Add Method (AddIns Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddAddInsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddAddInsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddAddInsObjA "}

Adds the file specified by ***FileName*** to the list of available add-ins.

Syntax

expression.**Add(*FileName, Install*)**

expression Required. An expression that returns an **AddIns** object.

FileName Required **String**. The path for the template or WLL.

Install Optional **Variant**. **True** to install the add-in. **False** to add the add-in to the list of add-ins but not install it. The default value is **True**.

Remarks

Use the **Installed** property of an add-in to see whether it's already installed.

Add Method (AddIns Collection) Example

This example installs a template named "MyFax.dot" and adds it to the list of add-ins in the **Templates and Add-ins** dialog box.

```
AddIns.Add FileName:="C:\Program Files\Microsoft Office\Templates\Letters & Faxes\MyFax.dot", Install:=True
```

This Macintosh example adds MYTEMPLATE to the list of add-ins in the **Templates and Add-ins** dialog box.

```
AddIns.Add FileName:="HD:TEMPLATE:LIB MYTEMPLATE"
```

Add Method (AutoCorrectEntries Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddAutoCorrectEntriesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddAutoCorrectEntriesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddAutoCorrectEntriesObjA "}

Adds a plain-text AutoCorrect entry to the list of available entries.

Syntax

expression.**Add(*Name, Value*)**

expression Required. An expression that returns an **AutoCorrectEntries** object.

Name Required **String**. The text you want to have automatically replaced with the text specified by ***Value***.

Value Required **String**. The text you want to have automatically inserted whenever the text specified by ***Name*** is typed.

Remarks

Use the **AddRichText** method to create a formatted AutoCorrect entry.

Add Method (AutoCorrectEntries Collection) Example

This example adds a plain-text AutoCorrect entry for a common misspelling of the word "their."

```
AutoCorrect.Entries.Add Name:= "thier", Value:= "their"
```

Add Method (AutoTextEntries Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddAutoTextEntriesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddAutoTextEntriesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddAutoTextEntriesObjA "}

Adds an AutoText entry to the list of available entries.

Syntax

expression.**Add(*Name, Range*)**

expression Required. An expression that returns an **AutoTextEntries** object.

Name Required **String**. The text that, when typed, initiates an AutoText entry.

Range Required **Range**. A range of text that will be inserted whenever ***Name*** is typed.

Add Method (AutoTextEntries Collection) Example

This example adds an AutoText entry named "MyText" that contains the text in the selection.

```
NormalTemplate.AutoTextEntries.Add Name:= "MyText", _
    Range:= Selection.Range
```

Add Method (Bookmarks Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddBookmarksObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddBookmarksObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddBookmarksObjA "}

Adds a bookmark to a range.

Syntax

expression.**Add(*Name, Range*)**

expression Required. An expression that returns a **Bookmarks** object.

Name Required **String**. The name of the bookmark. The name cannot be more than one word.

Range Optional **Variant**. The range of text marked by the bookmark. A bookmark can be set to a collapsed range (the insertion point).

Add Method (Bookmarks Collection) Example

This example adds a bookmark named "myplace" to the selection in the active document.

```
ActiveDocument.Bookmarks.Add Name:= "myplace", Range:= Selection.Range
```

This example adds a bookmark named "mark" at the insertion point.

```
ActiveDocument.Bookmarks.Add Name:= "mark"
```

This example adds a bookmark named "third_para" to the third paragraph in "Letter.doc," and then it displays all the bookmarks for the document in the active window.

```
Set myDoc = Documents("Letter.doc")
myDoc.Bookmarks.Add Name:="third_para", _
    Range:=myDoc.Paragraphs(3).Range
myDoc.ActiveWindow.View.ShowBookmarks = True
```

Add Method (CaptionLabels Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCaptionLabelsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCaptionLabelsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCaptionLabelsObjA "}

Adds a custom caption label.

Syntax

expression.**Add(*Name*)**

expression Required. An expression that returns a **CaptionLabels** object.

Name Required **String**. The name of the custom caption label.

Add Method (CaptionLabels Collection) Example

This example adds a custom caption label named "Demo Slide."

```
CaptionLabels.Add Name:="Demo Slide"
```

Add Method (CustomLabels Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCustomLabelsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCustomLabelsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCustomLabelsObjA "}

Adds a custom mailing label.

Syntax

expression.**Add(*Name, DotMatrix*)**

expression Required. An expression that returns a **CustomLabels** object.

Name Required **String**. The name for the custom mailing labels.

DotMatrix Optional **Variant**. **True** to have the mailing labels printed on a dot-matrix printer.

Add Method (CustomLabels Collection) Example

This example adds a custom mailing label named "My Label," sets the page size, and then creates a page of these labels.

```
Set ml = Application.MailingLabel.CustomLabels.Add(Name:="My Label", _
    DotMatrix:=False)
ml.PageSize = wdCustomLabelLetter
addr = "Dave Edson" & vbCr & "123 Skye St." & vbCr & "Our Town, WA  98004"
Application.MailingLabel.CreateNewDocument _
    Name:="My Label", Address:=addr, ExtractAddress:=False
```

Add Method (Cells, Rows, and Columns Collections)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCellsRowsColumnsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCellsRowsColumnsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCellsRowsColumnsObjA "}

Adds a cell, row, or column to a table.

Syntax 1

expression.**Add(*BeforeCell*)**

Syntax 2

expression.**Add(*BeforeColumn*)**

Syntax 3

expression.**Add(*BeforeRow*)**

expression Required. An expression that returns a **Cells** object (Syntax 1), **Columns** object (Syntax 2), or **Rows** object (Syntax 3).

BeforeCell Optional **Variant**. A **Cell** object that represents the cell that will appear immediately to the right of the new cell or cells.

BeforeColumn Optional **Variant**. A **Column** object that represents the column that will appear immediately to the right of the new column.

BeforeRow Optional **Variant**. A **Row** object that represents the row that will appear immediately below the new row.

Add Method (Cells, Rows, and Columns Collections) Example

This example inserts a new row before the first row in the selection.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows.Add BeforeRow:= Selection.Rows(1)
End If
```

This example creates a table with two columns and two rows in the active document and then adds another column before the first column. The width of the new column is set at 1.5 inches.

```
Set myTable = ActiveDocument.Tables.Add(Selection.Range, 2, 2)
Set newCol = myTable.Columns.Add(BeforeColumn:= myTable.Columns(1))
newCol.SetWidth ColumnWidth:=InchesToPoints(1.5), RulerStyle:=wdAdjustNone
```

This example adds a row to the first table and then inserts the text "Cell" into this row.

```
counter = 1
Set myTable = ActiveDocument.Tables(1)
Set newrow = myTable.Rows.Add(BeforeRow:=myTable.rows(1))
For Each c In newrow.Cells
   c.Range.InsertAfter Text:="Cell " & counter
   counter = counter + 1
Next c
```

Add Method (Comments Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCommentsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCommentsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCommentsObjA "}

Adds a comment to the specified range.

Syntax

expression.**Add(*Range, Text*)**

expression Required. An expression that returns a **Comments** object.

Range Required **Range**. The range to have a comment added to it.

Text Optional **Variant**. The text of the comment.

Add Method (Comments Collection) Example

This example adds a comment at the insertion point.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Comments.Add Range:=Selection.Range, Text:="review this"
```

This example adds a comment to the third paragraph in the active document.

```
Set myRange = ActiveDocument.Paragraphs(3).Range
ActiveDocument.Comments.Add Range:=myRange, _
    Text:="original third paragraph"
```

Add Method (Dictionaries Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddDictionariesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddDictionariesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddDictionariesObjA "}

Adds the dictionary specified by ***FileName*** to the collection of active custom spelling dictionaries. If a file with the name specified by ***FileName*** doesn't exist, Word creates one.

Syntax

expression.**Add(*FileName*)**

expression Required. An expression that returns a **Dictionaries** object. Use the **CustomDictionaries** property to return the collection of custom spelling dictionaries.

FileName Required **String**. The string name of the dictionary file. If no path is specified in the string, the proofing tools path is used.

Remarks

The **Dictionaries** collection includes only the active custom spelling dictionaries. **Dictionary** objects that are derived from the **Languages** collection don't have an **Add** method. These include the **Dictionary** objects returned by the **ActiveSpellingDictionary**, **ActiveGrammarDictionary**, **ActiveThesaurusDictionary**, and **ActiveHyphenationDictionary** properties.

Add Method (Dictionaries Collection) Example

This example removes all dictionaries from the list of custom spelling dictionaries and creates a new custom dictionary file. The new dictionary is assigned to be the active custom dictionary, to which new words are automatically added.

```
With CustomDictionaries
    .ClearAll
    .Add FileName:= "c:\My Documents\MyCustom.dic"
    .ActiveCustomDictionary = CustomDictionaries(1)
End With
```

This example creates a new custom dictionary and assigns it to a variable. The new custom dictionary is then set to be used for text that's marked as French (Canadian). Note that to run a spelling check for another language, you must have installed the proofing tools for that language.

```
Set myDictionary = CustomDictionaries.Add(FileName:="MyFrench.dic")
With myDictionary
    .LanguageSpecific = True
    .LanguageID = wdFrenchCanadian
End With
```

Add Method (Documents Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddDocumentsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddDocumentsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddDocumentsObjA "}

Adds a new, empty document to the collection of open documents.

Syntax

expression.**Add(*Template, NewTemplate*)**

expression Required. An expression that returns a **Documents** object.

Template Optional **Variant**. The name of the template to be used for the new document. If this argument is omitted, the Normal template is used.

NewTemplate Optional **Variant**. **True** to open the document as a template. The default value is **False**.

Add Method (Documents Collection) Example

This example creates a new document based on the Normal template.

```
Documents.Add
```

This example creates a new document based on the Professional Memo template in Windows.

```
Documents.Add Template:="C:\Program Files\Microsoft Office\Templates\Memos\
Professional Memo.dot"
```

This example creates a new document based on the Professional Memo template on the Macintosh.

```
Documents.Add Template:="Macintosh HD:Microsoft
Office:Templates:Memos:Professional Memo"
```

This example creates and opens a new template, using the template attached to the active document as a model.

```
tmpName = ActiveDocument.AttachedTemplate.FullName
Documents.Add Template:=tmpName, NewTemplate:=True
```

Add Method (FirstLetterExceptions and TwoInitialCapsExceptions Collections)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFirstLetterTwoInitialObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFirstLetterTwoInitialObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFirstLetterTwoInitialObjA "}

Adds a new exception to the list of AutoCorrect exceptions.

Syntax

expression.**Add(*Name*)**

expression Required. An expression that returns a **FirstLetterExceptions** or **TwoInitialCapsExceptions** object.

Name Required **String**. Either the word with two initial caps that you want Word to overlook or the abbreviation that you don't want Word to follow with a capital letter.

Remarks

If the **TwoInitialCapsAutoAdd** property is **True**, words are automatically added to the list of initial-capital exceptions. If the **FirstLetterAutoAdd** property is **True**, abbreviations are automatically added to the list of first-letter exceptions.

Add Method (FirstLetterExceptions and TwoInitialCapsExceptions Collections) Example

This example adds the abbreviation "addr." to the list of first-letter exceptions.

```
AutoCorrect.FirstLetterExceptions.Add Name:="addr."
```

This example adds "Msoffice" to the list of initial-capital exceptions.

```
AutoCorrect.TwoInitialCapsExceptions.Add Name:= "MSoffice"
```

Add Method (Endnotes and Footnotes Collections)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddEndnotesFootnotesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddEndnotesFootnotesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddEndnotesFootnotesObjA "}

Adds an endnote or footnote to the specified range.

Syntax

expression.**Add(*Range, Reference, Text*)**

expression Required. An expression that returns an **Endnotes** or **Footnotes** object.

Range Required **Range**. The range marked for the endnote or footnote. This can be a collapsed range.

Reference Optional **Variant**. The text for the custom reference mark. If this argument is omitted, Word inserts an automatically numbered reference mark.

Text Optional **Variant**. The text of the endnote or footnote.

Remarks

To specify a symbol for the ***Reference*** argument, use the syntax **{***FontName CharNum***}**. *FontName* is the name of the font that contains the symbol. Names of decorative fonts appear in the **Font** box in the **Symbol** dialog box (**Insert** menu). *CharNum* is the sum of 31 and the number corresponding to the position of the symbol you want to insert, counting from left to right in the table of symbols. For example, to specify an omega symbol () at position 56 in the table of symbols in the Symbol font, the argument would be "{Symbol 87}".

Add Method (Endnotes and Footnotes Collections) Example

This example adds an automatically numbered footnote at the end of the selection.

```
ActiveDocument.Footnotes.Add Range:= Selection.Range , _
    Text:= "The Willow Tree, (Lone Creek Press, 1996)."
```

This example adds an endnote to the third paragraph in the active document

```
Set myRange = ActiveDocument.Paragraphs(3).Range
ActiveDocument.Endnotes.Add Range:=myRange, _
    Text:= "Ibid., 314."
```

This example adds a footnote that uses a custom symbol () for the reference mark.

```
ActiveDocument.Footnotes.Add Range:= Selection.Range , _
    Text:= "More information in the full report.", _
    Reference:= "{MS LineDraw 225}"
```

Add Method (Fields Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFieldsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFieldsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFieldsObjA "}

Adds a field at the specified range.

Syntax

expression.**Add(*Range, Type, Text, PreserveFormatting*)**

expression Required. An expression that returns a **Fields** object.

Range Required **Range**. The range where you want to add the field. If the range isn't collapsed, the field replaces the range.

Type Optional **Variant**. The kind of field to be added. Can be one of the following **WdFieldType** constants:

wdFieldAddin
wdFieldAdvance
wdFieldAsk
wdFieldAuthor
wdFieldAutoNum
wdFieldAutoNumLegal
wdFieldAutoNumOutline
wdFieldAutoText
wdFieldAutoTextList
wdFieldBarCode
wdFieldComments
wdFieldCompare
wdFieldCreateDate
wdFieldData
wdFieldDatabase
wdFieldDate
wdFieldDDE
wdFieldDDEAuto
wdFieldDocProperty
wdFieldDocVariable
wdFieldEditTime
wdFieldEmbed
wdFieldEmpty
wdFieldExpression
wdFieldFileName
wdFieldFileSize
wdFieldFillIn
wdFieldFootnoteRef
wdFieldFormCheckBox
wdFieldFormDropDown
wdFieldFormTextInput
wdFieldFormula
wdFieldGlossary
wdFieldKeyWord
wdFieldLastSavedBy
wdFieldLink
wdFieldListNum
wdFieldMacroButton
wdFieldMergeField
wdFieldMergeRec
wdFieldMergeSeq
wdFieldNext
wdFieldNextIf
wdFieldNoteRef
wdFieldNumChars
wdFieldNumPages
wdFieldNumWords
wdFieldOCX
wdFieldPage
wdFieldPageRef
wdFieldPrint
wdFieldPrintDate
wdFieldPrivate
wdFieldQuote
wdFieldRef
wdFieldRefDoc
wdFieldRevisionNum
wdFieldSaveDate
wdFieldSection
wdFieldSectionPages
wdFieldSequence
wdFieldSet
wdFieldSkipIf
wdFieldStyleRef
wdFieldSubject
wdFieldSubscriber

wdFieldGotoButton	**wdFieldSymbol**
wdFieldHTMLActiveX	**wdFieldTemplate**
wdFieldHyperlink	**wdFieldTime**
wdFieldIf	**wdFieldTitle**
wdFieldImport	**wdFieldTOA**
wdFieldInclude	**wdFieldTOAEntry**
wdFieldIncludePicture	**wdFieldTOC**
wdFieldIncludeText	**wdFieldTOCEntry**
wdFieldIndex	**wdFieldUserAddress**
wdFieldIndexEntry	**wdFieldUserInitials**
wdFieldInfo	**wdFieldUserName**

The default value is **wdFieldEmpty**.

Text Optional **Variant**. Additional text needed for the field. For example, if you want to specify a switch for the field, you would add it here.

PreserveFormatting Optional **Variant**. **True** to have the formatting that's applied to the field preserved during updates.

Remarks

You cannot insert some fields (such as **wdFieldOCX** and **wdFieldFormCheckBox**) by using the **Add** method of the **Fields** collection. Instead, you must use specific methods such as the **AddOLEControl** method and the **Add** method for the **FormFields** collection.

Add Method (Fields Collection) Example

This example inserts a USERNAME field at the beginning of the selection.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldUserName)
```

This example inserts a LISTNUM field at the end of the selection. The starting switch is set to begin at 3.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Fields.Add Range:=Selection.Range, Type:=wdFieldListNum,
Text:="\s 3"
```

This example inserts a DATE field at the beginning of the selection and then displays the result.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldDate)
MsgBox myField.Result
```

Add Method (FormFields Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFormFieldsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFormFieldsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFormFieldsObjA "}

Adds a new form field at the specified range.

Syntax

expression.**Add(*Range, Type*)**

expression Required. An expression that returns a **FormFields** object.

Range Required **Range**. The range where you want to add the form field. If the range isn't collapsed, the form field replaces the range.

Type Required **Long**. The type of form field to add. Can be one of the following **WdFieldType** constants: **wdFieldFormCheckBox**, **wdFieldFormDropDown**, or **wdFieldFormTextInput**.

Add Method (FormFields Collection) Example

This example adds a check box at the end of the selection, gives it a name, and then selects it.

```
Selection.Collapse Direction:=wdCollapseEnd
Set ffield = ActiveDocument.FormFields.Add(Range:=Selection.Range, _
    Type:=wdFieldFormCheckBox)
With ffield
    .Name = "Check_Box_1"
    .CheckBox.Value = True
End With
```

Add Method (Frames Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFramesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFramesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFramesObjA "}

Adds a new frame to the document, selection, or range.

Syntax

expression.**Add(*Range*)**

expression Required. An expression that returns a **Frames** object.

Range Required **Range**. The range that you want the frame to surround.

Add Method (Frames Collection) Example

This example adds a frame around the selection.

```
ActiveDocument.Frames.Add Range:= Selection.Range
```

This example adds a frame around the third paragraph in the selection.

```
Set myFrame = Selection.Frames.Add(Range:= Selection.Paragraphs(3).Range)
```

Add Method (Hyperlinks Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddHyperlinksObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddHyperlinksObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddHyperlinksObjA "}

Adds a new hyperlink to the document, selection, or range.

Syntax

expression.**Add(*Anchor, Address, SubAddress*)**

expression Required. An expression that returns a **Hyperlinks** object.

Anchor Required **Object**. The text or graphic that you want turned into a hyperlink.

Address Optional **Variant**. The address for the link. The address can be an e-mail address, an Internet address, or a file name. Note that Word doesn't check the accuracy of the address.

SubAddress Optional **Variant**. The name of a location within the destination file, such as a bookmark, named range, or slide number.

Add Method (Hyperlinks Collection) Example

This example turns the selection into a hyperlink to the Microsoft address on the World Wide Web.

```
ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
    Address:="http:\\www.microsoft.com"
```

This example turns the selection into a hyperlink that links to the bookmark named "MyBookMark" in MyFile.doc.

```
ActiveDocument.Hyperlinks.Add Anchor:=Selection.Range, _
    Address:="C:\My Documents\MyFile.doc", SubAddress:="MyBookMark"
```

This example turns the first shape in the selection into a hyperlink.

```
ActiveDocument.Hyperlinks.Add Anchor:=Selection.ShapeRange(1), _
    Address:="http:\\www.microsoft.com"
```

Add Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddC "}

Adds a member to a collection, as shown in the following table. For more information about the **Add** method for a particular object, click the name of that object in the table.

Object	Description
AddIns	Adds a new add-in to the list of available add-ins.
AutoCorrectEntries	Adds a plain-text AutoCorrect entry to the list of available AutoCorrect entries.
AutoTextEntries	Adds an AutoText entry to the list of available AutoText entries.
Bookmarks	Adds a bookmark to a range.
CaptionLabels	Adds a custom caption label.
Cells, Rows, Columns	Adds a cell, row, or column to a table.
Comments	Adds a comment to a range.
CustomLabels	Adds a custom mailing label.
Dictionaries	Adds a new custom spelling dictionary to the collection of active custom spelling dictionaries.
Documents	Adds a new, empty document to the collection of open documents.
Endnotes, Footnotes	Adds an endnote or footnote to a range.
Fields	Adds a field at a range.
FirstLetterExceptions, TwoInitialCapsExceptions	Adds a new exception to the list of AutoCorrect exceptions.
FormFields	Adds a new form field at a range.
Frames	Adds a new frame to a range, selection, or document.
HeadingStyles	Adds a new heading style to a document. The new heading style will be included whenever you compile a table of contents or table of figures.
Hyperlinks	Adds a new hyperlink to a range, selection, or document.
Indexes	Adds a new index to a document.
KeyBindings	Adds a shortcut key for a macro, built-in command, font, AutoText entry, style, or symbol.
ListEntries	Adds an item to a drop-down form field.
ListTemplates	Creates a new list template.
MailMergeFields	Adds a mail merge field to the data source document.
PageNumbers	Adds page numbers to a header or footer in a section.
Panes	Adds a new pane to a window.
Paragraphs	Adds a new, blank paragraph to a document.
RecentFiles	Adds a file to the list of recently used files.
Sections	Adds a new section to a document.

Styles	Adds a new user-defined style to the list of styles.
Tables	Adds a new, blank table to a document.
TablesOfAuthorities	Adds a table of authorities to a document.
TablesOfContents	Adds a table of contents to a document.
TablesOfFigures	Adds a table of figures to a document.
TabStops	Adds a custom tab stop to a document.
TextColumns	Adds a new text column to a section or document.
Variables	Adds a variable to a document.
Windows	Adds a new window of a document.

Add Method (HeadingStyles Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddHeadingStylesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddHeadingStylesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddHeadingStylesObjA "}

Adds a new heading style to a document. The new heading style will be included whenever you compile a table of contents or table of figures.

Syntax

expression.**Add(*Style, Level*)**

expression Required. An expression that returns a **HeadingStyles** object.

Style Required **Variant**. The style you want to add. You can specify this argument by using either the string name for the style or a **Style** object.

Level Required **Integer**. A number that represents the level of the heading.

Add Method (HeadingStyles Collection) Example

This example adds a table of contents at the beginning of the active document and then adds the Title style to the list of styles used to build a table of contents.

