Property

Returns a value that indicates whether the segment associated with the specified node is straight or curved. Read-only [MsoSegmentType](#).

MsoSegmentType can be one of these MsoSegmentType constants.

msoSegmentCurve The SegmentType property returns this value if the specified node is a control point for a curved segment.

msoSegmentLine

expression.**SegmentType**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

This property is read-only. Use the [SetSegmentType](#) method to set the value of this property.

Example

This example changes all straight segments to curved segments in shape three on myDocument. Shape three must be a freeform drawing.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Nodes
    n = 1
    While n <= .Count
        If .Item(n).SegmentType = msoSegmentLine Then
            .SetSegmentType n, msoSegmentCurve
        End If
        n = n + 1
    Wend
End With
```



Selected Property

True if the specified table cell is selected. Read-only **Boolean**.

expression.**Selected**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example puts a border around the first cell in the specified table if the cell is selected.

```
Sub IsCellSelected()  
    Dim celSelected As Cell  
    Set celSelected = ActivePresentation.Slides(1).Shapes(1) _  
        .Table.Columns(1).Cells(1)  
    If celSelected.Selected Then  
        With celSelected  
            .Borders(ppBorderTop).DashStyle = msoLineRoundDot  
            .Borders(ppBorderBottom).DashStyle = msoLineRoundDot  
            .Borders(ppBorderLeft).DashStyle = msoLineRoundDot  
            .Borders(ppBorderRight).DashStyle = msoLineRoundDot  
        End With  
    End If  
End Sub
```



Selection Property

Returns a [Selection](#) object that represents the selection in the specified document window. Read-only.

Example

If there's text selected in the active window, this example makes the text italic.

```
With Application.ActiveWindow.Selection  
    If .Type = ppSelectionText Then  
        .TextRange.Font.Italic = True  
    End If  
End With
```



SetEffect Property

Returns a **SetEffect** object for the animation behavior. Read-only. You can use the **SetEffect** object to set the value of a property.

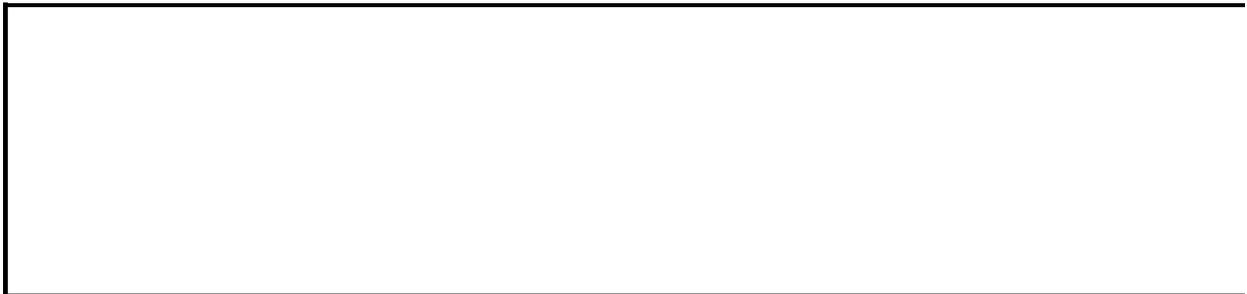
expression.**SetEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to the first slide of the active presentation and sets a set effect animation behavior.

```
Sub ChangeSetEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeSet)  
  
    With bhvEffect.SetEffect  
        .Property = msoAnimShapeFillColor  
        .To = RGB(Red:=0, Green:=255, Blue:=255)  
    End With  
End Sub
```



[Show All](#)

Shadow Property

 [Shadow property as it applies to the **Font** object.](#)

Determines whether the specified text has a shadow. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The specified text doesn't have a shadow.

msoTriStateMixed Some of the text has a shadow and some doesn't.

msoTriStateToggle

msoTrue The specified text has a shadow.

expression.**Shadow**

expression Required. An expression that returns one of the above objects.

 [Shadow property as it applies to the **Shape** and **ShapeRange** objects.](#)

Returns a read/only [ShadowFormat](#) object that contains shadow formatting properties for the specified shape or shapes.

expression.**Shadow**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **Font** object.](#)

This example adds a shadow to the title text on slide one in the active presentation.

```
Application.ActivePresentation.Slides(1).Shapes.Title _  
    .TextFrame.TextRange.Font.Shadow = True
```

 [As it applies to the **Shape** and **ShapeRange** objects.](#)

This example adds a shadowed rectangle to slide one in the active presentation. The blue, embossed shadow is offset 3 points to the right of and 2 points down from the rectangle.

```
Set myShap = Application.ActivePresentation.Slides(1).Shapes  
With myShap.AddShape(msoShapeRectangle, 10, 10, 150, 90).Shadow  
    .Type = msoShadow17  
    .ForeColor.RGB = RGB(0, 0, 128)  
    .OffsetX = 3  
    .OffsetY = 2  
End With
```



Shape Property

Returns a [Shape](#) object that represents a shape in a table cell (for the [Cell](#) object), a diagram node in a diagram (for the [DiagramNode](#) object), or an animated shape (for the [Effect](#) object).

expression.**Shape**

expression Required. An expression that returns one of the above objects.

Example

This example creates a 3x3 table in a new presentation and inserts a four-pointed star into the first cell of the table.

```
With Presentations.Add
    With .Slides.Add(1, ppLayoutBlank)
        .Shapes.AddTable(3, 3).Select
        .Shapes(1).Table.Cell(1, 1).Shape _
            .AutoShapeType = msoShape4pointStar
    End With
End With
```

The following example creates a diagram and adds child nodes to the root node. As each child is added, the root node displays the number of child nodes it has.

```
Sub CountChildNodes()

    Dim dgnNode As DiagramNode
    Dim shpDiagram As Shape
    Dim intNodes As Integer

    Set shpDiagram = ActivePresentation.Slides(1).Shapes.AddDiagram
        (Type:=msoDiagramRadial, Left:=10, Top:=15, _
        Width:=400, Height:=475)
    shpDiagram.DiagramNode.Children.AddNode

    Set dgnNode = shpDiagram.DiagramNode.Root

    For intNodes = 1 To 3
        dgnNode.Children.AddNode
        dgnNode.Shape.TextFrame.TextRange.Text = intNodes
    Next intNodes

End Sub
```



ShapeRange Property

Returns a [ShapeRange](#) object that represents all the slide objects that have been selected on the specified slide. This range can contain the drawings, shapes, OLE objects, pictures, text objects, titles, headers, footers, slide number placeholder, and date and time objects on a slide. Read-only.

Remarks

You can return a shape range from a selection when the presentation is in normal, slide, or any master view.

Example

This example sets the fill foreground color for all the selected shapes in window one.

```
Windows(1).Selection.ShapeRange.Fill _  
    .ForeColor.RGB = RGB(255, 0, 255)
```



Shapes Property

Returns a [Shapes](#) collection that represents all the elements that have been placed or inserted on the specified slide, slide master, or range of slides. This collection can contain the drawings, shapes, OLE objects, pictures, text objects, titles, headers, footers, slide numbers, and date and time objects on a slide, or on the slide image on a notes page. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example adds a rectangle that's 100 points wide and 50 points high, and whose upper-left corner is 5 points from the left edge of slide one in the active presentation and 25 points from the top of the slide.

```
Set firstSlide = ActivePresentation.Slides(1)
firstSlide.Shapes.AddShape msoShapeRectangle, 5, 25, 100, 50
```

This example sets the fill texture for shape three on slide one in the active presentation.

```
Set newRect = ActivePresentation.Slides(1).Shapes(3)
newRect.Fill.PresetTextured msoTextureOak
```

Assuming that slide one in the active presentation contains a title, both the second and third lines of code in the following example set the title text on slide one in the presentation.

```
Set firstSl = ActivePresentation.Slides(1)
firstSl.Shapes.Title.TextFrame.TextRange.Text = "Some title text"
firstSl.Shapes(1).TextFrame.TextRange.Text = "Other title text"
```

Assuming that shape two on slide two in the active presentation contains a text frame, the following example adds a series of paragraphs to the slide. Note that `Chr(13)` is used to insert paragraph marks within the text.

```
Set tShape = ActivePresentation.Slides(2).Shapes(2)
tShape.TextFrame.TextRange.Text = "First Item" & Chr(13) & _
    "Second Item" & Chr(13) & "Third Item"
```

For most slide layouts, the first shapes on the slide are text placeholders, and the following example accomplishes the same task as the preceding example.

```
Set testShape = ActivePresentation.Slides(2).Shapes.Placeholders(2)
testShape.TextFrame.TextRange.Text = "First Item" & _
    Chr(13) & "Second Item" & Chr(13) & "Third Item"
```



SharedWorkspace Property

Returns a [SharedWorkspace](#) object that represents the Document Workspace in which a specified presentation is located. Read-only.

expression.**SharedWorkspace**

expression Required. An expression that returns a **Presentation** object.

Remarks

Use the **SharedWorkspace** object to add the active Microsoft PowerPoint presentation to a Microsoft Windows SharePoint Services Document Workspace site on the server in order to take advantage of the workspace's collaboration features, or to disconnect or remove the document from the workspace. Use the **SharedWorkspace** object's collections to manage files, folders, links, members and tasks associated with the shared document.

The **SharedWorkspace** object model is available whether or not a document is stored in a workspace. The **SharedWorkspace** property of the **Presentation** object does not return **Nothing** when the document is not shared. Use the **Connected** property of the **SharedWorkspace** object to determine whether the active presentation is in fact saved in and connected to a shared workspace.

Users require appropriate permissions to use the objects, properties, and methods in the **SharedWorkspace** object hierarchy.

Use the **SharedWorkspaceFiles** collection, accessed through the **Files** property of the **SharedWorkspace** object, to manage presentations and files saved in a shared workspace.

Use the **SharedWorkspaceFolders** collection, accessed through the **Folders** property of the **SharedWorkspace** object, to manage subfolders within the main document library folder of a shared workspace.

Use the **SharedWorkspaceLinks** collection, accessed through the **Links** property of the **SharedWorkspace** object, to manage links to additional documents and information of interest to the members who are collaborating on the documents in the shared workspace.

Use the **SharedWorkspaceMembers** collection, accessed through the **Members** property of the **SharedWorkspace** object, to manage users who have rights to participate in a shared workspace and to collaborate on the shared documents saved in the workspace.

Use the **SharedWorkspaceTasks** collection, accessed through the **Tasks** property of the **SharedWorkspace** object, to manage tasks assigned to the

members who are collaborating on the documents in the shared workspace.

Use the **CreateNew** method to create a new Document Workspace and to add the active document to the workspace. Use the **Name** and **URL** properties to return information about the workspace.

The **SharedWorkspace** object uses a local cache of objects and properties from the server. The developer may need to update this cache before performing certain operations or to save cached property changes back to the server. Use the **Refresh** method of the **SharedWorkspace** object to refresh the local cache from the server, and the **LastRefreshed** property to determine when the refresh operation last took place. Use the **Save** method of the **SharedWorkspaceLink** and **SharedWorkspaceTask** objects after modifying their properties locally, in order to upload the changes to the server.

Use the **Disconnect** method to disconnect the local copy of the active document from the shared workspace, while leaving the shared copy intact in the workspace. Use the **RemoveDocument** method to remove the shared document from the shared workspace entirely.

Users require appropriate permissions to use the objects, properties, and methods in the **SharedWorkspace** object hierarchy. Use the **Role** argument when adding members to the **SharedWorkspaceMembers** collection to specify the set of permissions specific to each workspace member.

When using the **SharedWorkspace** object model, it is possible to create conditions where the **SharedWorkspace** object cache is not synchronized with the user interface displayed in the **Shared Workspace** pane of the active document. For example, if the **CreateNew** method programmatically adds the active document to a new workspace while the **Shared Workspace** pane is open, the **Shared Workspace** pane will continue to display the **Create New** button. In circumstances like these, if the user makes a selection in the **Shared Workspace** pane that is no longer valid, an error is raised and a refresh operation is carried out to synchronize the display with the current document state and shared workspace data.

The **Presentation** object also has a **Sync** property which returns a [Sync](#) object. Use the **Sync** object and its properties and methods to manage the synchronization of the local and the server copies of the shared document.

Example

The following example returns a reference to the Document Workspace in which the active presentation is stored. This example assumes that the active document belongs to a Document Workspace.

```
Dim objWorkspace As SharedWorkspace  
Set objWorkspace = ActivePresentation.SharedWorkspace
```



[Show All](#)

ShowScrollbar Property

MsoTrue to display the scroll bar during a slide show in browse mode.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue Not used with this property.

msoFalse

msoTriStateMixed Not used with this property.

msoTriStateToggle Not used with this property.

msoTrue

expression.**ShowScrollbar**

expression Required. An expression that returns one of the objects in the Applies To list.

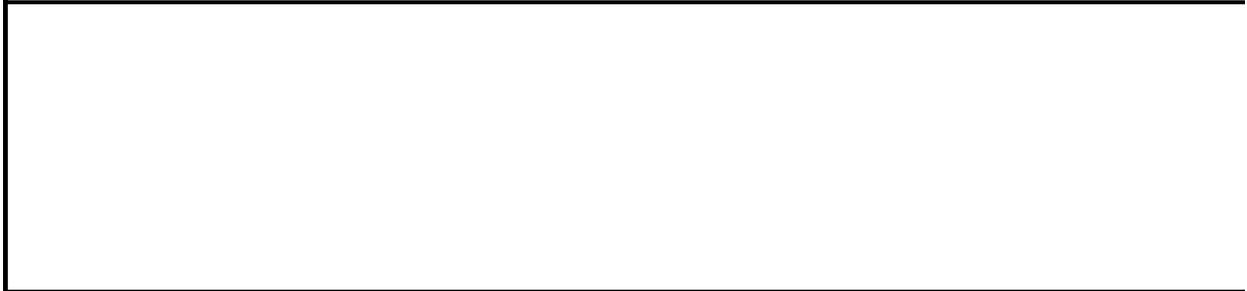
Remarks

Use the [ShowType](#) property prior to setting the **ShowScrollbar** property.

Example

This example specifies to display the slide show for the active presentation in a window and displays a scrollbar used for browsing through the slides during a slide show.

```
Sub ShowSlideShowScrollBar()  
    With ActivePresentation.SlideShowSettings  
        .ShowType = ppShowTypeWindow  
        .ShowScrollBar = msoTrue  
    End With  
End Sub
```



[Show All](#)

ShowSlideAnimation Property

Determines whether slide animations are enabled when previewing, saving, or publishing a Web presentation. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

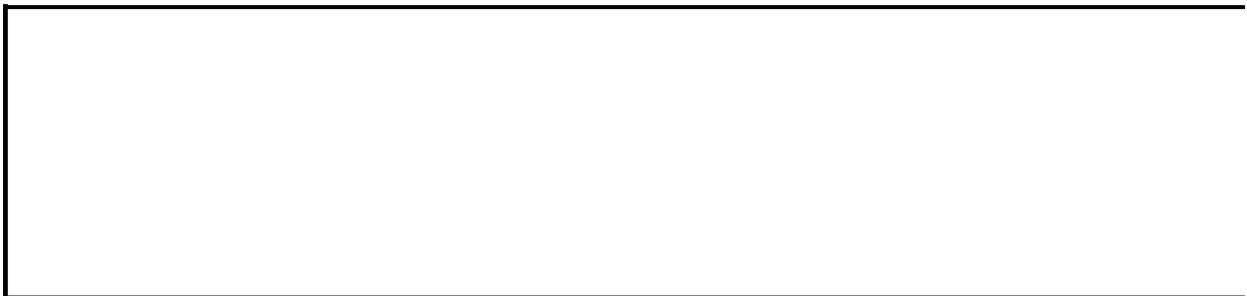
msoTriStateToggle

msoTrue Slide animations are enabled when previewing, saving, or publishing a Web presentation.

Example

This example specifies that slide animations in presentation two are enabled for the Web presentation. It then previews the Web page.

```
With Presentations(2)  
    .WebOptions.ShowSlideAnimation = msoTrue  
    .WebPagePreview  
End With
```



[Show All](#)

ShowStartupDialog Property

MsoTrue to display the **New Presentation** task pane when Microsoft PowerPoint is started. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Hides the **New Presentation** side pane.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue Default. Displays the **New Presentation** side pane.

expression.**ShowStartupDialog**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following line of code disables the **New Presentation** task pane when PowerPoint starts.

```
Sub DontShowStartup  
    Application.ShowStartupDialog = msoFalse  
End Sub
```



[Show All](#)

ShowType Property

Returns or sets the show type for the specified slide show. Read/write [PpSlideShowType](#).

PpSlideShowType can be one of these PpSlideShowType constants.

ppShowTypeKiosk

ppShowTypeSpeaker

ppShowTypeWindow

expression.**ShowType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example runs a slide show of the active presentation in a window, starting with slide two and ending with slide four. The new slide show window is placed in the upper-left corner of the screen, and its width and height are both 300 points.

```
With ActivePresentation.SlideShowSettings
    .RangeType = ppShowSlideRange
    .StartingSlide = 2
    .EndingSlide = 4
    .ShowType = ppShowTypeWindow
With .Run
    .Left = 0
    .Top = 0
    .Width = 300
    .Height = 300
End With
End With
```



[Show All](#)

ShowWindowsInTaskbar Property

Determines whether there is a separate Windows taskbar button for each open presentation. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Default. There is a separate Windows taskbar button for each open presentation.

Remarks

When set to **True**, this property simulates the look of a single-document interface (SDI), which makes it easier to navigate between open presentations. However, if you work with multiple presentations while other applications are open, you may want to set this property to **False** to avoid filling your taskbar with unnecessary buttons.

This property is available only when using Microsoft Office with Windows Update or Windows 2000.

Example

This example specifies that each open presentation doesn't have a separate Windows taskbar button.

```
Application.ShowWindowsInTaskbar = msoFalse
```



[Show All](#)

ShowWithAnimation Property

Determines whether the specified slide show displays shapes with assigned animation settings. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide show displays shapes with assigned animation settings.

Example

This example runs a slide show of the active presentation with animation and narration turned off.

```
With ActivePresentation.SlideShowSettings
    .ShowWithAnimation = msoFalse
    .ShowWithNarration = msoFalse
    .Run
End With
```



[Show All](#)

ShowWithNarration Property

Determines whether the specified slide show is shown with narration. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified slide show is shown with narration.

Example

This example runs a slide show of the active presentation with animation and narration turned off.

```
With ActivePresentation.SlideShowSettings
    .ShowWithAnimation = msoFalse
    .ShowWithNarration = msoFalse
    .Run
End With
```



Signatures Property

Returns a [SignatureSet](#) object that represents a collection of digital signatures.

expression.**Signatures**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following line of code displays the number of digital signatures.

```
Sub DisplayNumberOfSignatures
    MsgBox "Number of digital signatures: " & _
        ActivePresentation.Signatures.Count
End Sub
```



Size Property

Returns or sets the character size, in points. Read/write **Single**.

Example

This example sets the size of the text attached to shape one on slide one to 24 points.

```
Application.ActivePresentation.Slides(1) _  
    .Shapes(1).TextFrame.TextRange.Font _  
    .Size = 24
```



[Show All](#)

Slide Property

[Slide property as it applies to the **View** object.](#)

Returns or sets a **Slide** object that represents the slide that's currently displayed in the specified document window view. Read/write.

expression.**Slide**

expression Required. An expression that returns one of the above objects.

[Slide property as it applies to the **SlideShowView** object.](#)

Returns a **Slide** object that represents the slide that's currently displayed in the specified slide show window view. Read-only.

expression.**Slide**

expression Required. An expression that returns one of the above objects.

Remarks

If the currently displayed slide is from an embedded presentation, you can use the [Parent](#) property of the **Slide** object returned by the **Slide** property to return the embedded presentation that contains the slide. (The [Presentation](#) property of the **SlideShowWindow** object or **DocumentWindow** object returns the presentation from which the window was created, not the embedded presentation.)

Example

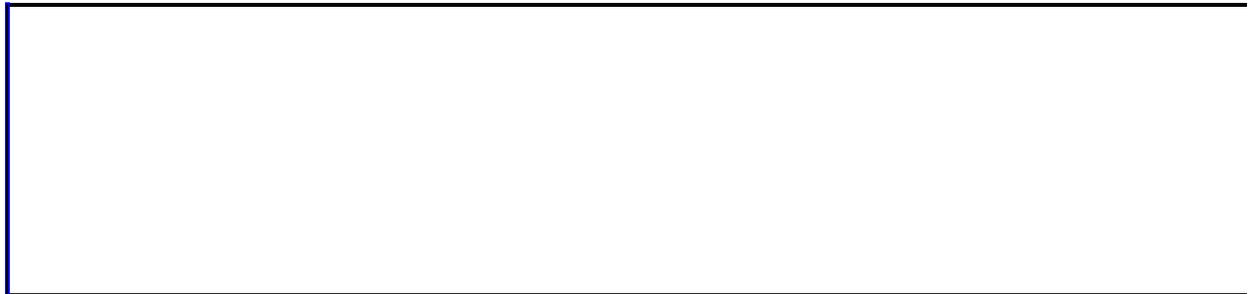
[As it applies to the **View** object.](#)

This example places on the Clipboard a copy of the slide that's currently displayed in slide show window one.

```
SlideShowWindows(1).View.Slide.Copy
```

This example displays the name of the presentation currently running in slide show window one.

```
MsgBox SlideShowWindows(1).View.Slide.Parent.Name
```



SlideElapsedTime Property

Returns the number of seconds that the current slide has been displayed.
Read/write **Long**.

Remarks

Use the [ResetSlideTime](#) method to reset the elapsed time for the slide that's currently displayed.

Example

This example sets a variable to the elapsed time for the slide that's currently displayed in slide show window one and then displays the value of the variable.

```
currTime = SlideShowWindows(1).View.SlideElapsedTime  
MsgBox currTime
```



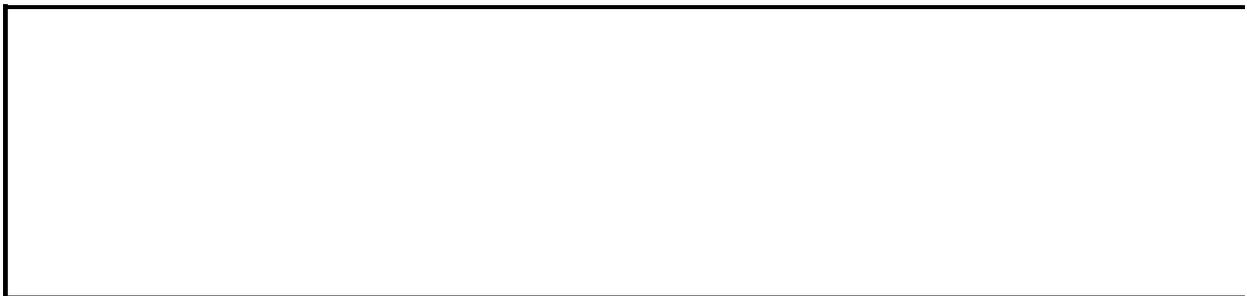
SlideHeight Property

Returns or sets the slide height, in points. Read/write **Single**.

Example

This example sets the slide height to 8.5 inches and the slide width to 11 inches for the active presentation.

```
With Application.ActivePresentation.PageSetup  
    .SlideWidth = 11 * 72  
    .SlideHeight = 8.5 * 72  
End With
```



SlideID Property

Returns a unique ID number for the specified slide. Read-only **Long**.

Remarks

Unlike the [SlideIndex](#) property, the **SlideID** property of a **Slide** object won't change when you add slides to the presentation or rearrange the slides in the presentation. Therefore, using the [FindBySlideID](#) method with the slide's ID number can be a more reliable way to return a specific **Slide** object from a **Slides** collection than using the [Item](#) method with the slide's index number.

Example

This example demonstrates how to retrieve the unique ID number for a **Slide** object and then use this number to return that **Slide** object from the **Slides** collection.

```
Set gslides = ActivePresentation.Slides
```

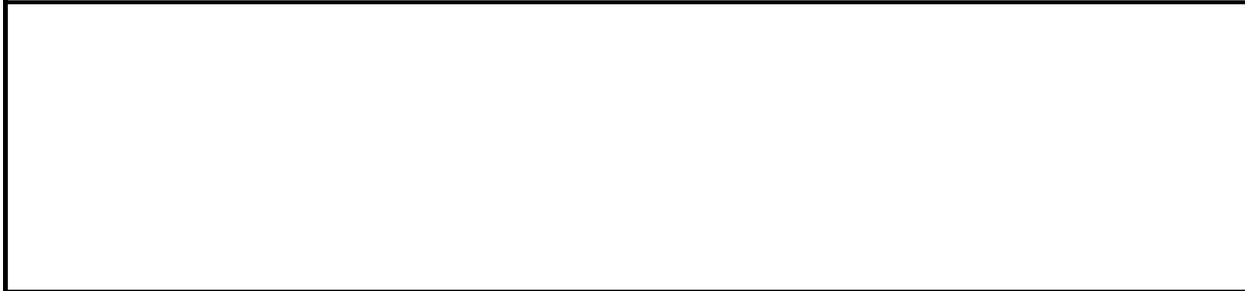
```
'Get slide ID
```

```
graphSlideID = gslides.Add(2, ppLayoutChart).SlideID
```

```
gslides.FindBySlideID(graphSlideID) _
```

```
    .SlideShowTransition.EntryEffect = _
```

```
    ppEffectCoverLeft      'Use ID to return specific slide
```



SlideIDs Property

Returns an array of slide IDs for the specified named slide show. Read-only
Variant.

Example

This example adds the current slide in the active window to the custom slide show named "Marketing Short Version." Note that to save a modified version of the custom slide show, you must delete the original custom show and then add it again, using the same name. Also note that if you want to resize an array contained in a **Variant** variable, you must explicitly declare the variable before attempting to resize its array.

```
'NOTE - The following code line is NOT optional.  
'Can't redim array without this  
Dim customShowSlideIDs As Variant  
Dim customShowToExpand As NamedSlideShow  
  
customShowName = "Marketing Short Version"  
Set customShowToExpand = ActivePresentation.SlideShowSettings _  
    .NamedSlideShows(customShowName)  
slideToAddID = ActiveWindow.View.Slide.SlideID  
customShowSlideIDs = customShowToExpand.SlideIDs  
numSlides = UBound(customShowSlideIDs)  
  
ReDim Preserve customShowSlideIDs(numSlides + 1)  
  
customShowSlideIDs(numSlides + 1) = slideToAddID  
customShowToExpand.Delete  
ActivePresentation.SlideShowSettings.NamedSlideShows _  
    .Add customShowName, customShowSlideIDs
```



SlideIndex Property

Returns the index number of the specified slide within the **Slides** collection.
Read-only **Long**.

Remarks

Unlike the [SlideID](#) property, the **SlideIndex** property of a **Slide** object can change when you add slides to the presentation or rearrange the slides in the presentation. Therefore, using the [FindBySlideID](#) method with the slide's ID number can be a more reliable way to return a specific **Slide** object from a **Slides** collection than using the [Item](#) method with the slide's index number.

Example

This example displays the index number of the currently displayed slide in slide show window one.

```
MsgBox SlideShowWindows(1).View.Slide.SlideIndex
```



SlideMaster Property

Returns a [Master](#) object that represents the slide master.

expression.**SlideMaster**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the background pattern for the slide master for the active presentation.

```
Application.ActivePresentation.SlideMaster.Background.Fill _  
    .PresetTextured msoTextureGreenMarble
```



[Show All](#)

SlideNumber Property

 [SlideNumber property as it applies to the **HeadersFooters** object.](#)

Returns a **HeaderFooter** object that represents the slide number in the lower-right corner of a slide, or the page number in the lower-right corner of a notes page or a page of a printed handout or outline. Read-only.

expression.**SlideNumber**

expression Required. An expression that returns one of the above objects.

 [SlideNumber property as it applies to the **Slide** and **SlideRange** objects.](#)

Returns the slide number. Read-only **Integer**.

expression.**SlideNumber**

expression Required. An expression that returns one of the above objects.

Remarks

The **SlideNumber** property of a **Slide** object is the actual number that appears in the lower-right corner of the slide when you display slide numbers. This number is determined by the number of the slide within the presentation (the [SlideIndex](#) property value) and the starting slide number for the presentation (the [FirstSlideNumber](#) property value). The slide number is always equal to the starting slide number + the slide index number – 1.

Example

[As it applies to the **HeadersFooters** object.](#)

This example hides the slide number on slide two in the active presentation if the number is currently visible, or it displays the slide number if it's currently hidden.

```
With Application.ActivePresentation.Slides(2) _
    .HeadersFooters.SlideNumber
    If .Visible Then
        .Visible = False
    Else
        .Visible = True
    End If
End With
```

[As it applies to the **Slide** and **SlideRange** objects.](#)

This example shows how changing the first slide number affects the slide number of a specific slide.