```
Set myToc = ActiveDocument.TablesOfContents _
    .Add(Range:=ActiveDocument.Range(0, 0), _
    UseHeadingStyles:=True, UpperHeadingLevel:=1, _
    LowerHeadingLevel:=3)
myToc.HeadingStyles.Add Style:="Title", Level:=2
```

Add Method (Indexes Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddIndexesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddIndexesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddIndexesObjA "}

Adds a new index to a document.

Syntax

expression.**Add(*Range, HeadingSeparator, RightAlignPageNumbers, Type, NumberOfColumns, AccentedLetters*)**

expression Required. An expression that returns an **Indexes** object.

Range Required **Range**. The range where you want the index to appear. The index replaces the range, if the range isn't collapsed.

HeadingSeparator Optional **Variant**. The text between alphabetic groups (entries that start with the same letter) in the index. Can be one of the following **WdHeadingSeparator** constants: **wdHeadingSeparatorBlankLine**, **wdHeadingSeparatorLetter**, **wdHeadingSeparatorLetterFull**, **wdHeadingSeparatorLetterLow**, or **wdHeadingSeparatorNone**.

RightAlignPageNumbers Optional **Variant**. **True** to align page numbers with the right margin.

Type Optional **Variant**. Specifies whether subentries are on the same line (run-in) as the main entry or on a separate line (indented) from the main entry. Can be either of the following **WdIndexType** constants: **wdIndexIndent** or **wdIndexRunin**.

NumberOfColumns Optional **Variant**. The number of columns for each page of the index. Specifying 0 (zero) sets the number of columns in the index to the same number as in the document.

AccentedLetters Optional **Variant**. **True** to include separate headings for accented letters in the index (for example, words that begin with "À" and words that begin with "A" are listed under separate headings).

Remarks

An index is built from XE (Index Entry) fields in a document. Use the **MarkEntry** method to mark index entries to be included in an index.

Add Method (Indexes Collection) Example

This example marks an index entry, and then it creates an index at the end of the active document.

```
ActiveDocument.Indexes.MarkEntry Range:=Selection.Range, Entry:="My Entry"
Set MyRange = ActiveDocument.Content
MyRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.Indexes.Add Range:=MyRange, Type:=wdIndexRunin
```

Add Method (ListEntries Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddListEntriesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddListEntriesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddListEntriesObjA "}

Adds an item to a drop-down form field.

Syntax

expression.**Add(*Name, Index*)**

expression Required. An expression that returns a **ListEntries** object.

Name Required **String**. The name of the drop-down form field item.

Index Optional **Variant**. A number that represents the position of the item in the list.

Add Method (ListEntries Collection) Example

This example inserts a drop-down form field in the active document and then adds the items "Red," "Blue," and "Green" to the form field.

```
Set myField = ActiveDocument.FormFields.Add(Range:= Selection.Range, _
     Type:= wdFieldFormDropDown)
With myField.DropDown.ListEntries
    .Add Name:="Red"
    .Add Name:="Blue"
    .Add Name:="Green"
End With
```

Add Method (ListTemplates Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddListTemplatesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddListTemplatesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddListTemplatesObjA "}

Creates a new list template.

Syntax

expression.**Add(*OutlineNumbered, Name*)**

expression Required. An expression that returns a **ListTemplates** object.

OutlineNumbered Optional **Variant**. **True** to apply outline numbering to the new list template.

Name Optional **Variant**. An optional name used for linking the list template to a LISTNUM field. You cannot use this name to index the list template in the collection.

Remarks

You cannot use the **Add** method on **ListTemplates** objects returned from a **ListGallery** object. You can, however, modify the existing list templates in the galleries.

Add Method (ListTemplates Collection) Example

This example adds a new, single-level list template to the active document. The example changes the numbering style for the new list template and then applies the list template to the selection.

```
Set myList = ActiveDocument.ListTemplates.Add(OutlineNumbered:=False)
myList.ListLevels(1).NumberStyle = wdListNumberStyleUpperCaseLetter
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=myList
```

Add Method (MailMergeFields Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddMailMergeFieldsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddMailMergeFieldsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddMailMergeFieldsObjA "}

Adds a mail merge field to the data source document.

Syntax

expression.**Add(*Range, Name*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range**. The range where you want the field to appear. This field replaces the range, if the range isn't collapsed.

Name Required **String**. The name of the field.

Add Method (MailMergeFields Collection) Example

This example replaces the selection with a mail merge field named "MiddleInitial."

```
ActiveDocument.MailMerge.Fields.Add Range:= Selection.Range, _
    Name:="MiddleInitial"
```

Add Method (PageNumbers Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddPageNumbersObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddPageNumbersObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddPageNumbersObjA "}

Adds page numbers to a header or footer in a section.

Syntax

expression.**Add(*PageNumberAlignment, FirstPage*)**

expression Required. An expression that returns a **PageNumbers** object.

PageNumberAlignment Optional **Variant**. Can be one of the following **WdPageNumberAlignment** constants: **wdAlignPageNumberCenter**, **wdAlignPageNumberInside**, **wdAlignPageNumberLeft**, **wdAlignPageNumberOutside**, or **wdAlignPageNumberRight**.

FirstPage Optional **Variant**. **False** to make the first-page header and the first-page footer different from the headers and footers on all subsequent pages in the document. If ***FirstPage*** is set to **False**, a page number isn't added to the first page. If this argument is omitted, the setting is controlled by the **DifferentFirstPageHeaderFooter** property.

Remarks

If the **LinkToPrevious** property for the **HeaderFooter** object is set to **True**, the page numbers will continue sequentially from one section to next throughout the document.

Add Method (PageNumbers Collection) Example

This example adds a page number to the primary footer in the first section of the active document.

```
With ActiveDocument.Sections(1)
    .Footers(wdHeaderFooterPrimary).PageNumbers.Add _
        PageNumberAlignment:= wdAlignPageNumberLeft, FirstPage:= True
End With
```

This example creates and formats page numbers in the header for the active document.

```
Set myPgNum = ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary) _
    .PageNumbers.Add(PageNumberAlignment:= wdAlignPageNumberCenter, _
    FirstPage:= True)
myPgNum.Select
With Selection.Range
    .Italic = True
    .Bold = True
End With
```

Add Method (Panes Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddPanesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddPanesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddPanesObjA "}

Adds a new pane to a window.

Syntax

expression.**Add(*SplitVertical*)**

expression Required. An expression that returns a **Panes** object.

SplitVertical Optional **Variant**. A number that represents the percentage of the window, from top to bottom, you want to appear above the split.

Remarks

This method will fail if it's applied to a window that's already been split.

Add Method (Panes Collection) Example

The following example splits the active window such that the top pane is 30 percent of the total window size.

```
ActiveWindow.Panes.Add SplitVertical:=30
```

Add Method (Paragraphs Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddParagraphsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddParagraphsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddParagraphsObjA "}

Adds a new, blank paragraph to a document.

Syntax

expression.**Add(*Range*)**

expression Required. An expression that returns a **Paragraphs** object.

Range Optional **Variant**. The range before which you want the new paragraph to be added. The new paragraph doesn't replace the range.

Remarks

If ***Range*** isn't specified, the new paragraph is added after the selection or range or at the end of the document, depending on *expression*.

Add Method (Paragraphs Collection) Example

This example adds a paragraph after the selection.

```
Selection.Paragraphs.Add
```

This example adds a paragraph mark before the first paragraph in the selection.

```
Selection.Paragraphs.Add Range:=Selection.Paragraphs(1).Range
```

This example adds a paragraph mark before the second paragraph in the active document.

```
ActiveDocument.Paragraphs.Add Range:=ActiveDocument.Paragraphs(2).Range
```

This example adds a new paragraph mark at the end of the active document.

```
ActiveDocument.Paragraphs.Add
```

Add Method (RecentFiles Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddRecentFilesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddRecentFilesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddRecentFilesObjA "}

Adds a file to the list of recently used files.

Syntax

expression.**Add(*Document, ReadOnly*)**

expression Required. An expression that returns a **RecentFiles** object.

Document Required **Variant**. The document you want to add to the list of recently used files. You can specify this argument by using either the string name for the document or a **Document** object.

ReadOnly Optional **Variant**. **True** to make the document read-only.

Add Method (RecentFiles Collection) Example

This example adds the active document to the list of recently used files.

```
If ActiveDocument.Saved = True Then
    RecentFiles.Add Document:=ActiveDocument.Name
End If
```

Add Method (Sections Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddSectionsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddSectionsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddSectionsObjA "}

Adds a new section to a document.

Syntax

expression.**Add(*Range, Start*)**

expression Required. An expression that returns a **Sections** object.

Range Optional **Variant**. The range before which you want to insert the section break. If this argument is omitted, the section break is inserted at the end of the document.

Start Optional **Variant**. The type of section break you want to add. Can be one of the following **WdSectionStart** constants: **wdSectionContinuous**, **wdSectionEvenPage**, **wdSectionNewColumn**, **wdSectionNewPage**, or **wdSectionOddPage**. If this argument is omitted, a Next Page section break is added.

Add Method (Sections Collection) Example

This example adds a Next Page section break before the third paragraph in the active document.

```
Set myRange = ActiveDocument.Paragraphs(3).Range
ActiveDocument.Sections.Add Range:=myRange
```

This example adds a Continuous section break at the selection.

```
Set myRange = Selection.Range
ActiveDocument.Sections.Add Range:=myRange, Start:=wdSectionContinuous
```

This example adds a Next Page section break at the end of the active document.

```
ActiveDocument.Sections.Add
```

Add Method (Styles Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddStylesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddStylesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddStylesObjA "}

Adds a new user-defined style to the collection of styles.

Syntax

expression.**Add(*Name, Type*)**

expression Required. An expression that returns a **Styles** object.

Name Required **String**. The string name for the new style.

Type Optional **Variant**. The style type of the new style. Can be either of the following **WdStyleType** constants: **wdStyleTypeParagraph** or **wdStyleTypeCharacter**.

Add Method (Styles Collection) Example

This example adds a new character style named "Introduction" and makes it 12-point Arial, with bold and italic formatting. The example then applies the new character style to the selection.

```
Set myStyle = ActiveDocument.Styles.Add(Name:="Introduction", _
    Type:=wdStyleTypeCharacter)
With myStyle.Font
    .Bold = True
    .Italic = True
    .Name = "Arial"
    .Size = 12
End With
Selection.Range.Style = "Introduction"
```

Add Method (Tables Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTablesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTablesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTablesObjA "}

Adds a new, blank table to a document.

Syntax

expression.**Add(*Range, NumRows, NumColumns*)**

expression Required. An expression that returns a **Tables** object.

Range Required **Range**. The range where you want the table to appear. The table replaces the range, if the range isn't collapsed.

NumRows Required **Long**. The number of rows you want to include in the table.

NumColumns Required **Long**. The number of columns you want to include in the table.

Add Method (Tables Collection) Example

This example adds a blank table with three rows and four columns at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(0, 0)
ActiveDocument.Tables.Add Range:=myRange, NumRows:=3, NumColumns:=4
```

This example adds a new, blank table with 6 rows and 10 columns at the end of the active document

```
Set MyRange = ActiveDocument.Content
MyRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.Tables.Add Range:=MyRange, NumRows:=6, NumColumns:=10
```

This example adds a table with three rows and five columns to a new document and then inserts data into each cell in the table.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 5)
With myTable
For x = 1 to 3
    For y = 1 to 5
        .Cell(x,y).Range.InsertAfter "Cell " & x & "," & y
    Next y
Next x
.Columns.AutoFit
End With
```

Add Method (TablesOfAuthorities Collection)

This item is not available for this version of Office

Add Method (TablesOfAuthorities Collection) Example

This item is not available for this version of Office

Add Method (TablesOfContents Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTablesOfContentsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTablesOfContentsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTablesOfContentsObjA "}

Adds a table of contents to a document.

Syntax

expression.**Add(*Range, UseHeadingStyles, UpperHeadingLevel, LowerHeadingLevel, UseFields, TableID, RightAlignPageNumbers, IncludePageNumbers, AddedStyles*)**

expression Required. An expression that returns a **TablesOfContents** object.

Range Required **Range**. The range where you want the table of contents to appear. The table of contents replaces the range, if the range isn't collapsed.

UseHeadingStyles Optional **Variant**. **True** to use built-in heading styles to create the table of contents. If this argument is omitted, ***UseHeadingStyles*** is set to **True**.

UpperHeadingLevel Optional **Variant**. The starting heading level for the table of contents. Corresponds to the starting value used with the \o switch for a TOC (Table of Contents) field. The default value is 1.

LowerHeadingLevel Optional **Variant**. The ending heading level for the table of contents. Corresponds to the ending value used with the \o switch for a TOC (Table of Contents) field. The default value is 9.

UseFields Optional **Variant**. **True** if TC (Table of Contents Entry) fields are used to create the table of contents. Use the **MarkEntry** method to mark entries to be included in the table of contents. If this argument is omitted, ***UseFields*** is set to **False**.

TableID Optional **Variant**. A one-letter identifier that's used to build a table of contents from TC fields. Corresponds to the \f switch for a TOC (Table of Contents) field. For example, "T" builds a table of contents from TC fields using the table identifier T. If this argument is omitted, TC fields aren't used.

RightAlignPageNumbers Optional **Variant**. **True** if page numbers in the table of contents are aligned with the right margin. If this argument is omitted, ***RightAlignPageNumbers*** is set to **True**.

IncludePageNumbers Optional **Variant**. **True** to include page numbers in the table of contents. If this argument is omitted, ***IncludePageNumbers*** is set to **True**.

AddedStyles Optional **Variant**. The sting name for additional styles used to compile the table of contents (styles other than the Heading 1 – Heading 9 styles). Use the **Add** method of a **HeadingStyles** object to create new heading styles.

Add Method (TablesOfContents Collection) Example

This example adds a table of contents at the beginning of the active document. The table of contents is built from paragraphs styled with the Heading 1, Heading 2, and Heading 3 styles or the custom styles myStyle and yourStyle.

```
Set myRange = ActiveDocument.Range(0, 0)
ActiveDocument.TablesOfContents.Add _
    Range:=myRange, _
    UseFields:=False, _
    UseHeadingStyles:=True, _
    LowerHeadingLevel:=3, _
    UpperHeadingLevel:=1, _
    AddedStyles:="myStyle, yourStyle"
```

Add Method (TablesOfFigures Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTablesOfFiguresObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTablesOfFiguresObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTablesOfFiguresObjA "}

Adds a table of figures to a document.

Syntax

expression.**Add(*Range, Caption, IncludeLabel, UseHeadingStyles, UpperHeadingLevel, LowerHeadingLevel, UseFields, TableId, RightAlignPageNumbers, IncludePageNumbers, AddedStyles*)**

expression Required. An expression that returns a **TablesOfFigures** object.

Range Required **Range**. The range where you want the table of figures to appear.

Caption Optional **Variant**. The label that identifies the items you want to include in the table of figures. Corresponds to the \c switch for a TOC (Table of Contents) field. If this argument is omitted, ***Caption*** is set to "Figure."

IncludeLabel Optional **Variant**. **True** to include the caption label and caption number in the table of figures. If this argument is omitted, ***IncludeLabel*** is set to **True**.

UseHeadingStyles Optional **Variant**. **True** to use built-in heading styles to create the table of figures. If this argument is omitted, ***UseHeadingStyles*** is set to **False**.

UpperHeadingLevel Optional **Variant**. The starting heading level for the table of figures, if ***UseHeadingStyles*** is set to **True**. Corresponds to the starting value used with the \o switch for a TOC (Table of Contents) field. The default value is 1.

LowerHeadingLevel Optional **Variant**. The ending heading level for the table of figures, if ***UseHeadingStyles*** is set to **True**. Corresponds to the ending value used with the \o switch for a TOC (Table of Contents) field. The default value is 9.

UseFields Optional **Variant**. **True** to use TC (Table of Contents Entry) fields to create the table of figures. Use the **MarkEntry** method to mark entries you want to include in the table of figures. If this argument is omitted, ***UseFields*** is set to **False**.

TableId Optional **Variant**. A one-letter identifier that's used to build a table of figures from TC (Table of Contents Entry) fields. Corresponds to the \f switch for a TOC (Table of Contents) field. For example, "i" builds a table of figures for an illustration.

RightAlignPageNumbers Optional **Variant**. **True** align page numbers with the right margin in the table of figures. If this argument is omitted, ***RightAlignPageNumbers*** is set to **True**.

IncludePageNumbers Optional **Variant**. **True** if page numbers are included in the table of figures. If this argument is omitted, ***IncludePageNumbers*** is set to **True**.

AddedStyles Optional **Variant**. The sting name for additional styles used to compile the table of figures (styles other than the Heading 1 – Heading 9 styles). Use the **Add** method to create new heading styles.

Add Method (TablesOfFigures Collection) Example

This example inserts a table of figures in the active document.

```
ActiveDocument.TablesOfFigures.Add Range:=Selection.Range
```

Add Method (TabStops Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTabStopsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTabStopsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTabStopsObjA "}

Adds a custom tab stop.

Syntax

expression.**Add(*Position, Alignment, Leader*)**

expression Required. An expression that returns a **TabStops** object.

Position Required **Single**. The position of the tab stop (in points) relative to the left margin.

Alignment Optional **Variant**. The alignment of the tab stop. Can be one of the following **WdTabAlignment** constants: **wdAlignTabBar**, **wdAlignTabCenter**, **wdAlignTabDecimal**, **wdAlignTabLeft**, **wdAlignTabList**, or **wdAlignTabRight**. If this argument is omitted, **wdAlignTabLeft** is used.

Leader Optional **Variant**. The type of leader for the tab stop. Can be one of the following **WdTabLeader** constants: **wdTabLeaderDashes**, **wdTabLeaderDots**, **wdTabLeaderHeavy**, **wdTabLeaderLines**, or **wdTabLeaderSpaces**. If this argument is omitted, **wdTabLeaderSpaces** is used.

Add Method (TabStops Collection) Example

This example adds a tab stop positioned at 2.5 inches (from the left edge of the page) to the selected paragraphs.

```
Selection.Paragraphs.TabStops.Add Position:=InchesToPoints(2.5)
```

This example adds two tab stops to the selected paragraphs. The first tab stop is a left aligned, has a dotted leader, and is positioned at 1 inch (72 points) from the left edge of the page. The second tab stop is centered and is positioned at 2 inches from the left edge.

```
With Selection.Paragraphs.TabStops
    .Add Position:=InchesToPoints(1), Leader:=wdTabLeaderDots,
Alignment:=wdAlignTabLeft
    .Add Position:=InchesToPoints(2), Alignment:=wdAlignTabCenter
End With
```

Add Method (TextColumns Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTextColumnsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTextColumnsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTextColumnsObjA "}

Adds a new text column to a document or section.

Note By default, there's always at least one text column in the **TextColumns** collection.

Syntax

expression.**Add(*Width, Spacing, EvenlySpaced*)**

expression Required. An expression that returns a **TextColumns** object.

Width Optional **Variant**. The width of the new text column in the document, in points.

Spacing Optional **Variant**. The spacing between the text columns in the document, in points.

EvenlySpaced Optional **Variant**. **True** to evenly space all the text columns be in the document.

Add Method (TextColumns Collection) Example

This example creates a new document and then adds another 2.5-inch-wide text column to it.

```
Set myDoc = Documents.Add
myDoc.PageSetup.TextColumns.Add Width:=InchesToPoints(2.5), _
    Spacing:=InchesToPoints(0.5), EvenlySpaced:=False
```

This example adds a new text column to the active document and then evenly spaces all the text columns in the document.

```
ActiveDocument.PageSetup.TextColumns.Add Width:=InchesToPoints(1.5), _
     EvenlySpaced:=True
```

Add Method (Variables Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddVariablesObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddVariablesObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddVariablesObjA "}

Adds a variable to a document.

Syntax

expression.**Add(*Name, Value*)**

expression Required. An expression that returns a **Variables** object.

Name Required **String**. The name of the document variable.

Value Optional **Variant**. The value for the document variable.

Remarks

Document variables are invisible to the user unless a DOCVARIABLE field is inserted with the appropriate variable name. If you try to add a variable with a name that already exists in the **Variables** collection, an error occurs. To avoid this error, you can enumerate the collection before adding a new variable to it.

Add Method (Variables Collection) Example

This example adds a variable named "`Temp`" to the active document and then inserts a DOCVARIABLE field to display the value in the `Temp` variable.

```
With ActiveDocument
    .Variables.Add Name:="Temp", Value:="12"
    .Fields.Add Range:=Selection.Range, Type:=wdFieldDocVariable,
Text:="Temp"
End With
ActiveWindow.View.ShowFieldCodes = False
```

This example sets the value of the `Blue` variable to 6. If this variable doesn't already exist, the example adds it to the document and sets it to 6.

```
For Each aVar In ActiveDocument.Variables
    If aVar.Name = "Blue" Then num = aVar.Index
Next aVar
If num = 0 Then
    ActiveDocument.Variables.Add Name:="Blue", Value:=6
Else
    ActiveDocument.Variables(num).Value = 6
End If
```

This example stores the user name (from the **Options** dialog box) in the template attached to the active document.

```
ScreenUpdating = False
With ActiveDocument.AttachedTemplate.OpenAsDocument
    .Variables.Add Name:="UserName", Value:= Application.UserName
    .Close SaveChanges:=wdSaveChanges
End With
```

Add Method (Windows Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddWindowsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddWindowsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddWindowsObjA "}

Opens another document window for the same document in an open window.

Syntax

expression.**Add(*Window*)**

expression Required. An expression that returns a **Windows** object.

Window Optional **Variant**. The **Window** object you want to open another window for. If this argument is omitted, a new window is opened for the active document.

Remarks

A colon (:) and a number appear in the window caption when more than one window is open for the document.

Add Method (Windows Collection) Example

This example opens a new window for the document that's displayed in the active window.

```
Windows.Add
```

This example opens a new window for MyDoc.doc.

```
Windows.Add Window:=Documents("MyDoc.doc").Windows(1)
```

Outdent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOutdentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOutdentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOutdentA"}

Removes one level of indent for one or more paragraphs.

Note Using this method is equivalent to clicking the **Decrease Indent** button on the **Formatting** toolbar.

Syntax

expression.**Outdent**

expression Required. An expression that returns a **Paragraph** or **Paragraphs** object.

See also

listoutdent method

indent method

Outdent Method Example

This example indents all the paragraphs in the active document twice, and then it removes one level of the indent for the first paragraph.

```
With ActiveDocument.Paragraphs
    .Indent
    .Indent
End With
ActiveDocument.Paragraphs(1).Outdent
```

Indent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIndentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIndentA"}

Indents one or more paragraphs by one level.

Note Using this method is equivalent to clicking the **Increase Indent** button on the **Formatting** toolbar.

Syntax

expression.**Indent**

expression Required. An expression that returns a **Paragraph** or **Paragraphs** object.

See also

listindent method

outdent method

firstlineindent

tabhangingindent method

selectcurrentindent method

rightindent property

leftindent property

Indent Method Example

This example indents all the paragraphs in the active document twice, and then it removes one level of the indent for the first paragraph.

```
With ActiveDocument.Paragraphs
    .Indent
    .Indent
End With
ActiveDocument.Paragraphs(1).Outdent
```

InsertFormula Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertFormulaC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertFormulaX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertFormulaA"}

Inserts an = (Formula) field that contains a formula at the selection.

Note The formula replaces the selection, if the selection isn't collapsed.

Syntax

expression.**Formula(*Formula, NumberFormat*)**

expression Required. An expression that returns a **Selection** object.

Formula Optional **Variant**. The mathematical formula you want the = (Formula) field to evaluate. Spreadsheet-type references to table cells are valid. For example, "=SUM(A4:C4)" specifies the first three values in the fourth row. For more information about the = (Formula) field, see Field codes:= (Formula) field.

NumberFormat Optional **Variant**. A format for the result of the = (Formula) field. For information about the types of formats you can apply, see Numeric Picture (\#) field switch.

Remarks

If you're using a spreadsheet application, such as Microsoft Excel, embedding all or part of a worksheet in a document is often easier than using the = (Formula) field in a table.

The ***Formula*** argument is optional only if the selection is in a cell and there's at least one cell that contains a value above or to the left of the cell that contains the insertion point. If the cells above the insertion point contain values, the inserted field is {=SUM(ABOVE)}; if the cells to the left of the insertion point contain values, the inserted field is {=SUM(LEFT)}. If both the cells above the insertion point and the cells to the left of it contain values, Word uses the following rules to determine which SUM function to insert:

- If the cell immediately above the insertion point contains a value, Word inserts {=SUM(ABOVE)}.
- If the cell immediately above the insertion point doesn't contain a value but the cell immediately to the left of the insertion point does, Word inserts {=SUM(LEFT)}.
- If neither cell immediatelyabove the insertion point nor the cell immediately below it contains a value, Word inserts {=SUM(ABOVE)}.
- If you don't specify ***Formula*** and all the cells above and to the left of the insertion point are empty, using the = (Formula) field causes an error.

InsertFormula Method Example

This example creates a table with three rows and three columns at the beginning of the active document and then calculates the average of all the numbers in the first column.

```
Set MyRange = ActiveDocument.Range(0, 0)
Set myTable = ActiveDocument.Tables.Add(MyRange, 3, 3)
With myTable
    .Cell(1, 1).Range.InsertAfter "100"
    .Cell(2, 1).Range.InsertAfter "50"
    .Cell(3, 1).Select
End With
Selection.InsertFormula Formula:="=Average(Above)"
```

The example inserts a formula field that's subtracted from a value represented by the bookmark named "GrossSales." The result is formatted with a dollar sign.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.InsertFormula Formula:= "=GrossSales-45,000.00", _
    NumberFormat:="$#,##0.00"
```

InsideColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsideColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsideColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsideColorIndexA "}

Returns or sets the color of the inside borders. Can be any of the following **WdColorIndex** constants: **wdAuto**, **wdBlack**, **wdBlue**, **wdBrightGreen**, **wdDarkBlue**, **wdDarkRed**, **wdDarkYellow**, **wdGray25**, **wdGray50**, **wdGreen**, **wdPink**, **wdRed**, **wdTeal**, **wdTurquoise**, **wdViolet**, **wdWhite**, or **wdYellow**. Read/write **Long**.

Remarks

If the **InsideLineStyle** property is set to either **wdLineStyleNone** or **False**, setting this property has no effect.

InsideColorIndex Property Example

This example adds borders between rows and between columns in the first table in the active document, and then it sets the colors for both the inside and outside borders.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    With myTable.Borders
        .InsideLineStyle = True
        .InsideColorIndex = wdBrightGreen
        .OutsideColorIndex = wdPink
    End With
End If
```

This example adds red borders between the first four paragraphs in the active document.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(1).Range.Start, _
    End:=myDoc.Paragraphs(4).Range.End)
With myRange.Borders
    .InsideLineStyle = wdLineStyleSingle
    .InsideLineWidth = wdLineWidth150pt
    .InsideColorIndex = wdRed
End With
```

OutsideColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutsideColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutsideColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutsideColorIndexA "}

Returns or sets the color of the outside borders. Can be any of the following **WdColorIndex** constants: **wdAuto**, **wdBlack**, **wdBlue**, **wdBrightGreen**, **wdDarkBlue**, **wdDarkRed**, **wdDarkYellow**, **wdGray25**, **wdGray50**, **wdGreen**, **wdPink**, **wdRed**, **wdTeal**, **wdTurquoise**, **wdViolet**, **wdWhite**, or **wdYellow**. Read/write **Long**.

Remarks

If the **OutsideLineStyle** property is set to either **wdLineStyleNone** or **False**, setting this property has no effect.

OutsideColorIndex Property Example

This example adds borders between rows and between columns in the first table in the active document, and then it sets the colors for both the inside and outside borders.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    With myTable.Borders
        .InsideLineStyle = True
        .InsideColorIndex = wdBrightGreen
        .OutsideColorIndex = wdPink
    End With
End If
```

This example adds a red, 0.75-point double border around the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).Borders
    .OutsideLineStyle = wdLineStyleDouble
    .OutsideLineWidth = wdLineWidth075pt
    .OutsideColorIndex = wdRed
End With
```

Execute Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddA "}

For information about the **Execute** method, click one of the following object names:

Dialog

Find

KeyBinding

MailMerge

AddIns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAddInsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAddInsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAddInsA "}

Returns an **AddIns** collection that represents all available add-ins, regardless of whether they're currently loaded. The **AddIns** collection includes the global templates and Word add-in libraries (WLLs) listed in the **Templates and 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.

AddIns Property Example

This example returns the total number of add-ins.

```
myCount = AddIns.Count
```

This example displays the name of each add-in in the **Addins** collection.

```
For Each aAddIn In AddIns
    MsgBox aAddIn.Name
Next aAddIn
```

Clear Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthClearC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthClearX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthClearA "}

DropCap object: Removes the dropped capital letter formatting.

KeyBinding object: Removes the key binding from the **KeyBindings** collection and resets a built-in command to its default key assignment.

ListEntries object: Removes all items from a drop-down form field.

TabStop object: Removes the specified custom tab stop.

TextInput object: Deletes the text from the specified text form field.

Syntax

expression.**Clear**

expression Required. An expression that returns a **DropCap**, **KeyBinding**, **ListEntries**, **TabStop**, or **TextInput** object.

Clear Method Example

This example clears the first custom tab in the first paragraph of the active document.

```
ActiveDocument.Paragraphs(1).TabStops(1).Clear
```

This example protects the document for forms and deletes the text from the first form field if the field is a text form field.

```
ActiveDocument.Protect Type:=wdAllowOnlyFormFields, NoReset:=True
If ActiveDocument.FormFields(1).Type = wdFieldFormTextInput Then
    ActiveDocument.FormFields(1).TextInput.Clear
End If
```

This example removes all items from the form field named "Colors" in Sales.doc.

```
Documents("Sales.doc").FormFields("Colors").DropDown.ListEntries.Clear
```

This example removes dropped capital letter formatting from the first letter in the active document.

```
Set drop = ActiveDocument.Paragraphs(1).DropCap
If Not (drop Is Nothing) Then drop.Clear
```

This example removes the ALT+F1 key assignment from the Normal template.

```
CustomizationContext = NormalTemplate
FindKey(BuildKeyCode(Arg1:=wdKeyAlt, Arg2:=wdKeyF1)).Clear
```

State Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStateA "}

Returns the current state of a mail merge operation. Can be one of the following **WdMailMergeState** constants: **wdNormalDocument**, **wdMainDocumentOnly**, **wdMainAndDataSource**, **wdMainAndHeader**, **wdMainAndSourceAndHeader**, or **wdDataSource**. Read-only **Long**.

State Property Example

This example executes a mail merge if the active document is a main document with an attached data source.

```
Set myMerge = ActiveDocument.MailMerge
If myMerge.State = wdMainAndDataSource Then myMerge.Execute
```

ID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIdC "}
{ewc HLP95EN.DLL, DYNALINK, "Example":"woproIdX":1}
{ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIdA "}

CaptionLabel object: If the **BuiltIn** property of the **CaptionLabel** object returns **True**, **ID** returns the type for the specified caption label. Can be one of the following **WdCaptionLabelID** constants: **wdCaptionEquation**, **wdCaptionFigure**, or **wdCaptionTable**. If the **BuiltIn** property returns **False**, **ID** returns **False**. Read-only **Long**.

Language object: Returns a number that identifies the specified language. Read-only **Long**.

Can be one of the following **WdLanguageID** constants:

wdAfrikaans
wdArabic
wdBasque
wdBelgianDutch
wdBelgianFrench
wdBrazilianPortuguese
wdBulgarian
wdByelorussian
wdCatalan
wdCroation
wdCzech
wdDanish
wdDutch
wdEnglishAUS
wdEnglishCanadian
wdEnglishNewZealand
wdEnglishSouthAfrica
wdEnglishUK
wdEnglishUS
wdEstonian
wdFarsi
wdFinnish
wdFrench
wdFrenchCanadian
wdGerman
wdGreek
wdHebrew
wdHungarian
wdItalian
wdIcelandic
wdJapanese
wdKorean
wdLanguageNone
wdLatvian
wdMacedonian
wdMalaysian
wdMexicanSpanish
wdNoProofing
wdNorwegianBokmol
wdNorwegianNynorsk
wdPolish
wdPortuguese
wdRomanian
wdRussian
wdSerbianCyrillic
wdSerbianLatin
wdSesotho
wdSimplifiedChinese
wdSlovak
wdSlovenian
wdSpanish
wdSpanishModernSort
wdSwedish
wdSwissFrench
wdSwissGerman
wdSwissItalian
wdTraditionalChinese
wdTsonga
wdTswana
wdTurkish
wdUkrainian
wdVenda
wdXhosa
wdZulu

ID Property Example

This example displays the built-in caption label names and ID values.

```
For Each cl In CaptionLabels
    If cl.BuiltIn = True Then MsgBox cl.Name & " " & cl.ID
Next cl
```

This example formats the selection with the Icelandic proofing tools language.

```
Selection.LanguageID = Languages("Icelandic").ID
```

Autoload Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoloadC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoloadX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoloadA"}

True if the specified add-in is automatically loaded when Word is started. Add-ins located in the Startup folder in the Word program folder are automatically loaded. Read-only **Boolean**.

Autoload Property Example

This example displays the name of each add-in that is automatically loaded when Word is started.

```
For Each myAddin In Addins
    If myAddin.Autoload = True Then
        MsgBox myAddin.Name
        afound = True
    End If
Next myAddin
If afound <> True Then MsgBox "No add-ins were loaded automatically."
```

This example determines whether the add-in named "Gallery.dot" was automatically loaded.

```
For Each aAddin In Addins
    If InStr(LCase$(aAddin.Name), "gallery.dot") > 0 Then
        If aAddin.Autoload = True Then Msgbox "Autoload"
    End If
Next aAddin
```

Compiled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCompiledC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCompiledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCompiledA"}

True if the specified add-in is a Word add-in library (WLL). **False** if the add-in is a template. Read-only **Boolean**.

Compiled Property Example

This example determines how many WLLs are currently loaded.

```
count = 0
For Each aAddin in Addins
    If aAddin.Compiled = True And aAddin.Installed = True Then
        count = count + 1
    End If
Next aAddin
MsgBox Str(count) & " WLL's are loaded"
```

If the first add-in is a template, this example unloads the template and opens it.

```
If Addins(1).Compiled = False Then
    Addins(1).Installed = False
    Documents.Open FileName:=AddIns(1).Path & Application.PathSeparator & _
        AddIns(1).Name
End If
```

Word Add-in Library (WLL)

A stand-alone, dynamic-link library (Windows) or code resource (Macintosh) that, when loaded, can add custom commands that extend the functionality of Word. Commands and functions in a loaded WLL are also available to Visual Basic projects. WLLs are written and compiled in programming languages such as C and C++.

Installed Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInstalledC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInstalledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInstalledA"}

True if the specified add-in is installed (loaded). Add-ins that are loaded are selected in the **Templates and Add-ins** dialog box (**Tools** menu). Read/write **Boolean**.

Note Uninstalled add-ins are included in the **AddIns** collection. To remove a template or WLL from the **AddIns** collection, apply the **Delete** method to the **AddIn** object (the add-in name is removed from the **Templates and Add-ins** dialog box). To unload all templates and WLLs, apply the **Unload** method to the **AddIns** collection.

Installed Property Example

This example unloads the global template named "Gallery.dot."

```
Addins("Gallery.dot").Installed = False
```

This example loads FindAll.wll.

```
Addins("C:\Templates\FindAll.wll").Installed = True
```

This example loads Custom.dot.

```
AddIns("C:\Program Files\Microsoft Office\Templates\Custom.dot").Installed
= True
```

This example displays a message on the status bar if Dot1.dot is loaded as a global template.

```
If AddIns("Dot1.dot").Installed = True Then StatusBar = "Dot1.dot is
loaded"
```

Unload Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUnloadC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUnloadX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUnloadA "}

Unloads all loaded add-ins and, depending on the value of the ***RemoveFromList*** argument, removes them from the **AddIns** collection.

Syntax

expression.**Unload(*RemoveFromList*)**

expression Required. An expression that returns an **AddIns** object.

RemoveFromList Required **Boolean**. **True** to remove the unloaded add-ins from the **AddIns** collection (the names are removed from the **Templates and Add-ins** dialog box). **False** to leave the unloaded add-ins in the collection.

If the **Autoload** property for an unloaded add-in returns **True**, **Unload** cannot remove that add-in from the **AddIns** collection, regardless of the value of ***RemoveFromList***.

Remarks

To unload a single template or WLL, set the **Installed** property of the **AddIn** object to **False**. To remove a single template or WLL from the **AddIns** collection, apply the **Delete** method to the **AddIn** object.

Unload Method Example

This example unloads all the add-ins listed in the **Templates and Add-ins** dialog box. The add-in names remain in the **AddIns** collection.

```
If AddIns.Count > 0 Then AddIns.UnLoad RemoveFromList:=False
```

Visible Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVisibleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVisibleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVisibleA "}

Application, **Task**, or **Border** object: **True** if the specified object is visible. Read/write **Boolean**.

FillFormat, **LineFormat**, **ShadowFormat**, **Shape**, **ShapeRange**, or **ThreeDFormat** object: **True** if the specified object, or the formatting applied to it, is visible. Read/write **Long**.

Remarks

To hide applications other than Word that are already running, use the **Hidden** property of the **Task** object. For information about starting a new task, see the **Task** object.

Visible Property Example

This example hides Word.

```
Application.Visible = False
```

This example hides the Calculator, if it's running. If it's not running, a message is displayed.

```
If Tasks.Exists("Calculator") Then
    Tasks("Calculator").Visible = False
Else
    Msgbox "Calculator is not running."
End If
```

This example creates a table in the active document and removes the default borders from the table.

```
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=12, NumColumns:=5)
For Each aBorder In myTable.Borders
    aBorder.Visible = False
Next aBorder
```

This example hides the shadow formatting for the first shape in the active document.

```
ActiveDocument.Shapes(1).Shadow.Visible = False
```

This example creates a new document, and then adds text and a rectangle to it. The example also sets Word to hide the rectangle while the document is being printed and then make it visible again after printing is completed.

```
Set myDoc = Documents.Add
Selection.TypeText Text:="This is some sample text."
With myDoc
    .Shapes.AddShape msoShapeRectangle, 200, 70, 150, 60
    .Shapes(1).Visible = False
    .PrintOut
    .Shapes(1).Visible = True
End With
```

Orientation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOrientationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOrientationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOrientationA "}

PageSetup object: Returns or sets the orientation of the page. Can be either of the following **WdOrientation** constants: **wdOrientLandscape** or **wdOrientPortrait**. Read/write **Long**.

Range or **Selection** object: Returns or sets the orientation of the text in the range or selection when the Text Direction feature is enabled. Can be one of the following **WdTextOrientation** constants: **wdTextOrientationDownward**, **wdTextOrientationHorizontal**, or **wdTextOrientationUpward**. Read/write **Long**.

TextFrame object: Returns or sets the orientation of the text inside the frame. Can be one of the following **MsoTextOrientation** constants: **msoTextOrientationDownward**, **msoTextOrientationHorizontal**, **msoTextOrientationMixed**, or **msoTextOrientationUpward**. Read/write **Long**.

Remarks

The following **MsoTextOrientation** constants aren't used in the U.S. English version of Word: **msoTextOrientationVertical**, **msoTextOrientationHorizontalRotatedFarEast**, and **msoTextOrientationVerticalFarEast**.

You can set the orientation for a text frame or for a range or selection that happens to occur inside a text frame. For information about the difference between a text frame and a text box, see the **TextFrame** object.

Orientation Property Example

This example changes the orientation of the document named "MyDocument.doc" and then prints the document. The example then changes the orientation of the document back to portrait.

```
Set myDoc = Documents("MyDocument.doc")
With myDoc
    .PageSetup.Orientation = wdOrientLandscape
    .PrintOut
    .PageSetup.Orientation = wdOrientPortrait
End With
```

This example creates a new document, inserts text into it, uses this text to create a text box, and then sets the orientation of the text frame so that the text slopes upward.

```
Set mydoc = Documents.Add
Selection.TypeText "This is some text."
mydoc.Content.Select
Selection.CreateTextbox
mydoc.Shapes(1).TextFrame.Orientation = msoTextOrientationUpward
```

ShapeRange Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShapeRangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShapeRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShapeRangeA"}

Returns a **ShapeRange** collection that represents all the **Shape** objects in the specified range or selection. The shape range can contain drawings, shapes, pictures, OLE objects, ActiveX controls, text objects, and callouts. Read-only.

ShapeRange Property Example

The following example sets the fill foreground color to purple for all the shapes in the active document.

```
ActiveDocument.Content.ShapeRange.Fill.ForeColor.RGB = RGB(255, 0, 255)
```

The following example applies shadow formatting to all the shapes in the selection.

```
Selection.ShapeRange.Shadow.Type = msoShadow6
```

LockAnchor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLockAnchorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLockAnchorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLockAnchorA "}

Frame object: **True** if the specified frame is locked. The frame anchor indicates where the frame will appear in Normal view. You cannot reposition a locked frame anchor. Read/write **Boolean**.

Shape or **ShapeRange** object: **True** if the specified **Shape** or **ShapeRange** object's anchor is locked to the anchoring range. When a shape has a locked anchor, you cannot move the shape's anchor by dragging it (the anchor doesn't move as the shape is moved). Read/write **Boolean**.

Remarks

A **Shape** objects is anchored to a range of text, but you can position it anywhere on the page. The shape is anchored to the beginning of the first paragraph that contains the anchoring range. A shape will always remain on the same page as its anchor.

LockAnchor Property Example

This example locks the anchor of the first frame in section two in the active document.

```
Set myRange = ActiveDocument.Sections(2).Range
If TypeName(myRange) <> "Nothing" And myRange.Frames.Count > 0 Then
    myRange.Frames(1).LockAnchor = True
End If
```

This example creates a new document, adds a shape to it, and then locks the shape's anchor.

```
Set myDoc = Documents.Add
Set myShape = myDoc.Shapes.AddShape(msoShapeBalloon, 0, 0, 140, 70)
myShape.LockAnchor = True
ActiveWindow.View.ShowObjectAnchors = True
```

This example returns a message that states the lock status for each shape in the active document.

```
For each s in ActiveDocument.Shapes
    Msgbox "Shape " & s.Count & " is locked - " & s.LockAnchor
Next s
```

Shapes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShapesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShapesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShapesA "}

Returns a **Shapes** collection that represents all the **Shape** objects in the specified document, header, or footer. This collection can contain drawings, shapes, pictures, OLE objects, ActiveX controls, text objects, and callouts. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Remarks

The **Shapes** property, when applied to a document, returns all the **Shape** objects in the main story of the document, excluding the headers and footers. When applied to a **HeaderFooter** object, the **Shapes** property returns all the **Shape** objects found in all the headers and footers in the document.

Shapes Property Example

This example creates a new document, adds a rectangle to it that's 100 points wide and 50 points high, and sets the upper-left corner of the rectangle to be 5 points from the left edge and 25 points from the upper-left corner of the page.

```
Set myDoc = Documents.Add
myDoc.Shapes.AddShape msoShapeRectangle, 5, 25, 100, 50
```

This example sets the fill texture for all the shapes in the active document.

```
For each s in ActiveDocument.Shapes
    s.Fill.PresetTextured msoTextureOak
Next s
```

This example adds a shadow to the first shape in the active document.

```
Set myShape = ActiveDocument.Shapes(1)
myShape.Shadow.Type = msoShadow6
```

This example displays a count of all the shapes in the headers and footers in the active document.

```
MsgBox
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).Shapes.Count
```

Shape Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShapeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShapeA "}

Returns a **Shape** object for the specified hyperlink. If the hyperlink isn't represented by a shape, an error occurs. Read-only.

Shape Property Example

This example changes the fill color for the shape that represents the first hyperlink in the active document. For this example to work, the hyperlink must be represented by a shape.

```
ActiveDocument.Hyperlinks(1).Shape.Fill.ForeColor.RGB = RGB(255, 255, 0)
```

CreateTextbox Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreateTextboxC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreateTextboxX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreateTextboxA"}

Adds a default-size text box around the selection. If the selection is an insertion point, this method changes the pointer to a cross-hair pointer so that the user can draw a text box.

Syntax

expression.**CreateTextbox**

expression Requried. An expression that returns a **Selection** object.

Remarks

Using this method is equivalent to clicking the **Text Box** button on the **Drawing** toolbar. A text box is a rectangle with an associated text frame.

CreateTextbox Method Example

This example adds a text box around the selection and then changes the text box's line style.

```
If Selection.Type = wdSelectionNormal Then
    Selection.CreateTextbox
    Selection.ShapeRange(1).Line.DashStyle =msoLineDashDot
End If
```

Left Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLeftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLeftA "}

Application, **Task**, or **Window** object: Returns or sets the horizontal position of the specified window or task, in points. Read/write **Long**.

Shape or **ShapeRange** object: Returns or sets the horizontal position of the specified shape or shape range, in points. Read/write **Single**.

Remarks

The position of a shape is measured from the upper-left corner of the shape's bounding box to the shape's anchor. The **RelativeHorizontalPosition** property controls whether the anchor is positioned alongside the column, the margin, or the edge of the page.

For a **ShapeRange** object that contains more than one shape, the **Left** property sets the horizontal position of each shape.

Left Property Example

This example sets the horizontal position of the active window to 100 points.

```
With ActiveWindow
    .WindowState = wdWindowStateNormal
    .Left = 100
    .Top = 0
End With
```

This example sets the horizontal position of the first shape in the active document to 1 inch from the left edge of the page.

```
With ActiveDocument.Shapes(1)
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
    .Left = InchesToPoints(1)
End With
```

This example sets the horizontal position of the first and second shapes in the active document to 1 inch from the left edge of the column.

```
With ActiveDocument.Shapes.Range(Array(1, 2))
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionColumn
    .Left = InchesToPoints(1)
End With
```

Top Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTopC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTopA "}

Application, **Task**, or **Window** object: Returns or sets the vertical position of the specified window or task, in points. Read/write **Long**.