```
With Application.ActivePresentation
    .PageSetup.FirstSlideNumber = 1 'starts slide numbering at 1
    MsgBox .Slides(2).SlideNumber 'returns 2

    .PageSetup.FirstSlideNumber = 10 'starts slide numbering at 10
    MsgBox .Slides(2).SlideNumber 'returns 11
End With
```



[Show All](#)

SlideOrientation Property

Returns or sets the on-screen and printed orientation of slides in the specified presentation. Read/write [MsoOrientation](#).

MsoOrientation can be one of these MsoOrientation constants.

msoOrientationHorizontal

msoOrientationMixed

msoOrientationVertical

expression.**SlideOrientation**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets orientation of all slides in the active presentation to vertical (portrait).

```
Application.ActivePresentation.PageSetup.SlideOrientation = _  
    msoOrientationVertical
```



SlideRange Property

Returns a [SlideRange](#) object that represents a range of selected slides. Read-only.

Remarks

A slide range can be constructed in slide view, slide sorter view, normal view, notes page view, or outline view. In slide view, **SlideRange** returns one slide — the current, displayed slide.

Example

This example sets the background scheme color for all the selected slides in window one.

```
Windows(1).Selection.SlideRange.ColorScheme _  
    .Colors(ppBackground).RGB = RGB(0, 0, 255)
```



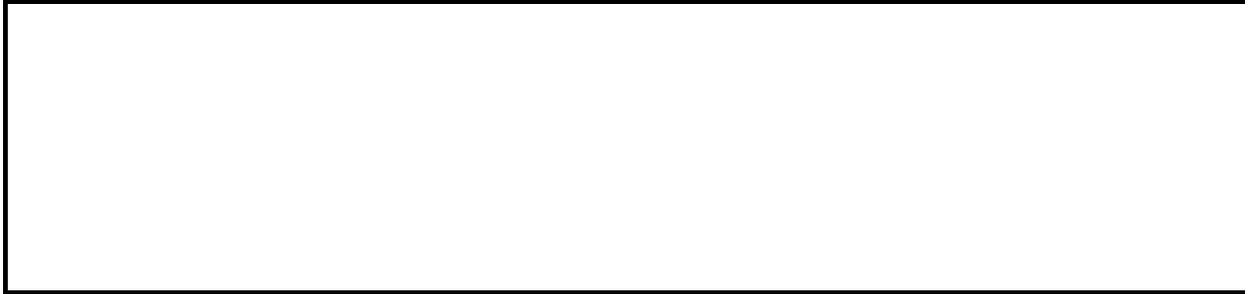
Slides Property

Returns a [Slides](#) collection that represents all slides in the specified presentation.
Read-only.

Example

This example adds a slide to the active presentation.

```
Application.ActivePresentation.Slides.Add 1, ppLayoutTitle
```



[Show All](#)

SlideShowName Property

 [SlideShowName property as it applies to the **ActionSetting**, **PrintOptions**, **PublishObject**, and **SlideShowSettings** objects.](#)

Returns or sets the name of the custom slide show to run in response to a mouse action on the shape during a slide show (**ActionSetting** object), returns or sets the name of the custom slide show to print (**PrintOptions** object), or returns or sets the name of the custom slide show published as a web presentation (**PublishObject** object). Read/write **String**.

expression.**SlideShowName**

expression Required. An expression that returns one of the above objects.

Remarks

The [RangeType](#) property must be set to **ppPrintNamedSlideShow** to print a custom slide show.

[SlideShowName](#) property as it applies to the [SlideShowView](#) object.

Returns the name of the custom slide show that's currently running in the specified slide show view. Read-only **String**.

expression.**SlideShowName**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **ActionSetting**, **PrintOptions**, **PublishObject**, and **SlideShowSettings** objects.](#)

This example prints an existing custom slide show named "tech talk."

```
With ActivePresentation.PrintOptions
    .RangeType = ppPrintNamedSlideShow
    .SlideShowName = "tech talk"
End With
ActivePresentation.PrintOut
```

The following example saves the current presentation as an HTML version 4.0 file with the name "mallard.htm." It then displays a message indicating that the current named presentation is being saved in both PowerPoint and HTML formats.

```
With Pres.PublishObjects(1)
    PresName = .SlideShowName
    .SourceType = ppPublishAll
    .FileName = "C:\HTMLPres\mallard.htm"
    .HTMLVersion = ppHTMLVersion4
    MsgBox ("Saving presentation " & "'" & PresName & "'" & " in PowerPoint" _
        & Chr(10) & Chr(13) _
        & " format and HTML version 4.0 format")
    .Publish
End With
```

 [As it applies to the **SlideShowView** object.](#)

If the slide show running in slide show window one is a custom slide show, this example displays its name.

```
With SlideShowWindows(1).View
    If .IsNamedShow Then
        MsgBox "Now showing in slide show window 1: " _
            & .SlideShowName
    End If
End With
```


SlideShowSettings Property

Returns a [SlideShowSettings](#) object that represents the slide show settings for the specified presentation. Read-only.

Example

This example starts a slide show meant to be presented by a speaker. The slide show will run with animation and narration turned off.

```
With Application.ActivePresentation.SlideShowSettings  
    .ShowType = ppShowTypeSpeaker  
    .ShowWithNarration = False  
    .ShowWithAnimation = False  
    .Run  
End With
```



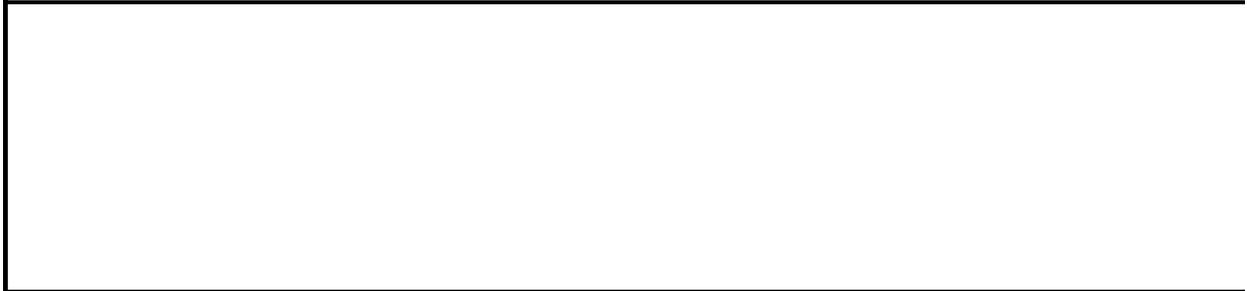
SlideShowTransition Property

Returns a [SlideShowTransition](#) object that represents the special effects for the specified slide transition. Read-only.

Example

This example sets slide two in the active presentation to advance automatically after 5 seconds during a slide show and to play a dog bark sound at the slide transition.

```
With ActivePresentation.Slides(2).SlideShowTransition
    .AdvanceOnTime = True
    .AdvanceTime = 5
    .SoundEffect.ImportFromFile "c:\windows\media\dogbark.wav"
End With
ActivePresentation.SlideShowSettings.AdvanceMode = _
    ppSlideShowUseSlideTimings
```



SlideShowWindow Property

Returns a [SlideShowWindow](#) object that represents the slide show window in which the specified presentation is running. Read-only.

Remarks

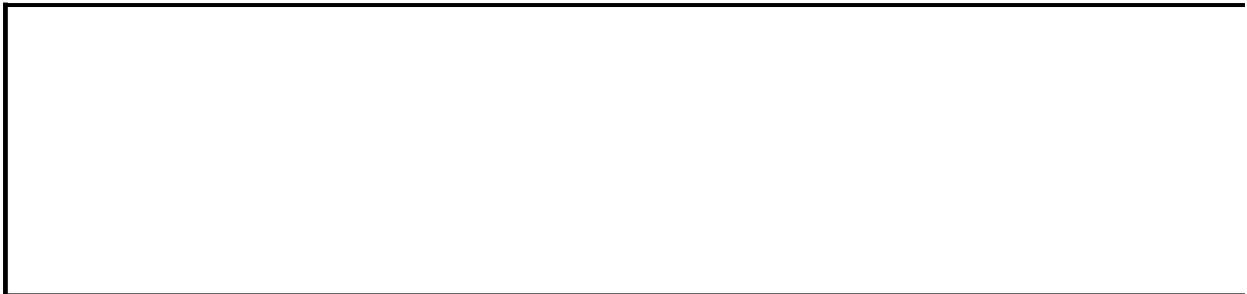
You can use this property in conjunction with the **Me** keyword and the **Parent** property to return the slide show window in which an ActiveX control event was fired, as shown in the example.

Example

The following example shows the Click event procedures for buttons named "cmdBack" and "cmdForward". If you add these buttons to the slide master and add these event procedures to them, all slides based on the master (and set to show master background graphics) will have these navigation buttons that will be active during a slide show. The **Me** keyword returns the **Master** object that represents the slide master that contains the control. If the control were on an individual slide, the **Me** keyword in an event procedure for that control would return a **Slide** object.

```
Private Sub cmdBack_Click()  
    Me.Parent.SlideShowWindow.View.Previous  
End Sub
```

```
Private Sub cmdForward_Click()  
    Me.Parent.SlideShowWindow.View.Next  
End Sub
```



SlideShowWindows Property

Returns a [SlideShowWindows](#) collection that represents all open slide show windows. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example runs a slide show in a window and sets the height and width of the slide show window.

```
With Application
    .Presentations(1).SlideShowSettings.Run
    With .SlideShowWindows(1)
        .Height = 250
        .Width = 250
    End With
End With
```



[Show All](#)

SlideSize Property

Returns or sets the slide size for the specified presentation. Read/write [PpSlideSizeType](#).

PpSlideSizeType can be one of these PpSlideSizeType constants.

ppSlideSize35MM

ppSlideSizeA3Paper

ppSlideSizeA4Paper

ppSlideSizeB4ISOPaper

ppSlideSizeB4JISPaper

ppSlideSizeB5ISOPaper

ppSlideSizeB5JISPaper

ppSlideSizeBanner

ppSlideSizeCustom

ppSlideSizeHagakiCard

ppSlideSizeLedgerPaper

ppSlideSizeLetterPaper

ppSlideSizeOnScreen

ppSlideSizeOverhead

expression.**SlideSize**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the slide size to overhead for the active presentation.

```
Application.ActivePresentation.PageSetup _  
    .SlideSize = ppSlideSizeOverhead
```



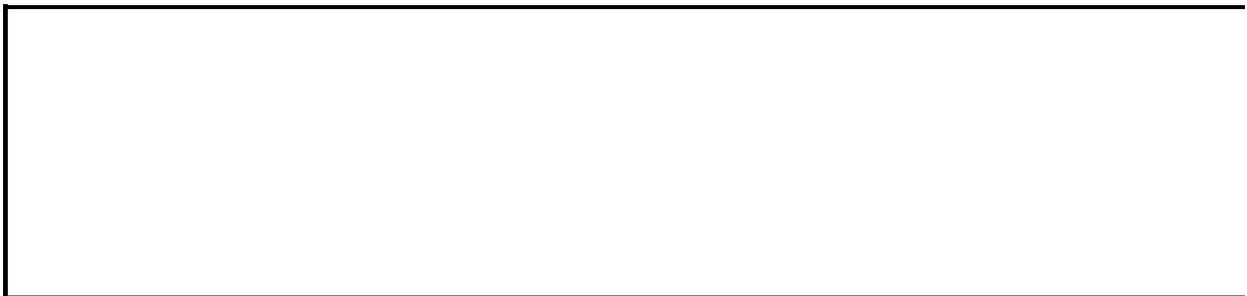
SlideWidth Property

Returns or sets the slide width, in points. Read/write **Single**.

Example

This example sets the slide height to 8.5 inches and the slide width to 11 inches for the active presentation.

```
With Application.ActivePresentation.PageSetup  
    .Slidewidth = 11 * 72  
    .SlideHeight = 8.5 * 72  
End With
```



[Show All](#)

Smooth Property

Sets or returns an [MsoTriState](#) that represents whether the transition from one animation point to another is smoothed. Read/write.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The animation point should not be smoothed.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. The animation should be smoothed.

expression.**Smooth**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example changes smoothing for an animation point.

```
Sub ChangeSmooth(ByVal ani As AnimationBehavior, ByVal bln As MsoTri  
    ani.PropertyEffect.Points.Smooth = bln  
End Sub
```



[Show All](#)

SmoothEnd Property

Sets or returns an [MsoTriState](#) constant that represents whether an animation should decelerate as it ends. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. An animation does not decelerate when it ends.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue An animation decelerates when it ends.

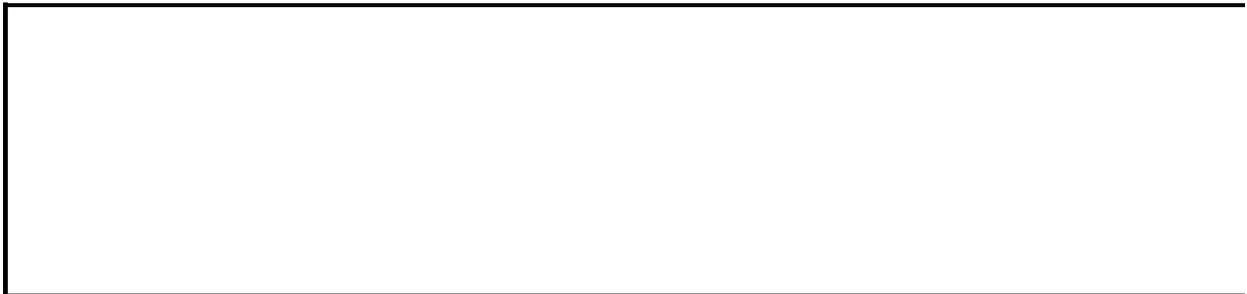
expression.**SmoothEnd**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to a slide, animates the shape, and instructs the shape to decelerate when it ends.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets animation effect  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    'Sets duration of effect and slows animation at end  
    With effDiamond.Timing  
        .Duration = 5  
        .SmoothEnd = msoTrue  
    End With  
  
End Sub
```



[Show All](#)

SmoothStart Property

Sets or returns an [MsoTriState](#) constant that represents whether an animation should accelerate when it starts. Read/write.

MsoTriState can be one of these **MsoTriState** constants.

msoCTrue Doesn't apply to this property.

msoFalse Default. The animation does not accelerate when it starts.

msoTriStateMixed Doesn't apply to this property.

msoTriStateToggle Doesn't apply to this property.

msoTrue The animation accelerates when it starts.

expression.**SmoothStart**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to a slide, animates the shape, and instructs the shape to accelerate when it starts.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    With effDiamond.Timing  
        .Duration = 5  
        .SmoothStart = msoTrue  
    End With  
  
End Sub
```



[Show All](#)

SnapToGrid Property

MsoTrue to snap shapes to the grid lines in the specified presentation.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue

expression.**SnapToGrid**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example toggles snapping shapes to the grid lines in the active presentation.

```
Sub ToggleSnapToGrid()  
  With ActivePresentation  
    If .SnapToGrid = msoTrue Then  
      .SnapToGrid = msoFalse  
    Else  
      .SnapToGrid = msoTrue  
    End If  
  End With  
End Sub
```



SoundEffect Property

[ActionSetting](#) object: Returns a [SoundEffect](#) object that represents the sound to be played when the specified shape is clicked or the mouse pointer passes over the shape. If you don't hear the sound that you assigned to the shape when you run the slide show, make sure that the [TextLevelEffect](#) property is set to a value other than **ppAnimateLevelNone** and that the [Animate](#) property is set to **True**.

[AnimationSettings](#) and [EffectInformation](#) objects: Returns a **SoundEffect** object that represents the sound to be played during the animation of the specified shape.

[SlideShowTransition](#) object: Returns a **SoundEffect** object that represents the sound to be played during the transition to the specified slide.

expression.**SoundEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the file Bass.wav to be played whenever shape one on slide one in the active presentation is animated.

```
With ActivePresentation.Slides(1).Shapes(1).AnimationSettings
    .Animate = True
    .TextLevelEffect = ppAnimateByAllLevels
    .SoundEffect.ImportFromFile "c:\bass.wav"
End With
```



SourceFullName Property

Returns or sets the name and path of the source file for the linked OLE object.
Read/write **String**.

Example

This example sets the source file for shape one on slide one in the active presentation to Wordtest.doc and specifies that the object's image be updated automatically.

```
With ActivePresentation.Slides(1).Shapes(1)
  If .Type = msoLinkedOLEObject Then
    With .LinkFormat
      .SourceFullName = "c:\my documents\wordtest.doc"
      .AutoUpdate = ppUpdateOptionAutomatic
    End With
  End If
End With
```



[Show All](#)

SourceType Property

Returns or sets the source type of the presentation to be published to HTML. Read/write [PpPublishSourceType](#).

PpPublishSourceType can be one of these PpPublishSourceType constants.

ppPublishAll

ppPublishNamedSlideShow Use this value to publish a custom slide show, specifying the name of the custom slide show with the [SlideShowName](#) property.

ppPublishSlideRange

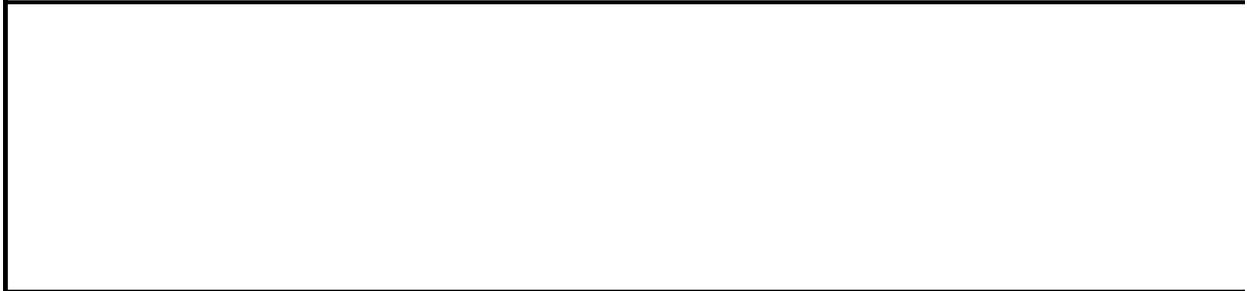
expression.**SourceType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example publishes the specified slide range (slides three through five) of the active presentation to HTML. It names the published presentation Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .Publish
End With
```



SpaceAfter Property

Returns or sets the amount of space after the last line in each paragraph of the specified text, in points or lines. Read/write **Single**.

Example

This example sets the spacing after paragraphs to 6 points for the text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat
    .LineRuleAfter = False
    .SpaceAfter = 6
  End With
End With
```



SpaceBefore Property

Returns or sets the amount of space before the first line in each paragraph of the specified text, in points or lines. Read/write **Single**.

Example

This example sets the spacing before paragraphs to 6 points for the text in shape two on slide in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat
    .LineRuleBefore = False
    .SpaceBefore = 6
  End With
End With
```



SpaceWithin Property

Returns or sets the amount of space between base lines in the specified text, in points or lines. Read/write **Single**.

Example

This example sets line spacing to 21 points for the text in shape two on slide two in the active presentation.

```
With Application.ActivePresentation.Slides(2).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat
    .LineRuleWithin = False
    .SpaceWithin = 21
  End With
End With
```



[Show All](#)

SpeakerNotes Property

Determines whether speaker notes are to be published with the presentation.
Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Speaker notes are to be published with the presentation.

Example

This example publishes slides three through five of the active presentation to HTML. It includes the associated speaker's notes with the published presentation and names it Mallard.htm.

```
With ActivePresentation.PublishObjects(1)
    .FileName = "C:\Test\Mallard.htm"
    .SourceType = ppPublishSlideRange
    .RangeStart = 3
    .RangeEnd = 5
    .SpeakerNotes = msOTrue
    .Publish
End With
```



[Show All](#)

Speed Property

 [Speed property as it applies to the **SlideShowTransition** object.](#)

Returns or sets a **PpTransitionSpeed** constant that represents the speed of the transition to the specified slide. Read/write.

PpTransitionSpeed can be one of these PpTransitionSpeed constants.

ppTransitionSpeedFast

ppTransitionSpeedMedium

ppTransitionSpeedMixed

ppTransitionSpeedSlow

expression.**Speed**

expression Required. An expression that returns a **SlideShowTransition** object.

 [Speed property as it applies to the **Timing** object.](#)

Returns or sets a **Single** that represents the speed, in seconds, of the specified animation. Read/write.

expression.**Speed**

expression Required. An expression that returns a **Timing** object.

Example

 [As it applies to the **SlideShowTransition** object.](#)

This example sets the special effect for the transition to the first slide in the active presentation and specifies that the transition be fast.

```
With ActivePresentation.Slides(1).SlideShowTransition
    .EntryEffect = ppEffectStripsDownLeft
    .Speed = ppTransitionSpeedFast
End With
```

 [As it applies to the **Timing** object.](#)

This example sets the animation for the main sequence to reverse and sets the speed to one second.

```
Sub AnimPoints()
    Dim tmlAnim As TimeLine
    Dim spdAnim As Timing

    Set tmlAnim = ActivePresentation.Slides(1).TimeLine
    Set spdAnim = tmlAnim.MainSequence(1).Timing
    With spdAnim
        .AutoReverse = msoTrue
        .Speed = 1
    End With
End Sub
```



SplitHorizontal Property

Returns or sets the percentage of the document window width that the outline pane occupies in normal view. Corresponds to the pane divider position between the slide and outline panes. Read/write **Long**.

Remarks

The maximum value of the **SplitHorizontal** property is always less than 100% because the slide pane has a minimum width that depends on a 10% zoom level. The actual maximum value may vary depending on the size of the application window.

Example

The following example sets the vertical pane divider for the active document window to divide at 15% outline pane and 85% slide pane.

```
ActiveWindow.SplitHorizontal = 15
```



SplitVertical Property

Returns or sets the percentage of the document window height that the slide pane occupies in normal view. Corresponds to the pane divider position between the slide and notes panes. Read/write **Long**.

Remarks

The minimum value of the **SplitVertical** property is always greater than 0% because the slide pane has a minimum height that depends on a 10% zoom level. The actual minimum value may vary depending on the size of the application window.

Example

The following example sets the horizontal pane divider for the active document window to divide at 60% slide pane and 40% notes pane.

```
ActiveWindow.SplitVertical = 60
```



[Show All](#)

Start Property

 [Start property as it applies to the **PrintRange** object.](#)

Returns the number of the first slide in the range of slides to be printed. Read-only **Integer**.

expression.**Start**

expression Required. An expression that returns one of the above objects.

 [Start property as it applies to the **TextRange** object.](#)

Returns the position of the first character in the specified text range relative to the first character in the shape that contains the text. Read-only **Long**.

expression.**Start**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **PrintRange** object.](#)

This example displays a message that indicates the starting and ending slide numbers for print range one in the active presentation.

```
With ActivePresentation.PrintOptions.Ranges
  If .Count > 0 Then
    With .Item(1)
      MsgBox "Print range 1 starts on slide " & .Start & _
        " and ends on slide " & .End
    End With
  End If
End With
```



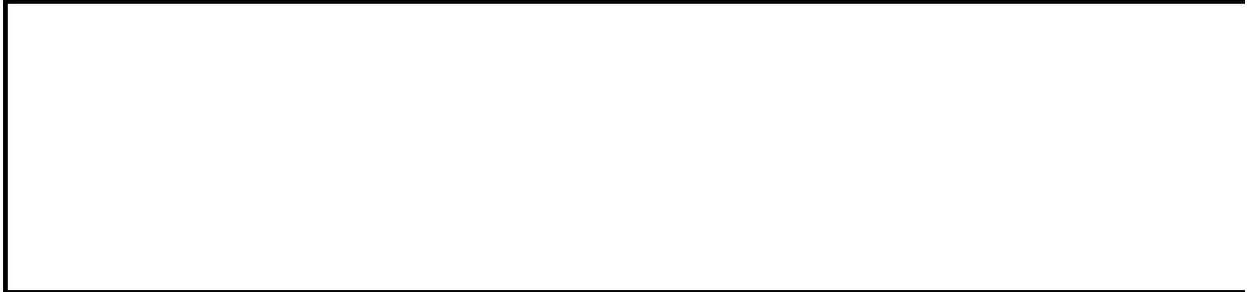
StartingSlide Property

Returns or sets the first slide to be displayed in the specified slide show.
Read/write **Long**.

Example

This example runs a slide show of the active presentation, starting with slide two and ending with slide four.

```
With ActivePresentation.SlideShowSettings
    .RangeType = ppShowSlideRange
    .StartingSlide = 2
    .EndingSlide = 4
    .Run
End With
```



StartValue Property

Returns or sets the beginning value of a bulleted list when the [Type](#) property of the **BulletFormat** object is set to **ppBulletNumbered**. The value must be in the range of 1 to 32767. Read/write **Integer**.

Example

This example sets the bullets in the text box specified by shape two on slide one to start with the number five.

```
With ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange.ParagraphFormat.Bullet
    .Type = ppBulletNumbered
    .StartValue = 5
  End With
End With
```



[Show All](#)

State Property

Returns or sets the state of the slide show. Read/write [PpSlideShowState](#).

PpSlideShowState can be one of these PpSlideShowState constants.

ppSlideShowBlackScreen

ppSlideShowDone

ppSlideShowPaused

ppSlideShowRunning

ppSlideShowWhiteScreen

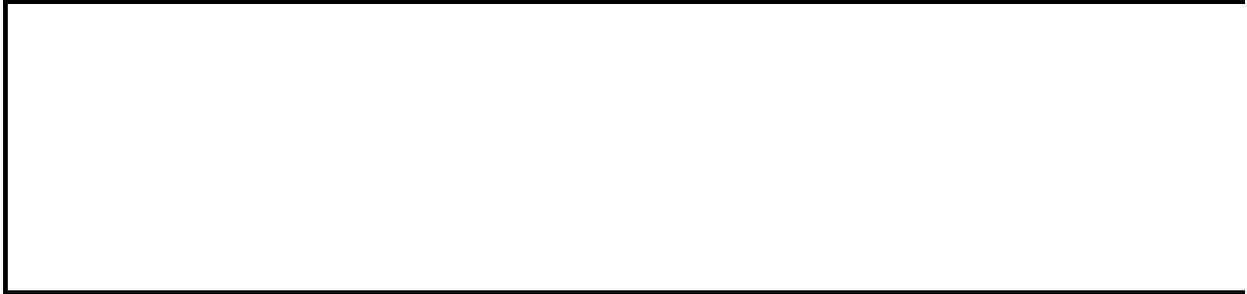
expression.**State**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the view state in slide show window one to a black screen.

```
SlideShowWindows(1).View.State = ppSlideShowBlackScreen
```



StopAfterSlides Property

Returns or sets the number of slides to be displayed before the media clip stops playing. Read/write **Long**.

Remarks

For the **StopAfterSlides** property setting to take effect, the [PauseAnimation](#) property of the specified slide must be set to **False**, and the [PlayOnEntry](#) property must be set to **True**.

The media clip will stop playing when the specified number of slides have been displayed or when the clip comes to an end— whichever comes first. A value of 0 (zero) specifies that the clip will stop playing after the current slide.

Example

This example specifies that the media clip represented by shape three on slide one in the active presentation will be played automatically when it's animated, that the slide show will continue while the media clip is playing in the background, and that the clip will stop playing after three slides are displayed or when the end of the clip is reached— whichever comes first. Shape three must be a sound or movie object.

```
Set OLEobj = ActivePresentation.Slides(1).Shapes(3)
With OLEobj.AnimationSettings.PlaySettings
    .PlayOnEntry = True
    .PauseAnimation = False
    .StopAfterSlides = 3
End With
```



[Show All](#)

Style Property

 [Style property as it applies to the **LineFormat** object.](#)

Returns or sets the line style. Read/write [MsoLineStyle](#).

MsoLineStyle can be one of these MsoLineStyle constants.

msoLineSingle

msoLineStyleMixed

msoLineThickBetweenThin

msoLineThickThin

msoLineThinThick

msoLineThinThin

expression.Style

expression Required. An expression that returns one of the above objects.

 [Style property as it applies to the **BulletFormat** object.](#)

Returns or sets the bullet style. Read/write [PpNumberedBulletStyle](#). Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.