Shape or **ShapeRange** object: Returns or sets the vertical position (in points) of the specified shape or shape range. Read/write **Single**.

Remarks

The position of a shape is measured from the upper-left corner of the shape's bounding box to the shape's anchor. The **RelativeVerticalPosition** property controls whether the shape's anchor is positioned alongside the paragraph, the margin, or the edge of the page.

For a **ShapeRange** object that contains more than one shape, the **Top** property sets the vertical position of each shape.

Top Property Example

This example positions the Word application window 100 points from the top of the screen.

```
Application.WindowState = wdWindowStateNormal
Application.Top = 100
```

This example starts the Calculator and positions its window 100 points from the top of the screen.

```
Shell "Calc.exe"
With Tasks("Calculator")
    .WindowState = wdWindowStateNormal
    .Top = 100
End With
```

This example sets the vertical position of the first shape in the active document to 1 inch from the top of the page.

```
With ActiveDocument.Shapes(1)
    .RelativeVerticalPosition = wdRelativeVerticalPositionPage
    .Top = InchesToPoints(1)
End With
```

This example sets the vertical position of the first and second shapes in the active document to 1 inch from the top of the page.

```
With ActiveDocument.Shapes.Range(Array(1, 2))
    .RelativeVerticalPosition = wdRelativeVerticalPositionPage
    .Top = InchesToPoints(1)
End With
```

Application Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproApplicationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproApplicationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproApplicationA "}

Used without an object qualifier, this property returns an **Application** object that represents the Microsoft Word application. Used with an object qualifier, this property returns an **Application** object that represents the creator of the specified object (you can use this property with an OLE Automation object to return that object's application). Read-only.

Note The **CreateObject** and **GetObject** functions give you access to an OLE Automation object.

Application Property Example

This example displays scroll bars, screen tips, and the status bar for Microsoft Word.

```
With Application
    .DisplayScrollBars = True
    .DisplayScreenTips = True
    .DisplayStatusBar = True
End With
```

This example displays the Microsoft Excel startup path if Microsoft Excel is running.

```
If Tasks.Exists(Name:="Microsoft Excel") = True Then
    Set myobject = GetObject("", "Excel.Application")
    MsgBox myobject.Application.StartupPath
    Set myobject = Nothing
End If
```

Count Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCountC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCountX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCountA "}

Returns the number of items in the specified collection. Read-only **Long**.

Count Property Example

This example displays the number of paragraphs in the active document.

```
MsgBox "The active document contains " & _
    ActiveDocument.Paragraphs.Count & " paragraphs."
```

This example displays the number of words in the selection.

```
If Selection.Words.Count >= 1 And Selection.Type <> wdSelectionIP Then
    MsgBox "The selection contains " & Selection.Words.Count _
        & " words."
End If
```

This example uses the `aFields()` array to store the field codes in the active document.

```
fcount = ActiveDocument.Fields.Count
If fcount >= 1 Then
    ReDim aFields(fcount)
    For Each aField In ActiveDocument.Fields
        aFields(aField.Index) = aField.Code.Text
    Next aField
End If
```

Creator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCreatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCreatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCreatorA "}

Returns a 32-bit integer that indicates the application in which the specified object was created. For example, if the object was created in Microsoft Word, this property returns the hexadecimal number 4D535744, which represents the string "MSWD." This value can also be represented by the constant **wdCreatorCode**. Read-only **Long**.

Remarks

The **Creator** property is designed to be used on the Macintosh, where each application has a four-character creator code. For example, Microsoft Word has the creator code MSWD.

Creator Property Example

This example displays a message about the creator of `myObject`.

```
Set myObject = ActiveDocument
If myObject.Creator = wdCreatorCode Then
    MsgBox "This is a Microsoft Word object"
Else
    MsgBox "This is not a Microsoft Word object"
End If
```

Delete Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDeleteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDeleteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDeleteA "}

Syntax 1: Deletes the specified object.

Syntax 2: Deletes a table cell or cells and optionally controls how the remaining cells are shifted.

Syntax 3: Deletes the specified number of characters or words. This method returns a value that indicates the number of items deleted, or it returns 0 (zero) if the deletion was unsuccessful.

Syntax 1

expression.**Delete**

Syntax 2

expression.**Delete(*ShiftCells*)**

Syntax 3

expression.**Delete(*Unit, Count*)**

expression Syntax 1: Required. An expression that returns an object in the Applies To list.

Syntax 2: Required. An expression that returns a **Cell** or **Cells** object.

Syntax 3: Required. An expression that returns a **Range** or **Selection** object.

ShiftCells Optional **Variant**. The direction in which the remaining cells are to be shifted. Can be one of the following **WdDeleteCells** constants: **wdDeleteCellsEntireColumn**, **wdDeleteCellsEntireRow**, **wdDeleteCellsShiftLeft**, or **wdDeleteCellsShiftUp**.

Unit Optional **Variant**. The unit by which the collapsed range or selection is to be deleted. Can be either of the following **WdUnits** constants: **wdCharacter** or **wdWord**. The default value is **wdCharacter**.

Count Optional **Variant**. The number of units to be deleted. To delete units after the range or selection, collapse the range or selection and use a positive number. To delete units before the range or selection, collapse the range or selection and use a negative number.

Delete Method Example

This example deletes the selection.

```
Selection.Delete
```

This example collapses the selection and deletes the two words following the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.Delete Unit:=wdWord, Count:=2
```

This example collapses `myRange` and deletes the two characters preceding the insertion point.

```
Set myRange = Selection.Words(1)
myRange.Collapse Direction:=wdCollapseStart
myRange.Delete Unit:=wdCharacter, Count:=-2
```

This example deletes the first cell in the first table in document one.

```
Documents(1).Tables(1).Cell(1, 1).Delete ShiftCells:=wdDeleteCellsShiftLeft
```

If the bookmark named "temp" exists in the active document, this example deletes the bookmark.

```
If ActiveDocument.Bookmarks.Exists("temp") Then
    ActiveDocument.Bookmarks("temp").Delete
End If
```

This example deletes the style named "Intro" from Sales.doc. Paragraphs using the Intro style will revert to using the Normal style.

```
Documents("Sales.doc").Styles("Intro").Delete
```

This example deletes the first word in the active document.

```
ActiveDocument.Words(1).Delete
```

Name Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNameA "}

Returns or sets the name for the specified object, as described in the following table.

Object	Description
AddIn, **Dictionary**, **Document**, **FileConverter**, **RecentFile**, **Subdocument**, **Template**, **MailMergeDataSource**	Returns the object's file name. Read-only **String**.
Application	Returns the name of the application (for example, "Microsoft Word"). Read-only **String**.
AutoCaption, **CaptionLabel**	Returns the name of the caption. Read-only **String**.
AutoCorrectEntry, **AutoTextEntry**	Returns or sets the name of the AutoCorrect or AutoText entry. Read/write **String**.
Bookmark	Returns the name of the bookmark. Read-only **String**.
CustomLabel	Returns or sets the name of the custom mailing label. Read/write **String**.
FirstLetterException, **TwoInitialCapsException**	Return the word that is excepted from AutoCorrect actions. Read-only **String**.
Font	Returns or sets the name of the font. Read/write **String**.
FormField	Returns or sets the name of the form field. Read/write **String**.
Language	Returns the name of the proofing tools language. Read-only **String**.
ListEntry	Returns or sets the name of the drop-down form field item. Read/write **String**.
ListTemplate	Returns or sets an optional list template name that can be used in conjunction with the Name instruction for a LISTNUM field. Read/write **String**.
MailMergeDataField, **MailMergeFieldName**	Returns the name of the mail merge field. Read-only **String**.
MailMergeDataSource	Returns the full name of the data source document. Read-only **String**.
Shape, **ShapeRange**	Returns or sets the shape name. Read/write **String**.
Hyperlink	Returns the friendly name (as it appears in the History folder) of the hyperlink. Read-only **String**.
ReadabilityStatistic	Returns the name of the readability statistic. Read-only **String**.
SpellingSuggestion	Returns the spelling suggestion. Read-only **String**.
TableOfAuthoritiesCategory	Returns or sets the name of the table of authorities category. Read/write **String**.
Task	Returns the task name. Read-only **String**.
Variable	Returns the document variable name. Read-only **String**.

Name Property Example

This example adds a document variable to the active document and then displays the name of the first document variable.

```
ActiveDocument.Variables.Add Name:="Temp", Value:="1"
MsgBox ActiveDocument.Variables(1).Name
```

This example returns the name of the first bookmark in Hello.doc.

```
abook = Documents("Hello.doc").Bookmarks(1).Name
```

This example displays the names of the form fields in the active document.

```
If ActiveDocument.FormFields.Count >= 1 Then
    For Each FF In ActiveDocument.FormFields
        FFNames = FFNames & FF.Name & vbCr
    Next FF
    MsgBox FFNames
End If
```

This example formats the selection as Arial bold.

```
With Selection.Font
    .Name = "Arial"
    .Bold = True
End With
```

This example sets the name of the first list template used in the active document to "myList." A LISTNUM field (linked to the myList template) is then added at the insertion point. The field adopts the formatting of the myList template.

```
If ActiveDocument.ListTemplates.Count >= 1 Then
    ActiveDocument.ListTemplates(1).Name = "myList"
    Selection.Collapse Direction:=wdCollapseEnd
    ActiveDocument.Fields.Add Range:=Selection.Range, _
        Type:=wdFieldListNum, Text:="myList"
End If
```

Parent Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproParentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproParentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproParentA "}

Returns the parent object of the specified object. Read-only.

Parent Property Example

This example sets the variable `myObject` to the parent object of the **Bookmarks** object. The message box displays the object type name of `myObject` (**Document**).

```
Set myObject = ActiveDocument.Bookmarks.Parent
MsgBox TypeName(myObject)
```

This example sets the variable `myRange` to cell one in table one in the active document. The width of this cell is changed to 36 points, and border formatting is removed from the table.

```
Set myRange = ActiveDocument.Tables(1).Cell(1, 1)
With myRange
    .SetWidth ColumnWidth:=36, RulerStyle:=wdAdjustNone
    .Parent.Borders.Enable = False
End With
```

Value Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproValueC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproValueX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproValueA "}

AutoCorrectEntry, **AutoTextEntry**, or **Variable** object: Returns or sets the value of the AutoCorrect entry, AutoText entry, or document variable. Read/write **String**.

CheckBox object: **True** if the check box is selected. Read/write **Boolean**.

DropDown object: Returns or sets the number of the selected item in a drop-down form field. Read/write **Long**.

MailMergeDataField object: Returns the contents of the mail merge data field for the current record. Use the **ActiveRecord** property to set the active record in a mail merge data source. Read-only **String**.

ReadabilityStatistic object: Returns the value of the grammar statistic. Read-only **Single**.

Value Property Example

This example adds a document variable to the active document and then displays the value of the new variable.

```
ActiveDocument.Variables.Add Name:="Temp2", Value:="10"
MsgBox ActiveDocument.Variables("Temp2").Value
```

This example displays the contents of the active data record in the data source attached to Main.doc.

```
For Each dataF In Documents("Main.doc").MailMerge.DataSource.DataFields
    If dataF.Value <> "" Then dRecord = dRecord & dataF.Value & vbCr
Next dataF
MsgBox dRecord
```

This example creates an AutoCorrent entry and then displays the value of the new entry.

```
AutoCorrect.Entries.Add Name:="i.e.", Value:="that is"
MsgBox AutoCorrect.Entries("i.e.").Value
```

This example checks the grammar in the active document and then displays the Flesch reading-ease index.

```
ActiveDocument.CheckGrammar
MsgBox ActiveDocument.ReadabilityStatistics("Flesch Reading Ease").Value
```

Item Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthItemC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthItemX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthItemA "}

Returns a member of a collection, either by position or by name.

Syntax

expression.**Item(*Index*)**

expression Required. An expression that returns an object in the Applies To list.

Index Required **Variant**. The name or index number of a member of the collection. The index can be a numeric expression (a number from 1 to the value of the collection's **Count** property), a constant, or a string. For more information about using the **Item** method with a specific collection, see the Help topic for that collection.

Remarks

If the value provided as ***Index*** doesn't match any existing member of the collection, an error occurs.

The **Item** method is the default method for collections. Therefore, the following two lines of code are equivalent.

```
Application.Documents(1)
Application.Documents.Item(1)
```

Item Method Example

This example displays the name of the first document in the **Documents** collection.

```
If Documents.Count >= 1 Then
    MsgBox Documents.Item(1).Name
End If
```

This example selects the bookmark named "temp" in the active document.

```
If ActiveDocument.Bookmarks.Exists("temp") = True Then
    ActiveDocument.Bookmarks.Item("temp").Select
End If
```

This example adds two names to the **MyNames** collection. The **Item** method is used to return each of the items in the collection.

```
Dim MyNames As New Collection
MyNames.Add Item:="Dave Edson"
MyNames.Add Item:="Tim O'Brien"
MsgBox MyNames.Item(1) & vbCr & MyNames.Item(2)
```

This example clears all the items from the drop-down form field named "Colors" and then adds two color names. The **Item** method is used to display the first color in the drop-down form field.

```
With ActiveDocument.FormFields("Colors").DropDown.ListEntries
    .Clear
    .Add Name:="Blue"
    .Add Name:="Red"
    MsgBox .Item(1).Name
End With
```

Next Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNextA "}

Returns the next object in the collection. Read-only.

Next Property Example

This example activates the next window.

```
If Windows.Count > 1 Then ActiveWindow.Next.Activate
```

If the selection is in a table, this example selects the contents of the next table cell.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).Next.Select
End If
```

This example updates the fields in the first section in the active document as long as the **Next** method returns a **Field** object and the field isn't a FILLIN field.

```
If ActiveDocument.Sections(1).Range.Fields.Count >= 1 Then
    Set myField = ActiveDocument.Fields(1)
    While Not (myField Is Nothing)
        If myField.Type <> wdFieldFillIn Then myField.Update
        Set myField = myField.Next
    Wend
End If
```

This example indents the second paragraph in the selection.

```
Selection.Paragraphs(1).Next.Indent
```

Previous Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPreviousC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPreviousX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPreviousA "}

Returns the previous object in the collection. Read-only.

Previous Property Example

This example sets the space-before and space-after formatting for the paragraph immediately preceding the selection.

```
Set myPara = Selection.Paragraphs(1).Previous
With myPara
    .SpaceAfter = 12
    .SpaceBefore = 6
End With
```

If the selection is in a table, this example selects the contents of the previous row.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows(1).Previous.Select
End If
```

This example displays the field code of the second-to-last field in the active document.

```
Set aField = ActiveDocument.Fields(ActiveDocument.Fields.Count).Previous
MsgBox "Field code = " & aField.Code
```

Height Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeightA "}

Returns or sets the height of the specified object (in points), as shown in the following table.

Object	Height
Application	Returns or sets the height of the Word application window. Read/write **Long**.
Cell, **Cells**	Returns or sets the height of the specified cell or cells in a table. If the **HeightRule** property of the specified row is **wdRowHeightAuto**, **Height** returns **wdUndefined**; setting the **Height** property sets **HeightRule** to **wdRowHeightAtLeast**. Read/write **Single**.
CustomLabel	Returns or sets the height of the specified custom mailing label. Read/write **Single**.
Frame	Returns or sets the height of the specified frame. Read/write **Single**.
InlineShape	Returns or sets the height of the specified inline shape. Read/write **Single**.
Row, **Rows**	Returns or sets the height of the specified row or rows in a table. If the **HeightRule** property of the specified row is **wdRowHeightAuto**, **Height** returns **wdUndefined**; setting the **Height** property sets **HeightRule** to **wdRowHeightAtLeast**. Read/write **Single**.
Shape, **ShapeRange**	Returns or sets the height of the specified shape. Read/write **Single**.
Task	Returns or sets the height of the specified task window. Read/write **Long**.
Window	Returns or sets the height of the window. You cannot set this property if the window is maximized or minimized. Use the **UsableHeight** property to determine the maximum size for the window. Use the **WindowState** property to determine the window state. Read/write **Long**.

Height Property Example

This example sets the height of the rows in the first table in the active document to at least 20 points.

```
ActiveDocument.Tables(1).Rows.Height = 20
```

This example displays the height (in points) of the table row that contains the insertion point.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox Selection.Rows(1).Height
End If
```

This example changes the height of the active window to fill the application window area.

```
With ActiveWindow
    .WindowState = wdWindowStateNormal
    .Height = Application.UsableHeight
End With
```

This example inserts a picture as an inline shape and changes the height and width of the image.

```
Set aInLine = ActiveDocument.InlineShapes.AddPicture(FileName:="C:\Windows\
Bubbles.bmp", _
Range:=Selection.Range)
With aInLine
    .Height = 100
    .Width = 200
End With
```

Split Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSplitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSplitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSplitA "}

Cell object: Splits a single table cell into multiple cells.

Cells object: Splits a range of table cells.

Subdocument object: Divides an existing subdocument into two subdocuments at the same level in master document view or outline view. The division is at the beginning of the specified range. If the active document isn't in either master document or outline view, or if the range isn't at the beginning of a paragraph in a subdocument, an error occurs.

Table object: Inserts an empty paragraph immediately above the specified row in the table, and returns a **Table** object that contains both the specified row and the rows that follow it.

Syntax 1

expression.**Split(*NumRows, NumColumns*)**

Syntax 2

expression.**Split(*NumRows, NumColumns, MergeBeforeSplit*)**

Syntax 3

expression.**Split(*Range*)**

Syntax 4

expression.**Split(*BeforeRow*)**

expression Syntax 1: An expression that returns a **Cell** object.

Syntax 2: An expression that returns a **Cells** object.

Syntax 3: An expression that returns a **Subdocument** object.

Syntax 4: An expression that returns a **Table** object.

NumColumns Optional **Variant**. The number of columns that the cell or group of cells is to be split into.

NumRows Optional **Variant**. The number of rows that the cell or group of cells is to be split into.

MergeBeforeSplit Optional **Variant**. **True** to merge the cells with one another before splitting them.

Range Required **Range** object. The range that, when the subdocument is split, becomes a separate subdocument.

BeforeRow Required **Variant**. The row that the table is to be split before. Can be a row number or a **Row** object.

Split Method Example

This example creates a 5x5 table in the active document and splits it before the third row. Shading is applied to the cells in the resulting table (the new 3x5 table).

```
Set newDoc = Documents.Add
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumColumns:=5, NumRows:=5)
myTable.Split(BeforeRow:=myTable.Rows(3)).Shading.Texture =
wdTexture10Percent
```

This example splits the first cell in the first table into two cells.

```
ActiveDocument.Tables(1).Cell(1, 1).Split NumColumns:=2
```

This example merges the selected cells into a single cell and then splits the cell into three cells in the same row.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells.Split NumRows:=1, NumColumns:=3, MergeBeforeSplit:=
True
End If
```

This example splits the selection from an existing subdocument into a separate subdocument.

```
Selection.Range.Subdocuments(1).Split Range:=Selection.Range
```

Split Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSplitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSplitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSplitA "}

True if the window is split into multiple panes. Read/write **Boolean**.

Split Property Example

This example splits the active window into two equal-sized window panes.

```
ActiveWindow.Split = True
```

If the Document1 window is split, this example closes the active pane.

```
If Windows("Document1").Split = True Then
    Windows("Document1").ActivePane.Close
End If
```

Reset Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetA "}

Font object: Removes manual character formatting (formatting not applied using a style). For example, if you manually format a word as bold and the underlying style is plain text (not bold), the **Reset** method removes the bold format.

Paragraph, **Paragraphs**, or **ParagraphFormat** object: Removes manual paragraph formatting (formatting not applied using a style). For example, if you manually right align a paragraph and the underlying style has a different alignment, the **Reset** method changes the alignment to match the formatting of the underlying style.

RoutingSlip object: Resets the routing slip so that a new routing can be initiated with the same recipient list and delivery information. The routing must be completed before you use this method.

InlineShape object: Removes changes that were made to an inline shape.

ListGallery object: Resets the list template specified by ***Index*** for the specified list gallery to the built-in list template format.

Syntax 1

expression.**Reset**

Syntax 2

expression.**Reset(*Index*)**

expression Syntax 1: Required. An expression that returns a **Font**, **RoutingSlip**, **InlineShape**, **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Syntax 2: Required. An expression that returns a **ListGallery** object.

Index Required **Long**. A number from 1 through 7, corresponding to a format in the specified list gallery. Skipping the **None** option, the formats are numbered from left to right, starting with the top row.

Reset Method Example

This example removes manual formatting from the selection.

```
Selection.Font.Reset
```

This example removes manual paragraph formatting from the second paragraph in the active document.

```
ActiveDocument.Paragraphs(2).Reset
```

This example prepares the active document to be rerouted to the same recipients as in the previous routing settings.

```
If ActiveDocument.HasRoutingSlip = True Then
    ActiveDocument.RoutingSlip.Reset
End If
```

This example inserts a picture as an inline shape, changes the brightness, and then resets the picture to its original brightness.

```
Set aInLine = ActiveDocument.InlineShapes.AddPicture _
    (FileName:="C:\Windows\Bubbles.bmp", Range:=Selection.Range)
aInLine.PictureFormat.Brightness = 0.5
MsgBox "Changing brightness back"
aInLine.Reset
```

This example sets the fourth format listed on the **Numbered** tab in the **Bullets and Numbering** dialog box back to the built-in numbering format, and then it applies the list template to the selection.

```
ListGalleries(wdNumberGallery).Reset(4)
Selection.Range.ListFormat.ApplyListTemplate _
    ListTemplate:=ListGalleries(2).ListTemplates(4)
```

This example resets all the list templates in the **Bullets and Numbering** dialog box back to the the built-in formats.

```
For Each lg In ListGalleries
    For i = 1 to 7
        lg.Reset Index:=i
    Next i
Next lg
```

Browser Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBrowserC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBrowserX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBrowserA "}

Returns a **Browser** object that represents the **Select Browse Object** tool on the vertical scroll bar. Read-only.

Browser Property Example

This example moves to the next footnote reference mark in the active document.

```
With Application.Browser
    .Target = wdBrowseFootnote
    .Next
End With
```

This example moves to the next field in the active document. The text from the initial selection to the next field is formatted as bold.

```
Selection.ExtendMode = True
With Application.Browser
    .Target = wdBrowseField
    .Next
End With
With Selection
    .Font.Bold = True
    .ExtendMode = False
    .Collapse Direction:=wdCollapseEnd
End With
```

Build Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBuildC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBuildX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBuildA "}

Returns the version and build number of the Word application. Read-only **String**.

Build Property Example

This example displays the version and build number of Word.