PpNumberedBulletStyle can be one of these PpNumberedBulletStyle constants.

ppBulletAlphaLCParenBoth Lowercase alphabetic characters with both parentheses.

ppBulletAlphaLCParenRight Lowercase alphabetic characters with right parenthesis.

ppBulletAlphaLCPeriod Lowercase alphabetic characters with a period.

ppBulletAlphaUCParenBoth Uppercase alphabetic characters with both parentheses.

ppBulletAlphaUCParenRight Uppercase alphabetic characters with right parenthesis.

ppBulletAlphaUCPeriod Uppercase alphabetic characters with a period.

ppBulletArabicAbjadDash Arabic Abjad alphabets with a dash.

ppBulletArabicAlphaDash Arabic language alphabetic characters with a dash.

ppBulletArabicDBPeriod Double-byte Arabic numbering scheme with double-byte period.

ppBulletArabicDBPlain Double-byte Arabic numbering scheme (no punctuation).

ppBulletArabicParenBoth Arabic numerals with both parentheses.

ppBulletArabicParenRight Arabic numerals with right parenthesis.

ppBulletArabicPeriod Arabic numerals with a period.

ppBulletArabicPlain Arabic numerals.

ppBulletCircleNumDBPlain Double-byte circled number for values up to 10.

ppBulletCircleNumWDBlackPlain Shadow color number with circular background of normal text color.

ppBulletCircleNumWDWhitePlain Text colored number with same color circle drawn around it.

ppBulletHebrewAlphaDash Hebrew language alphabetic characters with a dash.

ppBulletHindiAlphaPeriod

ppBulletHindiNumPeriod

ppBulletKanjiKoreanPeriod Japanese/Korean numbers with a period.

ppBulletKanjiKoreanPlain Japanese/Korean numbers without a period.

ppBulletRomanLCParenBoth Lowercase Roman numerals with both parentheses.

ppBulletRomanLCParenRight Lowercase Roman numerals with right parenthesis.

ppBulletRomanLCPeriod Lowercase Roman numerals with period.

ppBulletRomanUCParenBoth Uppercase Roman numerals with both parentheses.

ppBulletRomanUCParenRight Uppercase Roman numerals with right parenthesis.

ppBulletRomanUCPeriod Uppercase Roman numerals with period.

ppBulletSimpChinPeriod Simplified Chinese with a period.

ppBulletSimpChinPlain Simplified Chinese without a period.

ppBulletStyleMixed Any undefined style.

ppBulletThaiAlphaParenBoth

ppBulletThaiAlphaParenRight

ppBulletThaiAlphaPeriod

ppBulletThaiNumParenBoth

ppBulletThaiNumParenRight

ppBulletThaiNumPeriod

ppBulletTradChinPeriod Traditional Chinese with a period.

ppBulletTradChinPlain Traditional Chinese without a period.

expression.**Style**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **LineFormat** object.](#)

This example adds a thick, blue compound line to myDocument. The compound line consists of a thick line with a thin line on either side of it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .Style = msoLineThickBetweenThin
    .Weight = 8
    .ForeColor.RGB = RGB(0, 0, 255)
End With
```

 [As it applies to the **BulletFormat** object.](#)

This example sets the bullet style for the bulleted list, represented by shape one on the first slide, to a shadow color number with circular background of normal text color.

```
ActivePresentation.Slides(1).Shapes(1).TextFrame _
    .TextRange.ParagraphFormat.Bullet _
    .Style = ppBulletCircleNumWDBlackPlain
```



SubAddress Property

Returns or sets the location within a document— such as a bookmark in a word document, a range in a Microsoft Excel worksheet, or a slide in a PowerPoint presentation— associated with the specified hyperlink. Read/write **String**.

Example

This example sets shape one on slide one in the active presentation to jump to the slide named "Last Quarter" in Latest Figures.ppt when the shape is clicked during a slide show.

```
With ActivePresentation.Slides(1).Shapes(1) _  
    .ActionSettings(ppMouseClick)  
    .Action = ppActionHyperlink  
    With .Hyperlink  
        .Address = "c:\sales\latest figures.ppt"  
        .SubAddress = "last quarter"  
    End With  
End With
```

This example sets shape one on slide one in the active presentation to jump to range A1:B10 in Latest.xls when the shape is clicked during a slide show.

```
With ActivePresentation.Slides(1).Shapes(1) _  
    .ActionSettings(ppMouseClick)  
    .Action = ppActionHyperlink  
    With .Hyperlink  
        .Address = "c:\sales\latest.xls"  
        .SubAddress = "A1:B10"  
    End With  
End With
```



[Show All](#)

Subscript Property

Determines whether the specified text is subscript. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. The specified text is not subscript.

msoTriStateMixed Some characters are subscript and some aren't.

msoTriStateToggle

msoTrue The specified text is subscript.

Remarks

Setting the **BaselineOffset** property to a negative value automatically sets the **Subscript** property to **msoTrue** and the **Superscript** property to **msoFalse**.

Setting the **BaselineOffset** property to a positive value automatically sets the **Subscript** property to **msoFalse** and the **Superscript** property to **msoTrue**.

Setting the **Subscript** property to **msoTrue** automatically sets the **BaselineOffset** property to -0.25 (-25 percent).

Example

This example enlarges the first character in the title on slide one if that character is subscript.

```
With Application.ActivePresentation.Slides(1) _  
    .Shapes.Title.TextFrame.TextRange  
    With .Characters(1, 1).Font  
        If .Subscript Then  
            scaleChar = -20 * .BaselineOffset  
            .Size = .Size * scaleChar  
        End If  
    End With  
End With
```



Subtype Property

Sets or returns a **MsoAnimFilterEffectSubtype** constant that determines the subtype of the filter effect. Read/write.

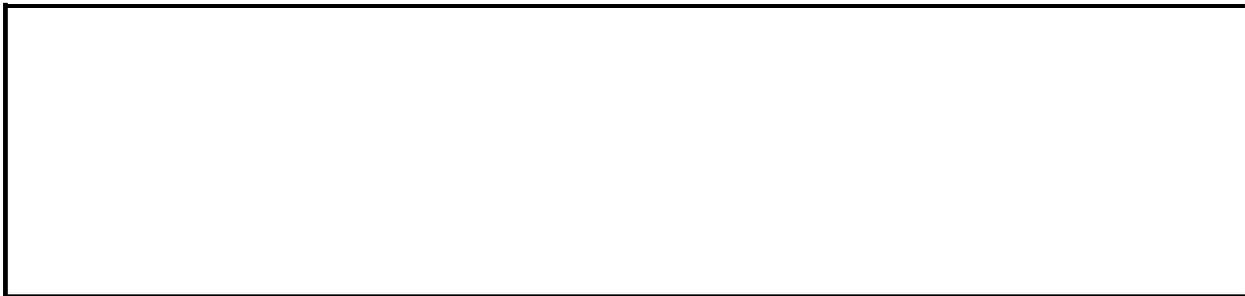
expression.**Subtype**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to the first slide of the active presentation and sets a filter effect animation behavior.

```
Sub ChangeFilterEffect()  
    Dim sldFirst As Slide  
    Dim shpHeart As Shape  
    Dim effNew As Effect  
    Dim bhvEffect As AnimationBehavior  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpHeart = sldFirst.Shapes.AddShape(Type:=msoShapeHeart, _  
        Left:=100, Top:=100, Width:=100, Height:=100)  
    Set effNew = sldFirst.TimeLine.MainSequence.AddEffect _  
        (Shape:=shpHeart, EffectID:=msoAnimEffectChangeFillColor, _  
        Trigger:=msoAnimTriggerAfterPrevious)  
    Set bhvEffect = effNew.Behaviors.Add(msoAnimTypeFilter)  
  
    With bhvEffect.FilterEffect  
        .Type = msoAnimFilterEffectTypeWipe  
        .Subtype = msoAnimFilterEffectSubtypeUp  
        .Reveal = msoTrue  
    End With  
End Sub
```



[Show All](#)

Superscript Property

Determines whether the specified text is superscript. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. The specified text is not superscript.

msoTriStateMixed Some characters are superscript and some aren't.

msoTriStateToggle

msoTrue The specified text is superscript.

Remarks

Setting the **BaselineOffset** property to a negative value automatically sets the **Subscript** property to **msoTrue** and the **Superscript** property to **msoFalse**.

Setting the **BaselineOffset** property to a positive value automatically sets the **Subscript** property to **msoFalse** and the **Superscript** property to **msoTrue**.

Setting the **Superscript** property to **msoTrue** automatically sets the **BaselineOffset** property to 0.3 (30 percent).

Example

This example sets the text for shape two on slide one and then makes the fifth character superscript with a 30-percent offset.

```
With Application.ActivePresentation.Slides(1).Shapes(2).TextFrame
  With .TextRange
    .Text = "E=mc2"
    .Characters(5, 1).Font.Superscript = msoTrue
  End With
End With
```



[Show All](#)

Sync Property

Returns a [Sync](#) object that enables you to manage the synchronization of the local and server copies of a shared presentation stored in a Microsoft Windows SharePoint Services shared workspace. Read-only.

expression.Sync

expression Required. An expression that returns a **Presentation** object.

Remarks

The **Status** property of the **Sync** object returns important information about the current state of synchronization. Use the **GetUpdate** method to refresh the sync status. Use the **LastSyncTime**, **ErrorType**, and **WorkspaceLastChangedBy** properties to return additional information.

For more information on the differences and conflicts that can exist between the local and server copies of shared presentations, see the **Status** property.

Use the **PutUpdate** method to save local changes to the server. Close and re-open the document to retrieve the latest version from the server when no local changes have been made. Use the **ResolveConflict** method to resolve differences between the local and the server copies, or the **OpenVersion** method to open a different version along with the currently open local version of the document.

The **GetUpdate**, **PutUpdate**, and **ResolveConflict** methods of the **Sync** object do not return status codes because they complete their tasks asynchronously. The **Sync** object provides important status information through a single event, called the **PresentationSync** event of the **Application** object.

The **Sync** event described above returns an [msoSyncEventType](#) value.

MsoSyncEventType can be one of the following **msoSyncEventType** constants.

- msoSyncEventDownloadInitiated** (0)
- msoSyncEventDownloadSucceeded** (1)
- msoSyncEventDownloadFailed** (2)
- msoSyncEventUploadInitiated** (3)
- msoSyncEventUploadSucceeded** (4)
- msoSyncEventUploadFailed** (5)
- msoSyncEventDownloadNoChange** (6)
- msoSyncEventOffline** (7)

The **Sync** object model is available whether sharing and synchronization are enabled or disabled on the active document. The **Sync** property of the **Presentation** object does not return **Nothing** when the active document is not

shared or synchronization is not enabled. Use the **Status** property to determine whether the document is shared and whether synchronization is enabled.

Not all document synchronization problems raise run-time errors that can be trapped. After using the methods of the **Sync** object, it is a good idea to check the **Status** property. If the **Status** property is **msoSyncStatusError**, check the **ErrorType** property for additional information on the type of error that has occurred.

In many circumstances, the best way to resolve an error condition is to call the **GetUpdate** method. For example, if a call to **PutUpdate** results in an error condition, then a call to **GetUpdate** will reset the status to **msoSyncStatusLocalChanges**.

Example

The following example displays the name of the last person to modify the active presentation if the active presentation is a shared document in a Document Workspace.

```
Dim eStatus As MsoSyncStatusType
Dim strLastUser As String

eStatus = ActivePresentation.Sync.Status

If eStatus = msoSyncStatusLatest Then
    strLastUser = ActivePresentation.Sync.WorkspaceLastChangedBy
    MsgBox "You have the most up-to-date copy." & _
        "This file was last modified by " & strLastUser
End If
```



Table Property

Returns a [Table](#) object that represents a table in a shape or in a shape range.
Read-only.

Example

This example sets the width of the first column in the table in shape five on the second slide to 80 points.

```
ActivePresentation.Slides(2).Shapes(5).Table _  
                                .Columns(1).Width = 80
```



[Show All](#)

TableDirection Property

Returns or sets the direction in which the table cells are ordered. Read/write [PpDirection](#).

PpDirection can be one of these PpDirection constants.

ppDirectionLeftToRight

ppDirectionMixed

ppDirectionRightToLeft

expression.**TableDirection**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The default value of the **TableDirection** property is **ppDirectionLefttToRight** unless the [LanguageSettings](#) property or the [DefaultLanguageID](#) property is set to a right-to-left language, in which case the default value is **ppDirectionRightToLeft**. The **ppDirectionMixed** constant may be returned when using the **TextDirection** property.

Example

This example sets the direction in which cells in the selected table are ordered to left to right (first column is the leftmost column).

```
ActiveWindow.Selection.ShapeRange.Table.TableDirection = _  
    ppDirectionLeftToRight
```



TabStops Property

Returns a [TabStops](#) collection that represents the tab stops for the specified text.
Read-only.

Example

This example adds a slide with two text columns to the active presentation, sets a left-aligned tab stop for the title on the new slide, aligns the title box to the left, and assigns title text utilizing the tab stop just created.

```
With Application.ActivePresentation.Slides _  
    .Add(2, ppLayoutTwoColumnText).Shapes  
    With .Title.TextFrame  
        With .Ruler  
            .Levels(1).FirstMargin = 0  
            .TabStops.Add ppTabStopLeft, 310  
        End With  
        .TextRange.ParagraphFormat.Alignment = ppAlignLeft  
        .TextRange = "first column" + Chr(9) + "second column"  
    End With  
End With
```



Tags Property

Returns a [Tags](#) object that represents the tags for the specified object. Read-only.

Example

Note Tag values are added and stored in uppercase text. You should perform tests on tag values using uppercase text, as shown in the second example.

This example adds a tag named "REGION" and a tag named "PRIORITY" to slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Tags
    .Add "Region", "East"      'Adds "Region" tag with value "East"
    .Add "Priority", "Low"    'Adds "Priority" tag with value "Low"
End With
```

This example searches through the tags for each slide in the active presentation. If there's a tag named "PRIORITY," a message box displays the tag value. If the object doesn't have a tag named "PRIORITY," the example adds this tag with the value "Unknown."

```
For Each s In Application.ActivePresentation.Slides
    With s.Tags
        found = False
        For i = 1 To .Count
            If .Name(i) = "PRIORITY" Then
                found = True
                slNum = .Parent.SlideIndex
                MsgBox "Slide " & slNum & " Priority: " & .Value(i)
            End If
        Next
        If Not found Then
            slNum = .Parent.SlideIndex
            .Add "Priority", "Unknown"
            MsgBox "Slide " & slNum & " Priority tag added: Unknown"
        End If
    End With
Next
```



[Show All](#)

TargetBrowser Property

Sets or returns an [MsoTargetBrowser](#) constant that represents the browser used with Microsoft PowerPoint. Read/write.

MsoTargetBrowser can be one of these **MsoTargetBrowser** constants.

msoTargetBrowserIE4

msoTargetBrowserIE5

msoTargetBrowserIE6

msoTargetBrowserV3

msoTargetBrowserV4

expression.**TargetBrowser**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the target browser for the active presentation to Microsoft Internet Explorer 6 if the current target browser is an earlier version.

```
Sub SetWebBrowser()  
    With ActivePresentation.WebOptions  
        If .TargetBrowser < msoTargetBrowserIE6 Then  
            .TargetBrowser = msoTargetBrowserIE6  
        End If  
    End With  
End Sub
```

This example sets the target browser for all presentations to Internet Explorer 6.

```
Sub GlobalTargetBrowser()  
    Application.DefaultWebOptions _  
        .TargetBrowser = msoTargetBrowserIE6  
End Sub
```



TemplateName Property

Returns the name of the design template associated with the specified presentation. Read-only **String**.

Remarks

The returned string includes the MS-DOS file name extension (for file types that are registered) but doesn't include the full path.

Example

The following example applies the design template Professional.pot to the presentation Pres1.ppt if it's not already applied to it.

```
With Presentations("Pres1.ppt")
  If .TemplateName <> "Professional.pot" Then
    .ApplyTemplate "c:\program files\microsoft office" & _
      "\templates\presentation designs\Professional.pot"
  End If
End With
```



[Show All](#)

Text Property

 [Text property as it applies to the **HeaderFooter**, **TextEffectFormat**, and **TextRange** objects.](#)

Returns or sets a **String** that represents the text contained in the specified object. Read/write.

expression.**Text**

expression Required. An expression that returns one of the above objects.

 [Text property as it applies to the **Comment** object.](#)

Returns a **String** that represents the text in a comment. Read-only.

expression.**Text**

expression Required. An expression that returns a [Comment](#) object.

Example

 [As it applies to the **TextRange** object.](#)

This example sets the text and font style for the title on slide one in the active presentation.

```
Set myPres = Application.ActivePresentation
With myPres.Slides(1).Shapes.Title.TextFrame.TextRange
    .Text = "Welcome!"
    .Font.Italic = True
End With
```



[Show All](#)

TextDirection Property

Returns or sets the text direction for the specified paragraph. Read/write [PpDirection](#). The default value depends on the language support you have selected or installed.

PpDirection can be one of these PpDirection constants.

ppDirectionLeftToRight

ppDirectionMixed

ppDirectionRightToLeft

expression.**TextDirection**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example displays the text direction for the paragraphs in shape two on slide one in the active presentation.

```
MsgBox ActivePresentation.Slides(1).Shapes(2).TextFrame.TextRange _  
    .ParagraphFormat.TextDirection
```



TextEffect Property

Returns a [TextEffectFormat](#) object that contains text-effect formatting properties for the specified shape. Applies to [Shape](#) or [ShapeRange](#) objects that represent WordArt.

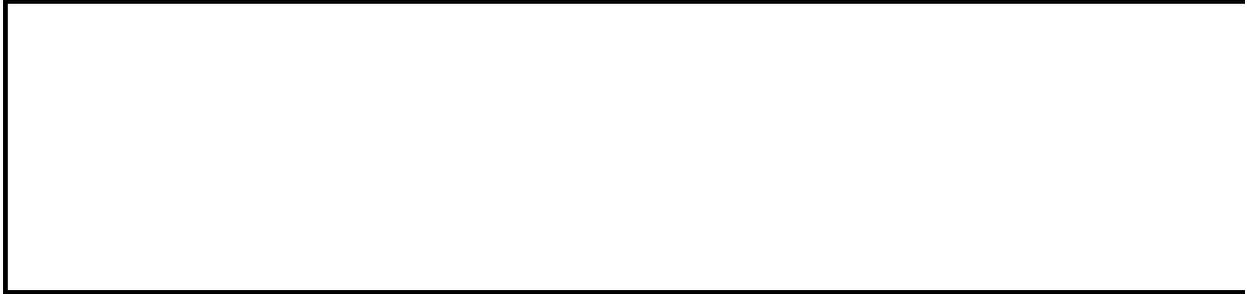
expression.**TextEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the font style to bold for shape three on myDocument if the shape is WordArt.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.FontBold = True
    End If
End With
```



TextFrame Property

Returns a [TextFrame](#) object that contains the alignment and anchoring properties for the specified shape or master text style.

Remarks

Use the [TextRange](#) property of the **TextFrame** object to return the text in the text frame.

Use the [HasTextFrame](#) property to determine whether a shape contains a text frame before you apply the **TextFrame** property.

Example

This example adds a rectangle to myDocument, adds text to the rectangle, and sets the top margin for the text frame.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes _
    .AddShape(msoShapeRectangle, 180, 175, 350, 140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginTop = 10
End With
```



[Show All](#)

TextLevelEffect Property

Returns or sets a [PpTextLevelEffect](#) constant that indicates whether the text in the specified shape is animated by first-level paragraphs, second-level paragraphs, or some other level (up to fifth-level paragraphs). Read/write.

PpTextLevelEffect can be one of these PpTextLevelEffect constants.

ppAnimateByAllLevels

ppAnimateByFifthLevel

ppAnimateByFirstLevel

ppAnimateByFourthLevel

ppAnimateBySecondLevel

ppAnimateByThirdLevel

ppAnimateLevelMixed

ppAnimateLevelNone

expression.**TextLevelEffect**

expression Required. An expression that returns one of the objects in the Applies To list.

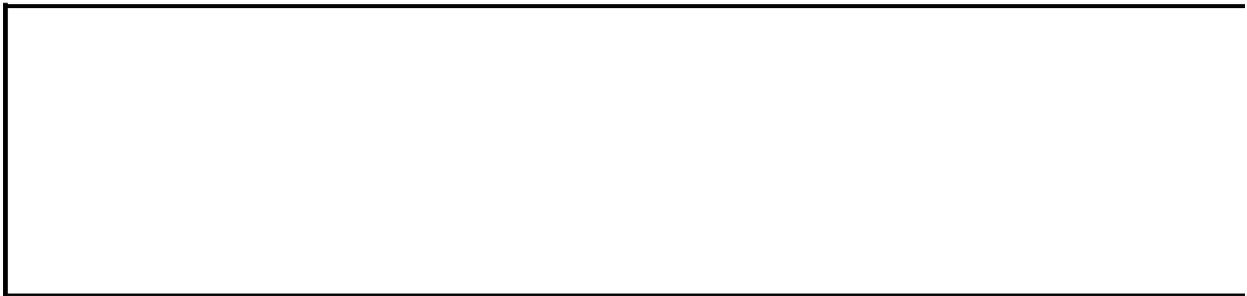
Remarks

For the **TextLevelEffect** property setting to take effect, the [Animate](#) property must be set to **True**.

Example

This example adds a title slide and title text to the active presentation and sets the title to be built letter by letter.

```
With ActivePresentation.Slides.Add(1, ppLayoutTitleOnly).Shapes(1)
    .TextFrame.TextRange.Text = "Sample title"
    With .AnimationSettings
        .Animate = True
        .TextLevelEffect = ppAnimateByFirstLevel
        .TextUnitEffect = ppAnimateByCharacter
        .EntryEffect = ppEffectFlyFromLeft
    End With
End With
```



TextRange Property

Returns a [TextRange](#) object that represents the selected text ([Selection](#) object) or the text in the specified text frame ([TextFrame](#) object).

expression.**TextRange**

expression Required. An expression that returns one of the above objects.

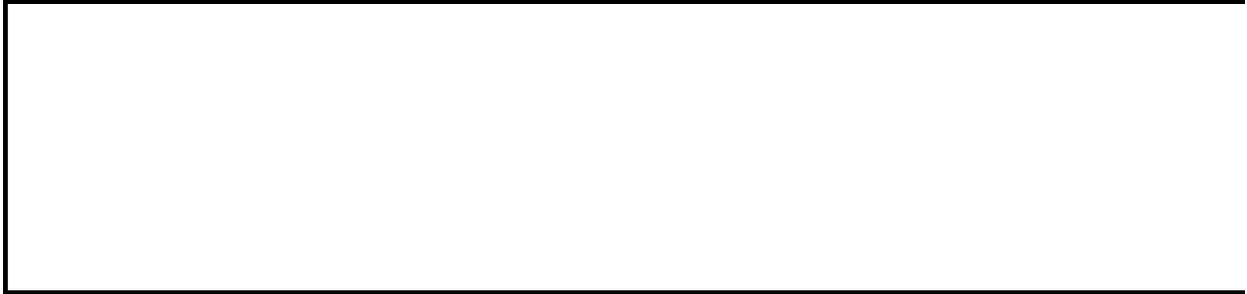
Remarks

You can construct a text range from a selection when the presentation is in slide view, normal view, outline view, notes page view, or any master view.

Example

This example makes the selected text bold in the first window.

```
Windows(1).Selection.TextRange.Font.Bold = True
```



TextRangeLength Property

Returns or sets an **Long** that represents the length of a text range. Read-only.

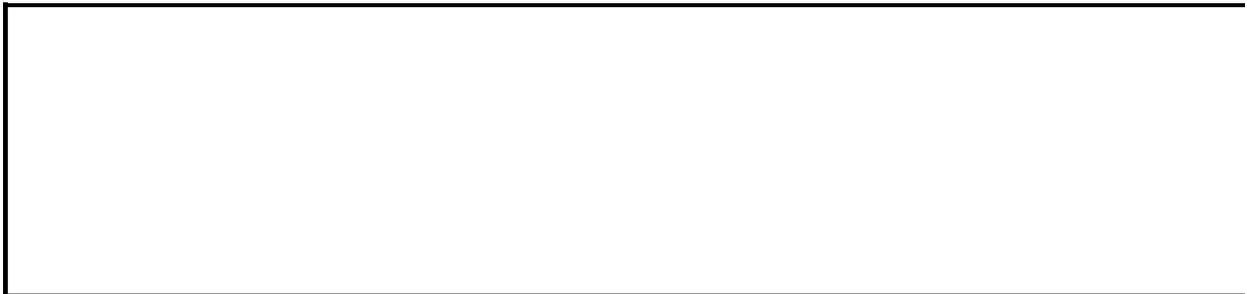
expression.**TextRangeLength**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a shape with text and rotates the shape without rotating the text.

```
Sub SetTextRange()  
    Dim shpStar As Shape  
    Dim sldOne As Slide  
    Dim effNew As Effect  
  
    Set sldOne = ActivePresentation.Slides(1)  
    Set shpStar = sldOne.Shapes.AddShape(Type:=msoShape5pointStar, _  
        Left:=32, Top:=32, Width:=300, Height:=300)  
  
    shpStar.TextFrame.TextRange.Text = "Animated shape."  
  
    Set effNew = sldOne.TimeLine.MainSequence.AddEffect(Shape:=shpSt  
        EffectId:=msoAnimEffectPath5PointStar, Level:=msoAnimateText  
        Trigger:=msoAnimTriggerAfterPrevious)  
    With effNew  
        If .TextRangeStart = 0 And .TextRangeLength > 0 Then  
            With .Behaviors.Add(Type:=msoAnimTypeRotation).RotationE  
                .From = 0  
                .To = 360  
            End With  
            .Timing.AutoReverse = msoTrue  
        End If  
    End With  
End Sub
```



TextRangeStart Property

Returns or sets an **Long** that represents the start of a text range. Read-only.

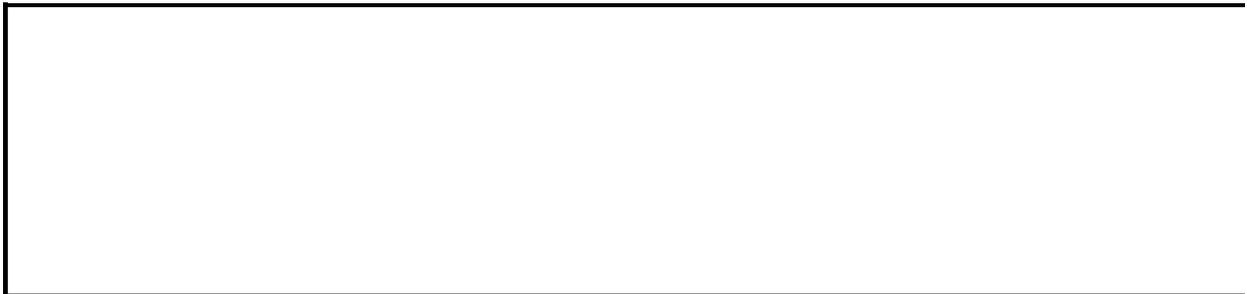
expression.**TextRangeStart**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example adds a shape with text and rotates the shape without rotating the text.

```
Sub SetTextRange()  
    Dim shpStar As Shape  
    Dim sldOne As Slide  
    Dim effNew As Effect  
  
    Set sldOne = ActivePresentation.Slides(1)  
    Set shpStar = sldOne.Shapes.AddShape(Type:=msoShape5pointStar, _  
        Left:=32, Top:=32, Width:=300, Height:=300)  
  
    shpStar.TextFrame.TextRange.Text = "Animated shape."  
  
    Set effNew = sldOne.TimeLine.MainSequence.AddEffect(Shape:=shpSt  
        EffectId:=msoAnimEffectPath5PointStar, Level:=msoAnimateText  
        Trigger:=msoAnimTriggerAfterPrevious)  
    With effNew  
        If .TextRangeStart = 0 And .TextRangeLength > 0 Then  
            With .Behaviors.Add(Type:=msoAnimTypeRotation).RotationE  
                .From = 0  
                .To = 360  
            End With  
            .Timing.AutoReverse = msoTrue  
        End If  
    End With  
End Sub
```



TextShape Property

Returns a [Shape](#) object representing the shape of the text box associated with a diagram node.

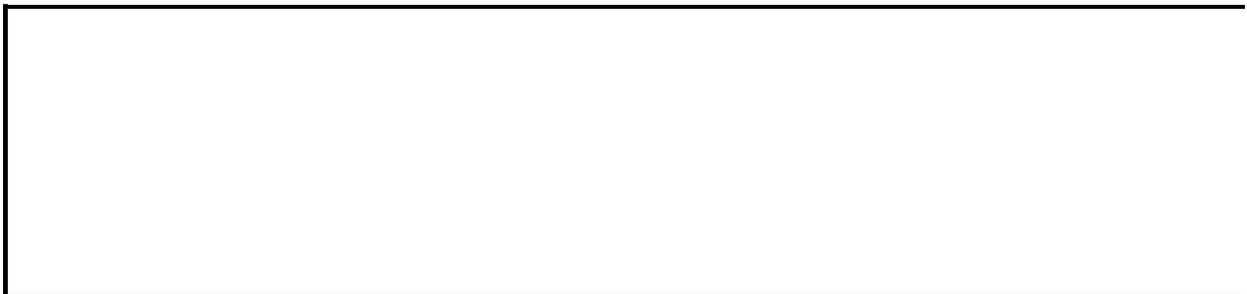
expression.**TextShape**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds child nodes to a parent node, and displays text in the parent node indicating the number of child nodes created.

```
Sub CountChildNodes()  
  
    Dim dgnNode As DiagramNode  
    Dim shpDiagram As Shape  
    Dim intNodes As Integer  
    Dim shpText As Shape  
  
    'Adds diagram and first node to first slide  
    Set shpDiagram = ActivePresentation.Slides(1).Shapes _  
        .AddDiagram(Type:=msoDiagramRadial, Left:=200, Top:=75, _  
            Width:=300, Height:=475)  
    Set dgnNode = shpDiagram.DiagramNode.Children.AddNode  
  
    'Adds three child nodes to first node  
    For intNodes = 1 To 3  
        dgnNode.Children.AddNode  
    Next intNodes  
  
    'Enters node number into each child node  
    For intNodes = 1 To dgnNode.Children.Count  
        Set shpText = shpDiagram.DiagramNode.Children(1) _  
            .Children(intNodes).TextShape  
        shpText.TextFrame.TextRange.Text = CStr(intNodes)  
    Next intNodes  
  
End Sub
```



TextStyles Property

Returns a [TextStyles](#) collection that represents three text styles— title text, body text, and default text— for the specified slide master. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

This example sets the font name and font size for level-one body text on slides in the active presentation.

```
With ActivePresentation.SlideMaster_  
    .TextStyles(ppBodyStyle).Levels(1)  
    With .Font  
        .Name = "arial"  
        .Size = 36  
    End With  
End With
```



TextToDisplay Property

Returns or sets the display text for a hyperlink not associated with a graphic.
Read/write **String**.