```
MsgBox Prompt:=Application.Build, Title:="Microsoft Word Version"
```

CapsLock Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCapsLockC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCapsLockX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCapsLockA "}

True if the CAPS LOCK key is turned on. Read-only **Boolean**.

CapsLock Property Example

This example retrieves the current state of the CAPS LOCK key.

```
theState = Application.CapsLock
```

DisplayRecentFiles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayRecentFilesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayRecentFilesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayRecentFilesA "}

True if the names of recently used files are displayed on the **File** menu. Read/write **Boolean**.

DisplayRecentFiles Property Example

This example sets Word to display a maximum of six file names on the **File** menu.

```
Application.DisplayRecentFiles = True
RecentFiles.Maximum = 6
```

This example removes the list of recently used files from the **File** menu.

```
Application.DisplayRecentFiles = False
```

DisplayStatusBar Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayStatusBarC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayStatusBarX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayStatusBarA "}

True if the status bar is displayed. Read/write **Boolean**.

DisplayStatusBar Property Example

This example toggles the status bar.

```
Application.DisplayStatusBar = Not Application.DisplayStatusBar
```

This example displays scroll bars and the status bar.

```
With Application
    .DisplayScrollBars = True
    .DisplayStatusBar = True
End With
```

ExitWindows Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExitWindowsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExitWindowsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExitWindowsA "}

Closes all open applications, quits Microsoft Windows, and logs the current user off. This method doesn't save changes to open Word documents; however, it does prompt you to save changes to open documents in other Windows-based applications. This method isn't available on the Macintosh.

Syntax

expression.**ExitWindows**

expression Required. An expression that returns a **Tasks** object.

ExitWindows Method Example

This example saves all open Word documents, quits Word, and then quits Microsoft Windows.

```
Documents.Save NoPrompt:=True, OriginalFormat:=wdOriginalFormat
Tasks.ExitWindows
```

FontNames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontNamesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontNamesA "}

Returns a **FontNames** object that includes the names of all the available fonts. Read-only.

FontNames Property Example

This example displays the font names in the **FontNames** collection.

```
For Each aFont In FontNames
    response = MsgBox(Prompt:=aFont, Buttons:=vbOKCancel)
    If response = vbCancel Then Exit For
Next aFont
```

LandscapeFontNames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLandscapeFontNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLandscapeFontNamesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLandscapeFontNamesA "}

Returns a **FontNames** object that includes the names of all the available landscape fonts. Read-only.

LandscapeFontNames Property Example

This example displays the landscape font names in the **FontNames** object at the end of the active document.

```
For Each aFont In LandscapeFontNames
    response = MsgBox(Prompt:=aFont, Buttons:=vbOKCancel)
    If response = vbCancel Then Exit For
Next aFont
```

Maximum Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMaximumC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMaximumX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMaximumA "}

RecentFiles object: Returns or sets the maximum number of recently used files that can appear on the **File** menu. Can be a number from 0 (zero) to 9. Read/write **Long**.

Dictionaries object: Returns the maximum number of custom dictionaries allowed. Read-only **Long**.

Maximum Property Example

This example disables the list of most recently used files.

```
RecentFiles.Maximum = 0
```

This example displays a message if the number of custom dictionaries is equal to the maximum number allowed. If the maximum number hasn't been reached, a custom dictionary named "MyDictionary.dic" is added.

```
If CustomDictionaries.Count = CustomDictionaries.Maximum Then
    MsgBox "Cannot add another dictionary file"
Else
    CustomDictionaries.Add "MyDictionary.dic"
End If
```

This example increases the number of items on the list of most recently used files by 1.

```
num = RecentFiles.Maximum
If num <> 9 Then RecentFiles.Maximum = num + 1
```

MouseAvailable Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMouseAvailableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMouseAvailableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMouseAvailableA "}

True if there's a mouse available for the system (always **True** on the Macintosh). Read-only **Boolean**.

MouseAvailable Property Example

This example displays a message if there's not a mouse available.

```
If Application.MouseAvailable = False Then
    Msgbox "Make sure your mouse is plugged in."
Else
    Msgbox "Mouse is available"
End If
```

NumLock Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumLockC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumLockX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumLockA "}

Returns the state of the NUM LOCK key. **True** if the keys on the numeric key pad insert numbers, **False** if the keys move the insertion point. Read-only **Boolean**.

NumLock Property Example

This example returns the current state of the NUM LOCK key.

```
theState = Application.NumLock
```

OrganizerCopy Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOrganizerCopyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOrganizerCopyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOrganizerCopyA "}

Copies the specified AutoText entry, toolbar, style, or macro project item from the source document or template to the destination document or template.

Syntax

expression.**OrganizerCopy**(***Source, Destination, Name, Object***)

expression Required. An expression that returns an **Application** object.

Source Required **String**. The document or template file name that contains the item you want to copy.

Destination Required **String**. The document or template file name to which you want to copy an item.

Name Required **String**. The name of the AutoText entry, toolbar, style, or macro you want to copy.

Object Required **Long**. The kind of item you want to copy. Can be one of the following **WdOrganizerObject** constants: **wdOrganizerObjectAutoText**, **wdOrganizerObjectCommandBars**, **wdOrganizerObjectProjectItems**, or **wdOrganizerObjectStyles**.

OrganizerCopy Method Example

This example copies all the AutoText entries in the template attached to the active document to the Normal template.

```
For Each aEntry In ActiveDocument.AttachedTemplate.AutoTextEntries
    Application.OrganizerCopy
Source:=ActiveDocument.AttachedTemplate.FullName, _
        Destination:=NormalTemplate.FullName, Name:=aEntry.Name, _
        Object:=wdOrganizerObjectAutoText
Next aEntry
```

If the style named "SubText" exists in the active document, this example copies the style to C:\Templates\MyTemplate.dot.

```
For Each sty In ActiveDocument.Styles
    If sty = "SubText" Then
        Application.OrganizerCopy Source:=ActiveDocument.Name, _
            Destination:="C:\Templates\MyTemplate.dot", Name:="SubText", _
            Object:=wdOrganizerObjectStyles
    End If
Next sty
```

OrganizerDelete Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOrganizerDeleteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOrganizerDeleteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOrganizerDeleteA "}

Deletes the specified style, AutoText entry, toolbar, or macro project item from a document or template.

Syntax

expression.**OrganizerDelete**(***Source, Name, Object***)

expression Required. An expression that returns an **Application** object.

Source Required **String**. The file name of the document or template that contains the item you want to delete.

Name Required **String**. The name of the style, AutoText entry, toolbar, or macro you want to delete.

Object Required **Long**. The kind of item you want to copy. Can be one of the following **WdOrganizerObject** constants: **wdOrganizerObjectAutoText**, **wdOrganizerObjectCommandBars**, **wdOrganizerObjectProjectItems**, or **wdOrganizerObjectStyles**.

OrganizerDelete Method Example

This example deletes the toolbar named "Custom 1" from the Normal template.

```
For Each cb In CommandBars
    If cb.Name = "Custom 1" Then
        Application.OrganizerDelete Source:=NormalTemplate.Name, _
            Name:="Custom 1", Object:=wdOrganizerObjectCommandBars
    End If
Next cb
```

This example prompts the user to delete each AutoText entry in the template attached to the active document. If the user clicks the Yes button, the AutoText entries are deleted.

```
For Each entry In ActiveDocument.AttachedTemplate.AutoTextEntries
    response = MsgBox("Do you want to delete the " & entry.Name & _
        " AutoText entry?", vbYesNoCancel)
    If response = vbYes Then
        Application.OrganizerDelete
Source:=ActiveDocument.AttachedTemplate.Name, _
            Name:=entry.Name, Object:=wdOrganizerObjectAutoText
    ElseIf response = vbCancel Then
        Exit For
    End If
Next entry
```

OrganizerRename Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOrganizerRenameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOrganizerRenameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOrganizerRenameA "}

Renames the specified style, AutoText entry, toolbar, or macro project item in a document or template.

Syntax

expression.**OrganizerRename(*Source, Name, NewName, Object*)**

expression Required. An expression that returns an **Application** object.

Source Required **String**. The file name of the document or template that contains the item you want to rename.

Name Required **String**. The name of the style, AutoText entry, toolbar, or macro you want to rename.

NewName Required **String**. The new name for the item.

Object Required **Long**. The kind of item you want to copy. Can be one of the following **WdOrganizerObject** constants: **wdOrganizerObjectAutoText**, **wdOrganizerObjectCommandBars**, **wdOrganizerObjectProjectItems**, or **wdOrganizerObjectStyles**.

OrganizerRename Method Example

This example changes the name of the style named "SubText" in the active document to "SubText2."

```
For Each sty In ActiveDocument.Styles
    If sty.NameLocal = "SubText" Then
        Application.OrganizerRename Source:=ActiveDocument.Name,
Name:="SubText", _
        NewName:="SubText2", Object:=wdOrganizerObjectStyles
    End If
Next sty
```

This example changes the name of the macro module named "Module1" in the attached template to "MyMacros."

```
dot = ActiveDocument.AttachedTemplate.Name
Application.OrganizerRename Source:=dot, Name:="Module1", _
    NewName:="MyMacros", Object:=wdOrganizerObjectProjectItems
```

Overtype Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOvertypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOvertypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOvertypeA "}

True if overtype mode is active. In overtype mode, the characters you type replace existing characters one by one. When overtype isn't active, the characters you type move existing text to the right. Read/write **Boolean**.

Overtype Property Example

If overtype mode is active, this example displays a message box asking whether overtype should be deactivated. If the user clicks the Yes button, overtype mode is made inactive.

```
If Options.Overtype = True Then
    aButton = MsgBox("Overtype is on. Turn off?", 4)
    If aButton = vbYes Then Options.Overtype = False
End If
```

PortraitFontNames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPortraitFontNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPortraitFontNamesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPortraitFontNamesA "}

Returns a **FontNames** object that includes the names of all the available portait fonts. Read-only.

PortraitFontNames Property Example

This example inserts a list of portrait fonts at the insertion point.

```
For Each aFont In PortraitFontNames
    With Selection
        .Collapse Direction:=wdCollapseEnd
        .InsertAfter aFont
        .InsertParagraphAfter
        .Collapse Direction:=wdCollapseEnd
    End With
Next aFont
```

PrintPreview Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPrintPreviewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPrintPreviewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPrintPreviewA "}

Toggles the view to or from print preview.

Note The **PrintPreview** property can be set to **True** or **False** to switch to or from print preview, respectively. You can also change the view by setting the **Type** property for the **View** object to **wdPrintPreview**.

Syntax

expression.**PrintPreview**

expression Required. An expression that returns an **Document** object.

PrintPreview Method Example

This example switches the active document to print preview if it's currently in some other view.

```
If PrintPreview = False Then
    ActiveDocument.PrintPreview
End If
```

PrintPreview Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintPreviewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintPreviewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintPreviewA "}

True if print preview is the current view. Read/write **Boolean**.

PrintPreview Property Example

This example switches the view to print preview.

```
PrintPreview = True
```

This example switches the active window from print preview to normal view.

```
PrintPreview = False
ActiveWindow.View.Type = wdNormalView
```

Quit Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthQuitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthQuitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthQuitA "}

Quits Word and optionally saves or routes the open documents.

Syntax

expression.**Quit(*SaveChanges, Format, RouteDocument*)**

expression Required. An expression that returns an **Application** object.

SaveChanges Optional **Variant**. Specifies whether Word saves changed documents before quitting. Can be one of the following **WdSaveOptions** constants: **wdDoNotSaveChanges**, **wdPromptToSaveChanges**, or **wdSaveChanges**.

OriginalFormat Optional **Variant**. Specifies the way Word saves documents whose original format was not Word Document format. Can be one of the following **WdOriginalFormat** constants: **wdOriginalDocumentFormat**, **wdPromptUser**, or **wdWordDocument**.

RouteDocument Optional **Variant**. **True** to route the document to the next recipient. If the document doesn't have a routing slip attached, this argument is ignored.

Quit Method Example

This example quits Word and prompts the user to save each document that has changed since it was last saved.

```
Application.Quit SaveChanges:=wdPromptToSaveChanges
```

This example prompts the user to save all documents. If the user clicks the Yes button, all documents are saved in the Word format before Word quits.

```
response = MsgBox("Do you want to save all documents?", vbYesNo)
If response = vbYes Then Application.Quit _
    SaveChanges:=wdSaveChanges, OriginalFormat:=wdWordDocument
```

RecentFiles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecentFilesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecentFilesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecentFilesA "}

Returns a **RecentFiles** collection that represents the most recently accessed files. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

RecentFiles Property Example

This example opens the first item in the **RecentFiles** collection (the first document name at the bottom of the **File** menu).

```
If RecentFiles.Count >= 1 Then RecentFiles(1).Open
```

This example displays the name of each file in the **RecentFiles** collection.

```
For Each rFile In RecentFiles
    MsgBox rFile.Name
Next rFile
```

Repeat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRepeatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRepeatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRepeatA "}

Repeats the most recent editing action one or more times. Returns **True** if the commands were repeated successfully.

Note Using this method is the equivalent to using the **Repeat** command on the **Edit** menu.

Syntax

expression.**Repeat(*Times*)**

expression Optional. An expression that returns an **Application** object.

Times Optional **Variant**. The number of times you want to repeat the last command.

Repeat Method Example

This example inserts the text "Hello" followed by two paragraphs (the second typing action is repeated once).

```
Selection.TypeText "Hello"
Selection.TypeParagraph
Repeat
```

This example repeats the last command three times (if it can be repeated).

```
On Error Resume Next
If Repeat(3) = True Then StatusBar = "Action repeated"
```

Resize Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResizeA "}

Sizes the Word application window or the specified task window. If the window is maximized or minimized, an error occurs. On the Macintosh, this method generates an error.

Note Use the **Width** or **Height** property to set the window width and height independently.

Syntax

expression.**Resize(*Width, Height*)**

expression Required. An expression that returns an **Application** or **Task** object.

Width Required **Long**. The width of the window, in points.

Height Required **Long**. The height of the window, in points.

Resize Method Example

This example resizes the Microsoft Excel application window to 6 inches wide by 4 inches high.

```
If Tasks.Exists("Microsoft Excel") = True Then
    With Tasks("Microsoft Excel")
        .WindowState = wdWindowStateNormal
        .Resize Width:=InchesToPoints(6), Height:=InchesToPoints(4)
    End With
End If
```

This example resizes the Word application window to 7 inches wide by 6 inches high.

```
With Application
    .WindowState = wdWindowStateNormal
    .Resize Width:=InchesToPoints(7), Height:=InchesToPoints(6)
End With
```

BackgroundPrintingStatus Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackgroundPrintingStatusC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackgroundPrintingStatusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackgroundPrintingStatusA "}

Returns the number of print jobs in the background printing queue. Read-only **Long**.

BackgroundPrintingStatus Property Example

This example returns the number of Word print jobs currently queued up for background printing.

```
If Options.PrintBackground = True Then
    jobs = Application.BackgroundPrintingStatus
End If
```

If the number of print jobs is greater than 0 (zero), this example displays a message in the status bar.

```
If Application.BackgroundPrintingStatus > 0 Then
    StatusBar = Application.BackgroundPrintingStatus & " print jobs are
queued up"
End If
```

Languages Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguagesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguagesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguagesA "}

Returns a **Languages** collection that represents the proofing languages from the **Language** dialog box (**Tools** menu). Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Languages Property Example

This example returns the full path and file name of the active spelling dictionary.

```
Set mySpell = Languages(Selection.LanguageID).ActiveSpellingDictionary
MsgBox mySpell.Path & Application.PathSeparator & mySpell.Name
```

This example uses the `aLang()` array to store the proofing language names.

```
ReDim aLang(Languages.Count - 1)
i = 0
For Each myLanguage In Languages
    aLang(i) = myLanguage.NameLocal
    i = i + 1
Next myLanguage
```

ScreenUpdating Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproScreenUpdatingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproScreenUpdatingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproScreenUpdatingA "}

True if screen updating is turned on. Read/write **Boolean**.

Remarks

The **ScreenUpdating** property controls most display changes on the monitor while a procedure is running. When screen updating is turned off, toolbars remain visible and Word still allows the procedure to display or retrieve information using status bar prompts, input boxes, dialog boxes, and message boxes. You can increase the speed of some procedures by keeping screen updating turned off. You must set the **ScreenUpdating** property to **True** when the procedure finishes or stops after an error.

ScreenUpdating Property Example

This example turns off screen updating and then adds a new document. Five hundred lines of text are added to the document. At every fiftieth line, the macro selects the line and refreshes the screen.

```
Application.ScreenUpdating = False
Documents.Add
For x = 1 To 500
    With ActiveDocument.Content
        .InsertAfter "This is line ." & x
        .InsertParagraphAfter
    End With
If x Mod 50 = 0 Then
   ActiveDocument.Paragraphs(x).Range.Select
   Application.ScreenRefresh
End If
Next x
Application.ScreenUpdating = True
```

SpecialMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpecialModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpecialModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpecialModeA "}

True if Word is in a special mode (for example, CopyText mode or MoveText mode). Read-only **Boolean**.

Remarks

Word enters a special copy or move mode if you press F2 or SHIFT+F2 while text is selected.

SpecialMode Property Example

This example checks to see whether Word is in a special mode. If it is, ESC is activated before the current selection is cut and pasted.

```
If Application.SpecialMode = True Then SendKeys "ESC"
With Selection
    .Cut
    .EndKey Unit:=wdStory
    .Paste
End With
```

SubstituteFont Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSubstituteFontC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSubstituteFontX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSubstituteFontA "}

Sets font-mapping options, which are reflected in the **Font Substitution** dialog box (**Compatibility** tab, **Options** dialog box, **Tools** menu).

Syntax

expression.**SubstituteFont(*UnavailableFont, SubstituteFont*)**

expression Required. An expression that returns an **Application** object.

UnavailableFont Required **String**. The name of a font not available on your computer that you want to map to a different font for display and printing.

SubstituteFont Required **String**. The name of a font available on your computer that you want to substitute for the unavailable font.

SubstituteFont Method Example

This example substitutes Courier for Myfont.

```
Application.SubstituteFont UnavailableFont:= "Myfont", _
    SubstituteFont:= "Courier"
```

Target Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTargetC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTargetX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTargetA "}

Returns or sets the document item that the **Previous** and **Next** methods locate when they're applied to the **Browser** object. Read/write **Long**.

Can be one of the following **WdBrowseTarget** constants:

wdBrowseComment
wdBrowseEdit
wdBrowseEndnote
wdBrowseField
wdBrowseFind
wdBrowseFootnote
wdBrowseGoTo
wdBrowseGraphic
wdBrowseHeading
wdBrowsePage
wdBrowseSection
wdBrowseTable

Target Property Example

This example moves the insertion point to the next comment in the active document.

```
With Application.Browser
    .Target = wdBrowseComment
    .Next
End With
```

MacroContainer Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMacroContainerC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMacroContainerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMacroContainerA "}

Returns a **Template** object or **Document** object that represents the template or document in which the module that contains the running procedure is stored. Read-only.

MacroContainer Property Example

This example displays the name of the document or template in which the module that contains the running procedure is stored.

```
Set cntnr = MacroContainer
MsgBox cntnr.Name
```

Version Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVersionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVersionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVersionA "}

Application object: Returns the Microsoft Word version number. Read-only **String**.

System object: Returns the version number of the operating system. Read-only **String**.

Version Property Example

This example displays the Word version number in a message box.

```
Msgbox "The version of Word is " & Application.Version
```

This example displays the version number of the operating system in a message box.

```
Msgbox "The system version is " & System.Version
```

ViewType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproViewTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproViewTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproViewTypeA "}

Returns or sets the view for the **TextRetrievalMode** object. Can be one of the following **WdViewType** constants: **wdMasterView**, **wdNormalView**, **wdOnlineView**, **wdOutlineView**, **wdPageView**, or **wdPrintPreview**. Read/write **Long**.

Remarks

Changing the view for the **TextRetrievalMode** object doesn't change the display of a document on the screen. Instead, it determines what characters in the document will be included when a range is retrieved.

ViewType Property Example

This example sets the view for text retrieval to outline view and then displays the contents of the active document in a dialog box. Note that only the text displayed in outline view is retrieved.

```
Set myText = ActiveDocument.Content
myText.TextRetrievalMode.ViewType = wdOutlineView
Msgbox myText
```

Comment Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCommentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCommentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCommentA "}

Returns the comment associated with the specified version of a document. Read-only **String**.

Comment Property Example

This example displays the comment text for the first version of the active document.

```
If ActiveDocument.Versions.Count >= 1 Then
    MsgBox Prompt:=ActiveDocument.Versions(1).Comment, _
        Title:="First Version Comment"
End If
```

This example saves a version of the document with the user's comment and then displays the comment.

```
Set myVersions = ActiveDocument.Versions
aComment = InputBox("Type a comment")
myVersions.Save Comment:=aComment
last = myVersions.Count
MsgBox Prompt:=myVersions(last).Comment, Title:=myVersions(last).SavedBy
```

GoBack Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGoBackC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGoBackX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGoBackA "}

Moves the insertion point among the last three locations where editing occurred in the active document (the same as pressing SHIFT+F5).

Syntax

expression.**GoBack**

expression Required. An expression that returns an **Application** object.

GoBack Method Example

This example opens the most recently used file and then moves the insertion point to the location where editing last occurred.

```
RecentFiles(1).Open
Application.GoBack
```

NameLocal Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNameLocalC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNameLocalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNameLocalA "}

Language object: Returns the name of a proofing tool language in the language of the user. Read-only **String**.

Style object: Returns the name of a built-in style in the language of the user. Setting this property renames a user-defined style or adds an alias to a built-in style. Read/write **String**.

NameLocal Property Example

This example displays the style name (in the language of the user) applied to the selected paragraphs. If more than one style has been applied to the selection, the first style name is displayed.

```
MsgBox Selection.Paragraphs.Style.NameLocal
```

This example adds the name "MyH1" as the alias for the Heading 1 style in the active document.

```
ActiveDocument.Styles("Heading 1").NameLocal = "MyH1"
```

This example renames the style named "Test" to "Intro."

```
ActiveDocument.Styles("Test").NameLocal = "Intro"
```

This example displays the name of the German language two different ways – first in the language of the user, and then in German.

```
MsgBox Languages(wdGerman).NameLocal
MsgBox Languages(wdGerman).Name
```

Options Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOptionsA "}

Returns an **Options** object that represents application settings you can set in Word. Read-only.

Options Property Example

This example disables fast saves and then saves the active document.

```
Options.AllowFastSave = False
ActiveDocument.Save
```

This example prints Sales.doc with comments and field results.

```
With Options
    .PrintFieldCodes = False
    .PrintComments = True
End With
Documents("Sales.doc").PrintOut
```

ResetIgnoreAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetIgnoreAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetIgnoreAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetIgnoreAllA "}

Clears the list of words that were previously ignored during a spelling check. After you run this method, previously ignored words are checked along with all the other words.

Syntax

expression.**ResetIgnoreAll**

expression Required. An expression that returns an **Application** object.

ResetIgnoreAll Method Example

This example clears the list of words that were ignored during a previous spelling check and then begins a new spelling check on the active document.

```
Application.ResetIgnoreAll
ActiveDocument.CheckSpelling
```

VBE Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVBEC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVBEX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVBEA "}

Returns a **VBE** object that represents the VB Editor. Read-only.

VBE Property Example

This example displays the number of references available for the active project.

```
MsgBox "References = " & VBE.ActiveVBProject.References.Count
```

ListCommands Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthListCommandsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthListCommandsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthListCommandsA "}

Creates a new document and then inserts a table of Word commands along with their associated shortcut keys and menu assignments.

Syntax

expression.**ListCommands(*ListAllCommands*)**

expression Required. An expression that returns an **Application** object.

ListAllCommands Required **Boolean**. **True** to include all Word commands and their assignments (whether customized or built-in). **False** to include only commands with customized assignments.

ListCommands Method Example

This example creates a new document that lists all Word commands along with their associated shortcut keys and menu assignments. The example then prints and closes the new document without saving changes.

```
Application.ListCommands ListAllCommands:=True
With ActiveDocument
    .PrintOut
    .Close SaveChanges:=wdDoNotSaveChanges
End With
```

SendWindowMessage Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSendWindowMessageC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSendWindowMessageX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSendWindowMessageA "}

Sends a Windows message and its associated parameters to the specified task. On the Macintosh, this method isn't available and generates an error.

Syntax

expression.**SendWindowMessage(*Message, wParam, lParam*)**

expression Required. An expression that returns a **Task** object.

Message Required **Long**. A hexidecimal number that corresponds to the message you want to send. If you have the Microsoft Win32 Software Development Kit, you can look up the name of the message in the header files (WINUSER.H, for example) to find the associated hexadecimal number (precede the hexidecimal value with &h).

wParam, lParam Required **Long**. Parameters appropriate for the message you're sending. For information about what these values represent, see the reference topic for that message in the Win32 online Help included with the Microsoft Win32 Software Developement Kit. To retrieve the appropriate values, you may need to use the Spy utility (which comes with the kit).

SendWindowMessage Method Example

If Notepad is running, this example displays the **About** dialog box (in Notepad) by sending a WM_COMMAND message to Notepad. The **SendWindowMessage** method is used to send the WM_COMMAND message (111 is the hexidecimal value for WM_COMMAND), with the parameters 11 and 0. The Spy utility was used to determine the ***wParam*** and ***lParam*** values.

```
For Each myTask In Tasks
    If InStr(myTask.Name, "Notepad") > 0 Then
        myTask.Activate
        myTask.SendWindowMessage &h111, 11, 0
    End If
Next myTask
```

PathSeparator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPathSeparatorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPathSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPathSeparatorA"}

Returns the character used to separate folder names. In Windows, this property returns a backslash (\);on the Macintosh, this property returns a colon (:).

Note The **FullName** property returns the path and file name as a single string.

PathSeparator Property Example

This example displays the path and file name of the active document.

```
MsgBox ActiveDocument.Path & Application.PathSeparator & _
    ActiveDocument.Name
```

If the first add-in is a template, this example unloads the template and opens it.

```
If Addins(1).Compiled = False Then
    Addins(1).Installed = False
    Documents.Open FileName:=AddIns(1).Path & Application.PathSeparator & _
        AddIns(1).Name
End If
```

QuickDrawInstalled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproQuickDrawInstalledC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproQuickDrawInstalledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproQuickDrawInstalledA"}

True if Macintosh QuickDraw GX is installed. Read-only **Boolean**.

Note In Windows, **QuickDrawInstalled** isn't available and generates an error.

QuickDrawInstalled Property Example

This example displays a message box that indicates whether or not QuickDraw GX is installed.

```
If System.QuickDrawInstalled = True Then
    MsgBox "QuickDraw GX is installed on this system."
Else
    Msgbox "QuickDraw GX is not installed on this system."
End If
```

DefaultTab Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultTabC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultTabX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultTabA"}

Returns or sets the active tab when the specified dialog box is displayed. Can be any of the **WdWordDialogTab** constants. Read/write **Long**.

DefaultTab Property Example

This example displays the **Page Setup** dialog box with the **Paper Source** tab selected.

```
With Dialogs(wdDialogFilePageSetup)
    .DefaultTab = wdDialogFilePageSetupTabPaperSource
    .Show
End With
```

StatusBar Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStatusBarC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStatusBarX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStatusBarA"}

Displays the specified text in the status bar. Write-only **String**.

StatusBar Property Example

This example displays a message in the status bar.

```
StatusBar = "Please wait..."
```

This example displays in the status bar the name of the template attached to the active document.

```
aName = ActiveDocument.AttachedTemplate.Name
StatusBar = aName & " template is attached to the active document"
```

StartupPath Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStartupPathC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStartupPathX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStartupPathA"}

Returns or sets the complete path of the Startup folder, excluding the final separator. Read/write **String**.

Note Templates and add-ins located in the Startup folder are automatically loaded when you start Word.

StartupPath Property Example

This example displays the complete path of the Startup folder.

```
MsgBox Application.StartupPath
```

This example enables the user to change the path of the Startup folder.

```
x = MsgBox("Do you want to change the startup path?", vbYesNo, _
    "Current path = " & Application.StartupPath)
If x = vbYes Then
    newStartup = InputBox("Type a startup path")
    Application.StartupPath = newStartup
End If
```

OperatingSystem Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOperatingSystemC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOperatingSystemX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOperatingSystemA"}

Returns the name of the current operating system (for example, "Windows" or "Windows NT"). Read-only **String**.

OperatingSystem Property Example

This example displays a message that includes the name of the current operating system.

```
MsgBox "This computer is running " & System.OperatingSystem
```

CentimetersToPoints Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCentimetersToPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCentimetersToPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCentimetersToPointsA"}

Converts a measurement from centimeters to points (1 cm = 28.35 points). Returns the converted measurement as a **Single**.

Syntax

expression.**CentimetersToPoints(*Centimeters*)**

expression Optional. An expression that returns an **Application** object.

Centimeters Required **Single**. The centimeter value to be converted to points.

CentimetersToPoints Method Example

This example adds a centered tab stop to all the paragraphs in the selection. The tab stop is positioned at 1.5 centimeters from the left margin.

```
Selection.Paragraphs.TabStops.Add Position:=CentimetersToPoints(1.5), _
    Alignment:=wdAlignTabCenter
```

This example sets a first-line indent of 2.5 centimeters for the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).FirstLineIndent = CentimetersToPoints(2.5)
```

Display Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDisplayC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDisplayX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDisplayA"}

Displays the specified built-in Word dialog box until either the user closes it or the specified amount of time has passed. Returns a **Long** that indicates which button was clicked to close the dialog box.

Return value	Description
-2	The **Close** button.
-1	The **OK** button.
0 (zero)	The **Cancel** button.
> 0 (zero)	A command button: 1 is the first button, 2 is the second button, and so on.

Note Any actions initiated or settings specified while a dialog box is displayed using this method aren't carried out. Use the **Show** method to display a dialog box and carry out actions or apply settings.

Syntax

expression.**Display**(***TimeOut***)

expression Required. An expression that returns a **Dialog** object.

TimeOut Optional **Variant**. The amount of time that Word will wait before closing the dialog box automatically. One unit is approximately 0.001 second. Concurrent system activity may increase the effective time value. If this argument is omitted, the dialog box is closed when the user closes it.

Display Method Example

This example displays the **About** dialog box.

```
Set aDialog = Dialogs(wdDialogHelpAbout)
aDialog.Display
```

This example displays the **Zoom** dialog box for approximately nine seconds.

```
Dialogs(wdDialogViewZoom).Display TimeOut:=9000
```

EnableCancelKey Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnableCancelKeyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnableCancelKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnableCancelKeyA"}

Returns or sets the way that Word handles CTRL+BREAK user interruptions. Can be either one of the following **WdEnableCancelKey** constants: **wdCancelDisabled** (prevents CTRL+BREAK from interrupting a macro) or **wdCancelInterrupt** (allows a macro to be interrupted by CTRL+BREAK). Read/write **Long**.

Remarks

Use this property very carefully. If you use **wdCancelDisabled**, there's no way to interrupt a runaway loop or other non – self-terminating code. The **EnableCancelKey** property is reset to **wdCancelInterrupt** when your code stops running.

EnableCancelKey Property Example

This example disables CTRL+BREAK from interrupting a counter loop.

```
Application.EnableCancelKey = wdCancelDisabled
For i = 1 To 10000
    StatusBar = i
Next i
Application.EnableCancelKey = wdCancelInterrupt
```

International Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInternationalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInternationalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInternationalA"}

Returns information about the current country and international settings. Read-only **Variant**.

Syntax

expression.**International(*Index*)**

expression Required. An expression that returns an **Application** object.

Index Required **Long**. Specifies a single item to be returned. Can be one of the following **WdInternationalIndex** constants.

Constant	Description
wd24HourClock	Returns **True** if you're using 24-hour time; returns **False** if you're using 12-hour time.
wdCurrencyCode	Returns the currency symbol ($ in U.S. English).
wdDateSeparator	Returns the date separator (/ in U.S. English).
wdDecimalSeparator	Returns the decimal separator (. in U.S. English).
wdInternationalAM	Returns the string used to indicate morning hours (for example, 10 AM).
wdInternationalPM	Returns the string used to indicate afternoon and evening hours (for example, 2 PM).
wdListSeparator	Returns the list separator (, in U.S. English).
wdProductLanguageID	Returns the language version of Word.
wdThousandsSeparator	Returns the thousands separator (, in U.S. English).
wdTimeSeparator	Returns the time separator (: in U.S. English).

International Property Example

This example displays the currency format in the status bar.

```
StatusBar = "Currency Format: " & _
    Application.International(wdCurrencyCode)
```

LinesToPoints Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthLinesToPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthLinesToPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthLinesToPointsA"}

Converts a measurement from lines to points (1 line = 12 points). Returns the converted measurement as a **Single**.

Syntax

expression.**LinesToPoints(*Lines*)**

expression Optional. An expression that returns an **Application** object.

Lines Required **Single**. The line value to be converted to points.

LinesToPoints Method Example

This example sets the paragraph line spacing in the selection to three lines.

```
With Selection.ParagraphFormat
    .LineSpacingRule = wdLineSpaceMultiple
    .LineSpacing = LinesToPoints(3)
End With
```

MillimetersToPoints Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMillimetersToPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMillimetersToPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMillimetersToPointsA"}

Converts a measurement from millimeters to points (1 mm = 2.85 points). Returns the converted measurement as a **Single**.

Syntax

expression.**MillimetersToPoints(*Millimeters*)**

expression Optional. An expression that returns an **Application** object.

Millimeters Required **Single**. The millimeter value to be converted to points.

MillimetersToPoints Method Example

This example sets the hyphenation zone in the active document to 8.8 millimeters.

```
ActiveDocument.HyphenationZone = MillimetersToPoints(8.8)
```

This example expands the spacing of the selected characters to 2.8 points.

```
Selection.Font.Spacing = MillimetersToPoints(1)
```

PicasToPoints Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPicasToPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPicasToPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPicasToPointsA"}

Converts a measurement from picas to points (1 pica = 12 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PicasToPoints(*Picas*)**

expression Optional. An expression that returns an **Application** object.

Picas Required **Single**. The pica value to be converted to points.

PicasToPoints Method Example

This example adds line numbers to the active document and sets the distance between the line numbers and the document text to 4 picas.

```
With ActiveDocument.PageSetup.LineNumbering
    .Active = True
    .DistanceFromText = PicasToPoints(4)
End With
```

This example sets the first-line indent for the selected paragraphs to 3 picas.

```
Selection.ParagraphFormat.FirstLineIndent = PicasToPoints(3)
```

UsableHeight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUsableHeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUsableHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUsableHeightA"}

Returns the height (in points) of the space that can be used by a window in the application window area. Read-only **Long**.

Note If the **UsableHeight** property returns 1, there's no space available for a document window. When there's no vertical space available, the horizontal (**UsableWidth**) value isn't valid. To determine the actual available height, subtract 1 from the **UsableHeight** value.

UsableHeight Property Example

This example increases the size of the active document window to fill the application window area except for a 5-point border on the top and the left side of the window.

```
With ActiveWindow
    .WindowState = wdWindowStateNormal
    .Top = 5
    .Left = 5
    .Height = Application.UsableHeight
    .Width = Application.UsableWidth
End With
```

UsableWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUsableWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUsableWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUsableWidthA"}

Returns the width (in points) of the space that can be used by a window in the application window area. Read-only **Long**.

Note The value of the **UsableWidth** property is valid only when the value of the **UsableHeight** property is greater than 1, which indicates that there's space available for a document window.

UsableWidth Property Example

This example displays the height and width of the application window area.

```
vertSpace = Format(PointsToInches(Application.UsableHeight), "Fixed")
horizSpace = Format(PointsToInches(Application.UsableWidth), "Fixed")
MsgBox "Usable Height: " & vertSpace & " inches" & vbCr & _
    "Usable Width: " & horizSpace & " inches"
```

InchesToPoints Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInchesToPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInchesToPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInchesToPointsA"}

Converts a measurement from inches to points (1 inch = 72 points). Returns the converted measurement as a **Single**.

Syntax

expression.**InchesToPoints(*Inches*)**

expression Optional. An expression that returns an **Application** object.

Inches Required **Single**. The inch value to be converted to points.

InchesToPoints Method Example

This example sets the space before for the selected paragraphs to 0.25 inch.

```
Selection.ParagraphFormat.SpaceBefore = InchesToPoints(0.25)
```

This example prints each open document after setting the left and right margins to 0.65 inch.

```
For Each openDoc in Documents
    With openDoc
        .PageSetup.LeftMargin = InchesToPoints(0.65)
        .PageSetup.RightMargin = InchesToPoints(0.65)
        .PrintOut
    End With
Next openDoc
```

CleanString Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCleanStringC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCleanStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCleanStringA"}

Removes nonprinting characters (character codes 1 – 29) and special Word characters from the specified string or changes them to spaces (character code 32), as described in the "Remarks" section. Returns the result as a string.

Syntax

expression.**CleanString(*String*)**

expression Optional. An expression that returns an **Application** object.

String Required **String**. The source string.

Remarks

In Windows, the following characters are converted as described in this table.

Character code	Description
7 (beep)	Removed unless preceded by character 13 (paragraph), then converted to character 9 (tab).
10 (line feed)	Converted to character 13 (paragraph) unless preceded by character 13, then removed.
13 (paragraph)	Unchanged.
31 (optional hyphen)	Removed.
160 (nonbreaking space)	Converted to character 32 (space).
172 (optional hyphen)	Removed.
176 (nonbreaking space)	Converted to character 32 (space).
182 (paragraph mark)	Removed.
183 (bullet)	Converted to character 32 (space).

On the Macintosh, the following characters are converted as described in this table.

Character code	Description
7 (beep)	Removed unless preceded by character 13 (paragraph), then converted to character 9 (tab).
10 (line feed)	Converted to character 13 (paragraph) unless preceded by character 13, then removed.
13 (paragraph)	Unchanged.
31 (optional hyphen)	Removed.
194 (optional hyphen)	Removed.
202 (nonbreaking space)	Converted to character 32 (space).

CleanString Method Example

This example removes nonprinting characters from the selected text and inserts the result into a new document.

```
cleanUp = Application.CleanString(Selection.Text)
Set myDoc = Documents.Add
myDoc.Content.InsertAfter cleanUp
```

This example removes nonprinting characters from the selected field code and then displays the result.

```
ActiveWindow.View.ShowFieldCodes = True
ActiveDocument.Fields(1).Select
MsgBox Application.CleanString(Selection.Text)
```

OnTime Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOnTimeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOnTimeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOnTimeA"}

Starts a background timer that runs a macro on the specified date and at the specified time.

Syntax

expression.**OnTime(*When, Name, Tolerance*)**

expression Required. An expression that returns an **Application** object.

When Required **Variant**. The time at which the macro is to be run. Can be a string that specifies a time (for example, `"4:30 pm"` or `"16:30"`), or it can be a serial number returned by a function such as **TimeValue** or **TimeSerial** (for example, `TimeValue("2:30 pm")` or `TimeSerial(14, 30, 00)`). You can also include the date (for example, `"6/30 4:15 pm"` or `TimeValue("6/30 4:15 pm")`).

Use the sum of the return values of the **Now** function and either the **TimeValue** or **TimeSerial** function to set a timer to run a macro a specified amount of time after the statement is run. For example, use `Now+TimeValue("00:05:30")` to run a macro 5 minutes and 30 seconds after the statement is run.

Name Required **String**. The name of the macro to be run. Use the complete macro path to ensure that the correct macro is run (for example, `"Project.Module1.Macro1"`). For the macro to run, the document or template must be available both when the **OnTime** instruction is run and when the time specified by ***When*** arrives. For this reason, it's best to store the macro in Normal.dot or another global template that's loaded automatically.

Tolerance Optional **Variant**. The maximum time (in seconds) that can elapse before a macro that wasn't run at the time specified by ***When*** is canceled. Macros may not always run at the specified time. For example, if a sort operation is under way or a dialog box is being displayed, the macro will be delayed until Word has completed the task. If this argument is 0 (zero) or omitted, the macro is run regardless of how much time has elapsed since the time specified by ***When***.

Remarks

Word can maintain only one background timer set by **OnTime**. If you start another timer before an existing timer runs, the existing timer is canceled.

OnTime Method Example

This example runs the macro named "MyMacro" in the current module at 3:55 P.M.

```
Application.OnTime When:="15:55:00", Name:="MyMacro"
```

This example runs the macro named "Macro1" 15 seconds from the time the example is run. The macro name includes the project and module name.

```
Application.OnTime When:=Now + TimeValue("00:00:15"),
Name:="Project1.Module1.Macro1"
```

This example runs the macro named "Start" at 1:30 P.M. The macro name includes the project and module name.

```
Application.OnTime When:=TimeValue("1:30 pm"),
Name:="VBAProj.MyModule.Start"
```

Show Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShowC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShowA"}

Displays and carries out actions initiated in the specified built-in Word dialog box. Returns a **Long** that indicates which button was clicked to close the dialog box.

Return value	Description
-2	The **Close** button.
-1	The **OK** button.
0 (zero)	The **Cancel** button.
> 0 (zero)	A command button: 1 is the first button, 2 is the second button, and so on.

Note Use the **Display** method to display a dialog box but not have any actions carried out or settings applied when the dialog box is closed.

Syntax

expression.**Show(*TimeOut*)**

expression Required. An expression that returns a **Dialog** object.

TimeOut Optional **Variant**. The amount of time that Word will wait before closing the dialog box automatically. One unit is approximately 0.001 second. Concurrent system activity may increase the effective time value. If this argument is omitted, the dialog box is closed when the user dismisses it.

Show Method Example

This example displays the **Find and Replace** dialog box with the word "Blue" preset in the **Find what** text box.

```
With  Dialogs(wdDialogEditFind)
    .Find = "Blue"
    .Show
End With
```

This example displays and carries out any action initiated in the **Open** dialog box. The file name is set to *.* so that all file names are displayed.

```
With Dialogs(wdDialogFileOpen)
    .Name = "*.*"
    .Show
End With
```

This example displays and carries out any action initiated in the **Zoom** dialog box. If there are no actions initiated for approximately 9 seconds, the dialog box is closed.

```
Dialogs(wdDialogViewZoom).Show TimeOut:=9000
```

HelpTool Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthHelpToolC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthHelpToolX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthHelpToolA "}

Changes the pointer from an arrow to a question mark, indicating that you'll get context-sensitive Help information about the next command or screen element you click. If you click text, Word displays a box describing current paragraph and character formats. Pressing ESC turns the pointer back to an arrow.

Syntax

expression.**HelpTool()**

expression Required. An expression that returns an **Application** object.

HelpTool Method Example

This example changes the mouse pointer from an arrow to a question mark.

```
Application.HelpTool
```

AutomaticChange Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutomaticChangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutomaticChangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutomaticChangeA "}

Performs an **AutoFormat** action when there's a change suggested by the Office Assistant. If no AutoFormat action is active, this method generates an error.

Syntax

expression.**AutomaticChange()**

expression Required. An expression that returns an **Application** object.

AutomaticChange Method Example

This example completes an Office Assistant AutoFormat action if one is active.

```
Application.AutomaticChange
```

ShowMe Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShowMeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShowMeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShowMeA "}

Displays the Office Assistant or the Help window when there's more information available. If additional information isn't available, this method generates a message that no associated Help topic exists.

Syntax

expression.**ShowMe()**

expression An expression that returns an **Application** object.

ShowMe Method Example

This examples completes a TipWizard Show Me action if one's available.

```
Application.ShowMe
```

AccentedLetters Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAccentedLettersC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAccentedLettersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAccentedLettersA"}

True if the specified index contains separate headings for accented letters (for example, words that begin with "À" are under one heading and words that begin with "A" are under another). Read/write **Boolean**.

AccentedLetters Property Example

This example formats the first index in the active document in a single column, with the appropriate letter preceding each alphabetic group and separate headings for accented letters.

```
If ActiveDocument.Indexes.Count >= 1 Then
    With ActiveDocument.Indexes(1)
        .HeadingSeparator = wdHeadingSeparatorLetter
        .NumberOfColumns = 1
        .AccentedLetters = True
    End With
End If
```

BrowseExtraFileTypes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBrowseExtraFileTypesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBrowseExtraFileTypesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBrowseExtraFileTypesA"}

Set this property to "text/html" to allow hyperlinked HTML files to be opened in Word (instead of the default Internet browser). Read/write **String**.

BrowseExtraFileTypes Property Example

This example allows hyperlinked HTML files to be opened in Word (instead of the default Internet browser).

```
Application.BrowseExtraFileTypes = "text/html"
```

ChangeFileOpenDirectory Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthChangeFileOpenDirectoryC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthChangeFileOpenDirectoryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthChangeFileOpenDirectoryA"}

Sets the folder in which Word searches for documents. The specified folder's contents are listed the next time the **Open** dialog box (**File** menu) is displayed.

Note Word searches the specified folder for documents until the user changes the folder in the **Open** dialog box or the current Word session ends. Use the **DefaultFilePath** property to change the default folder for documents in every Word session.

Syntax

expression.**ChangeFileOpenDirectory(*Path*)**

expression Optional. An expression that returns an **Application** object.

Path Required **String**. The path to the folder in which Word searches for documents.