Remarks

This property will cause a run-time error if used with a hyperlink that is not associated with a text range. You can use code similar to the following to test whether or not a given hyperlink, represented here by `myHyperlink`, is associated with a text range.

```
If TypeName(myHyperlink.Parent.Parent) = "TextRange" Then  
    strTRtest = "True"  
End If
```

Example

This example creates an associated hyperlink with the text in shape two on slide one. It then sets the display text to "Microsoft Home Page" and sets the hyperlink address to the correct URL.

```
With ActivePresentation.Slides(1).Shapes(2) _  
    .TextFrame.TextRange  
    With .ActionSettings(ppMouseClick)  
        .Action = ppActionHyperlink  
        .Hyperlink.TextToDisplay = "Microsoft Home Page"  
        .Hyperlink.Address = "http://www.microsoft.com/"  
    End With  
End With
```



[Show All](#)

TextUnitEffect Property

 [TextUnitEffect](#) property as it applies to the [AnimationSettings](#) object.

Returns or sets a [PpTextUnitEffect](#) constant that indicates whether the text in the specified shape is animated paragraph by paragraph, word by word, or letter by letter. Read/write.

PpTextUnitEffect can be one of these PpTextUnitEffect constants.

ppAnimateByCharacter

ppAnimateByParagraph

ppAnimateByWord

ppAnimateUnitMixed

expression.**TextUnitEffect**

expression Required. An expression that returns an [AnimationSettings](#) object.

Remarks

For the **TextUnitEffect** property setting to take effect, the [TextLevelEffect](#) property for the specified shape must have a value other than **ppAnimateLevelNone** or **ppAnimateByAllLevels**, and the [Animate](#) property must be set to **True**.

[TextUnitEffect](#) property as it applies to the [EffectInformation](#) object.

Returns an [MsoAnimTextUnitEffect](#) constant that indicates whether the text in the specified shape is animated paragraph by paragraph, word by word, or letter by letter. Read-only.

MsoAnimTextUnitEffect can be one of these MsoAnimTextUnitEffect constants.

msoAnimTextUnitEffectByCharacter
msoAnimTextUnitEffectByParagraph
msoAnimTextUnitEffectByWord
msoAnimTextUnitEffectMixed

expression.**TextUnitEffect**

expression Required. An expression that returns an [EffectInformation](#) object.

Example

 [As it applies to the **AnimationSettings** object.](#)

This example adds a title slide and title text to the active presentation and sets the title to be built letter by letter.

```
With ActivePresentation.Slides.Add(Index:=1, _  
    Layout:=ppLayoutTitleOnly).Shapes(1)  
    .TextFrame.TextRange.Text = "Sample title"  
    With .AnimationSettings  
        .Animate = True  
        .TextLevelEffect = ppAnimateByFirstLevel  
        .TextUnitEffect = ppAnimateByCharacter  
        .EntryEffect = ppEffectFlyFromLeft  
    End With  
End With
```



TextureName Property

Returns the name of the custom texture file for the specified fill. Read-only **String**.

This property is read-only. Use the [UserTextured](#) method to set the texture file for the fill.

Example

This example adds an oval to myDocument. If shape one on myDocument has a user-defined textured fill, the new oval will have the same fill as shape one. If shape one has any other type of fill, the new oval will have a green marble fill.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    Set newFill = .AddShape(msoShapeOval, 0, 0, 200, 90).Fill
    With .Item(1).Fill
        If .Type = msoFillTextured And _
            .TextureType = msoTextureUserDefined Then
            newFill.UserTextured .TextureName
        Else
            newFill.PresetTextured msoTextureGreenMarble
        End If
    End With
End With
```



[Show All](#)

TextureType Property

Returns the texture type for the specified fill. Read-only [MsoTextureType](#). Use the [PresetTextured](#) or [UserTextured](#) method to set the texture type for the fill.

MsoTextureType can be one of these MsoTextureType constants.

msoTexturePreset

msoTextureTypeMixed

msoTextureUserDefined

expression.**TextureType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example changes the fill to canvas for all shapes on myDocument that have a custom textured fill.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
  With s.Fill
    If .TextureType = msoTextureUserDefined Then
      .PresetTextured msoTextureCanvas
    End If
  End With
Next
```



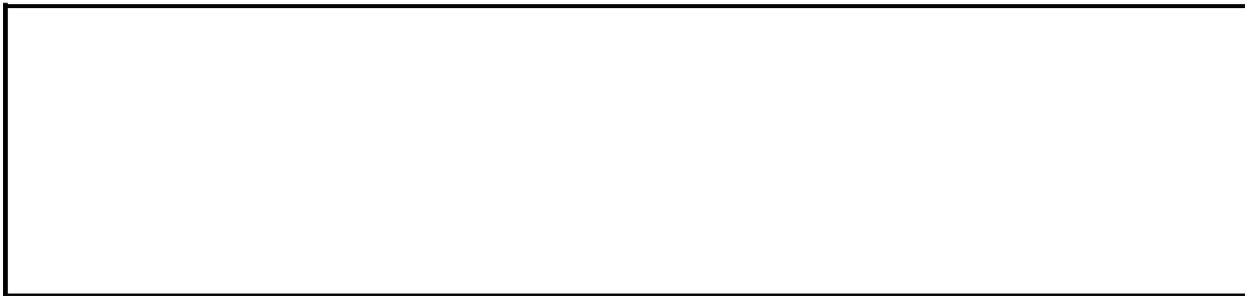
ThreeD Property

Returns a [ThreeDFormat](#) object that contains 3-D – effect formatting properties for the specified shape. Read-only.

Example

This example sets the depth, extrusion color, extrusion direction, and lighting direction for the 3-D effects applied to shape one on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .Depth = 50
    'RGB value for purple
    .ExtrusionColor.RGB = RGB(255, 100, 255)
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```



Time Property

Sets or returns a **Single** that represents the time at a given animation point.
Read/write.

expression.**Time**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The value of the **Time** property can be any floating-point value between 0 and 1, representing a percentage of the entire timeline from 0% to 100%. For example, a value of 0.2 would correspond to a point in time at 20% of the entire timeline duration from left to right.

Example

This example inserts three fill color animation points in the main sequence animation timeline on the first slide.

```
Sub BuildTimeLine()  
  
    Dim shpFirst As Shape  
    Dim effMain As Effect  
    Dim tmlMain As TimeLine  
    Dim aniBhvr As AnimationBehavior  
    Dim aniPoint As AnimationPoint  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlMain = ActivePresentation.Slides(1).TimeLine  
    Set effMain = tmlMain.MainSequence.AddEffect(Shape:=shpFirst, _  
        EffectId:=msoAnimEffectBlinds)  
    Set aniBhvr = tmlMain.MainSequence(1).Behaviors.Add _  
        (Type:=msoAnimTypeProperty)  
  
    With aniBhvr.PropertyEffect  
        .Property = msoAnimShapeFillColor  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 0.2  
        aniPoint.Value = RGB(0, 0, 0)  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 0.5  
        aniPoint.Value = RGB(0, 255, 0)  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 1  
        aniPoint.Value = RGB(0, 255, 255)  
    End With  
End Sub
```



TimeLine Property

Returns a [TimeLine](#) object representing the animation timeline for the slide.

expression.**TimeLine**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a bouncing animation to the first shape on the first slide.

```
Sub NewTimeLineEffect()  
    Dim sldFirst As Slide  
    Dim shpFirst As Shape  
  
    Set sldFirst = ActivePresentation.Slides(1)  
    Set shpFirst = sldFirst.Shapes(1)  
  
    sldFirst.TimeLine.MainSequence.AddEffect _  
        Shape:=shpFirst, EffectId:=msoAnimEffectBounce  
  
End Sub
```



Timing Property

Returns a [Timing](#) object that represents the timing properties for an animation sequence.

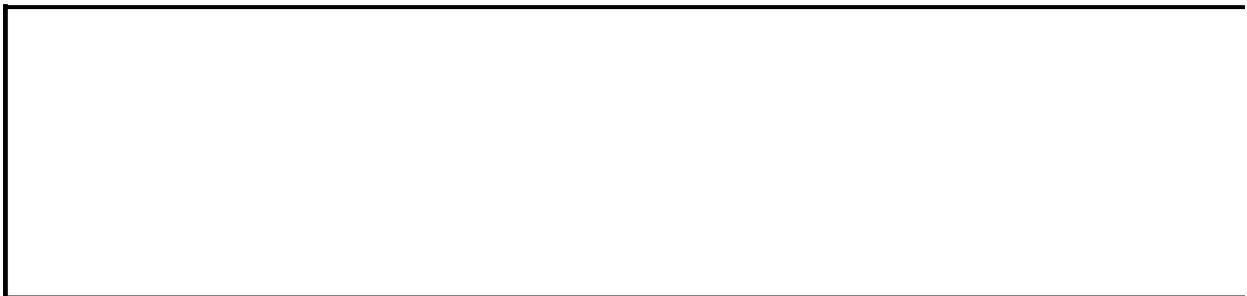
expression.**Timing**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example sets the duration of the first animation sequence on the first slide.

```
Sub SetTiming()  
    ActivePresentation.Slides(1).TimeLine _  
        .MainSequence(1).Timing.Duration = 1  
End Sub
```



TintAndShade Property

Returns a **Single** that represents the lightening or darkening of the the color of a specified shape. Read/write.

expression.**TintAndShade**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

You can enter a value from -1 (darkest) to 1 (lightest) for the **TintAndShade** property, 0 (zero) being neutral.

Example

This example creates a new shape in the active document, sets the fill color, and lightens the color shade.

```
Sub PrinterPlate()  
    Dim shpHeart As Shape  
  
    Set shpHeart = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeHeart, Left:=150, _  
        Top:=150, Width:=250, Height:=250)  
  
    With shpHeart.Fill.ForeColor  
        .CMYK = 16111872  
        .TintAndShade = 0.3  
        .OverPrint = msoTrue  
        .Ink(Index:=1) = 0  
        .Ink(Index:=2) = 1  
        .Ink(Index:=3) = 1  
        .Ink(Index:=4) = 0  
    End With  
End Sub
```



Title Property

Returns a [Shape](#) object that represents the slide title. Read-only.

Remarks

You can also use the [Item](#) method of the [Shapes](#) or [Placeholders](#) collection to return the slide title.

Example

This example sets the title text on myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.Title.TextFrame.TextRange.Text = "Welcome!"
```



TitleMaster Property

Returns a [Master](#) object that represents the title master for the specified presentation. If the presentation doesn't have a title master, an error occurs.

expression.**TitleMaster**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

Use the [AddTitleMaster](#) method to add a title master to a presentation.

Example

If the active presentation has a title master, this example sets the footer text for the title master.

```
With Application.ActivePresentation
  If .HasTitleMaster Then
    .TitleMaster.HeadersFooters.Footer.Text = "Introduction"
  End If
End With
```



[Show All](#)

To Property

 [To property as it applies to the **ColorEffect** object.](#)

Sets or returns a **ColorFormat** object that represents the RGB color value of an animation behavior. Read/write.

expression.**To**

expression Required. An expression that returns a [ColorEffect](#) object.

Remarks

Use this property in conjunction with the **From** property to transition from one color to another.

[To property as it applies to the **RotationEffect** object.](#)

Sets or returns a **Single** that represents the ending rotation of an object in degrees, specified relative to the screen (for example, 90 degrees is completely horizontal). Read/write.

expression.**To**

expression Required. An expression that returns a [RotationEffect](#) object.

Remarks

Use this property in conjunction with the **From** property to transition from one rotation angle to another.

The default value is **Empty** in which case the current position of the object is used.

[To property as it applies to the **PropertyEffect** object.](#)

Sets or returns a **Variant** that represents the ending value of an object's property.
Read/write.

expression.**To**

expression Required. An expression that returns a [PropertyEffect](#) object.

Remarks

The default value is **Empty**, in which case the current position of the object is used.

[To property as it applies to the **SetEffect** object.](#)

Sets or returns a **Variant** that represents the value or ending value of the **SetEffect** object's **Type** property. Read/write.

expression.**To**

expression Required. An expression that returns a [SetEffect](#) object.

Remarks

Do not confuse this property with the [ToX](#) or [ToY](#) properties of the [ScaleEffect](#) and [MotionEffect](#) objects, which are only used for scaling or motion effects.

Example

 [As it applies to the **ColorEffect** object.](#)

The following example adds a color effect and changes its color from a light bluish green to yellow.

```
Sub AddAndChangeColorEffect()  
    Dim effBlinds As Effect  
    Dim tmlTiming As TimeLine  
    Dim shpRectangle As Shape  
    Dim animColor As AnimationBehavior  
    Dim clrEffect As ColorEffect  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set tmlTiming = ActivePresentation.Slides(1).TimeLine  
    Set effBlinds = tmlTiming.MainSequence.AddEffect(Shape:=shpRecta  
        effectId:=msoAnimEffectBlinds)  
    Set animColor = tmlTiming.MainSequence(1).Behaviors _  
        .Add(Type:=msoAnimTypeColor)  
    Set clrEffect = animColor.ColorEffect  
  
    clrEffect.From.RGB = RGB(Red:=255, Green:=255, Blue:=0)  
    clrEffect.To.RGB = RGB(Red:=0, Green:=255, Blue:=255)  
End Sub
```

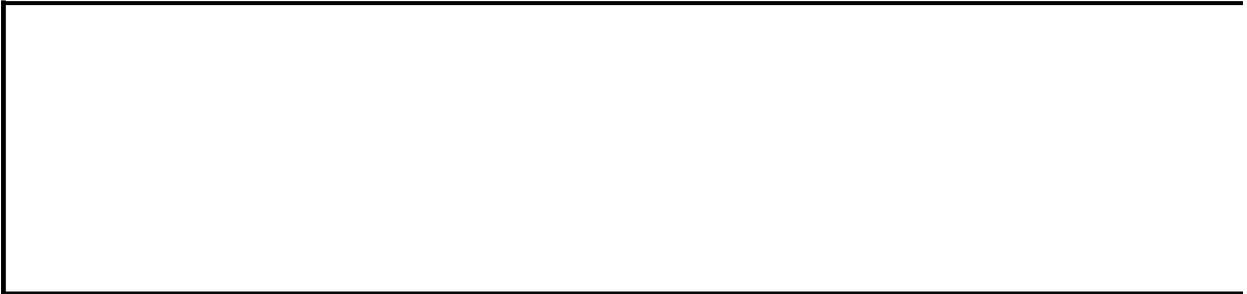
 [As it applies to the **RotationEffect** object.](#)

The following example adds a rotation effect and immediately changes its rotation angle from 90 degrees to 270 degrees.

```
Sub AddAndChangeRotationEffect()  
    Dim effBlinds As Effect  
    Dim tmlTiming As TimeLine  
    Dim shpRectangle As Shape  
    Dim animColor As AnimationBehavior  
    Dim rtnEffect As RotationEffect  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlTiming = ActivePresentation.Slides(1).TimeLine  
    Set effBlinds = tmlTiming.MainSequence.AddEffect(Shape:=shpRecta
```

```
        effectId:=msoAnimEffectBlinds)
Set animColor = tmlTiming.MainSequence(1).Behaviors.Add(Type:=ms
Set rtnEffect = animColor.RotationEffect

rtnEffect.From = 90
rtnEffect.To = 270
End Sub
```



[Show All](#)

Top Property

 [Top property as it applies to the **Application**, **DocumentWindow**, **Shape**, **ShapeRange**, and **SlideShowWindow** objects.](#)

Application, **DocumentWindow** and **SlideShowWindow** objects: Returns or sets a **Single** that represents the distance in points from the top edge of the document, application, and slide show window to the top edge of the application window's client area. Setting this property to a very large positive or negative value may position the window completely off the desktop. Read/write.

Shape object: Returns or sets a **Single** that represents the distance from the top edge of the shape's bounding box to the top edge of the document. Read/write.

ShapeRange object: Returns or sets a **Single** that represents the distance from the top edge of the topmost shape in the shape range to the top edge of the document. Read/write.

expression.Top

expression Required. An expression that returns one of the above objects.

 [Top property as it applies to the **Comment** object.](#)

Comment object: Returns a **Single** that represents the distance in points from the left edge of the comment to the left edge of the slide. Read-only.

expression.Top

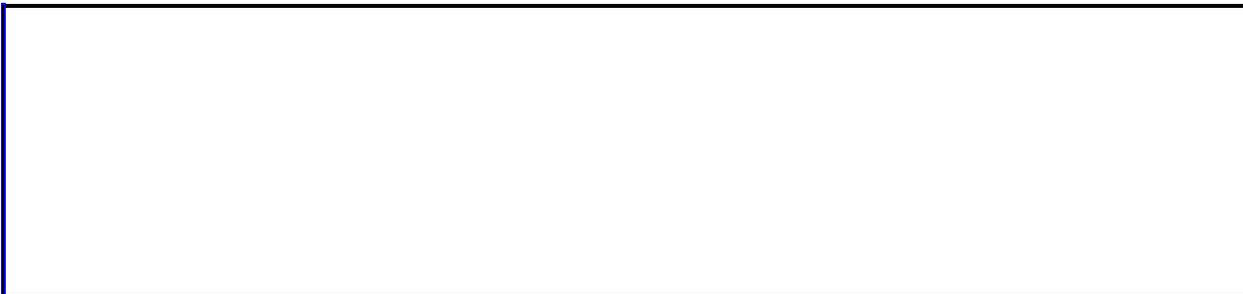
expression Required. An expression that returns a **Comment** object.

Example

 As it applies to the [Application](#), [DocumentWindow](#), [Shape](#), [ShapeRange](#), and [SlideShowWindow](#) objects.

This example arranges windows one and two horizontally; in other words, each window occupies half the available vertical space and all the available horizontal space in the application window's client area. For this example to work, there must be only two document windows open.

```
Windows.Arrange ppArrangeTiled
sngHeight = Windows(1).Height           ' available height
sngWidth = Windows(1).Width + Windows(2).Width ' available width
With Windows(1)
    .Width = sngWidth
    .Height = sngHeight / 2
    .Left = 0
End With
With Windows(2)
    .Width = sngWidth
    .Height = sngHeight / 2
    .Top = sngHeight / 2
    .Left = 0
End With
```



ToX Property

Sets or returns a **Single** that represents the ending width or horizontal position of a [ScaleEffect](#) or [MotionEffect](#) object, respectively, specified as a percent of the screen width. Read/write.

expression.**ToX**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The default value of this property is **Empty**, in which case the current position of the object is used.

Use this property in conjunction with the [FromX](#) property to resize or jump from one position to another.

Do not confuse this property with the [To](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set or change colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds an animation path and sets the starting and ending horizontal and vertical positions.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animMotion As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets effect and animation properties  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectCusto  
    Set animMotion = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Sets starting and ending horizontal and vertical positions  
    With animMotion.MotionEffect  
        .FromX = 0  
        .FromY = 0  
        .ToX = 50  
        .ToY = 50  
    End With  
  
End Sub
```



ToY Property

Returns or sets a **Single** that represents the ending height or vertical position of a [ScaleEffect](#) or [MotionEffect](#) object, respectively, specified as a percentage of the screen width. Read/write.

expression.**ToY**

expression Required. An expression that returns one of the objects in the Applies To list.

Remarks

The default value of this property is **Empty**, in which case the current position of the object is used.

Use this property in conjunction with the [FromY](#) property to resize or jump from one position to another.

Do not confuse this property with the [To](#) property of the [ColorEffect](#), [RotationEffect](#), or [PropertyEffect](#) objects, which is used to set or change colors, rotations, or other properties of an animation behavior, respectively.

Example

The following example adds an animation path and sets the starting and ending horizontal and vertical positions.

```
Sub AddMotionPath()  
  
    Dim effCustom As Effect  
    Dim animMotion As AnimationBehavior  
    Dim shpRectangle As Shape  
  
    'Adds shape and sets effect and animation properties  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
            Top:=100, Width:=50, Height:=50)  
    Set effCustom = ActivePresentation.Slides(1).TimeLine.MainSequen  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectCusto  
    Set animMotion = effCustom.Behaviors.Add(msoAnimTypeMotion)  
  
    'Sets starting and ending horizontal and vertical positions  
    With animMotion.MotionEffect  
        .FromX = 0  
        .FromY = 0  
        .ToX = 50  
        .ToY = 50  
    End With  
  
End Sub
```



Tracking Property

Returns or sets the ratio of the horizontal space allotted to each character in the specified WordArt to the width of the character. Can be a value from 0 (zero) through 5. (Large values for this property specify ample space between characters; values less than 1 can produce character overlap.) Read/write **Single**.

The following table gives the values of the **Tracking** property that correspond to the settings available in the user interface.

User interface setting	Equivalent Tracking property value
Very Tight	0.8
Tight	0.9
Normal	1.0
Loose	1.2
Very Loose	1.5

Example

This example adds WordArt that contains the text "Test" to myDocument and specifies that the characters be very tightly spaced.

```
Set myDocument = ActivePresentation.Slides(1)
Set newWordArt = myDocument.Shapes.AddTextEffect _
    (PresetTextEffect:=msoTextEffect1, Text:="Test", _
    FontName:="Arial Black", FontSize:=36, _
    FontBold:=False, FontItalic:=False, Left:=100, Top:=100)
newWordArt.TextEffect.Tracking = 0.8
```



Transparency Property

Returns or sets the degree of transparency of the specified fill, shadow, or line as a value between 0.0 (opaque) and 1.0 (clear). Read/write **Single**.

Remarks

The value of this property affects the appearance of solid-colored fills and lines only; it has no effect on the appearance of patterned lines or patterned, gradient, picture, or textured fills.

Example

This example sets the shadow for shape three on myDocument to semitransparent red. If the shape doesn't already have a shadow, this example adds one to it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Shadow
    .Visible = True
    .ForeColor.RGB = RGB(255, 0, 0)
    .Transparency = 0.5
End With
```



[Show All](#)

TransparencyColor Property

Returns or sets the transparent color for the specified picture as a [red-green-blue \(RGB\) value](#). For this property to take effect, the [TransparentBackground](#) property must be set to **True**. Applies to bitmaps only. Read/write **Long**.

Remarks

If you want to be able to see through the transparent parts of the picture all the way to the objects behind the picture, you must set the **Visible** property of the picture's **FillFormat** object to **False**. If your picture has a transparent color and the **Visible** property of the picture's **FillFormat** object is set to **True**, the picture's fill will be visible through the transparent color, but objects behind the picture will be obscured.

Example

This example sets the color that has the RGB value returned by the function RGB(0, 0, 255) as the transparent color for shape one on myDocument. For the example to work, shape one must be a bitmap.

```
blueScreen = RGB(0, 0, 255)
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1)
    With .PictureFormat
        .TransparentBackground = True
        .TransparencyColor = blueScreen
    End With
    .Fill.Visible = False
End With
```



[Show All](#)

TransparentBackground Property

Determines whether parts of the picture that are the color defined as the transparent color appear transparent. Read/write [MsoTriState](#). Applies to bitmaps only.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Parts of the picture that are the color defined as the transparent color appear transparent.

Remarks

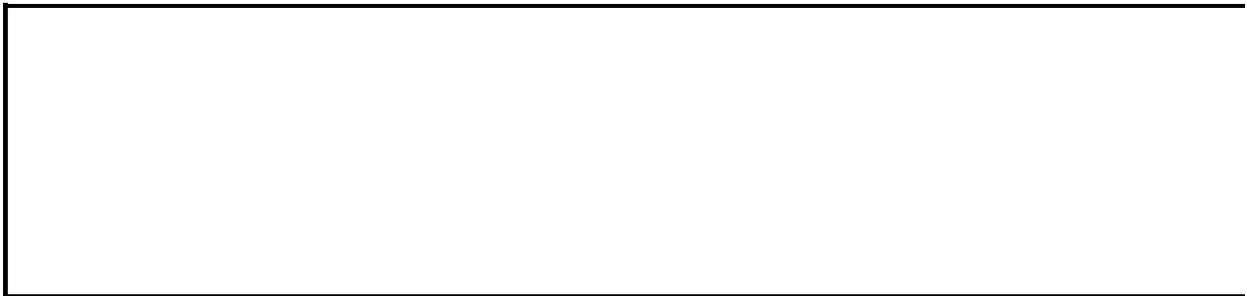
Use the [TransparencyColor](#) property to set the transparent color.

If you want to be able to see through the transparent parts of the picture all the way to the objects behind the picture, you must set the **Visible** property of the picture's **FillFormat** object to **msoFalse**. If your picture has a transparent color and the **Visible** property of the picture's **FillFormat** object is set to **msoTrue**, the picture's fill will be visible through the transparent color, but objects behind the picture will be obscured.

Example

This example sets the color that has the RGB value returned by the function RGB(0, 24, 240) as the transparent color for shape one on myDocument. For the example to work, shape one must be a bitmap.

```
blueScreen = RGB(0, 0, 255)
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1)
    With .PictureFormat
        .TransparentBackground = msoTrue
        .TransparencyColor = blueScreen
    End With
    .Fill.Visible = msoFalse
End With
```



TriggerDelayTime Property

Sets or returns a **Single** that represents the delay, in seconds, from when an animation trigger is enabled. Read/write.

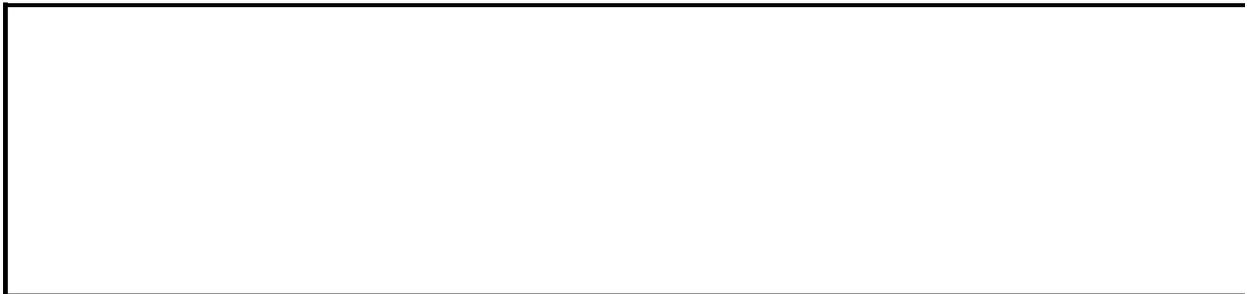
expression.**TriggerDelayTime**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to a slide, adds an animation to the shape, and instructs the shape to begin the animation three seconds after it is clicked.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    With effDiamond.Timing  
        .Duration = 5  
        .TriggerShape = shpRectangle  
        .TriggerType = msoAnimTriggerOnShapeClick  
        .TriggerDelayTime = 3  
    End With  
  
End Sub
```



TriggerShape Property

Sets or returns a **Shape** object that represents the shape associated with an animation trigger. Read/write.

expression.**TriggerShape**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds two shapes to a slide, adds an animation to a shape, and begins the animation when the other shape is clicked.

```
Sub AddShapeSetTiming()  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
    Set shpOval = _  
        ActivePresentation.Slides(1).Shapes. _  
        AddShape(Type:=msoShapeOval, Left:=400, Top:=100, Width:=100,  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes. _  
        AddShape(Type:=msoShapeRectangle, Left:=100, Top:=100, Width:=  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine. _  
        InteractiveSequences.Add().AddEffect(Shape:=shpRectangle, _  
        effectId:=msoAnimEffectPathDiamond, trigger:=msoAnimTriggerOnS  
  
    With effDiamond.Timing  
        .Duration = 5  
        .TriggerShape = shpOval  
    End With  
End Sub
```



[Show All](#)

TriggerType Property

Sets or returns an [MsoAnimTriggerType](#) constant that represents the trigger that starts an animation. Read/write.

MsoAnimTriggerType can be one of these MsoAnimTriggerType constants.

msoAnimTriggerAfterPrevious

msoAnimTriggerMixed

msoAnimTriggerNone

msoAnimTriggerOnPageClick Default.

msoAnimTriggerOnShapeClick

msoAnimTriggerWithPrevious

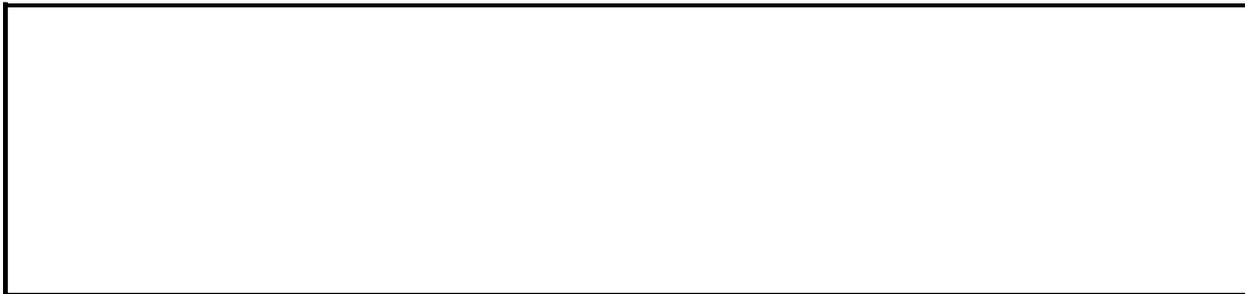
expression.**TriggerType**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

The following example adds a shape to a slide, adds an animation to the shape, and instructs the shape to begin the animation three seconds after it is clicked.

```
Sub AddShapeSetTiming()  
  
    Dim effDiamond As Effect  
    Dim shpRectangle As Shape  
  
    Set shpRectangle = ActivePresentation.Slides(1).Shapes _  
        .AddShape(Type:=msoShapeRectangle, Left:=100, _  
        Top:=100, Width:=50, Height:=50)  
    Set effDiamond = ActivePresentation.Slides(1).TimeLine.MainSeque  
        .AddEffect(Shape:=shpRectangle, effectId:=msoAnimEffectPathD  
  
    With effDiamond.Timing  
        .Duration = 5  
        .TriggerType = msoAnimTriggerOnShapeClick  
        .TriggerDelayTime = 3  
    End With  
  
End Sub
```



[Show All](#)

Type Property

 [Type property as it applies to the **AnimationBehavior** object.](#)

Returns or sets an **MsoAnimType** constant that represents the type of animation.
Read/write.

MsoAnimType can be one of these MsoAnimType constants.

MsoAnimTypeColor

MsoAnimTypeMixed

MsoAnimTypeMotion

MsoAnimTypeNone

MsoAnimTypeProperty

MsoAnimTypeRoatation

MsoAnimTypeScale

MsoAnimTypeTransition

expression.Type

expression Required. An expression that returns an **AnimationBehavior** object.

 [Type property as it applies to the **BulletFormat** object.](#)

Returns or sets a **PpBulletType** constant that represents the type of bullet.
Read/write.

PpBulletType can be one of these PpBulletType constants.

ppBulletMixed

ppBulletNone

ppBulletNumbered

ppBulletPicture

ppBulletUnnumbered

expression.Type

expression Required. An expression that returns a [BulletFormat](#) object.

[Type property as it applies to the CalloutFormat object.](#)

Returns or sets an [MsoCalloutType](#) constant that represents the type of callout. Read/write.

MsoCalloutType can be one of these MsoCalloutType constants.

msoCalloutFour

msoCalloutMixed

msoCalloutOne

msoCalloutThree

msoCalloutTwo

expression.Type

expression Required. An expression that returns a [CalloutFormat](#) object.

[Type property as it applies to the ColorFormat object.](#)

Returns the [MsoColorType](#) constant that represents the type of color. Read-only.

MsoColorType can be one of these MsoColorType constants.

msoColorTypeCMS

msoColorTypeCMYK

msoColorTypeInk

msoColorTypeMixed

msoColorTypeRGB

msoColorTypeScheme

expression.Type

expression Required. An expression that returns one of the above objects.

[Type property as it applies to the ConnectorFormat object.](#)

Returns or sets an [MsoConnectorType](#) constant that represents the type of connector. Read/write.

MsoConnectorType can be one of these MsoConnectorType constants.

msoConnectorCurve

msoConnectorElbow

msoConnectorStraight

msoConnectorTypeMixed

expression.Type

expression Required. An expression that returns a [ConnectorFormat](#) object.

[Type](#) property as it applies to the [Diagram](#) object.

Returns an [MsoDiagramType](#) constant that represents the type of diagram. Read-only.

MsoDiagramType can be one of these MsoDiagramType constants.

msoDiagramCycle

msoDiagramMixed

msoDiagramOrgChart

msoDiagramPyramid

msoDiagramRadial

msoDiagramTarget

msoDiagramVenn

expression.Type

expression Required. An expression that returns a [Diagram](#) object.

[Type](#) property as it applies to the [FillFormat](#) object.

Returns an [MsoFillType](#) constant that represent the type of fill. Read-only.

MsoFillType can be one of these MsoFillType constants.

msoFillBackground

msoFillGradient
msoFillMixed
msoFillPatterned
msoFillPicture
msoFillSolid
msoFillTextured

expression.Type

expression Required. An expression that returns a [FillFormat](#) object.

[Type property as it applies to the **Hyperlink** object.](#)

Returns an [MsoHyperlinkType](#) constant that represents the type of hyperlink.
Read-only.

MsoHyperlinkType can be one of these MsoHyperlinkType constants.

msoHyperlinkInlineShape For use in Microsoft Word only.

msoHyperlinkRange

msoHyperlinkShape

expression.Type

expression Required. An expression that returns a [Hyperlink](#) object.

[Type property as it applies to the **PlaceholderFormat** object.](#)

Returns a [PpPlaceholderType](#) constant that represents the type of placeholder.
Read-only.

PpPlaceholderType can be one of these PpPlaceholderType constants.

ppPlaceholderBitmap

ppPlaceholderBody

ppPlaceholderCenterTitle

ppPlaceholderChart

ppPlaceholderDate

ppPlaceholderFooter
ppPlaceholderHeader
ppPlaceholderMediaClip
ppPlaceholderMixed
ppPlaceholderObject
ppPlaceholderOrgChart
ppPlaceholderSlideNumber
ppPlaceholderSubtitle
ppPlaceholderTable
ppPlaceholderTitle
ppPlaceholderVerticalBody
ppPlaceholderVerticalTitle

expression.Type

expression Required. An expression that returns a [PlaceholderFormat](#) object.

[Type property as it applies to the Selection object.](#)

Returns a [PpSelectionType](#) constant that represents the type of objects in a selection. Read-only.

PpSelectionType can be one of these PpSelectionType constants.

ppSelectionNone
ppSelectionShapes
ppSelectionSlides
ppSelectionText

expression.Type

expression Required. An expression that returns a [Selection](#) object.

[Type property as it applies to the ShadowFormat object.](#)

Returns or sets an [MsoShadowType](#) constant that represents the type of shadow. Read/write.

MsoShadowType can be one of these MsoShadowType constants.

msoShadow1

msoShadow10

msoShadow11

msoShadow12

msoShadow13

msoShadow14

msoShadow15

msoShadow16

msoShadow17

msoShadow18

msoShadow19

msoShadow2

msoShadow20

msoShadow3

msoShadow4

msoShadow5

msoShadow6

msoShadow7

msoShadow8

msoShadow9

msoShadowMixed

expression.Type

expression Required. An expression that returns a [ShadowFormat](#) object.

 [Type property as it applies to the Shape and ShapeRange objects.](#)

Returns an [MsoShapeType](#) constant that represents the type of shape or shapes in a range of shapes. Read-only.

MsoShapeType can be one of these MsoShapeType constants.

msoAutoShape

msoCallout

msoCanvas
msoChart
msoComment
msoDiagram
msoEmbeddedOLEObject
msoFormControl
msoFreeform
msoGroup
msoLine
msoLinkedOLEObject
msoLinkedPicture
msoMedia
msoOLEControlObject
msoPicture
msoPlaceholder
msoScriptAnchor
msoShapeTypeMixed
msoTable
msoTextBox
msoTextEffect

expression.**Type**

expression Required. An expression that returns one of the above objects.

[Type property as it applies to the **SoundEffect** object.](#)

Returns or sets a **PpSoundEffectType** constant that represents the type of sound effect. Read/write.

PpSoundEffectType can be one of these PpSoundEffectType constants.

ppSoundEffectsMixed
ppSoundFile
ppSoundNone
ppSoundStopPrevious

expression.Type

expression Required. An expression that returns a [SoundEffect](#) object.

[Type property as it applies to the TabStop object.](#)

Returns or sets a [PpTabStopType](#) constant that represents the formatting of a tab stop. Read/write.

PpTabStopType can be one of these PpTabStopType constants.

ppTabStopCenter

ppTabStopDecimal

ppTabStopLeft

ppTabStopMixed

ppTabStopRight

expression.Type

expression Required. An expression that returns a [TabStop](#) object.

[Type property as it applies to the View object.](#)

Returns a [PpViewType](#) constant that represents the type of view. Read-only.

PpViewType can be one of these PpViewType constants.

ppViewHandoutMaster

ppViewMasterThumbnails

ppViewNormal

ppViewNotesMaster

ppViewNotesPage

ppViewOutline

ppViewPrintPreview

ppViewSlide

ppViewSlideMaster

ppViewSlideSorter

ppViewThumbnails
ppViewTitleMaster

expression.Type

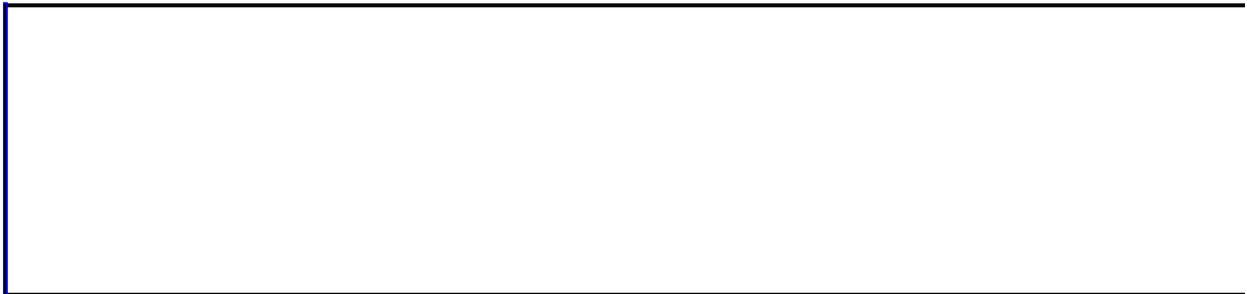
expression Required. An expression that returns a [View](#) object.

Example

 [As it applies to the **Shape** object.](#)

This example loops through all the shapes on all the slides in the active presentation and sets all linked Microsoft Excel worksheets to be updated manually.

```
For Each sld In ActivePresentation.Slides
  For Each sh In sld.Shapes
    If sh.Type = msoLinkedOLEObject Then
      If sh.OLEFormat.ProgID = "Excel.Sheet" Then
        sh.LinkFormat.AutoUpdate = ppUpdateOptionManual
      End If
    End If
  Next
Next
Next
```



[Show All](#)

Underline Property

Determines whether the specified text (for the **Font** object) or the font style (for the **FontInfo** object) is underlined. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The specified text (or font style) isn't underlined.

msoTriStateMixed Some characters are underlined (for the specified text) and some aren't.

msoTriStateToggle

msoTrue The specified text (or font style) is underlined.

expression.**Underline**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the formatting for the text in shape two on slide one in the active presentation.

```
With Application.ActivePresentation.Slides(1).Shapes(2)
    With .TextFrame.TextRange.Font
        .Size = 32
        .Name = "Palatino"
        .Underline = msoTrue
    End With
End With
```



[Show All](#)

UpdateLinksOnSave Property

Determines whether hyperlinks and paths to all supporting files are automatically updated before you save or publish the presentation as a Web page. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Hyperlinks and paths to all supporting files are not automatically updated before you save or publish the presentation as a Web page.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Hyperlinks and paths to all supporting files are automatically updated before you save or publish the presentation as a Web page, ensuring that the links are up-to-date at the time the presentation is saved.

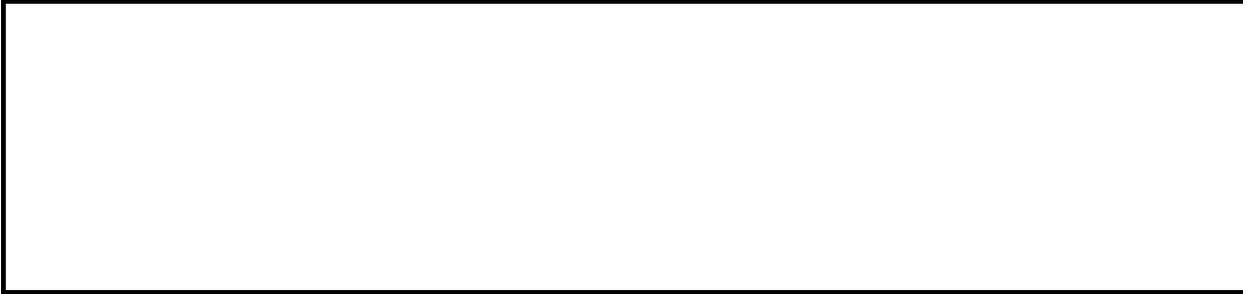
Remarks

You should set this property to **False** if the location where the presentation is saved is different from the final location on the Web server and the supporting files are not available at the first location.

Example

This example specifies that links are not updated before the presentation is saved.

```
Application.DefaultWebOptions.UpdateLinksOnSave = msoFalse
```



[Show All](#)

UseFormat Property

Determines whether the date and time object contains automatically updated information. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The date and time object is a fixed string.

msoTriStateMixed

msoTriStateToggle

msoTrue The date and time object contains automatically updated information.

Remarks

This property applies only to a [HeaderFooter](#) object that represents a date and time (returned by the [DateAndTime](#) property). Set the **UseFormat** property of a date and time **HeaderFooter** object to **True** when you want to set or return the date and time format by using the [Format](#) property. Set the **UseFormat** property to **msoFalse** when you want to set or return the text string for the fixed date and time.

Example

This example sets the date and time for the slide master of the active presentation to be updated automatically and then it sets the date and time format to show hours, minutes, and seconds.

```
Set myPres = Application.ActivePresentation
With myPres.SlideMaster.HeadersFooters.DateAndTime
    .UseFormat = msoTrue
    .Format = ppDateTimeHmss
End With
```



[Show All](#)

UseLongFileNames Property

Determines whether long file names are used. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Long file names are not used and the DOS file name format (8.3) is used.

msoTriStateMixed

msoTriStateToggle

msoTrue Default. Long file names are used when you save or publish a complete or partial presentation as a Web page.

Remarks

If you don't use long file names and your presentation has supporting files, Microsoft PowerPoint automatically organizes those files in a separate folder. Otherwise, use the [**OrganizeInFolder**](#) property to determine whether supporting files are organized in a separate folder.

Example

This example disallows the use of long file names as the global default for the application.

```
Application.DefaultWebOptions.UseLongFileNames = msoFalse
```



[Show All](#)

UseTextColor Property

Determines whether the specified bullets are set to the color of the first text character in the paragraph. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The specified bullets are set to any other color.

msoTriStateMixed

msoTriStateToggle

msoTrue The specified bullets are set to the color of the first text character in the paragraph.

Remarks

You cannot explicitly set this property to **msoFalse**. Setting the bullet format color (using the [Color](#) property of the **Font** object) sets this property to **msoFalse**. When **UseTextColor** is **msoFalse**, you can set it to **msoTrue** to reset the bullet format to the default color.

Example

This example resets bullets in shape two on slide one in the active presentation to their default character, font, and color.

```
With ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat.Bullet
    .RelativeSize = 1
    .UseTextColor = msoTrue
    .UseTextFont = msoTrue
    .Character = 8226
  End With
End With
```



[Show All](#)

UseTextFont Property

Determines whether the specified bullets are set to the font of the first text character in the paragraph. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse The specified bullets are set to a custom font.

msoTriStateMixed

msoTriStateToggle

msoTrue The specified bullets are set to the font of the first text character in the paragraph.

Remarks

You cannot explicitly set this property to **msoFalse**. Setting the bullet format font (using the [Name](#) property of the **Font** object) sets this property to **msoFalse**. When **UseTextFont** is **msoFalse**, you can set it to **msoTrue** to reset the bullet format to the default font.

Example

This example resets bullets in shape two on slide one in the active presentation to their default character, font, and color.

```
With ActivePresentation.Slides(1).Shapes(2)
  With .TextFrame.TextRange.ParagraphFormat.Bullet
    .RelativeSize = 1
    .UseTextColor = msoTrue
    .UseTextFont = msoTrue
    .Character = 8226
  End With
End With
```



Value Property

Sets or returns a **Variant** that represents the value of a property for an animation point.

expression.**Value**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example inserts three fill color animation points in the main sequence animation timeline on the first slide.

```
Sub BuildTimeLine()  
  
    Dim shpFirst As Shape  
    Dim effMain As Effect  
    Dim tmlMain As TimeLine  
    Dim aniBhvr As AnimationBehavior  
    Dim aniPoint As AnimationPoint  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set tmlMain = ActivePresentation.Slides(1).TimeLine  
    Set effMain = tmlMain.MainSequence.AddEffect(Shape:=shpFirst, _  
        EffectId:=msoAnimEffectBlinds)  
    Set aniBhvr = tmlMain.MainSequence(1).Behaviors.Add _  
        (Type:=msoAnimTypeProperty)  
  
    With aniBhvr.PropertyEffect  
        .Property = msoAnimShapeFillColor  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 0.2  
        aniPoint.Value = RGB(0, 0, 0)  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 0.5  
        aniPoint.Value = RGB(0, 255, 0)  
        Set aniPoint = .Points.Add  
        aniPoint.Time = 1  
        aniPoint.Value = RGB(0, 255, 255)  
    End With  
End Sub
```



[Show All](#)

VBA Signed Property

Determines whether the Visual Basic for Applications (VBA) project for the specified document has been digitally signed. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The VBA project for the specified document has been digitally signed.

Example

This example loads a presentation called MyPres.ppt and tests to see whether or not it has a digital signature. If there's no digital signature, the code displays a warning message.

```
Presentations.Open FileName:="c:\My Documents\MyPres.ppt", _  
    ReadOnly:=msoFalse, WithWindow:=msoTrue  
With ActivePresentation  
    If .VBA Signed = msoFalse And _  
        .VBProject.VBComponents.Count > 0 Then  
        MsgBox "Warning! The Visual Basic project for" _  
            & vbCrLf & "this presentation has not" _  
            & vbCrLf & " been digitally signed." _  
            , vbCritical, "Digital Signature Warning"  
    End If  
End With
```



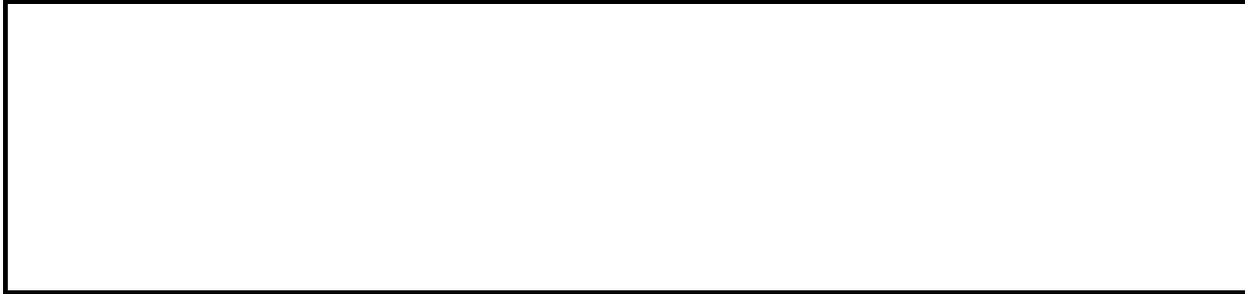
VBE Property

Returns a **VBE** object that represents the Visual Basic Editor. Read-only.

Example

This example sets the name of the active project in the Visual Basic Editor.

```
Application.VBE.ActiveVBProject.Name = "TestProject"
```



VBProject Property

Returns a **VBProject** object that represents the individual Visual Basic project for the presentation. Read-only.

Example

This example changes the name of the Visual Basic project for the active presentation.

```
ActivePresentation.VBProject.Name = "TestProject"
```



Version Property

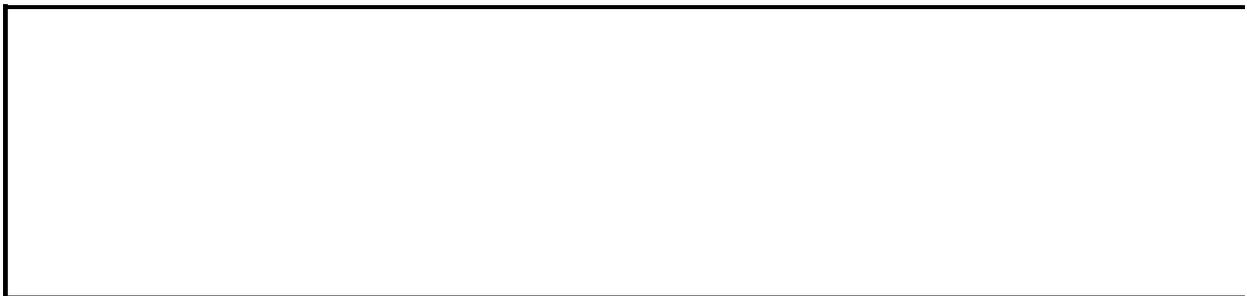
Returns the PowerPoint version number. Read-only **String**.

Example

This example displays a message box that contains the PowerPoint version number and build number, and the name of the operating system.

With Application

```
MsgBox "Welcome to PowerPoint version " & .Version & _  
      ", build " & .Build & ", running on " & .OperatingSystem & "!"  
End With
```



[Show All](#)

VerticalAnchor Property

Returns or sets the vertical alignment of text in a text frame. Read/write [MsoVerticalAnchor](#).

MsoVerticalAnchor can be one of these MsoVerticalAnchor constants.

msoAnchorBottom

msoAnchorBottomBaseLine Anchors the bottom of the text string to the current position regardless of the resizing of text. When you resize text without baseline anchoring, the text centers itself on the previous position.

msoAnchorMiddle

msoAnchorTop

msoAnchorTopBaseline Anchors the bottom of the text string to the current position regardless of the resizing of text. When you resize text without baseline anchoring, the text centers itself on the previous position.

msoVerticalAnchorMixed Read-only. Returned when two or more text boxes within a shape range have this property set to different values.

expression.**VerticalAnchor**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the alignment of the text in shape one on myDocument to top centered.

```
Set myDocument = ActivePresentation.SlideMaster
With myDocument.Shapes(1)
    .TextFrame.HorizontalAnchor = msoAnchorCenter
    .TextFrame.VerticalAnchor = msoAnchorTop
End With
```



[Show All](#)

VerticalFlip Property

Determines whether the specified shape is flipped around the vertical axis. Read-only [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified shape is flipped around the vertical axis.

Example

This example restores each shape on myDocument to its original state if it's been flipped horizontally or vertically.

```
Set myDocument = ActivePresentation.Slides(1)
For Each s In myDocument.Shapes
    If s.HorizontalFlip Then s.Flip msoFlipHorizontal
    If s.VerticalFlip Then s.Flip msoFlipVertical
Next
```



[Show All](#)

Vertices Property

Returns the coordinates of the specified freeform drawing's vertices (and control points for Bézier curves) as a series of [coordinate pairs](#). You can use the array returned by this property as an argument to the [AddCurve](#) method or [AddPolyline](#) method. Read-only **Variant**.

The following table shows how the **Vertices** property associates the values in the array `vertArray()` with the coordinates of a triangle's vertices.

VertArray element	Contains
<code>vertArray(1, 1)</code>	The horizontal distance from the first vertex to the left side of the slide
<code>vertArray(1, 2)</code>	The vertical distance from the first vertex to the top of the slide
<code>vertArray(2, 1)</code>	The horizontal distance from the second vertex to the left side of the slide
<code>vertArray(2, 2)</code>	The vertical distance from the second vertex to the top of the slide
<code>vertArray(3, 1)</code>	The horizontal distance from the third vertex to the left side of the slide
<code>vertArray(3, 2)</code>	The vertical distance from the third vertex to the top of the slide

Example

This example assigns the vertex coordinates for shape one on myDocument to the array variable vertArray() and displays the coordinates for the first vertex.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(1)
    vertArray = .Vertices
    x1 = vertArray(1, 1)
    y1 = vertArray(1, 2)
    MsgBox "First vertex coordinates: " & x1 & ", " & y1
End With
```

This example creates a curve that has the same geometric description as shape one on myDocument. Shape one must contain $3n+1$ vertices for this example to succeed.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes
    .AddCurve .Item(1).Vertices
End With
```



[Show All](#)

View Property

[View property as it applies to the **SlideShowWindow** object.](#)

Returns a [SlideShowView](#) object. Read-only.

expression.**View**

expression Required. An expression that returns one of the above objects.

[View property as it applies to the **DocumentWindow** object.](#)

Returns a [View](#) object that represents the view in the specified document window. Read-only.

expression.**View**

expression Required. An expression that returns one of the above objects.

Example

 [As it applies to the **SlideShowWindow** object.](#)

This example uses the **View** property to exit the current slide show, sets the view in the active window to slide view, and then displays slide three.

```
Application.SlideShowWindows(1).View.Exit  
With Application.ActiveWindow  
    .ViewType = ppViewSlide  
    .View.GotoSlide 3  
End With
```



[Show All](#)

ViewType Property

 [ViewType property as it applies to the **DocumentWindow** object.](#)

Returns or sets the type of the view contained in the specified document window.
Read/write **PpViewType**.

PpViewType can be one of these PpViewType constants.

ppViewHandoutMaster

ppViewMasterThumbnails

ppViewNormal

ppViewNotesMaster

ppViewNotesPage

ppViewOutline

ppViewPrintPreview

ppViewSlide

ppViewSlideMaster

ppViewSlideSorter

ppViewThumbnails

ppViewTitleMaster

expression.**ViewType**

expression Required. An expression that returns one of the above objects.

 [ViewType property as it applies to the **Pane** object.](#)

In any view with panes, returns the type of view for the specified pane. When referencing a view without panes, returns the type of view for the parent **DocumentWindow** object. Read-only **PpViewType**.

PpViewType can be one of these PpViewType constants.

ppViewHandoutMaster

ppViewMasterThumbnails

ppViewNormal
ppViewNotesMaster
ppViewNotesPage
ppViewOutline
ppViewPrintPreview
ppViewSlide
ppViewSlideMaster
ppViewSlideSorter
ppViewThumbnails
ppViewTitleMaster

expression.**ViewType**

expression Required. An expression that returns one of the above objects.

Example

[As it applies to the **DocumentWindow** object.](#)

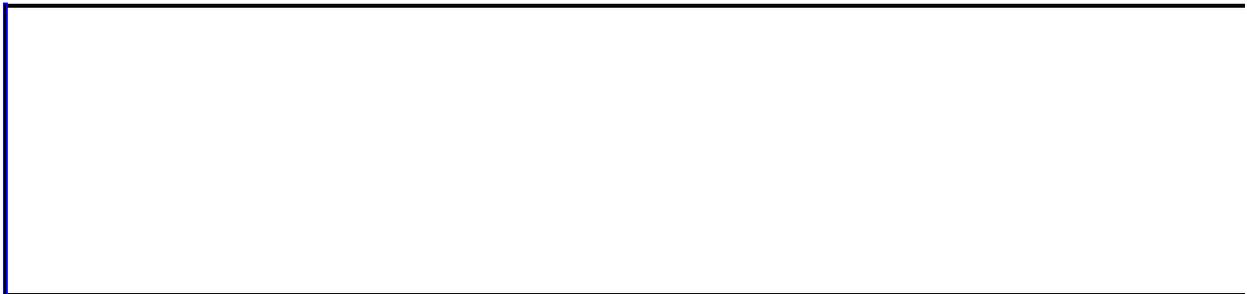
This example changes the view in the active window to slide sorter view if the window is currently displayed in normal view.

```
With Application.ActiveWindow
  If .ViewType = ppViewNormal Then
    .ViewType = ppViewSlideSorter
  End If
End With
```

[As it applies to the **Pane** object.](#)

If the view in the active pane is slide view, this example makes the notes pane the active pane. The notes pane is the third member of the **Panes** collection.

```
With ActiveWindow
  If .ActivePane.ViewType = ppViewSlide Then
    .Panes(3).Activate
  End If
End With
```



[Show All](#)

Visible Property

Returns or sets the visibility of the specified object or the formatting applied to the specified object. Read/write [MsoTriState](#).

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The specified object or object formatting is visible.