ChangeFileOpenDirectory Method Example

This example changes the folder in which Word searches for documents, and then opens a file named "Test.doc."

```
ChangeFileOpenDirectory "C:\My Documents"
Documents.Open FileName:="Test.doc"
```

This example changes the the folder in which Word searches for documents, and then displays the **Open** dialog box.

```
ChangeFileOpenDirectory "C:\"
Dialogs(wdDialogFileOpen).Show
```

CustomizationContext Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCustomizationContextC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCustomizationContextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCustomizationContextA"}

Returns or sets a **Template** or **Document** object that represents the template or document in which changes to menu bars, toolbars, and key bindings are stored. Corresponds to the value of the **Save in** box on the **Commands** tab in the **Customize** dialog box (**Tools** menu). Read/write.

CustomizationContext Property Example

This example adds the ALT+CTRL+W key combination to the **FileClose** command. The keyboard customization is saved in the Normal template.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyW), _
    KeyCategory:=wdKeyCategoryCommand, Command:="FileClose"
```

This example adds the **File Versions** button to the **Standard** toolbar. The command bar customization is saved in the template attached to the active document.

```
CustomizationContext = ActiveDocument.AttachedTemplate
Application.CommandBars("Standard").Controls.Add Type:=msoControlButton, _
        ID:=2522, Before:=8
```

DefaultTableSeparator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultTableSeparatorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultTableSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultTableSeparatorA"}

Returns or sets the single character used to separate text into cells when text is converted to a table. Read/write **String**.

Note The value of the **DefaultTableSeparator** property is used if the ***Separator*** argument is omitted from the **ConvertToTable** method.

DefaultTableSeparator Property Example

This example changes the default table separator character.

```
Application.DefaultTableSeparator = "%"
```

Extensions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExtensionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExtensionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExtensionsA"}

Returns the file name extensions associated with the specified **FileConverter** object. Read-only **String**.

Extensions Property Example

This example displays the name and file name extensions for first file converter.

```
Set aConverter = FileConverters(1)
MsgBox "The file extensions for " & aConverter.FormatName & _
    " files are: " & aConverter.Extensions
```

GoForward Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGoForwardC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGoForwardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGoForwardA"}

Moves the insertion point forward among the last three locations where editing occurred in the active document.

Syntax

expression.**GoForward**

expression Required. An expression that returns an **Application** object.

GoForward Method Example

This example moves the insertion point to the next location where editing occurred.

```
Application.GoForward
```

IsObjectValid Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsObjectValidC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsObjectValidX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsObjectValidA"}

True if the specified variable that references an object is valid. **False** if the object referenced by the variable has been deleted. Read-only **Boolean**.

Syntax

expression.**IsObjectValid(*Object*)**

expression Optional. An expression that returns an **Application** object.

Object Required **Object**. A variable that references an object.

IsObjectValid Property Example

This example adds a table to the active document and assigns it to the variable `aTable`. The example deletes the first table from the document. If the table that `aTable` refers to was not the first table in the document (that is, if `aTable` is still a valid object), the example also removes any borders from that table.

```
Set aTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=2, NumColumns:=3)
ActiveDocument.Tables(1).Delete
If IsObjectValid(aTable) = True Then aTable.Borders.Enable = False
```

PointsToCentimeters Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPointsToCentimetersC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPointsToCentimetersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPointsToCentimetersA"}

Converts a measurement from points to centimeters (1 centimeter = 28.35 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PointsToCentimeters(*Points*)**

expression Optional. An expression that returns an **Application** object.

Points Required **Single**. The measurement, in points.

PointsToCentimeters Method Example

This example converts a measurement of 30 points to the corresponding number of centimeters.

```
MsgBox PointsToCentimeters(30) & " centimeters"
```

This example converts the value of the variable `data` (a measurement in points) to centimeters, inches, lines, millimeters, or picas, depending on the value of the variable `unit` (a value from 1 through 5 that indicates the resulting unit of measurement).

```
Select Case unit
    Case 1
        result = PointsToCentimeters(data)
    Case 2
        result = PointsToInches(data)
    Case 3
        result = PointsToLines(data)
    Case 4
        result = PointsToMillimeters(data)
    Case 5
        result = PointsToPicas(data)
    Case Else
        Error 5
End Select
```

PointsToInches Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPointsToInchesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPointsToInchesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPointsToInchesA"}

Converts a measurement from points to inches (1 inch = 72 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PointsToInches(*Points*)**

expression Optional. An expression that returns an **Application** object.

Points Required **Single**. The measurement, in points.

PointsToInches Method Example

This example converts the measurement of the top margin for the active document to inches and displays the result in a message box.

```
MsgBox PointsToInches(ActiveDocument.Sections(1).PageSetup.TopMargin)
```

This example converts the value of the variable `data` (a measurement in points) to centimeters, inches, lines, millimeters, or picas, depending on the value of the variable `unit` (a value from 1 through 5 that indicates the resulting unit of measurement).

```
Select Case unit
    Case 1
        result = PointsToCentimeters(data)
    Case 2
        result = PointsToInches(data)
    Case 3
        result = PointsToLines(data)
    Case 4
        result = PointsToMillimeters(data)
    Case 5
        result = PointsToPicas(data)
    Case Else
        Error 5
End Select
```

PointsToLines Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPointsToLinesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPointsToLinesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPointsToLinesA"}

Converts a measurement from points to lines (1 line = 12 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PointsToLines(*Points*)**

expression Optional. An expression that returns an **Application** object.

Points Required **Single**. The measurement, in points.

PointsToLines Method Example

This example converts the line spacing value of the first paragraph in the selection from points to lines.

```
MsgBox PointsToLines(Selection.Paragraphs(1).LineSpacing) & " lines"
```

This example converts the value of the variable `data` (a measurement in points) to centimeters, inches, lines, millimeters, or picas, depending on the value of the variable `unit` (a value from 1 through 5 that indicates the resulting unit of measurement).

```
Select Case unit
    Case 1
        result = PointsToCentimeters(data)
    Case 2
        result = PointsToInches(data)
    Case 3
        result = PointsToLines(data)
    Case 4
        result = PointsToMillimeters(data)
    Case 5
        result = PointsToPicas(data)
    Case Else
        Error 5
End Select
```

PointsToMillimeters Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPointsToMillimetersC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPointsToMillimetersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPointsToMillimetersA"}

Converts a measurement from points to millimeters (1 millimeter = 2.835 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PointsToMillimeters(*Points*)**

expression Optional. An expression that returns an **Application** object.

Points Required **Single**. The measurement, in points.

PointsToMillimeters Method Example

This example converts 72 points to the corresponding number of millimeters.

```
MsgBox PointsToMillimeters(72) & " millimeters"
```

This example converts the value of the variable `data` (a measurement in points) to centimeters, inches, lines, millimeters, or picas, depending on the value of the variable `unit` (a value from 1 through 5 that indicates the resulting unit of measurement).

```
Select Case unit
    Case 1
        result = PointsToCentimeters(data)
    Case 2
        result = PointsToInches(data)
    Case 3
        result = PointsToLines(data)
    Case 4
        result = PointsToMillimeters(data)
    Case 5
        result = PointsToPicas(data)
    Case Else
        Error 5
End Select
```

PointsToPicas Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPointsToPicasC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPointsToPicasX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPointsToPicasA"}

Converts a measurement from points to picas (1 pica = 12 points). Returns the converted measurement as a **Single**.

Syntax

expression.**PointsToPicas(*Points*)**

expression Optional. An expression that returns an **Application** object.

Points Required **Single**. The measurement, in points.

PointsToPicas Method Example

This example converts 36 points to the corresponding number of picas.

```
MsgBox PointsToPicas(36) & " picas"
```

This example converts the value of the variable `data` (a measurement in points) to centimeters, inches, lines, millimeters, or picas, depending on the value of the variable `unit` (a value from 1 through 5 that indicates the resulting unit of measurement).

```
Select Case unit
    Case 1
        result = PointsToCentimeters(data)
    Case 2
        result = PointsToInches(data)
    Case 3
        result = PointsToLines(data)
    Case 4
        result = PointsToMillimeters(data)
    Case 5
        result = PointsToPicas(data)
    Case Else
        Error 5
End Select
```

Run Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRunC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRunX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRunA"}

Runs a Visual Basic macro.

Syntax

expression.**Run(*MacroName*)**

expression Required. An expression that returns an **Application** object.

MacroName Required **String**. The name of the macro. Can be any combination of template, module, and macro name. For example, the following statements are all valid.

```
Application.Run "Normal.Module1.MAIN"
Application.Run "MyProject.MyModule.MyProcedure"
Application.Run "'My Document.doc'!ThisModule.ThisProcedure"
```

If you specify the document name, your code can only run macros in documents related to the current context – not just any macro in any document.

Remarks

Although Visual Basic code can call a macro directly (without this method being used), this method is useful when the macro name is stored in a variable (for more information, see the example for this topic). The following two statements are functionally equivalent:

```
Normal.Module2.Macro1
Application.Run MacroName:="Normal.Module2.Macro1"
```

You cannot pass parameters to a macro by using the **Run** method.

Run Method Example

This example prompts the user to enter a template name, module name, and macro name, and then it runs that macro.

```
t = InputBox("Enter the template name")
md = InputBox("Enter the module name")
m = InputBox("Enter the macro name")
Application.Run MacroName:=t & "." & md & "." & m
```

ShowVisualBasicEditor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowVisualBasicEditorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowVisualBasicEditorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowVisualBasicEditorA"}

True if the Visual Basic Editor window is visible. Read/write **Boolean**.

ShowVisualBasicEditor Property Example

This example makes the Visual Basic Editor window visible.

```
Application.ShowVisualBasicEditor = True
```

AutoCaptions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoCaptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoCaptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoCaptionsA "}

Returns an **AutoCaptions** collection that represents the captions that are automatically added when items such as tables and pictures are inserted into a document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

AutoCaptions Property Example

This example displays the name of each item that automatically gets a caption when inserted into the document.

```
For Each aCaption In AutoCaptions
    If aCaption.AutoInsert Then MsgBox aCaption.Name
Next aCaption
```

AutoInsert Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoInsertC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoInsertX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoInsertA "}

True if a caption is automatically added when the item is inserted into a document. Read/write **Boolean**.

AutoInsert Property Example

This example enables Word to add captions to tables automatically. Then the example collapses the selection to an insertion point, and inserts a table. A caption is automatically added to the new table.

```
AutoCaptions("Microsoft Word Table").AutoInsert = True
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.Tables.Add Range:=Selection.Range, NumRows:=2, _
NumColumns:=2
```

CancelAutoInsert Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCancelAutoInsertC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCancelAutoInsertX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCancelAutoInsertA "}

Prevents Word from automatically adding captions to any type of item.

Syntax

expression.**CancelAutoInsert**

expression Required. An expression that returns an **AutoCaptions** object.

CancelAutoInsert Method Example

This example prevents Word from automatically adding captions to any type of item.

```
AutoCaptions.CancelAutoInsert
```

CaptionLabels Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCaptionLabelsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCaptionLabelsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCaptionLabelsA "}

Returns a **CaptionLabels** collection that represents all the available caption labels. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

CaptionLabels Property Example

This example sets the numbering style for table captions.

```
CaptionLabels(wdCaptionTable).NumberStyle =
wdCaptionNumberStyleLowercaseRoman
```

This example adds a new caption label named "Photo" and then inserts a photo caption.

```
CaptionLabels.Add Name:="Photo"
With Selection
    .InsertParagraphAfter
    .InsertCaption Label:="Photo"
End With
```

CaptionLabel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCaptionLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCaptionLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCaptionLabelA "}

Returns or sets the caption label ("Figure," "Table," or "Equation," for exapmle) of the specified caption. Read/write **Variant**.

Note This property can be set to a string or a **WdCaptionLabelID** constant.

CaptionLabel Property Example

This example displays the name ("Microsoft Excel Worksheet," for example) and caption label ("Figure," for example) for each item that has a caption added automatically when inserted.

```
For Each myCaption In AutoCaptions
    If myCaption.AutoInsert = True Then MsgBox myCaption.Name & _
        vbCr & "Label = " & myCaption.CaptionLabel.Name
Next myCaption
```

This example sets the caption label for Word tables to "Table" and then inserts a new table immediately after the selection.

```
With AutoCaptions("Microsoft Word Table")
    .AutoInsert = True
    .CaptionLabel = wdCaptionTable
End With
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Tables.Add Range:=Selection.Range, NumRows:=2, _
    NumColumns:=3
```

ChapterStyleLevel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproChapterStyleLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproChapterStyleLevelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproChapterStyleLevelA "}

Returns or sets the heading style that marks a new chapter when chapter numbers are included with the specified caption label. The number 1 corresponds to Heading 1, 2 corresponds to Heading 2, and so on. Read/write **Long**.

Note The **IncludeChapterNumber** property must be set to **True** for chapter numbers to be included with caption labels.

ChapterStyleLevel Property Example

This example formats the table's caption label to include a chapter number. The chapter number is taken from paragraphs formatted with the Heading 2 style.

```
With CaptionLabels(wdCaptionTable)
    .IncludeChapterNumber = True
    .ChapterStyleLevel = 2
End With
```

IncludeChapterNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludeChapterNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludeChapterNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludeChapterNumberA "}

True if a chapter number is included with page numbers or a caption label. Read/write **Boolean**.

IncludeChapterNumber Property Example

This example adds page numbers in the footer for section one in the active document. The page numbers include the chapter number.

```
With ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary).PageNumbers
    .Add
    .IncludeChapterNumber = True
    .HeadingLevelForChapter = 1
End With
```

This example adds the chapter number from the Heading 2 style to figure captions, sets the caption numbering style, and then inserts a new figure caption. The document should already contain a Heading 2 style with numbering.

```
With CaptionLabels(wdCaptionFigure)
    .IncludeChapterNumber = True
    .ChapterStyleLevel = 2
    .NumberStyle = wdCaptionNumberStyleUppercaseLetter
End With
Selection.InsertCaption Label:="Figure", Title:=": History"
```

AutoCorrect Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoCorrectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoCorrectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoCorrectA "}

Returns an **AutoCorrect** object that contains the current AutoCorrect options, entries, and exceptions. Read-only.

AutoCorrect Property Example

This example adds an AutoCorrect replacement entry. After this code runs, every instance of "sr" that's typed in a document will automatically be replaced with "Stella Richards."

```
AutoCorrect.Entries.Add Name:= "sr", Value:= "Stella Richards"
```

This example deletes the specified AutoCorrect entry it if it exists.

```
myValue = InputBox("Enter the AutoCorrect entry to delete.")
For Each anEntry in AutoCorrect.Entries
    If anEntry.Name = myValue Then
        myMatch = True
        myConfirm = MsgBox("Are you sure you want to delete " & _
            anEntry.Name, 4)
        If myConfirm = vbYes Then
            anEntry.Delete
        End If
    End If
Next anEntry
If myMatch <> True Then
    MsgBox "There was no AutoCorrect entry: " & myValue
End If
```

CorrectDays Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCorrectDaysC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCorrectDaysX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCorrectDaysA "}

True if Word automatically capitalizes the first letter of days of the week. Read/write **Boolean**.

CorrectDays Property Example

This example sets Word to automatically capitalize the first letter of days of the week.

```
AutoCorrect.CorrectDays = True
```

This example toggles the value of the **CorrectDays** property.

```
AutoCorrect.CorrectDays = Not AutoCorrect.CorrectDays
```

CorrectInitialCaps Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCorrectInitialCapsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCorrectInitialCapsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCorrectInitialCapsA "}

True if Word automatically makes the second letter lowercase if the first two letters of a word are typed in uppercase. For example, "WOrd" is corrected to "Word." Read/write **Boolean**.

CorrectInitialCaps Property Example

This example sets Word to automatically correct errors in initial capitalization.

```
AutoCorrect.CorrectInitialCaps = True
```

CorrectSentenceCaps Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCorrectSentenceCapsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCorrectSentenceCapsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCorrectSentenceCapsA "}

True if Word automatically capitalizes the first letter in each sentence. Read/write **Boolean**.

CorrectSentenceCaps Property Example

This example toggles the value of the **CorrectSenetenceCaps** property.

```
AutoCorrect.CorrectSentenceCaps = Not AutoCorrect.CorrectSentenceCaps
```

Entries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEntriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEntriesA "}

Returns an **AutoCorrectEntries** collection that represents the current list of AutoCorrect entries. This list corresponds to the list of AutoCorrect entries on the **AutoCorrect** tab in the **AutoCorrect** dialog box (**Tools** menu). Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Entries Property Example

This example displays the total number of AutoCorrect entries.

```
MsgBox AutoCorrect.Entries.Count
```

This example deletes the specified AutoCorrect entry if it exists.

```
myValue = InputBox("Enter the AutoCorrect entry to delete.")
For Each anEntry in AutoCorrect.Entries
    If anEntry.Name = myValue Then
        myMatch = True
        myConfirm = MsgBox("Are you sure you want to delete " & _
            anEntry.Name, 4)
        If myConfirm = vbYes Then
            anEntry.Delete
        End If
    End If
Next anEntry
If myMatch <> True Then
    MsgBox "There was no AutoCorrect entry: " & myValue
End If
```

ReplaceText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReplaceTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReplaceTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReplaceTextA "}

True if Word automatically replaces specified text with entries from the AutoCorrect list. Read/write **Boolean**.

ReplaceText Property Example

This example sets Word to automatically replace specified text with entries from the AutoCorrect list as you type.

```
AutoCorrect.ReplaceText = True
```

This example toggles the value of the **ReplaceText** property.

```
AutoCorrect.ReplaceText = Not AutoCorrect.ReplaceText
```

AddRichText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddRichTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddRichTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddRichTextA "}

Creates a formatted AutoCorrect entry, preserving all text attributes of the specified range. Returns an **AutoCorrectEntry** object. The **RichText** property for entries added by using this method returns **True**. If **AddRichText** isn't used, inserted **AutoCorrect** entries conform to the current style.

Syntax

expression.**AddRichText(*Name, Range*)**

expression Required. An expression that returns an **AutoCorrectEntries** object.

Name Required **String**. The text to replace automatically with ***Range***.

Range Required **Range** object. The formatted text that Word will insert automatically whenever ***Name*** is typed.

AddRichText Method Example

This example stores the selected text as a formatted AutoCorrect entry that will be inserted automatically whenever "NewText" is typed.

```
If Selection.Type = wdSelectionNormal Then 'checks for text selected
    AutoCorrect.Entries.AddRichText "NewText", Selection.Range
Else
    MsgBox "You need to select some text."
End If
```

This example stores the third word in the active document as a formatted AutoCorrect entry that will be inserted automatically whenever "NewText" is typed.

```
AutoCorrect.Entries.AddRichText "NewText", ActiveDocument.Words(3)
```

CorrectCapsLock Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCorrectCapsLockC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCorrectCapsLockX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCorrectCapsLockA "}

Windows only. **True** if Word automatically corrects instances in which you use the CAPS LOCK key inadvertently as you type. Read/write **Boolean**.

CorrectCapsLock Property Example

This example determines whether Word is set to automatically correct CAPS LOCK key errors.

```
If AutoCorrect.CorrectCapsLock = True Then
    MsgBox "Correct CAPS LOCK is active."
Else
    MsgBox "Correct CAPS LOCK is not active."
End If
```

FirstLetterAutoAdd Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstLetterAutoAddC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstLetterAutoAddX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstLetterAutoAddA "}

True if Word automatically adds abbreviations to the list of AutoCorrect First Letter exceptions. Word adds an abbreviation to this list if you delete and then retype the letter that Word capitalized immediately after the period following the abbreviation. Read/write **Boolean**.

FirstLetterAutoAdd Property Example

This example prevents Word from automatically adding abbreviations to the list of AutoCorrect First Letter exceptions.

```
AutoCorrect.FirstLetterAutoAdd = False
```

FirstLetterExceptions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstLetterExceptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstLetterExceptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstLetterExceptionsA "}

Returns a **FirstLetterExceptions** collection that represents the list of abbreviations after which Word won't automatically capitalize the next letter. This list corresponds to the list of AutoCorrect exceptions on the **First Letter** tab in the **AutoCorrect Exceptions** dialog box (**AutoCorrect** command, **Tools** menu).Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

FirstLetterExceptions Property Example

This example adds "apt." to the list of AutoCorrect First Letter exceptions.

```
AutoCorrect.FirstLetterExceptions.Add "apt."
```

This example deletes the specified AutoCorrect First Letter exception if it exists.

```
myValue = InputBox("Enter the First Letter exception to delete.")
For Each anEntry in AutoCorrect.FirstLetterExceptions
    If anEntry.Name = myValue Then
        myMatch = True
        myConfirm = MsgBox("Are you sure you want to delete " & _
            anEntry.Name, 4)
        If myConfirm = vbYes Then
            anEntry.Delete
        End If
    End If
Next anEntry
If myMatch <> True Then
    MsgBox "There was no First Letter exception: " & myValue
End If
```

RichText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRichTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRichTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRichTextA "}

True if formatting is stored with the AutoCorrect entry replacement text. Read-only **Boolean**.

RichText Property Example

This example determines whether AutoCorrect entry one is formatted.

```
MsgBox AutoCorrect.Entries(1).RichText
```

TwoInitialCapsAutoAdd Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTwoInitialCapsAutoAddC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTwoInitialCapsAutoAddX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTwoInitialCapsAutoAddA "}

True if Word automatically adds words to the list of AutoCorrect Initial Caps exceptions. A word is added to this list if you delete and then retype the uppercase letter (following the initial uppercase letter) that Word changed to lowercase. Read/write **Boolean**.

TwoInitialCapsAutoAdd Property Example

This example sets Word to automatically add words to the list of AutoCorrect Initial Caps exceptions.

```
AutoCorrect.TwoInitialCapsAutoAdd = True
```

Bookmark Property

This item is not available for this version of Office

Bookmark Property Example

This item is not available for this version of Office

Bookmarks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBookmarksC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBookmarksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBookmarksA "}

Returns a **Bookmarks** collection that represents all the bookmarks in a document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Bookmarks Property Example

This example retrieves the starting and ending character positions for first bookmark in the active document.

```
With ActiveDocument.Bookmarks(1)
    BookStart = .Start
    BookEnd = .End
End With
```

This example uses the `aMarks()` array to store the name of each bookmark contained in the active document.

```
If ActiveDocument.Bookmarks.Count >= 1 Then
    ReDim aMarks(ActiveDocument.Bookmarks.Count - 1)
    i = 0
    For Each aBookmark In ActiveDocument.Bookmarks
        aMarks(i) = aBookmark.Name
        i = i + 1
    Next aBookmark
End If
```

This example applies bold formatting to the first range of bookmarked text in the selection.