Example

This example sets the horizontal and vertical offsets for the shadow of shape three on the first slide in the active presentation. The shadow is offset 5 points to the right of the shape and 3 points above it. If the shape doesn't already have a shadow, this example adds one to it.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes(3).Shadow
    .Visible = msoTrue
    .OffsetX = 5
    .OffsetY = -3
End With
```



WebOptions Property

Returns the [WebOptions](#) object, which contains presentation-level attributes used by Microsoft PowerPoint when you save or publish a complete or partial presentation as a Web page or open a Web page. Read-only.

Example

This example specifies that when saving or publishing the active presentation as a Web page, Portable Network Graphics (PNG) are allowed, and the text color for the outline pane is white and the background color for the outline and slide panes is black.

```
With ActivePresentation.WebOptions  
    .FrameColors = ppFrameColorsWhiteTextOnBlack  
    .AllowPNG = True  
End With
```



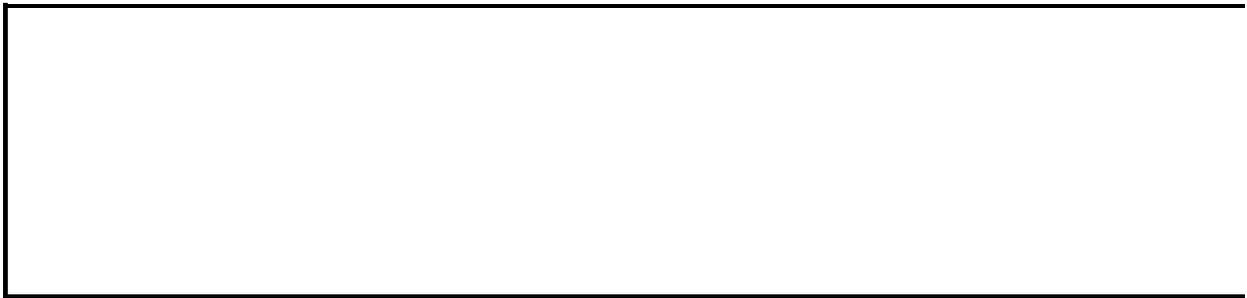
Weight Property

Returns or sets the thickness of the specified line, in points. Read/write **Single**.

Example

This example adds a green dashed line two points thick to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(0, 255, 255)
    .Weight = 2
End With
```



Width Property

Returns or sets the width of the specified object, in points. Read-only **Single** for the [Master](#) object, read/write **Single** for all other objects.

Example

This example arranges windows one and two horizontally; in other words, each window occupies half the available vertical space and all the available horizontal space in the application window's client area. For this example to work, there must be only two document windows open.

```
Windows.Arrange ppArrangeTiled
ah = Windows(1).Height           ' available height
aw = Windows(1).Width + Windows(2).Width ' available width
With Windows(1)
    .Width = aw
    .Height = ah / 2
    .Left = 0
End With
With Windows(2)
    .Width = aw
    .Height = ah / 2
    .Top = ah / 2
    .Left = 0
End With
```

This example sets the width for column one in the specified table to 80 points (72 points per inch).

```
ActivePresentation.Slides(2).Shapes(5).Table.Columns(1).Width = 80
```



[Show All](#)

Windows Property

 [Windows property as it applies to the **Application** object.](#)

Returns a [DocumentWindows](#) collection that represents all open document windows. Read-only.

 [Windows property as it applies to the **Presentation** object.](#)

Returns a [DocumentWindows](#) collection that represents all document windows associated with the specified presentation. This property doesn't return any slide show windows associated with the presentation. Read-only.

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).

Example

[As it applies to the **Application** object.](#)

This example closes all windows except the active window.

```
With Application.Windows
  For i = .Count To 2 Step -1
    .Item(i).Close
  Next
End With
```



[Show All](#)

WindowState Property

Returns or sets the state of the specified window. Read/write [PpWindowState](#).

PpWindowState can be one of these PpWindowState constants.

ppWindowMaximized

ppWindowMinimized

ppWindowNormal

expression.**WindowState**

expression Required. An expression that returns one of the objects in the Applies To list.

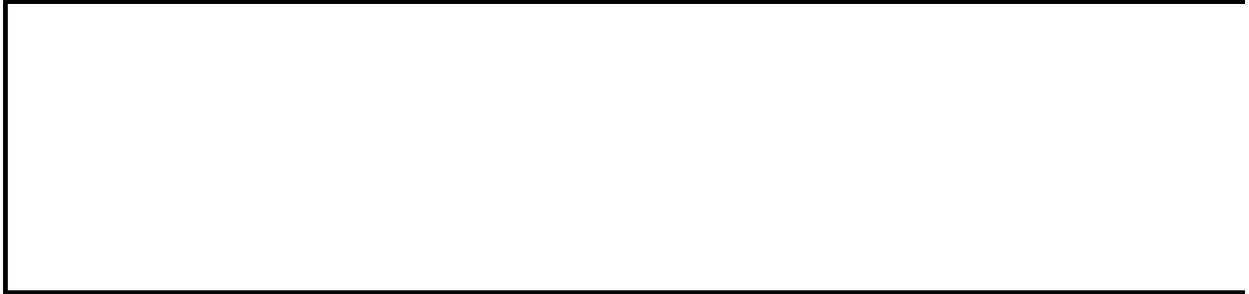
Remarks

When the state of the window is **ppWindowNormal**, the window is neither maximized nor minimized.

Example

This example maximizes the active window.

```
Application.ActiveWindow.WindowState = ppWindowMaximized
```



[Show All](#)

WordWrap Property

 [WordWrap property as it applies to the **TextFrame** object.](#)

Determines whether lines break automatically to fit inside the shape. Read/write **MsoTriState**.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue Lines break automatically to fit inside the shape.

expression.**WordWrap**

expression Required. An expression that returns one of the objects in the Applies To list.

 [WordWrap property as it applies to the **ParagraphFormat** object.](#)

Used only with Kanji characters. Read/write **Long**.

Example

 [As it applies to the **TextFrame** object.](#)

This example adds a rectangle that contains text to myDocument, and then turns off word wrapping in the new rectangle.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeRectangle, _
    0, 0, 100, 300).TextFrame
    .TextRange.Text = _
        "Here is some test text that is too long for this box"
    .WordWrap = False
End With
```



WritePassword Property

Sets or returns a **String** that represents a password for saving changes to the specified document. Read/write.

expression.**WritePassword**

expression Required. An expression that returns one of the objects in the Applies To list.

Example

This example sets the password for saving changes to the active presentation.

```
Sub SetSavePassword()  
    ActivePresentation.WritePassword = complexstrPWD 'global variabl  
End Sub
```



[Show All](#)

Zoom Property

 [Zoom property as it applies to the **View** object.](#)

Returns or sets the zoom setting of the specified view as a percentage of normal size. Can be a value from 10 to 400 percent. Read/write **Integer**.

expression.**Zoom**

expression Required. An expression that returns one of the above objects.

 [Zoom property as it applies to the **SlideShowView** object.](#)

Returns the zoom setting of the specified slide show window view as a percentage of normal size. Can be a value from 10 to 400 percent. Read-only **Integer**.

expression.**Zoom**

expression Required. An expression that returns one of the above objects.

Example

[As it applies to the **View** object.](#)

The following example sets the zoom to 30 percent for the view in document window one.

```
Windows(1).View.Zoom = 30
```



[Show All](#)

ZoomToFit Property

Determines whether the view is zoomed to fit the dimensions of the document window every time the document window is resized. Read/write [MsoTriState](#). This property applies only to slide view, notes page view, or master view.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse

msoTriStateMixed

msoTriStateToggle

msoTrue The view is zoomed to fit the dimensions of the document window every time the document window is resized.

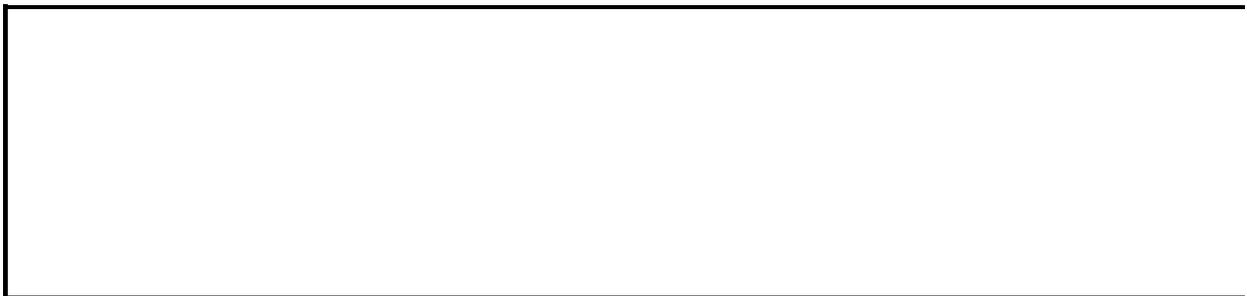
Remarks

When the value of the [Zoom](#) property is explicitly set, the value of the [ZoomToFit](#) property is automatically set to **msoFalse**.

Example

The following example sets the view in document window one to slide view, with the zoom automatically set to fit the dimensions of the window.

```
With Windows(1)  
    .ViewType = ppViewSlide  
    .View.ZoomToFit = msoTrue  
End With
```



ZOrderPosition Property

Returns the position of the specified shape in the z-order. `Shapes(1)` returns the shape at the back of the z-order, and `Shapes(Shapes.Count)` returns the shape at the front of the z-order. Read-only **Long**.

This property is read-only. To set the shape's position in the z-order, use the [ZOrder](#) method.

Remarks

A shape's position in the z-order corresponds to the shape's index number in the **Shapes** collection. For example, if there are four shapes on the slide, the expression `myDocument.Shapes(1)` returns the shape at the back of the z-order, and the expression `myDocument.Shapes(4)` returns the shape at the front of the z-order.

Whenever you add a new shape to a collection, it's added to the front of the z-order by default.

Example

This example adds an oval to myDocument and then places the oval second from the back in the z-order if there is at least one other shape on the slide.

```
Set myDocument = ActivePresentation.Slides(1)
With myDocument.Shapes.AddShape(msoShapeOval, 100, 100, 100, 300)
    While .ZOrderPosition > 2
        .ZOrder msoSendBackward
    Wend
End With
```



[Show All](#)

AfterNewPresentation Event

Occurs after a presentation is created.

expression.**AfterNewPresentation**(*Pres*)

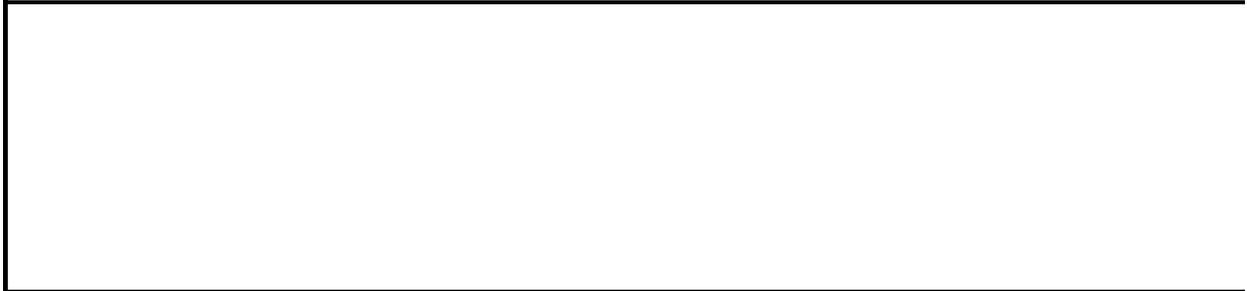
expression Required. An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres Required **Presentation**. The presentation that is created.

Example

This example uses the **RGB** function to set the slide master background color for the new presentation to salmon pink, and then applies the third color scheme to the new presentation.

```
Private Sub App_AfterNewPresentation(ByVal Pres As Presentation)
    With Pres
        Set CS3 = .ColorSchemes(3)
        CS3.Colors(ppBackground).RGB = RGB(240, 115, 100)
        .SlideMaster.ColorScheme = CS3
    End With
End Sub
```



[Show All](#)

AfterPresentationOpen Event

Occurs after an existing presentation is opened.

expression.**AfterPresentationOpen**(*Pres*)

expression Required. An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres Required **Presentation**. The presentation that is opened.

Example

This example modifies the background color for color scheme three, applies the modified color scheme to the presentation that was opened, and displays the presentation in Slide view.

```
Private Sub App_AfterPresentationOpen(ByVal Pres As Presentation)
    With Pres
        Set CS3 = .ColorSchemes(3)
        CS3.Colors(ppBackground).RGB = RGB(240, 115, 100)
        With Windows(1)
            .Selection.SlideRange.ColorScheme = CS3
            .ViewType = ppViewSlide
        End With
    End With
End Sub
```



ColorSchemeChanged Event

Occurs after a color scheme is changed.

Private Sub *object_ColorSchemeChanged*(ByVal *SldRange* As SlideRange)

object A variable that references an object of type [Application](#) declared with events in a class module.

SldRange The range of slides affected by the change.

Remarks

Actions which trigger this event would include actions such as modifying a slide's or slide master's color scheme, or applying a template.

To access the **Application** events, declare an **Application** variable in the General Declarations section of your code. Then set the variable equal to the **Application** object for which you want to access events. For information about using events with the Microsoft PowerPoint **Application** object, see [Using Events with the Application Object](#).

Example

This example displays a message when the color scheme for the selected slide or slides is changed. This example assumes an **Application** object called PPTApp has been declared using the **WithEvents** keyword.

```
Private Sub PPTApp_ColorSchemeChanged(ByVal SldRange As SlideRange)

    If SldRange.Count = 1 Then
        MsgBox "You've changed the color scheme for " _
            & SldRange.Name & "."
    Else
        MsgBox "You've changed the color scheme for " _
            & SldRange.Count & " slides."
    End If
End Sub
```



[Show All](#)

NewPresentation Event

Occurs after a presentation is created, as it is added to the [Presentations](#) collection.

Private Sub *application*_NewPresentation(ByVal *Pres* As Presentation)

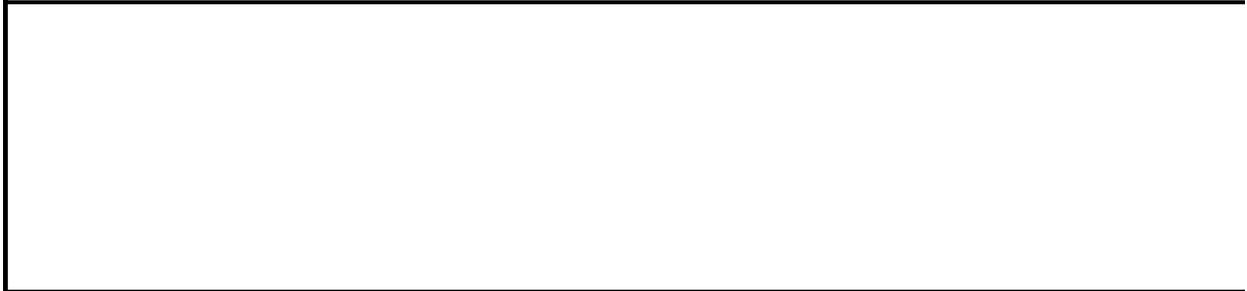
application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The new presentation.

Example

This example uses the **RGB** function to set the slide master background color for the new presentation to salmon pink, and then applies the third color scheme to the new presentation.

```
Private Sub App_NewPresentation(ByVal Pres As Presentation)
    With Pres
        Set CS3 = .ColorSchemes(3)
        CS3.Colors(ppBackground).RGB = RGB(240, 115, 100)
        .SlideMaster.ColorScheme = CS3
    End With
End Sub
```



PresentationBeforeSave Event

Occurs before a presentation is saved.

Private Sub *object*_PresentationBeforeSave(ByVal *Pres* As Presentation, *Cancel* As Boolean)

object A variable that references an object of type [Application](#) declared with events in a class module.

Pres The presentation being saved.

Cancel **True** to cancel the save process.

Remarks

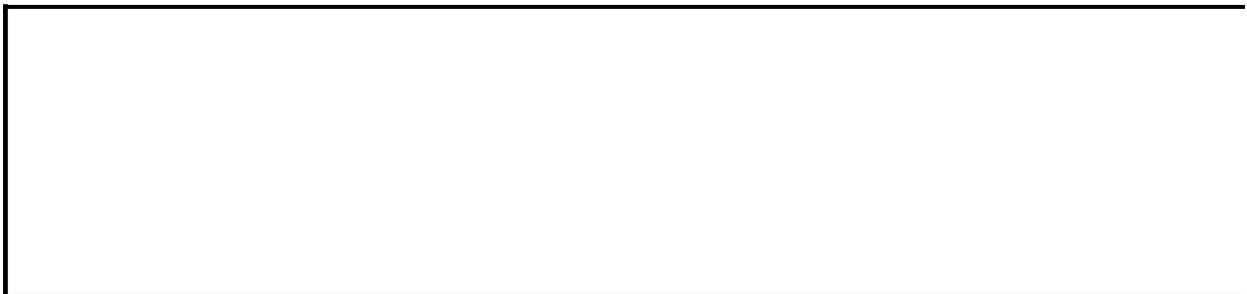
This event is triggered as the **Save As** dialog box appears.

To access the **Application** events, declare an **Application** variable in the General Declarations section of your code. Then set the variable equal to the **Application** object for which you want to access events. For information about using events with the Microsoft PowerPoint **Application** object, see [Using Events with the Application Object](#).

Example

This example checks if there are revisions in a presentation and, if there are, asks whether to save the presentation. If a user's response is no, the save process is cancelled. This example assumes an **Application** object called PPTApp has been declared using the **WithEvents** keyword.

```
Private Sub PPTApp_PresentationBeforeSave(ByVal Pres As Presentation
    Cancel As Boolean)
    Dim intResponse As Integer
    Set Pres = ActivePresentation
    If Pres.HasRevisionInfo Then
        intResponse = MsgBox(Prompt:="The presentation contains revisions.
            "Do you want to accept the revisions before saving?", Buttons:=vbYesNo)
        If intResponse = vbYes Then
            Cancel = True
            MsgBox "Your presentation was not saved."
        End If
    End If
End Sub
```



[Show All](#)

PresentationClose Event

Occurs immediately before any open presentation closes, as it is removed from the [Presentations](#) collection.

Private Sub *application*_PresentationClose(ByVal *Pres* As Presentation)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation that is being closed.

Example

This example saves a copy of the active presentation as an HTML file, with the same name and within the same folder.

```
Private Sub App_PresentationClose(ByVal Pres As Presentation)
    FindNum = InStr(1, Pres.FullName, ".")
    HTMLName = Mid(Pres.FullName, 1, FindNum - 1) _
        & ".htm"
    Pres.SaveCopyAs HTMLName, ppSaveAsHTML
End Sub
```



[Show All](#)

PresentationNewSlide Event

Occurs when a new slide is created in any open presentation, as the slide is added to the [Slides](#) collection.

Private Sub *application*_PresentationNewSlide(**ByVal** *Sld* As Slide)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Sld The new slide.

Example

This example modifies the background color for color scheme three and then applies the modified color scheme to the new slide. Next, it adds default text to shape one if it has a text frame.

```
Private Sub App_PresentationNewSlide(ByVal Sld As Slide)
    With ActivePresentation
        Set CS3 = .ColorSchemes(3)
        CS3.Colors(ppBackground).RGB = RGB(240, 115, 100)
        Windows(1).Selection.SlideRange.ColorScheme = CS3
    End With

    If Sld.Layout <> ppLayoutBlank Then
        With Sld.Shapes(1)
            If .HasTextFrame = msoTrue Then
                .TextFrame.TextRange.Text = "King Salmon"
            End If
        End With
    End If
End Sub
```



[Show All](#)

PresentationOpen Event

Occurs after an existing presentation is opened, as it is added to the [Presentations](#) collection.

Private Sub *application*_PresentationOpen(ByVal *Pres* As Presentation)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation that is opened.

Example

This example modifies the background color for color scheme three, applies the modified color scheme to the presentation that was just opened, and displays the presentation in slide view.

```
Private Sub App_PresentationOpen(ByVal Pres As Presentation)
    With Pres
        Set CS3 = .ColorSchemes(3)
        CS3.Colors(ppBackground).RGB = RGB(240, 115, 100)
        With Windows(1)
            .Selection.SlideRange.ColorScheme = CS3
            .ViewType = ppViewSlide
        End With
    End With
End Sub
```



[Show All](#)

PresentationPrint Event

Occurs before a presentation is printed.

Private Sub *application_PresentationPrint*(ByVal *Pres* As Presentation)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation to be printed.

Example

This example sets the **PrintHiddenSlides** property to **True** so that every time the active presentation is printed, the hidden slides are printed as well.

```
Private Sub App_PresentationPrint(ByVal Pres As Presentation)
    Pres.PrintOptions.PrintHiddenSlides = True
End Sub
```



[Show All](#)

PresentationSave Event

Occurs before any open presentation is saved.

Private Sub *application_PresentationSave*(ByVal *Pres* As Presentation)

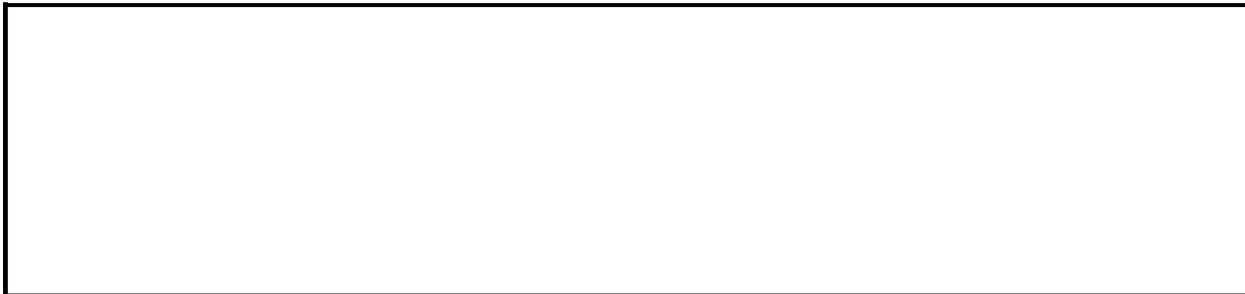
application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation to be saved.

Example

This example saves the current presentation as an HTML version 4.0 file with the name "mallard.htm." It then displays a message indicating that the current named presentation is being saved in both PowerPoint and HTML formats.

```
Private Sub App_PresentationSave(ByVal Pres As Presentation)
    With Pres.PublishObjects(1)
        PresName = .SlideShowName
        .SourceType = ppPublishAll
        .FileName = "C:\HTMLPres\mallard.htm"
        .HTMLVersion = ppHTMLVersion4
        MsgBox ("Saving presentation " & "" _
            & PresName & "" & " in PowerPoint" _
            & Chr(10) & Chr(13) _
            & " format and HTML version 4.0 format")
        .Publish
    End With
End Sub
```



[Show All](#)

PresentationSync Event

Occurs when the local copy of a presentation that is part of a Document Workspace is synchronized with the copy on the server. Provides important status information regarding the success or failure of the synchronization of the presentation.

expression.PresentationSync(Pres, SyncEventType)

expression Required. An expression that returns the **Application** object.

Pres The presentation that is being synchronized.

SyncEventType An [msoSyncEventType](#) value. The status of the synchronization.

MsoSyncEventType can be one of the following **msoSyncEventType** constants.

- msoSyncEventDownloadInitiated** (0)
- msoSyncEventDownloadSucceeded** (1)
- msoSyncEventDownloadFailed** (2)
- msoSyncEventUploadInitiated** (3)
- msoSyncEventUploadSucceeded** (4)
- msoSyncEventUploadFailed** (5)
- msoSyncEventDownloadNoChange** (6)
- msoSyncEventOffline** (7)

Example

The following example displays a message if the synchronization of a presentation in a Document Workspace fails.

```
Private Sub app_PresentationSync(ByVal Pres As Presentation, _  
    ByVal SyncEventType As Office.MsoSyncEventType)  
  
    If SyncEventType = msoSyncEventDownloadFailed Or _  
        SyncEventType = msoSyncEventUploadFailed Then  
  
        MsgBox "Synchronization failed. " & _  
            "Please contact your administrator, " & vbCrLf & _  
            "or try again later."  
  
    End If  
  
End Sub
```



SlideSelectionChanged Event

This event occurs at different times depending on the current view.

View	Description
Normal, Master	Occurs when the slide in the slide pane changes.
Slide Sorter	Occurs when the selection changes.
Slide, Notes	Occurs when the slide changes.
Outline	Does not occur.

Private Sub *object_SlideSelectionChanged*(ByVal *SldRange* As SlideRange)

object A variable that references an object of type [Application](#) declared with events in a class module.

SldRange The selection of slides. In most cases this would be a single slide (for example, in Slide View you navigate to the next slide), but in some cases this could be multiple slides (for example, a marquee selection in Slide Sorter View).

Remarks

To access the **Application** events, declare an **Application** variable in the General Declarations section of your code. Then set the variable equal to the **Application** object for which you want to access events. For information about using events with the Microsoft PowerPoint **Application** object, see [Using Events with the Application Object](#).

Example

This example displays a message every time a user selects a different slide. This example assumes an **Application** object called PPTApp has been declared using the **WithEvents** keyword.

```
Private Sub PPTApp_SlideSelectionChanged(ByVal SldRange As SlideRang
    MsgBox "Slide selection changed."
End Sub
```



[Show All](#)

SlideShowBegin Event

Occurs when you start a slide show. Microsoft PowerPoint creates the slide show window and passes it to this event. If one slide show branches to another, the SlideShowBegin event does not occur again when the second slide show begins.

Private Sub *application_SlideShowBegin*(ByVal *Wn* As SlideShowWindow)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Wn The slide show window initialized prior to this event.

Example

This example adjusts the size and position of the slide show window and then reactivates it.

```
Private Sub App_SlideShowBegin(ByVal Wn As SlideShowWindow)
    With Wn
        .Height = 325
        .Width = 400
        .Left = 100
        .Activate
    End With
End Sub
```



[Show All](#)

SlideShowEnd Event

Occurs after a slide show ends— immediately after the last [SlideShowNextSlide](#) event occurs.

Private Sub *application_SlideShowEnd*(ByVal *Pres* As Presentation)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation closed when this event occurs.

Remarks

The SlideShowEnd event always occurs before a slide show ends if the [SlideShowBegin](#) event has occurred. You can use the SlideShowEnd event to return any property settings and variable initializations that occur in the SlideShowBegin event to their original settings.

Example

This example turns off the entry effect and automatic advance timing slide show transition effects for slides one through four at the end of the slide show. It also sets the slides to advance manually.

```
Private Sub App_SlideShowEnd(ByVal Pres As Presentation)
    With Pres.Slides.Range(Array(1, 4)) _
        .SlideShowTransition
            .EntryEffect = ppEffectNone
            .AdvanceOnTime = msoFalse
        End With

    With Pres.SlideShowSettings
        .AdvanceMode = ppSlideShowManualAdvance
    End With
End Sub
```



[Show All](#)

SlideShowNextBuild Event

Occurs upon mouse-click or timing animation, but before the animated object becomes visible.

Private Sub *application_SlideShowNextBuild*(ByVal *Wn* As SlideShowWindow)

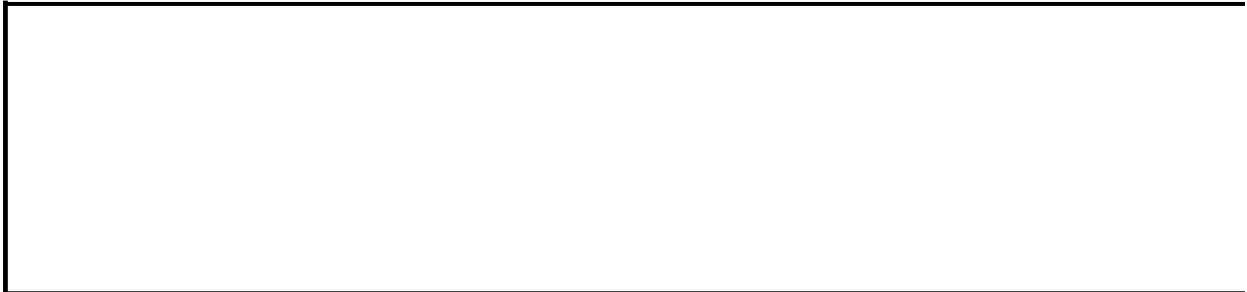
application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Wn The active slide show window.

Example

If the current shape on slide one is a movie, this example plays the movie continuously until stopped manually by the presenter. This code is designed to be used with the second `SlideShowNextSlide` event example.

```
Private Sub App_SlideShowNextBuild(ByVal Wn As SlideShowWindow)
    If EvtCounter <> 0 Then
        With ActivePresentation.Slides(1) _
            .Shapes(shpAnimArray(2, EvtCounter))
            If .Type = msoMedia Then
                If .MediaType = ppMediaTypeMovie
                    .AnimationSettings.PlaySettings _
                        .LoopUntilStopped
                End If
            End If
        End With
    End If
    EvtCounter = EvtCounter + 1
End Sub
```



[Show All](#)

SlideShowNextClick Event

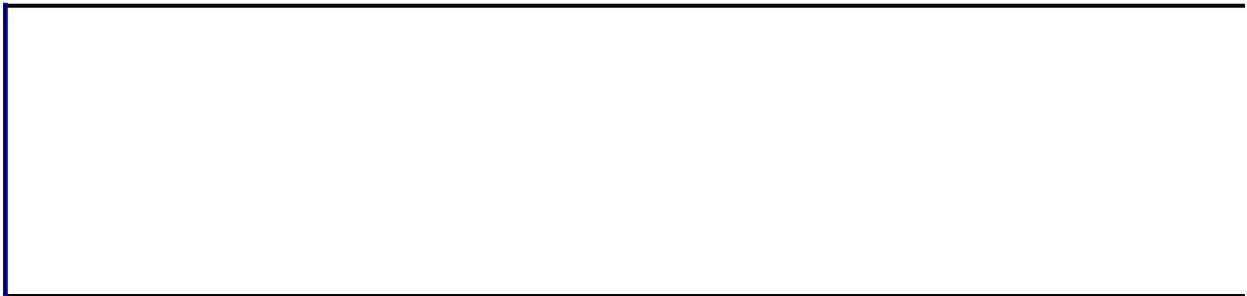
Occurs on the next click on the slide.

Private Sub *application_SlideShowNextClick*(ByVal *Wn* As SlideShowWindow, ByVal *nEffect* As Effect)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Wn The slide show window initialized prior to this event.

nEffect The effect to animate on next click.



[Show All](#)

SlideShowNextSlide Event

Occurs immediately before the transition to the next slide. For the first slide, occurs immediately after the [SlideShowBegin](#) event.

Private Sub *application_SlideShowNextSlide*(ByVal *Wn* As **SlideShowWindow**)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Wn The active slide show window.

Example

This example determines the slide position for the slide following the `SlideShowNextSlide` event. If the next slide is slide three, the example changes the type of pointer to a pen and the pen color to red.

```
Private Sub App_SlideShowNextSlide(ByVal Wn As SlideShowWindow)

    Dim Showpos As Integer

    Showpos = Wn.View.CurrentShowPosition + 1

    If Showpos = 3 Then
        With ActivePresentation.SlideShowSettings.Run.View
            .PointerColor.RGB = RGB(255, 0, 0)
            .PointerType = ppSlideShowPointerPen
        End With
    Else
        With ActivePresentation.SlideShowSettings.Run.View
            .PointerColor.RGB = RGB(0, 0, 0)
            .PointerType = ppSlideShowPointerArrow
        End With
    End If
End Sub
```

This example sets a global counter variable to zero. Then it calculates the number of shapes on the slide following this event, determines which shapes have animation, and fills a global array with the animation order and the number of each shape.

Note The array created in this example is also used in the `SlideShowNextBuild` event example.

```
Private Sub App_SlideShowNextSlide(ByVal Wn As SlideShowWindow)

    Dim i as Integer, j as Integer, numShapes As Integer
    Dim objSld As Slide

    Set objSld = ActivePresentation.Slides _
        (ActivePresentation.SlideShowWindow.View _
            .CurrentShowPosition + 1)
    With objSld.Shapes
```

```
numShapes = .Count
If numShapes > 0 Then
    j = 1
    ReDim shpAnimArray(1 To 2, 1 To numShapes)
    For i = 1 To numShapes
        If .Item(i).AnimationSettings.Animate Then
            shpAnimArray(1, j) = _
                .Item(i).AnimationSettings.AnimationOrder
            shpAnimArray(2, j) = i
            j = j + 1
        End If
    Next
End If
End With
End Sub
```



[Show All](#)

WindowActivate Event

Occurs when the application window or any document window is activated.

Private Sub *application_WindowActivate*(ByVal *Pres* As Presentation, ByVal *Wn* As DocumentWindow)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation displayed in the activated window.

Wn The activated document window.

Example

This example opens every activated presentation in slide sorter view.

```
Private Sub App_WindowActivate _ (ByVal Pres As Presentation)
    Wn.ViewType = ppViewSlideSorter
End Sub
```



[Show All](#)

WindowBeforeDoubleClick Event

Occurs when you double-click the items in the views listed in the following table.

View	Item
Normal or slide view	Shape
Slide sorter view	Slide
Notes page view	Slide image

The default double-click action occurs after this event unless the *Cancel* argument is set to **True**.

Private Sub *application_WindowBeforeDoubleClick*(**ByVal** *Sel* **As** **Selection**, **ByVal** *Cancel* **As** **Boolean**)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Sel The selection below the mouse pointer when the double-click occurs.

Cancel **False** when the event occurs. If the event procedure sets this argument to **True**, the default double-click action isn't performed when the procedure is finished.

Example

In slide sorter view, the default double-click event for any slide is to change to slide view. In this example, if the active presentation is displayed in slide sorter view, the default action is preempted by the `WindowBeforeDoubleClick` event. The event procedure changes the view to normal view and then cancels the change to slide view by setting the *Cancel* argument to **True**.

```
Private Sub App_WindowBeforeDoubleClick _           (ByVal Sel As Sele  
    With Application.ActiveWindow  
        If .ViewType = ppViewSlideSorter Then  
            .ViewType = ppViewNormal  
            Cancel = True  
        End If  
    End With  
End Sub
```



[Show All](#)

WindowBeforeRightClick Event

Occurs when you right-click a shape, a slide, a notes page, or some text. This event is triggered by the MouseUp event.

Private Sub *application*_WindowBeforeRightClick(ByVal *Sel* As Selection, ByVal *Cancel* As Boolean)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Sel The selection below the mouse pointer when the right-click occurred.

Cancel **False** when the event occurs. If the event procedure sets this argument to **True**, the default context menu does not appear when the procedure is finished.

Example

This example creates a duplicate of the selected shape. If the shape has a text frame, it adds the text "Duplicate Shape" to the new shape. Setting the *Cancel* argument to **True** then prevents the default context menu from appearing.

```
Private Sub App_WindowBeforeRightClick _ (ByVal Sel As Selec
    With ActivePresentation.Selection.ShapeRange
        If .HasTextFrame Then
            .Duplicate.TextFrame.TextRange.Text = "Duplicate Shape"
        Else
            .Duplicate
        End If
        Cancel = True
    End With
End Sub
```



[Show All](#)

WindowDeactivate Event

Occurs when the application window or any document window is deactivated.

Private Sub *application*_WindowDeactivate(ByVal *Pres* As Presentation, ByVal *Wn* As DocumentWindow)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Pres The presentation displayed in the deactivated window.

Wn The deactivated document window.

Example

This example finds the file name (without its extension) for the presentation in the window that is being deactivated. It then appends the .htm extension to the file name and saves it as a Web page in the same folder as the presentation.

```
Private Sub App_WindowDeactivate _ (ByVal Pres As Presentati
    FindNum = InStr(1, Wn.Presentation.FullName, ".")
    If FindNum = 0 Then
        HTMLName = Wn.Presentation.FullName & ".htm"
    Else
        HTMLName = Mid(Wn.Presentation.FullName, 1, FindNum - 1) _
            & ".htm"
    End If
    Wn.Presentation.SaveCopyAs HTMLName, ppSaveAsHTML
    MsgBox "Presentation being saved in HTML format as " _
        & HTMLName & " ."
End Sub
```



[Show All](#)

WindowSelectionChange Event

Occurs when the selection of text, a shape, or a slide in the active document window changes, whether through the user interface or through code.

Private Sub *application*_WindowSelectionChange(ByVal *Sel* As Selection)

application An object of type **Application** declared with events in a [class module](#). For information about using events with the **Application** object, see [Using Events with the Application Object](#).

Sel Represents the object selected.

Example

This example determines when a different slide is being selected and changes the background color of the newly selected slide.

```
Private Sub App_WindowSelectionChange(ByVal Sel As Selection)
    With Sel
        If .Type = ppSelectionNone Then
            With .SlideRange(1)
                .ColorScheme.Colors(ppBackground).RGB = _
                    RGB(240, 115, 100)
            End With
        End If
    End With
End Sub
```



[Show All](#)

PowerPoint Constants

This topic provides a list of all constants in the PowerPoint object model.

[MsoAnimAccumulate](#)

Constant	Value
msoAnimAccumulateAlways	2
msoAnimAccumulateNone	1

[MsoAnimAdditive](#)

Constant	Value
msoAnimAdditiveAddBase	1
msoAnimAdditiveAddSum	2

[MsoAnimAfterEffect](#)

Constant	Value
msoAnimAfterEffectDim	1
msoAnimAfterEffectHide	2
msoAnimAfterEffectHideOnNextClick	3
msoAnimAfterEffectMixed	-1
msoAnimAfterEffectNone	0

[MsoAnimateByLevel](#)

Constant	Value
msoAnimateChartAllAtOnce	7
msoAnimateChartByCategory	8
msoAnimateChartByCategoryElements	9
msoAnimateChartBySeries	10
msoAnimateChartBySeriesElements	11

msoAnimateDiagramAllAtOnce	12
msoAnimateDiagramBreadthByLevel	16
msoAnimateDiagramBreadthByNode	15
msoAnimateDiagramClockwise	17
msoAnimateDiagramClockwiseIn	18
msoAnimateDiagramClockwiseOut	19
msoAnimateDiagramCounterClockwise	20
msoAnimateDiagramCounterClockwiseIn	21
msoAnimateDiagramCounterClockwiseOut	22
msoAnimateDiagramDepthByBranch	14
msoAnimateDiagramDepthByNode	13
msoAnimateDiagramDown	26
msoAnimateDiagramInByRing	23
msoAnimateDiagramOutByRing	24
msoAnimateDiagramUp	25
msoAnimateLevelMixed	-1
msoAnimateLevelNone	0
msoAnimateTextByAllLevels	1
msoAnimateTextByFifthLevel	6
msoAnimateTextByFirstLevel	2
msoAnimateTextByFourthLevel	5
msoAnimateTextBySecondLevel	3
msoAnimateTextByThirdLevel	4

 **MsoAnimCommandType**

Constant	Value
msoAnimCommandTypeCall	1
msoAnimCommandTypeEvent	0
msoAnimCommandTypeVerb	2

 **MsoAnimDirection**

Constant	Value
msoAnimDirectionAcross	18
msoAnimDirectionBottom	11
msoAnimDirectionBottomLeft	15
msoAnimDirectionBottomRight	14
msoAnimDirectionCenter	28
msoAnimDirectionClockwise	21
msoAnimDirectionCounterclockwise	22
msoAnimDirectionCycleClockwise	43
msoAnimDirectionCycleCounterclockwise	44
msoAnimDirectionDown	3
msoAnimDirectionDownLeft	9
msoAnimDirectionDownRight	8
msoAnimDirectionFontAllCaps	40
msoAnimDirectionFontBold	35
msoAnimDirectionFontItalic	36
msoAnimDirectionFontShadow	39
msoAnimDirectionFontStrikethrough	38
msoAnimDirectionFontUnderline	37
msoAnimDirectionGradual	42
msoAnimDirectionHorizontal	16
msoAnimDirectionHorizontalIn	23
msoAnimDirectionHorizontalOut	24
msoAnimDirectionIn	19
msoAnimDirectionInBottom	31
msoAnimDirectionInCenter	30
msoAnimDirectionInSlightly	29
msoAnimDirectionInstant	41
msoAnimDirectionLeft	4
msoAnimDirectionNone	0
msoAnimDirectionOrdinalMask	5
msoAnimDirectionOut	20
msoAnimDirectionOutBottom	34

msoAnimDirectionOutCenter	33
msoAnimDirectionOutSlightly	32
msoAnimDirectionRight	2
msoAnimDirectionSlightly	27
msoAnimDirectionTop	10
msoAnimDirectionTopLeft	12
msoAnimDirectionTopRight	13
msoAnimDirectionUp	1
msoAnimDirectionUpLeft	6
msoAnimDirectionUpRight	7
msoAnimDirectionVertical	17
msoAnimDirectionVerticalIn	25
msoAnimDirectionVerticalOut	26

MsoAnimEffect

Constant	Value
msoAnimEffectAppear	1
msoAnimEffectArcUp	47
msoAnimEffectAscend	39
msoAnimEffectBlast	64
msoAnimEffectBlinds	3
msoAnimEffectBoldFlash	63
msoAnimEffectBoldReveal	65
msoAnimEffectBoomerang	25
msoAnimEffectBounce	26
msoAnimEffectBox	4
msoAnimEffectBrushOnColor	66
msoAnimEffectBrushOnUnderline	67
msoAnimEffectCenterRevolve	40
msoAnimEffectChangeFillColor	54
msoAnimEffectChangeFont	55
msoAnimEffectChangeFontColor	56

msoAnimEffectChangeFontSize	57
msoAnimEffectChangeFontStyle	58
msoAnimEffectChangeLineColor	60
msoAnimEffectCheckerboard	5
msoAnimEffectCircle	6
msoAnimEffectColorBlend	68
msoAnimEffectColorReveal	27
msoAnimEffectColorWave	69
msoAnimEffectComplementaryColor	70
msoAnimEffectComplementaryColor2	71
msoAnimEffectContrastingColor	72
msoAnimEffectCrawl	7
msoAnimEffectCredits	28
msoAnimEffectCustom	0
msoAnimEffectDarken	73
msoAnimEffectDesaturate	74
msoAnimEffectDescend	42
msoAnimEffectDiamond	8
msoAnimEffectDissolve	9
msoAnimEffectEaseIn	29
msoAnimEffectExpand	50
msoAnimEffectFade	10
msoAnimEffectFadedSwivel	41
msoAnimEffectFadedZoom	48
msoAnimEffectFlashBulb	75
msoAnimEffectFlashOnce	11
msoAnimEffectFlicker	76
msoAnimEffectFlip	51
msoAnimEffectFloat	30
msoAnimEffectFly	2
msoAnimEffectFold	53
msoAnimEffectGlide	49
msoAnimEffectGrowAndTurn	31

msoAnimEffectGrowShrink	59
msoAnimEffectGrowWithColor	77
msoAnimEffectLighten	78
msoAnimEffectLightSpeed	32
msoAnimEffectMediaPause	84
msoAnimEffectMediaPlay	83
msoAnimEffectMediaStop	85
msoAnimEffectPath4PointStar	101
msoAnimEffectPath5PointStar	90
msoAnimEffectPath6PointStar	96
msoAnimEffectPath8PointStar	102
msoAnimEffectPathArcDown	122
msoAnimEffectPathArcLeft	136
msoAnimEffectPathArcRight	143
msoAnimEffectPathArcUp	129
msoAnimEffectPathBean	116
msoAnimEffectPathBounceLeft	126
msoAnimEffectPathBounceRight	139
msoAnimEffectPathBuzzsaw	110
msoAnimEffectPathCircle	86
msoAnimEffectPathCrescentMoon	91
msoAnimEffectPathCurvedSquare	105
msoAnimEffectPathCurvedX	106
msoAnimEffectPathCurvyLeft	133
msoAnimEffectPathCurvyRight	146
msoAnimEffectPathCurvyStar	108
msoAnimEffectPathDecayingWave	145
msoAnimEffectPathDiagonalDownRight	134
msoAnimEffectPathDiagonalUpRight	141
msoAnimEffectPathDiamond	88
msoAnimEffectPathDown	127
msoAnimEffectPathEqualTriangle	98
msoAnimEffectPathFigure8Four	113

msoAnimEffectPathFootball	97
msoAnimEffectPathFunnel	137
msoAnimEffectPathHeart	94
msoAnimEffectPathHeartbeat	130
msoAnimEffectPathHexagon	89
msoAnimEffectPathHorizontalFigure8	111
msoAnimEffectPathInvertedSquare	119
msoAnimEffectPathInvertedTriangle	118
msoAnimEffectPathLeft	120
msoAnimEffectPathLoopdeLoop	109
msoAnimEffectPathNeutron	114
msoAnimEffectPathOctagon	95
msoAnimEffectPathParallelogram	99
msoAnimEffectPathPeanut	112
msoAnimEffectPathPentagon	100
msoAnimEffectPathPlus	117
msoAnimEffectPathPointyStar	104
msoAnimEffectPathRight	149
msoAnimEffectPathRightTriangle	87
msoAnimEffectPathSCurve1	144
msoAnimEffectPathSCurve2	124
msoAnimEffectPathSineWave	125
msoAnimEffectPathSpiralLeft	140
msoAnimEffectPathSpiralRight	131
msoAnimEffectPathSpring	138
msoAnimEffectPathSquare	92
msoAnimEffectPathStairsDown	147
msoAnimEffectPathSwoosh	115
msoAnimEffectPathTeardrop	103
msoAnimEffectPathTrapezoid	93
msoAnimEffectPathTurnDown	135
msoAnimEffectPathTurnRight	121
msoAnimEffectPathTurnUp	128

msoAnimEffectPathTurnUpRight	142
msoAnimEffectPathUp	148
msoAnimEffectPathVerticalFigure8	107
msoAnimEffectPathWave	132
msoAnimEffectPathZigzag	123
msoAnimEffectPeek	12
msoAnimEffectPinwheel	33
msoAnimEffectPlus	13
msoAnimEffectRandomBars	14
msoAnimEffectRandomEffects	24
msoAnimEffectRiseUp	34
msoAnimEffectShimmer	52
msoAnimEffectSling	43
msoAnimEffectSpin	61
msoAnimEffectSpinner	44
msoAnimEffectSpiral	15
msoAnimEffectSplit	16
msoAnimEffectStretch	17
msoAnimEffectStretchy	45
msoAnimEffectStrips	18
msoAnimEffectStyleEmphasis	79
msoAnimEffectSwish	35
msoAnimEffectSwivel	19
msoAnimEffectTeeter	80
msoAnimEffectThinLine	36
msoAnimEffectTransparency	62
msoAnimEffectUnfold	37
msoAnimEffectVerticalGrow	81
msoAnimEffectWave	82
msoAnimEffectWedge	20
msoAnimEffectWheel	21
msoAnimEffectWhip	38
msoAnimEffectWipe	22

msoAnimEffectZip	46
msoAnimEffectZoom	23

 [MsoAnimEffectAfter](#)

Constant	Value
msoAnimEffectAfterFreeze	1
msoAnimEffectAfterHold	3
msoAnimEffectAfterRemove	2
msoAnimEffectAfterTransition	4

 [MsoAnimEffectRestart](#)

Constant	Value
msoAnimEffectRestartAlways	1
msoAnimEffectRestartNever	3
msoAnimEffectRestartWhenOff	2

 [MsoAnimFilterEffectSubtype](#)

Constant	Value
msoAnimFilterEffectSubtypeAcross	9
msoAnimFilterEffectSubtypeDown	25
msoAnimFilterEffectSubtypeDownLeft	14
msoAnimFilterEffectSubtypeDownRight	16
msoAnimFilterEffectSubtypeFromBottom	13
msoAnimFilterEffectSubtypeFromLeft	10
msoAnimFilterEffectSubtypeFromRight	11
msoAnimFilterEffectSubtypeFromTop	12
msoAnimFilterEffectSubtypeHorizontal	5
msoAnimFilterEffectSubtypeIn	7
msoAnimFilterEffectSubtypeInHorizontal	3
msoAnimFilterEffectSubtypeInVertical	1
msoAnimFilterEffectSubtypeLeft	23

msoAnimFilterEffectSubtypeNone	0
msoAnimFilterEffectSubtypeOut	8
msoAnimFilterEffectSubtypeOutHorizontal	4
msoAnimFilterEffectSubtypeOutVertical	2
msoAnimFilterEffectSubtypeRight	24
msoAnimFilterEffectSubtypeSpokes1	18
msoAnimFilterEffectSubtypeSpokes2	19
msoAnimFilterEffectSubtypeSpokes3	20
msoAnimFilterEffectSubtypeSpokes4	21
msoAnimFilterEffectSubtypeSpokes8	22
msoAnimFilterEffectSubtypeUp	26
msoAnimFilterEffectSubtypeUpLeft	15
msoAnimFilterEffectSubtypeUpRight	17
msoAnimFilterEffectSubtypeVertical	6

 **MsoAnimFilterEffectType**

Constant	Value
msoAnimFilterEffectTypeBarn	1
msoAnimFilterEffectTypeBlinds	2
msoAnimFilterEffectTypeBox	3
msoAnimFilterEffectTypeCheckerboard	4
msoAnimFilterEffectTypeCircle	5
msoAnimFilterEffectTypeDiamond	6
msoAnimFilterEffectTypeDissolve	7
msoAnimFilterEffectTypeFade	8
msoAnimFilterEffectTypeImage	9
msoAnimFilterEffectTypeNone	0
msoAnimFilterEffectTypePixelate	10
msoAnimFilterEffectTypePlus	11
msoAnimFilterEffectTypeRandomBar	12
msoAnimFilterEffectTypeSlide	13
msoAnimFilterEffectTypeStretch	14

msoAnimFilterEffectTypeStrips	15
msoAnimFilterEffectTypeWedge	16
msoAnimFilterEffectTypeWheel	17
msoAnimFilterEffectTypeWipe	18

 **MsoAnimProperty**

Constant	Value
msoAnimColor	7
msoAnimHeight	4
msoAnimHeigth	4
msoAnimNone	0
msoAnimOpacity	5
msoAnimRotation	6
msoAnimShapeFillBackColor	1007
msoAnimShapeFillColor	1005
msoAnimShapeFillOn	1004
msoAnimShapeFillOpacity	1006
msoAnimShapeLineColor	1009
msoAnimShapeLineOn	1008
msoAnimShapePictureBrightness	1001
msoAnimShapePictureContrast	1000
msoAnimShapePictureCropFromBottom	1001
msoAnimShapePictureCropFromLeft	1002
msoAnimShapePictureCropFromRight	1003
msoAnimShapePictureCropFromTop	1000
msoAnimShapePictureGamma	1002
msoAnimShapePictureGrayscale	1003
msoAnimShapeShadowColor	1012
msoAnimShapeShadowOffsetX	1014
msoAnimShapeShadowOffsetY	1015
msoAnimShapeShadowOn	1010
msoAnimShapeShadowOpacity	1013

msoAnimShapeShadowType	1011
msoAnimTextBulletCharacter	111
msoAnimTextBulletColor	114
msoAnimTextBulletFontName	112
msoAnimTextBulletNumber	113
msoAnimTextBulletRelativeSize	115
msoAnimTextBulletStyle	116
msoAnimTextBulletType	117
msoAnimTextFontBold	100
msoAnimTextFontColor	101
msoAnimTextFontEmboss	102
msoAnimTextFontItalic	103
msoAnimTextFontName	104
msoAnimTextFontShadow	105
msoAnimTextFontSize	106
msoAnimTextFontStrikeThrough	110
msoAnimTextFontSubscript	107
msoAnimTextFontSuperscript	108
msoAnimTextFontUnderline	109
msoAnimVisibility	8
msoAnimWidth	3
msoAnimX	1
msoAnimY	2

 [**MsoAnimTextUnitEffect**](#)

Constant	Value
msoAnimTextUnitEffectByCharacter	1
msoAnimTextUnitEffectByParagraph	0
msoAnimTextUnitEffectByWord	2
msoAnimTextUnitEffectMixed	-1

 [**MsoAnimTriggerType**](#)

Constant	Value
msoAnimTriggerAfterPrevious	3
msoAnimTriggerMixed	-1
msoAnimTriggerNone	0
msoAnimTriggerOnPageClick	1
msoAnimTriggerOnShapeClick	4
msoAnimTriggerWithPrevious	2

[MsoAnimType](#)

Constant	Value
msoAnimTypeColor	2
msoAnimTypeCommand	6
msoAnimTypeFilter	7
msoAnimTypeMixed	-2
msoAnimTypeMotion	1
msoAnimTypeNone	0
msoAnimTypeProperty	5
msoAnimTypeRotation	4
msoAnimTypeScale	3
msoAnimTypeSet	8

[PpActionType](#)

Constant	Value
ppActionEndShow	6
ppActionFirstSlide	3
ppActionHyperlink	7
ppActionLastSlide	4
ppActionLastSlideViewed	5
ppActionMixed	-2
ppActionNamedSlideShow	10
ppActionNextSlide	1
ppActionNone	0

ppActionOLEVerb	11
ppActionPlay	12
ppActionPreviousSlide	2
ppActionRunMacro	8
ppActionRunProgram	9

PpAdvanceMode

Constant	Value
ppAdvanceModeMixed	-2
ppAdvanceOnClick	1
ppAdvanceOnTime	2

PpAfterEffect

Constant	Value
ppAfterEffectDim	2
ppAfterEffectHide	1
ppAfterEffectHideOnClick	3
ppAfterEffectMixed	-2
ppAfterEffectNothing	0

PpAlertLevel

Constant	Value
ppAlertsAll	2
ppAlertsNone	1

PpArrangeStyle

Constant	Value
ppArrangeCascade	2
ppArrangeTiled	1

PpAutoSize

Constant	Value
ppAutoSizeMixed	-2
ppAutoSizeNone	0
ppAutoSizeShapeToFitText	1

PpBaselineAlignment

Constant	Value
ppBaselineAlignBaseline	1
ppBaselineAlignCenter	3
ppBaselineAlignFarEast50	4
ppBaselineAlignMixed	-2
ppBaselineAlignTop	2

PpBorderType

Constant	Value
ppBorderBottom	3
ppBorderDiagonalDown	5
ppBorderDiagonalUp	6
ppBorderLeft	2
ppBorderRight	4
ppBorderTop	1

PpBulletType

Constant	Value
ppBulletMixed	-2
ppBulletNone	0
ppBulletNumbered	2
ppBulletPicture	3
ppBulletUnnumbered	1

[PpChangeCase](#)

Constant	Value
ppCaseLower	2
ppCaseSentence	1
ppCaseTitle	4
ppCaseToggle	5
ppCaseUpper	3

[PpChartUnitEffect](#)

Constant	Value
ppAnimateByCategory	2
ppAnimateByCategoryElements	4
ppAnimateBySeries	1
ppAnimateBySeriesElements	3
ppAnimateChartAllAtOnce	5
ppAnimateChartMixed	-2

[PpColorSchemeIndex](#)

Constant	Value
ppAccent1	6
ppAccent2	7
ppAccent3	8
ppBackground	1
ppFill	5
ppForeground	2
ppNotSchemeColor	0
ppSchemeColorMixed	-2
ppShadow	3
ppTitle	4

[PpDateTimeFormat](#)

Constant	Value
ppDateTimeddddMMMMddyyyy	2
ppDateTimedMMMMyyyy	3
ppDateTimedMMMyy	5
ppDateTimeFigureOut	14
ppDateTimeFormatMixed	-2
ppDateTimeHmm	10
ppDateTimehmAMPM	12
ppDateTimeHmss	11
ppDateTimehmssAMPM	13
ppDateTimeMdy	1
ppDateTimeMMddyHmm	8
ppDateTimeMMddyhmmAMPM	9
ppDateTimeMMMMdyyyy	4
ppDateTimeMMMMyy	6
ppDateTimeMMyy	7

PpDirection

Constant	Value
ppDirectionLeftToRight	1
ppDirectionMixed	-2
ppDirectionRightToLeft	2

PpEntryEffect

Constant	Value
ppEffectAppear	3844
ppEffectBlindsHorizontal	769
ppEffectBlindsVertical	770
ppEffectBoxIn	3074
ppEffectBoxOut	3073
ppEffectCheckerboardAcross	1025

ppEffectCheckerboardDown	1026
ppEffectCircleOut	3845
ppEffectCombHorizontal	3847
ppEffectCombVertical	3848
ppEffectCoverDown	1284
ppEffectCoverLeft	1281
ppEffectCoverLeftDown	1287
ppEffectCoverLeftUp	1285
ppEffectCoverRight	1283
ppEffectCoverRightDown	1288
ppEffectCoverRightUp	1286
ppEffectCoverUp	1282
ppEffectCrawlFromDown	3344
ppEffectCrawlFromLeft	3341
ppEffectCrawlFromRight	3343
ppEffectCrawlFromUp	3342
ppEffectCut	257
ppEffectCutThroughBlack	258
ppEffectDiamondOut	3846
ppEffectDissolve	1537
ppEffectFade	1793
ppEffectFadeSmoothly	3849
ppEffectFlashOnceFast	3841
ppEffectFlashOnceMedium	3842
ppEffectFlashOnceSlow	3843
ppEffectFlyFromBottom	3332
ppEffectFlyFromBottomLeft	3335
ppEffectFlyFromBottomRight	3336
ppEffectFlyFromLeft	3329
ppEffectFlyFromRight	3331
ppEffectFlyFromTop	3330
ppEffectFlyFromTopLeft	3333
ppEffectFlyFromTopRight	3334

ppEffectMixed	-2
ppEffectNewsflash	3850
ppEffectNone	0
ppEffectPeekFromDown	3338
ppEffectPeekFromLeft	3337
ppEffectPeekFromRight	3339
ppEffectPeekFromUp	3340
ppEffectPlusOut	3851
ppEffectPushDown	3852
ppEffectPushLeft	3853
ppEffectPushRight	3854
ppEffectPushUp	3855
ppEffectRandom	513
ppEffectRandomBarsHorizontal	2305
ppEffectRandomBarsVertical	2306
ppEffectSpiral	3357
ppEffectSplitHorizontalIn	3586
ppEffectSplitHorizontalOut	3585
ppEffectSplitVerticalIn	3588
ppEffectSplitVerticalOut	3587
ppEffectStretchAcross	3351
ppEffectStretchDown	3355
ppEffectStretchLeft	3352
ppEffectStretchRight	3354
ppEffectStretchUp	3353
ppEffectStripsDownLeft	2563
ppEffectStripsDownRight	2564
ppEffectStripsLeftDown	2567
ppEffectStripsLeftUp	2565
ppEffectStripsRightDown	2568
ppEffectStripsRightUp	2566
ppEffectStripsUpLeft	2561
ppEffectStripsUpRight	2562

ppEffectSwivel	3356
ppEffectUncoverDown	2052
ppEffectUncoverLeft	2049
ppEffectUncoverLeftDown	2055
ppEffectUncoverLeftUp	2053
ppEffectUncoverRight	2051
ppEffectUncoverRightDown	2056
ppEffectUncoverRightUp	2054
ppEffectUncoverUp	2050
ppEffectWedge	3856
ppEffectWheel1Spoke	3857
ppEffectWheel2Spokes	3858
ppEffectWheel3Spokes	3859
ppEffectWheel4Spokes	3860
ppEffectWheel8Spokes	3861
ppEffectWipeDown	2820
ppEffectWipeLeft	2817
ppEffectWipeRight	2819
ppEffectWipeUp	2818
ppEffectZoomBottom	3350
ppEffectZoomCenter	3349
ppEffectZoomIn	3345
ppEffectZoomInSlightly	3346
ppEffectZoomOut	3347
ppEffectZoomOutSlightly	3348

 **PpFarEastLineBreakLevel**

Constant	Value
ppFarEastLineBreakLevelCustom	3
ppFarEastLineBreakLevelNormal	1
ppFarEastLineBreakLevelStrict	2

[PpFollowColors](#)

Constant	Value
ppFollowColorsMixed	-2
ppFollowColorsNone	0
ppFollowColorsScheme	1
ppFollowColorsTextAndBackground	2

[PpFrameColors](#)

Constant	Value
ppFrameColorsBlackTextOnWhite	5
ppFrameColorsBrowserColors	1
ppFrameColorsPresentationSchemeAccentColor	3
ppFrameColorsPresentationSchemeTextColor	2
ppFrameColorsWhiteTextOnBlack	4

[PpHTMLVersion](#)

Constant	Value
ppHTMLAutodetect	4
ppHTMLDual	3
ppHTMLv3	1
ppHTMLv4	2

[PpIndentControl](#)

Constant	Value
ppIndentControlMixed	-2
ppIndentKeepAttr	2
ppIndentReplaceAttr	1

[PpMediaType](#)

Constant	Value
ppMediaTypeMixed	-2
ppMediaTypeMovie	3
ppMediaTypeOther	1
ppMediaTypeSound	2

[PpMouseActivation](#)

Constant	Value
ppMouseClicked	1
ppMouseOver	2

[PpNumberedBulletStyle](#)

Constant	Value
ppBulletAlphaLCParenBoth	8
ppBulletAlphaLCParenRight	9
ppBulletAlphaLCPeriod	0
ppBulletAlphaUCParenBoth	10
ppBulletAlphaUCParenRight	11
ppBulletAlphaUCPeriod	1
ppBulletArabicAbjadDash	24
ppBulletArabicAlphaDash	23
ppBulletArabicDBPeriod	29
ppBulletArabicDBPlain	28
ppBulletArabicParenBoth	12
ppBulletArabicParenRight	2
ppBulletArabicPeriod	3
ppBulletArabicPlain	13
ppBulletCircleNumDBPlain	18
ppBulletCircleNumWDBlackPlain	20
ppBulletCircleNumWDWhitePlain	19
ppBulletHebrewAlphaDash	25
ppBulletHindiAlpha1Period	40

ppBulletHindiAlphaPeriod	36
ppBulletHindiNumParenRight	39
ppBulletHindiNumPeriod	37
ppBulletKanjiKoreanPeriod	27
ppBulletKanjiKoreanPlain	26
ppBulletKanjiSimpChinDBPeriod	38
ppBulletRomanLCParenBoth	4
ppBulletRomanLCParenRight	5
ppBulletRomanLCPeriod	6
ppBulletRomanUCParenBoth	14
ppBulletRomanUCParenRight	15
ppBulletRomanUCPeriod	7
ppBulletSimpChinPeriod	17
ppBulletSimpChinPlain	16
ppBulletStyleMixed	-2
ppBulletThaiAlphaParenBoth	32
ppBulletThaiAlphaParenRight	31
ppBulletThaiAlphaPeriod	30
ppBulletThaiNumParenBoth	35
ppBulletThaiNumParenRight	34
ppBulletThaiNumPeriod	33
ppBulletTradChinPeriod	22
ppBulletTradChinPlain	21

PpParagraphAlignment

Constant	Value
ppAlignCenter	2
ppAlignDistribute	5
ppAlignJustify	4
ppAlignJustifyLow	7
ppAlignLeft	1
ppAlignmentMixed	-2

ppAlignRight 3
ppAlignThaiDistribute 6

PpPasteDataType

Constant	Value
ppPasteBitmap	1
ppPasteDefault	0
ppPasteEnhancedMetafile	2
ppPasteGIF	4
ppPasteHTML	8
ppPasteJPG	5
ppPasteMetafilePicture	3
ppPasteOLEObject	10
ppPastePNG	6
ppPasteRTF	9
ppPasteShape	11
ppPasteText	7

PpPlaceholderType

Constant	Value
ppPlaceholderBitmap	9
ppPlaceholderBody	2
ppPlaceholderCenterTitle	3
ppPlaceholderChart	8
ppPlaceholderDate	16
ppPlaceholderFooter	15
ppPlaceholderHeader	14
ppPlaceholderMediaClip	10
ppPlaceholderMixed	-2
ppPlaceholderObject	7
ppPlaceholderOrgChart	11
ppPlaceholderSlideNumber	13

ppPlaceholderSubtitle	4
ppPlaceholderTable	12
ppPlaceholderTitle	1
ppPlaceholderVerticalBody	6
ppPlaceholderVerticalTitle	5

[PpPrintColorType](#)

Constant	Value
ppPrintBlackAndWhite	2
ppPrintColor	1
ppPrintPureBlackAndWhite	3

[PpPrintHandoutOrder](#)

Constant	Value
ppPrintHandoutHorizontalFirst	2
ppPrintHandoutVerticalFirst	1

[PpPrintOutputType](#)

Constant	Value
ppPrintOutputBuildSlides	7
ppPrintOutputFourSlideHandouts	8
ppPrintOutputNineSlideHandouts	9
ppPrintOutputNotesPages	5
ppPrintOutputOneSlideHandouts	10
ppPrintOutputOutline	6
ppPrintOutputSixSlideHandouts	4
ppPrintOutputSlides	1
ppPrintOutputThreeSlideHandouts	3
ppPrintOutputTwoSlideHandouts	2

[PpPrintRangeType](#)

Constant	Value
ppPrintAll	1
ppPrintCurrent	3
ppPrintNamedSlideShow	5
ppPrintSelection	2
ppPrintSlideRange	4

[**PpPublishSourceType**](#)

Constant	Value
ppPublishAll	1
ppPublishNamedSlideShow	3
ppPublishSlideRange	2

[**PpRevisionInfo**](#)

Constant	Value
ppRevisionInfoBaseline	1
ppRevisionInfoMerged	2
ppRevisionInfoNone	0

[**PpSaveAsFileType**](#)

Constant	Value
ppSaveAsAddIn	8
ppSaveAsBMP	19
ppSaveAsDefault	11
ppSaveAsEMF	23
ppSaveAsGIF	16
ppSaveAsHTML	12
ppSaveAsHTMLDual	14
ppSaveAsHTMLv3	13
ppSaveAsJPG	17
ppSaveAsMetaFile	15

ppSaveAsPNG	18
ppSaveAsPowerPoint3	4
ppSaveAsPowerPoint4	3
ppSaveAsPowerPoint4FarEast	10
ppSaveAsPowerPoint7	2
ppSaveAsPresentation	1
ppSaveAsPresForReview	22
ppSaveAsRTF	6
ppSaveAsShow	7
ppSaveAsTemplate	5
ppSaveAsTIF	21
ppSaveAsWebArchive	20

[PpSelectionType](#)

Constant	Value
ppSelectionNone	0
ppSelectionShapes	2
ppSelectionSlides	1
ppSelectionText	3

[PpSlideLayout](#)

Constant	Value
ppLayoutBlank	12
ppLayoutChart	8
ppLayoutChartAndText	6
ppLayoutClipartAndText	10
ppLayoutClipArtAndVerticalText	26
ppLayoutFourObjects	24
ppLayoutLargeObject	15
ppLayoutMediaClipAndText	18
ppLayoutMixed	-2
ppLayoutObject	16

ppLayoutObjectAndText	14
ppLayoutObjectAndTwoObjects	30
ppLayoutObjectOverText	19
ppLayoutOrgchart	7
ppLayoutTable	4
ppLayoutText	2
ppLayoutTextAndChart	5
ppLayoutTextAndClipart	9
ppLayoutTextAndMediaClip	17
ppLayoutTextAndObject	13
ppLayoutTextAndTwoObjects	21
ppLayoutTextOverObject	20
ppLayoutTitle	1
ppLayoutTitleOnly	11
ppLayoutTwoColumnText	3
ppLayoutTwoObjects	29
ppLayoutTwoObjectsAndObject	31
ppLayoutTwoObjectsAndText	22
ppLayoutTwoObjectsOverText	23
ppLayoutVerticalText	25
ppLayoutVerticalTitleAndText	27
ppLayoutVerticalTitleAndTextOverChart	28

 [**PpSlideShowAdvanceMode**](#)

Constant	Value
ppSlideShowManualAdvance	1
ppSlideShowRehearseNewTimings	3
ppSlideShowUseSlideTimings	2

 [**PpSlideShowPointerType**](#)

Constant	Value
ppSlideShowPointerAlwaysHidden	3

ppSlideShowPointerArrow	1
ppSlideShowPointerAutoArrow	4
ppSlideShowPointerEraser	5
ppSlideShowPointerNone	0
ppSlideShowPointerPen	2

 **PpSlideShowRangeType**

Constant	Value
ppShowAll	1
ppShowNamedSlideShow	3
ppShowSlideRange	2

 **PpSlideShowState**

Constant	Value
ppSlideShowBlackScreen	3
ppSlideShowDone	5
ppSlideShowPaused	2
ppSlideShowRunning	1
ppSlideShowWhiteScreen	4

 **PpSlideShowType**

Constant	Value
ppShowTypeKiosk	3
ppShowTypeSpeaker	1
ppShowTypeWindow	2

 **PpSlideSizeType**

Constant	Value
ppSlideSize35MM	4
ppSlideSizeA3Paper	9

ppSlideSizeA4Paper	3
ppSlideSizeB4ISOPaper	10
ppSlideSizeB4JISPaper	12
ppSlideSizeB5ISOPaper	11
ppSlideSizeB5JISPaper	13
ppSlideSizeBanner	6
ppSlideSizeCustom	7
ppSlideSizeHagakiCard	14
ppSlideSizeLedgerPaper	8
ppSlideSizeLetterPaper	2
ppSlideSizeOnScreen	1
ppSlideSizeOverhead	5

 [**PpSoundEffectType**](#)

Constant	Value
ppSoundEffectsMixed	-2
ppSoundFile	2
ppSoundNone	0
ppSoundStopPrevious	1

 [**PpSoundFormatType**](#)

Constant	Value
ppSoundFormatCDAudio	3
ppSoundFormatMIDI	2
ppSoundFormatMixed	-2
ppSoundFormatNone	0
ppSoundFormatWAV	1

 [**PpTabStopType**](#)

Constant	Value
ppTabStopCenter	2

ppTabStopDecimal 4
ppTabStopLeft 1
ppTabStopMixed -2
ppTabStopRight 3

[PpTextLevelEffect](#)

Constant	Value
ppAnimateByAllLevels	16
ppAnimateByFifthLevel	5
ppAnimateByFirstLevel	1
ppAnimateByFourthLevel	4
ppAnimateBySecondLevel	2
ppAnimateByThirdLevel	3
ppAnimateLevelMixed	-2
ppAnimateLevelNone	0

[PpTextStyleType](#)

Constant	Value
ppBodyStyle	3
ppDefaultStyle	1
ppTitleStyle	2

[PpTextUnitEffect](#)

Constant	Value
ppAnimateByCharacter	2
ppAnimateByParagraph	0
ppAnimateByWord	1
ppAnimateUnitMixed	-2

[PpTransitionSpeed](#)

Constant	Value
ppTransitionSpeedFast	3
ppTransitionSpeedMedium	2
ppTransitionSpeedMixed	-2
ppTransitionSpeedSlow	1

PpUpdateOption

Constant	Value
ppUpdateOptionAutomatic	2
ppUpdateOptionManual	1
ppUpdateOptionMixed	-2

PpViewType

Constant	Value
ppViewHandoutMaster	4
ppViewMasterThumbnails	12
ppViewNormal	9
ppViewNotesMaster	5
ppViewNotesPage	3
ppViewOutline	6
ppViewPrintPreview	10
ppViewSlide	1
ppViewSlideMaster	2
ppViewSlideSorter	7
ppViewThumbnails	11
ppViewTitleMaster	8

PpWindowState

Constant	Value
ppWindowMaximized	3
ppWindowMinimized	2

ppWindowNormal 1

[Show All](#)

Adding controls to a document

To add [controls](#) to a document, display the **Control Toolbox**, click the control you want to add, and then click on the document. Drag an adjustment handle of the control until the control's outline is the size and shape you want.

Note Dragging a control (or a number of "grouped" controls) from the form back to the **Control Toolbox** creates a template of that control, which can be reused. This is a useful feature for implementing a standard "look and feel" for your applications.



[Show All](#)

Setting control properties

You can set some [control](#) properties at design time (before any macro is running). In [design mode](#), right-click a control and click **Properties** to display the Properties window. Property names are shown in the left column in the window, property values in the right column. You set a property value by entering the new value to the right of the property name.



[Show All](#)

Initializing control properties

You can initialize [controls](#) at run time by using Visual Basic code in a macro. For example, you could fill a list box, set text values, or set option buttons.

The following example uses the **AddItem** method to add data to a list box. Then it sets the value of a text box and displays the form.

```
Private Sub GetUserName()  
    With UserForm1  
        .lstRegions.AddItem "North"  
        .lstRegions.AddItem "South"  
        .lstRegions.AddItem "East"  
        .lstRegions.AddItem "West"  
        .txtSalesPersonID.Text = "00000"  
        .Show  
        ' ...  
    End With  
End Sub
```

You can also use code in the Initialize event of a form to set initial values for controls on the form. An advantage to setting initial control values in the Initialize event is that the initialization code stays with the form. You can copy the form to another project, and when you run the **Show** method to display the dialog box, the controls will be initialized.

```
Private Sub UserForm_Initialize()  
    With UserForm1  
        With .lstRegions  
            .AddItem "North"  
            .AddItem "South"  
            .AddItem "East"  
            .AddItem "West"  
        End With  
        .txtSalesPersonID.Text = "00000"  
    End With  
End Sub
```



[Show All](#)

Control and dialog box events

After you have added [controls](#) to your dialog box or document, you add event procedures to determine how the controls respond to user actions.

UserForms and controls have a predefined set of events. For example, a command button has a Click event that occurs when the user clicks the command button, and UserForms have an Initialize event that runs when the form is loaded.

To write a control or form event procedure, open a module by double-clicking the form or control, and select the event from the **Procedure** drop-down list box.

Event procedures include the name of the control. For example, the name of the Click event procedure for a command button named Command1 is Command1_Click.

If you add code to an event procedure and then change the name of the control, your code remains in procedures with the previous name.

For example, assume you add code to the Click event for Command1 and then rename the control to Command2. When you double-click Command2, you will not see any code in the Click event procedure. You will need to move code from Command1_Click to Command2_Click.

To simplify development, it is a good practice to name your controls before writing code.



[Show All](#)

Using control values while code is running

Some [control](#) properties can be set and returned while Visual Basic code is running. The following example sets the [Text](#) property of a text box to "Hello."

```
TextBox1.Text = "Hello"
```

The data entered on a form by a user is lost when the form is closed. If you return the values of controls on a form after the form has been unloaded, you get the initial values for the controls rather than the values the user entered.

If you want to save the data entered on a form, you can save the information to module-level variables while the form is still running. The following example displays a form and saves the form data.

```
'Code in module to declare public variables
Public strRegion As String
Public intSalesPersonID As Integer
Public blnCancelled As Boolean

'Code in form
Private Sub cmdCancel_Click()
    Module1.blnCancelled = True
    Unload Me
End Sub

Private Sub cmdOK_Click()
    'Save data
    intSalesPersonID = txtSalesPersonID.Text
    strRegion = lstRegions.List(lstRegions.ListIndex)
    Module1.blnCancelled = False
    Unload Me
End Sub

Private Sub UserForm_Initialize()
    Module1.blnCancelled = True
End Sub

'Code in module to display form
Sub LaunchSalesPersonForm()
```

```
frmSalesPeople.Show
If blnCancelled = True Then
    MsgBox "Operation Cancelled!", vbExclamation
Else
    MsgBox "The Salesperson's ID is: " &
        intSalesPersonID & _
        "The Region is: " & strRegion
End If
End Sub
```



Creating a UserForm

To create a custom dialog box, you must create a UserForm. To create a UserForm, click **UserForm** on the **Insert** menu in the Visual Basic Editor.

Use the Properties window to change the name, behavior, and appearance of the form. For example, to change the caption on a form, set the **Caption** property.



[Show All](#)

Adding controls to a UserForm

To add [controls](#) to a user form, find the control you want to add in the **Toolbox**, drag the control onto the form, and then drag an adjustment handle on the control until the control's outline is the size and shape you want.

Note Dragging a control (or a number of "grouped" controls) from the form back to the **Toolbox** creates a template of that control, which can be reused. This is a useful feature for implementing a standard "look and feel" for your applications.

When you've added controls to the form, use the commands on the **Format** menu in the Visual Basic Editor to adjust the control alignment and spacing.



Displaying a custom dialog box

To test your dialog box in the Visual Basic Editor, click **Run Sub/UserForm** on the **Run** menu in the Visual Basic Editor.

To display a dialog box from Visual Basic, use the **Show** method. The following example displays the dialog box named UserForm1.

```
Private Sub GetUserName()  
    UserForm1.Show  
End Sub
```



[Show All](#)

AddOLEObject Method

Creates an OLE object. Returns a [Shape](#) object that represents the new OLE object.

expression.AddOLEObject(*Left*, *Top*, *Width*, *Height*, *ClassName*, *FileName*, *DisplayAsIcon*, *IconFileName*, *IconIndex*, *IconLabel*, *Link*)

expression Required. An expression that returns a **Shapes** object.

Left , **Top** Optional **Float**. The position (in points) of the upper-left corner of the new object relative to the upper-left corner of the slide. The default value is 0 (zero).

Width , **Height** Optional **Float**. The initial dimensions of the OLE object, in points.

ClassName Optional **String**. The OLE long class name or the ProgID for the object that's to be created. You must specify either the **ClassName** or **FileName** argument for the object, but not both.

FileName Optional **String**. The file from which the object is to be created. If the path isn't specified, the current working folder is used. You must specify either the **ClassName** or **FileName** argument for the object, but not both.

DisplayAsIcon Optional [MsoTriState](#). Determines whether the OLE object will be displayed as an icon.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default.

msoTriStateMixed

msoTriStateToggle

msoTrue Displays the OLE object as an icon.

IconFileName Optional **String**. The file that contains the icon to be displayed.

IconIndex Optional **Integer**. The index of the icon within **IconFileName**. The order of icons in the specified file corresponds to the order in which the icons appear in the **Change Icon** dialog box (accessed from the **Insert Object** dialog box when the **Display as icon** check box is selected). The first icon in the file has the index number 0 (zero). If an icon with the given index number doesn't exist in **IconFileName**, the icon with the index number 1 (the second icon in the file) is used. The default value is 0 (zero).

IconLabel Optional **String**. A label (caption) to be displayed beneath the icon.

Link Optional **MsoTriState**. Determines whether the OLE object will be linked to the file from which it was created. If you specified a value for **ClassName**, this argument must be **msoFalse**.

MsoTriState can be one of these MsoTriState constants.

msoCTrue

msoFalse Default. Makes the OLE object an independent copy of the file.

msoTriStateMixed

msoTriStateToggle

msoTrue Links the OLE object to the file from which it was created.

Example

This example adds a linked Word document to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddOLEObject Left:=100, Top:=100, _
    Width:=200, Height:=300, _
    FileName:="c:\my documents\testing.doc", Link:=msoTrue
```

This example adds a new Microsoft Excel worksheet to myDocument. The worksheet will be displayed as an icon.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddOLEObject Left:=100, Top:=100, _
    Width:=200, Height:=300, _
    ClassName:="Excel.Sheet", DisplayAsIcon:=True
```

This example adds a command button to myDocument.

```
Set myDocument = ActivePresentation.Slides(1)
myDocument.Shapes.AddOLEObject Left:=100, Top:=100, _
    Width:=150, Height:=50, ClassName:="Forms.CommandButton.1"
```



Hyperlink Object

Multiple objects [└Hyperlinks](#)
[└Hyperlink](#)

Represents a hyperlink associated with a non-placeholder shape or text. You can use a hyperlink to jump to an Internet or intranet site, to another file, or to a slide within the active presentation. The **Hyperlink** object is a member of the [Hyperlinks](#) collection. The **Hyperlinks** collection contains all the hyperlinks on a slide or a master.

Using the Hyperlink Object

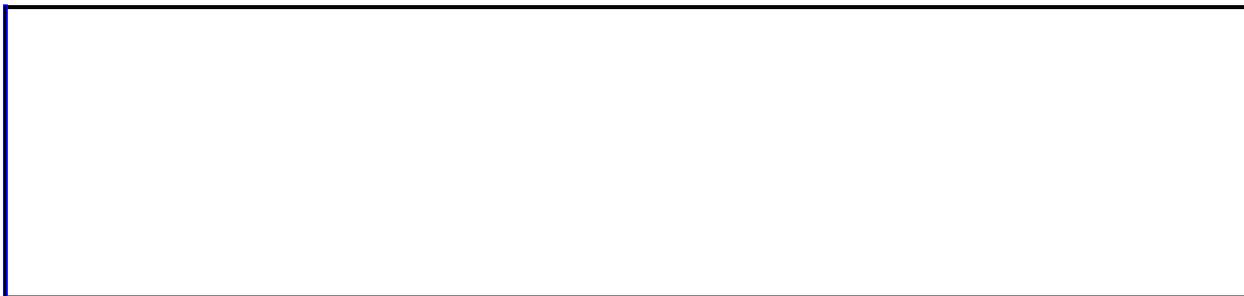
Use the [Hyperlink](#) property to return a hyperlink for a shape. A shape can have two different hyperlinks assigned to it: one that's followed when the user clicks the shape during a slide show, and another that's followed when the user passes the mouse pointer over the shape during a slide show. For the hyperlink to be active during a slide show, the **Action** property must be set to **ppActionHyperlink**. The following example sets the mouse-click action for shape three on slide one in the active presentation to an Internet link.

```
With ActivePresentation.Slides(1).Shapes(3) _  
    .ActionSettings(ppMouseClick)  
    .Action = ppActionHyperlink  
    .Hyperlink.Address = "http://www.microsoft.com"  
End With
```

A slide can contain more than one hyperlink. Each non-placeholder shape can have a hyperlink; the text within a shape can have its own hyperlink; and each individual character can have its own hyperlink. Use **Hyperlinks(index)**, where *index* is the hyperlink number, to return a single **Hyperlink** object. The following example adds the shape three mouse-click hyperlink to the Favorites folder.

```
ActivePresentation.Slides(1).Shapes(3) _  
    .ActionSettings(ppMouseClick).Hyperlink.AddToFavorites
```

Note When you use this method to add a hyperlink to the Microsoft Internet Explorer Favorites folder, an icon is added to the **Favorites** menu without a corresponding name. You must add the name from within Internet Explorer.



[Show All](#)

Returning an Object from a Collection

The **Item** method returns a single object from a collection. The following example sets the `firstPres` variable to a **Presentation** object that represents presentation one.

```
Set firstPres = Presentations.Item(1)
```

The **Item** method is the [default method](#) for most collections, so you can write the same statement more concisely by omitting the **Item** keyword.

```
Set firstPres = Presentations(1)
```

For more information about a specific collection, see the Help topic for that collection or the **Item** method for the collection.

Named Objects

Although you can usually specify an integer value with the **Item** method, it may be more convenient to return an object by name. Many objects are given automatically generated names when they are created. For example, the first slide you create will be automatically named "Slide1." If the first two shapes you create are a rectangle and an oval, their default names will be "Rectangle 1" and "Oval 2". You may want to give an object a more meaningful name to make it easier to refer to later. Most often, this is done by setting the object's **Name** property. The following example sets a meaningful name for a slide as it is added. The name can then be used instead of the index number to refer to the slide.

```
ActivePresentation.Slides.Add(1, 1).Name = "Home Page Slide"
With ActivePresentation.Slides("Home Page Slide")
    .FollowMasterBackground = False
    .Background.Fill.PresetGradient _
        msoGradientDiagonalDown, 1, msoGradientBrass
End With
```

Predefined Index Values

Some collections have predefined index values you can use to return single objects. Each predefined index value is represented by a constant. For example, you specify a **PpTextStyleType** constant with the **Item** method of the **TextStyles** collection to return a single text style.

The following example sets the margins for the body area on slides in the active presentation.

```
With ActivePresentation.SlideMaster _
    .TextStyles(ppBodyStyle).TextFrame
    .MarginBottom = 50
    .MarginLeft = 50
    .MarginRight = 50
    .MarginTop = 50
End With
```



ScaleEffect Object

[AnimationBehavior](#) \perp [ScaleEffect](#)

Represents a scaling effect for an [AnimationBehavior](#) object.

Using the ScaleEffect object

Use the [ScaleEffect](#) property of the **AnimationBehavior** object to return a **ScaleEffect** object. The following example refers to the scale effect for a given animation behavior.

```
ActivePresentation.Slides(1).TimeLine.MainSequence.Item.Behaviors(1)
```

Use the [ByX](#), [ByY](#), [FromX](#), [FromY](#), [ToX](#), and [ToY](#) properties of the **ScaleEffect** object to manipulate an object's scale. This example scales the first shape on the first slide starting at zero increasing in size until it reaches 100 percent of its original size. This example assumes that there is a shape on the first slide.

```
Sub ChangeScale()  
  
    Dim shpFirst As Shape  
    Dim effNew As Effect  
    Dim aniScale As AnimationBehavior  
  
    Set shpFirst = ActivePresentation.Slides(1).Shapes(1)  
    Set effNew = ActivePresentation.Slides(1).TimeLine.MainSequence  
        .AddEffect(Shape:=shpFirst, effectId:=msoAnimEffectCustom)  
    Set aniScale = effNew.Behaviors.Add(msoAnimTypeScale)  
  
    With aniScale.ScaleEffect  
  
        'Starting size  
        .FromX = 0  
        .FromY = 0  
  
        'Size after scale effect  
        .ToX = 100  
        .ToY = 100  
    End With  
  
End Sub
```



Hyperlinks Collection Object

Multiple objects [└Hyperlinks](#)
[└Hyperlink](#)

A collection of all the [Hyperlink](#) objects on a slide or master.

Using the Hyperlinks Collection

Use the [Hyperlinks](#) property to return the **Hyperlinks** collection. The following example updates all hyperlinks on slide one in the active presentation that have the specified address.

```
For Each hl In ActivePresentation.Slides(1).Hyperlinks
    If hl.Address = "c:\current work\sales.ppt" Then
        hl.Address = "c:\new\nwsales.ppt"
    End If
Next
```

Use the [Hyperlink](#) property to create a hyperlink and add it to the **Hyperlinks** collection. The following example sets a hyperlink that will be followed when the user clicks shape three on slide one in the active presentation during a slide show and adds the new hyperlink to the collection. Note that if shape three already has a mouse-click hyperlink defined, the following example will delete this hyperlink from the collection when it adds the new one, so the number of items in the **Hyperlinks** collection won't change.

```
With ActivePresentation.Slides(1).Shapes(3) _
    .ActionSettings(ppMouseClick)
    .Action = ppActionHyperlink
    .Hyperlink.Address = "http://www.microsoft.com"
End With
```