```
If Selection.Bookmarks.Count >= 1 Then
    Selection.Bookmarks(1).Range.Bold = True
End If
```

DefaultSorting Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultSortingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultSortingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultSortingA "}

Returns or sets the sorting option for bookmark names displayed in the **Bookmark** dialog box (**Insert** menu). Can be one of the following **WdBookmarkSortBy** constants: **wdSortByLocation** or **wdSortByName**. Read/write **Long**.

Remarks

This property doesn't affect the order of **Bookmark** objects in the **Bookmarks** collection.

DefaultSorting Property Example

This example sorts bookmarks by location and then displays the **Bookmark** dialog box.

```
ActiveDocument.Bookmarks.DefaultSorting = wdSortByLocation
Dialogs(wdDialogInsertBookmark).Show
```

Empty Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEmptyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEmptyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEmptyA "}

True if the specified bookmark is empty. An empty bookmark marks a location (a collapsed selection); it doesn't mark any text. Read-only **Boolean**.

Note An error occurs if the specified bookmark doesn't exist. Use the **Exists** property to determine whether the bookmark exists.

Empty Property Example

This example determines whether the bookmark named "temp" exists and whether it is empty.

```
If ActiveDocument.Bookmarks.Exists("temp") = True Then
    If ActiveDocument.Bookmarks("temp").Empty = True Then _
    MsgBox "The Temp bookmark is empty"
End If
```

ShowHidden Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowHiddenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowHiddenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowHiddenA "}

True if hidden bookmarks are included in the **Bookmarks** collection. This property also controls whether hidden bookmarks are listed in the **Bookmark** dialog box (**Insert** menu). Read/write **Boolean**.

Remarks

Hidden bookmarks are automatically inserted when cross-references are inserted into the document.

ShowHidden Property Example

This example displays the **Bookmark** dialog box with both visible and hidden bookmarks listed.

```
ActiveDocument.Bookmarks.ShowHidden = True
Dialogs(wdDialogInsertBookmark).Show
```

This example displays the name of each hidden bookmark in the document. Hidden bookmarks in a Word document begin with an underscore (_).

```
ActiveDocument.Bookmarks.ShowHidden = True
For Each aBookmark In ActiveDocument.Bookmarks
    If Left(aBookmark.Name, 1) = "_" Then MsgBox aBookmark.Name
Next aBookmark
```

BackgroundPatternColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackgroundPatternColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackgroundPatternColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackgroundPatternColorIndexA "}

Returns or sets the color that's applied to the background of the **Shading** object. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto	**wdGreen**
wdBlack	**wdNoHighlight**
wdBlue	**wdPink**
wdBrightGreen	**wdRed**
wdDarkBlue	**wdTeal**
wdDarkRed	**wdTurquoise**
wdDarkYellow	**wdViolet**
wdGray25	**wdWhite**
wdGray50	**wdYellow**

BackgroundPatternColorIndex Property Example

This example applies cyan background shading to the first paragraph in the active document.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.Shading.BackgroundPatternColorIndex = wdTurquoise
```

This example adds a table at the insertion point and then applies light gray background shading to the first cell.

```
Selection.Collapse Direction:=wdCollapseStart
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=2, NumColumns:=2)
myTable.Cell(1, 1).Shading.BackgroundPatternColorIndex = wdGray25
```

DistanceFromText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromTextA "}

DropCap object: Returns or sets the distance (in points) between the dropped capital letter and the paragraph text. Read/write **Single**.

LineNumbering object: Returns or sets the distance (in points) between the right edge of line numbers and the left edge of the document text. Read/write **Single**.

DistanceFromText Property Example

This example adds line numbers to the active document. The distance between the line numbers and the left margin is 36 points (0.5 inch).

```
With ActiveDocument.PageSetup.LineNumbering
    .Active = True
    .CountBy = 5
    .DistanceFromText = 36
End With
```

This example sets a dropped capital letter for the first paragraph in the active document. The offset for the dropped capital letter is then set to 12 points.

```
With ActiveDocument.Paragraphs(1).DropCap
    .Enable
    .FontName= "Arial"
    .Position = wdDropNormal
    .DistanceFromText = 12
End With
```

Enable Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnableA "}

Returns or sets border formatting for the specified object. Returns **True** or **wdUndefined** if border formatting is applied to all or part of the specified object. Can be set to **True**, **False**, or a **WdLineStyle** constant. Read/write **Long**.

Remarks

The **Enable** property applies to all borders for the specified object. **True** sets the line style to the default line style and sets the line width to the default line width. The default line style and line width can be set using the **DefaultBorderLineWidth** and **DefaultBorderLineStyle** properties.

To remove all the borders from an object, set the **Enable** property to **False**, as shown in the following example.

```
ActiveDocument.Tables(1).Borders.Enable = False
```

To remove or apply a single border, use **Borders**(*index*), where *index* is a **WdBorderType** constant, to return a single border, and then set the **LineStyle** property. The following example removes the bottom border from `myRange`.

```
myRange.Borders(wdBorderBottom).LineStyle = wdLineStyleNone
```

Enable Property Example

This example removes all borders from the first cell in table one.

```
If ActiveDocument.Tables.Count >= 1 Then
    ActiveDocument.Tables(1).Cell(1, 1).Borders.Enable = False
End If
```

This example applies a dashed border around the first paragraph in the selection.

```
Options.DefaultBorderLineWidth = wdLineWidth025pt
Selection.Paragraphs(1).Borders.Enable = wdLineStyleDashSmallGap
```

This example applies a border around the first character in the selection. If nothing is selected, the border is applied to the first character after the insertion point.

```
Selection.Characters(1).Borders.Enable = True
```

ForegroundPatternColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproForegroundPatternColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproForegroundPatternColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproForegroundPatternColorIndexA "}

Returns or sets the color that's applied to the foreground of the **Shading** object. This color is applied to the dots and lines in the shading pattern. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto	**wdGreen**
wdBlack	**wdNoHighlight**
wdBlue	**wdPink**
wdBrightGreen	**wdRed**
wdDarkBlue	**wdTeal**
wdDarkRed	**wdTurquoise**
wdDarkYellow	**wdViolet**
wdGray25	**wdWhite**
wdGray50	**wdYellow**

ForegroundPatternColorIndex Property Example

This example applies shading with different foreground and background colors to the selection.

```
With Selection.Shading
    .Texture = wdTexture30Percent
    .ForegroundPatternColorIndex = wdBlue
    .BackgroundPatternColorIndex = wdYellow
End With
```

InsideLineStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsideLineStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsideLineStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsideLineStyleA "}

Returns or sets the inside border for the specified object. Returns **wdUndefined** if more than one kind of border is applied to the specified object; otherwise, returns **False** or a **WdLineStyle** constant. Can be set to **True**, **False**, or a **WdLineStyle** constant. Read/write **Long**.

Remarks

True sets the line style to the default line style and sets the line width to the default line width. The default line style and line width can be set using the **DefaultBorderLineWidth** and **DefaultBorderLineStyle** properties.

Use either of the following instructions to remove the inside border from the first table in the active document.

```
ActiveDocument.Tables(1).Borders.InsideLineStyle = wdLineStyleNone
ActiveDocument.Tables(1).Borders.InsideLineStyle = False
```

InsideLineStyle Property Example

This example adds borders between rows and between columns in the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    myTable.Borders.InsideLineStyle = True
End If
```

This example adds borders between the first four paragraphs in the document.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(1).Range.Start, _
    End:=myDoc.Paragraphs(4).Range.End)
With myRange.Borders
    .InsideLineStyle = wdLineStyleSingle
    .InsideLineWidth = wdLineWidth150pt
End With
```

LineStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineStyleA "}

Returns or sets the border line style for the specified object. Read/write **Long**.

Can be one of the following **WdLineStyle** constants:

wdLineStyleNone
wdLineStyleSingle
wdLineStyleDot
wdLineStyleDashSmallGap
wdLineStyleDashLargeGap
wdLineStyleDashDot
wdLineStyleDashDotDot
wdLineStyleDouble
wdLineStyleTriple
wdLineStyleThinThickSmallGap
wdLineStyleThickThinSmallGap
wdLineStyleThinThickThinSmallGap
wdLineStyleThinThickMedGap
wdLineStyleThickThinMedGap
wdLineStyleThinThickThinMedGap
wdLineStyleThinThickLargeGap
wdLineStyleThickThinLargeGap
wdLineStyleThinThickThinLargeGap
wdLineStyleSingleWavy
wdLineStyleDoubleWavy
wdLineStyleDashDotStroked
wdLineStyleEmboss3D
wdLineStyleEngrave3D

Remarks

Setting the **LineStyle** property for a range that refers to individual characters or words applies a character border.

Setting the **LineStyle** property for a paragraph or range of paragraphs applies a paragraph border. Use the **InsideLineStyle** property to apply a border between consecutive paragraphs.

Setting the **LineStyle** property for a section applies a page border around the pages in the section.

LineStyle Property Example

If the selection is a paragraph or a collapsed selection, this example adds a single 0.75-point paragraph border above the selection. If the selection doesn't include a paragraph, a border is applied around the selected text.

```
With Selection.Borders(wdBorderTop)
    .LineStyle = wdLineStyleSingle
    .LineWidth = wdLineWidth075pt
End With
```

This example adds a double 1.5-point border below each frame in the active document.

```
For Each aFrame In ActiveDocument.Frames
    With aFrame.Borders(wdBorderBottom)
        .LineStyle = wdLineStyleDouble
        .LineWidth = wdLineWidth150pt
    End With
Next aFrame
```

The following example applies a border around the fourth word in the active document. Applying a single border (in this example, a top border) to text applies a border around the text.

```
ActiveDocument.Words(4).Borders(wdBorderTop).LineStyle = wdLineStyleSingle
```

OutsideLineStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutsideLineStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutsideLineStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutsideLineStyleA "}

Returns or sets the outside border for the specified object. Returns **wdUndefined** if more than one kind of border is applied to the specified object; otherwise, returns **False** or a **WdLineStyle** constant. Can be set to **True**, **False**, or a **WdLineStyle** constant. Read/write **Long**.

Remarks

True sets the line style to the default line style and sets the line width to the default line width. The default line style and line width can be set using the **DefaultBorderLineWidth** and **DefaultBorderLineStyle** properties.

Use either of the following instructions to remove the outside border from the first table in the active document.

```
ActiveDocument.Tables(1).Borders.OutsideLineStyle = wdLineStyleNone
ActiveDocument.Tables(1).Borders.OutsideLineStyle = False
```

OutsideLineStyle Property Example

This example adds a double 0.75-point border around the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).Borders
    .OutsideLineStyle = wdLineStyleDouble
    .OutsideLineWidth = wdLineWidth075pt
End With
```

This example adds a border around the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    myTable.Borders.OutsideLineStyle = wdLineStyleSingle
End If
```

Shading Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShadingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShadingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShadingA "}

Returns a **Shading** object that refers to the shading formatting for the specified object. Read-only.

Shading Property Example

This example applies yellow shading to the first paragraph in the selection.

```
With Selection.Paragraphs(1).Shading
    .Texture = wdTexture12Pt5Percent
    .BackgroundPatternColorIndex = wdYellow
    .ForegroundPatternColorIndex = wdBlack
End With
```

This example applies horizontal line texture to the first row in table one.

```
If ActiveDocument.Tables.Count >= 1 Then
    With ActiveDocument.Tables(1).Rows(1).Shading
        .Texture = wdTextureHorizontal
    End With
End If
```

This example applies 10 percent shading to the first word in the active document.

```
ActiveDocument.Words(1).Shading.Texture = wdTexture10Percent
```

Texture Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextureC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextureX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextureA "}

Returns or sets the shading texture for the specified object. Read/write **Long**.

Can be one of the following **WdTextureIndex** constants:

wdTextureAuto
wdTextureSolid
wdTexture2Pt5Percent
wdTexture5Percent
wdTexture7Pt5Percent
wdTexture10Percent
wdTexture12Pt5Percent
wdTexture15Percent
wdTexture17Pt5Percent
wdTexture20Percent
wdTexture22Pt5Percent
wdTexture25Percent
wdTexture27Pt5Percent
wdTexture30Percent
wdTexture32Pt5Percent
wdTexture35Percent
wdTexture37Pt5Percent
wdTexture40Percent
wdTexture42Pt5Percent
wdTexture45Percent
wdTexture47Pt5Percent
wdTexture50Percent
wdTexture52Pt5Percent
wdTexture55Percent
wdTexture57Pt5Percent
wdTexture60Percent
wdTexture62Pt5Percent
wdTexture65Percent
wdTexture67Pt5Percent
wdTexture70Percent
wdTexture72Pt5Percent
wdTexture75Percent
wdTexture77Pt5Percent
wdTexture80Percent
wdTexture82Pt5Percent
wdTexture85Percent
wdTexture87Pt5Percent
wdTexture90Percent
wdTexture92Pt5Percent
wdTexture95Percent
wdTexture97Pt5Percent
wdTextureDarkHorizontal
wdTextureDarkVertical
wdTextureDarkCross
wdTextureDarkDiagonalCross
wdTextureDarkDiagonalDown
wdTextureDarkDiagonalUp
wdTextureHorizontal
wdTextureVertical
wdTextureCross
wdTextureDiagonalCross
wdTextureDiagonalDown
wdTextureDiagonalUp
wdTextureNone
wdTextureSolid

Texture Property Example

This example sets a range that references the first paragraph in the active document and then applies a grid texture to that range.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.Shading.Texture = wdTextureCross
```

This example adds a table at the insertion point and then applies a vertical line texture to the first row in the table.

```
Selection.Collapse Direction:=wdCollapseStart
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=2, NumColumns:=2)
myTable.Rows(1).Shading.Texture = wdTextureVertical
```

This example applies 10 percent shading to the first word in the active document.

```
ActiveDocument.Words(1).Shading.Texture = wdTexture10Percent
```

DefaultBorderLineWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultBorderLineWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultBorderLineWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultBorderLineWidthA"}

Returns or sets the default line width of borders. Can be one of the following **WdLineWidth** constants: **wdLineWidth025pt**, **wdLineWidth050pt**, **wdLineWidth075pt**, **wdLineWidth100pt**, **wdLineWidth150pt**, **wdLineWidth225pt**, **wdLineWidth300pt**, **wdLineWidth450pt**, or **wdLineWidth600pt**. Read/write **Long**.

Note If the **Enable** property of the **Borders** object is set to **True**, the default line width and line style of borders are used.

DefaultBorderLineWidth Property Example

This example changes the default line width of borders and then adds a border around each paragraph in the selection.

```
Options.DefaultBorderLineWidth = wdLineWidth050pt
Selection.Borders.Enable = True
```

Inside Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsideC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsideX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsideA"}

True if an inside border can be applied to the specified object. Read-only **Boolean**.

Inside Property Example

If the current selection supports inside borders (that is, if multiple paragraphs or cells are selected), this example applies a single inside border.

```
For Each aBorder In Selection.Borders
    If aBorder.Inside = True Then aBorder.LineStyle = wdLineStyleSingle
Next aBorder
```

InsideLineWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsideLineWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsideLineWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsideLineWidthA"}

Returns or sets the line width of the inside border of an object. Returns **wdUndefined** if the object has inside borders with more than one line width; otherwise, returns **False** or a **WdLineWidth** constant. Can be set to **True**, **False**, or one of the following **WdLineWidth** constants: **wdLineWidth025pt**, **wdLineWidth050pt**, **wdLineWidth075pt**, **wdLineWidth100pt**, **wdLineWidth150pt**, **wdLineWidth225pt**, **wdLineWidth300pt**, **wdLineWidth450pt**, or **wdLineWidth600pt**. Read/write **Long**.

InsideLineWidth Property Example

This example adds borders between rows and between columns in the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    myTable.Borders.InsideLineStyle = wdLineStyleDot
    myTable.Borders.InsideLineWidth = wdLineWidth050pt
End If
```

This example adds dotted borders between the first four paragraphs in the active document.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(1).Range.Start, _
    End:=myDoc.Paragraphs(4).Range.End)
myRange.Borders.InsideLineStyle = wdLineStyleDot
myRange.Borders.InsideLineWidth = wdLineWidth075pt
```

OutsideLineWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutsideLineWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutsideLineWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutsideLineWidthA"}

Returns or sets the line width of the outside border of an object. Returns **wdUndefined** if the object has outside borders with more than one line width; otherwise, returns **False** or a **WdLineWidth** constant. Can be set to **True**, **False**, or one of the following **WdLineWidth** constants: **wdLineWidth025pt**, **wdLineWidth050pt**, **wdLineWidth075pt**, **wdLineWidth100pt**, **wdLineWidth150pt**, **wdLineWidth225pt**, **wdLineWidth300pt**, **wdLineWidth450pt**, or **wdLineWidth600pt**. Read/write **Long**.

OutsideLineWidth Property Example

This example adds a wavy border around the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    With ActiveDocument.Tables(1).Borders
        .OutsideLineStyle = wdLineStyleSingleWavy
        .OutSideLineWidth = wdLineWidth050pt
    End With
End If
```

This example adds dotted borders around the first four paragraphs in the active document.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(1).Range.Start, _
    End:=myDoc.Paragraphs(4).Range.End)
myRange.Borders.OutsideLineStyle = wdLineStyleDot
myRange.Borders.OutsideLineWidth = wdLineWidth075pt
```

Enable Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEnableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEnableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEnableA "}

Formats the first character in the specified paragraph as a dropped capital letter.

Syntax

expression.**Enable**

expression Required. An expression that returns a **DropCap** object.

Enable Method Example

This example formats the first paragraph in the selection to begin with a dropped capital letter.

```
With Selection.Paragraphs(1).DropCap
    .Enable
    .LinesToDrop = 2
    .FontName = "Arial"
End With
```

LineWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineWidthA"}

Returns or sets the line width of the border of an object. Returns a **WdLineWidth** constant or **wdUndefined** if the object either has no borders or has borders with more than one line width. Can be set to **True**, **False**, or one of the following **WdLineWidth** constants: **wdLineWidth025pt**, **wdLineWidth050pt**, **wdLineWidth075pt**, **wdLineWidth100pt**, **wdLineWidth150pt**, **wdLineWidth225pt**, **wdLineWidth300pt**, **wdLineWidth450pt**, or **wdLineWidth600pt**. Read/write **Long**.

Note If the specified line width isn't available for the border's line style, this property generates an error. To determine the line widths available for a particular line style, see the **Borders and Shading** dialog box (**Format** menu).

LineWidth Property Example

This example adds a borders below the first row in table one in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    With ActiveDocument.Tables(1).Rows(1).Borders(wdBorderBottom)
        .LineStyle = wdLineStyleSingle
        .LineWidth = wdLineWidth050pt
    End With
End If
```

This example adds a wavy, red line to the left of the selection.

```
With Selection.Borders(wdBorderLeft)
    .LineStyle = wdLineStyleSingleWavy
    .LineWidth = wdLineWidth075pt
    .ColorIndex = wdRed
End With
```

Borders Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBordersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBordersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBordersA "}

Returns a **Borders** collection that represents all the borders for the specified object. Read/write.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Borders Property Example

This example applies inside and outside borders to the first table in the active document.

```
Set myTable = ActiveDocument.Tables(1)
With myTable.Borders
    .InsideLineStyle = wdLineStyleSingle
    .OutsideLineStyle = wdLineStyleDouble
End With
```

This example applies a border around the first character in the selection. If nothing is selected, the border is applied to the first character after the insertion point.

```
Selection.Characters(1).Borders.Enable = True
```

This example applies a bottom border below all centered paragraphs in the active document.

```
For Each para In ActiveDocument.Paragraphs
    If para.Alignment = wdAlignParagraphCenter Then
        para.Borders(wdBorderBottom).LineStyle = wdLineStyleSingle
        para.Borders(wdBorderBottom).LineWidth = wdLineWidth300pt
    End If
Next para
```

This example adds a border around all of the pages in the current section.

```
For Each aBorder In Selection.Sections(1).Borders
    aBorder.ArtStyle = wdArtBasicBlackDots
    aBorder.ArtWidth = 6
Next aBorder
```

AlwaysInFront Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAlwaysInFrontC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAlwaysInFrontX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAlwaysInFrontA"}

True if page borders are displayed in front of the document text. Read/write **Boolean**.

AlwaysInFront Property Example

This example adds a graphical page border in front of text in the first section in the active document.

```
With ActiveDocument.Sections(1)
    .Borders.AlwaysInFront = True
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtPeople
        aBord.ArtWidth = 15
    Next aBord
End With
```

ApplyPageBordersToAllSections Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyPageBordersToAllSectionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyPageBordersToAllSectionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyPageBordersToAllSectionsA"}

Applies the specified page-border formatting to all sections in a document.

Syntax

expression.**ApplyPageBordersToAllSections**

expression Required. An expression that returns a **Borders** object.

ApplyPageBordersToAllSections Method Example

This example adds a single-line page border to all sections in the active document.

```
With ActiveDocument.Sections(1)
    For Each aBord In .Borders
        aBord.LineStyle = wdLineStyleSingle
        aBord.LineWidth = wdLineWidth050pt
    Next aBord
    .Borders.ApplyPageBordersToAllSections
End With
```

ArtStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproArtStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproArtStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproArtStyleA"}

Returns or sets the graphical page-border design for a document. Read/write **Long**.

Can be one of the following **WdPageBorderArt** constants:

wdArtApples
wdArtArchedScallops
wdArtBabyPacifier
wdArtBabyRattle
wdArtBalloons3Colors
wdArtBalloonsHotAir
wdArtBasicBlackDashes
wdArtBasicBlackDots
wdArtBasicBlackSquares
wdArtBasicThinLines
wdArtBasicWhiteDashes
wdArtBasicWhiteDots
wdArtBasicWhiteSquares
wdArtBasicWideInline
wdArtBasicWideMidline
wdArtBasicWideOutline
wdArtBats
wdArtBirds
wdArtBirdsFlight
wdArtCabins
wdArtCakeSlice
wdArtCandyCorn
wdArtCelticKnotwork
wdArtCertificateBanner
wdArtChainLink
wdArtChampagneBottle
wdArtCheckedBarBlack
wdArtCheckedBarColor
wdArtCheckered
wdArtChristmasTree
wdArtCirclesLines
wdArtCirclesRectangles
wdArtClassicalWave
wdArtClocks
wdArtCompass
wdArtConfetti
wdArtConfettiGrays
wdArtConfettiOutline
wdArtConfettiStreamers
wdArtHolly
wdArtHouseFunky
wdArtHypnotic
wdArtIceCreamCones
wdArtLightBulb
wdArtLightning1
wdArtLightning2
wdArtMapleLeaf
wdArtMapleMuffins
wdArtMapPins
wdArtMarquee
wdArtMarqueeToothed
wdArtMoons
wdArtMosaic
wdArtMusicNotes
wdArtNorthwest
wdArtOvals
wdArtPackages
wdArtPalmsBlack
wdArtPalmsColor
wdArtPaperClips
wdArtPapyrus
wdArtPartyFavor
wdArtPartyGlass
wdArtPencils
wdArtPeople
wdArtPeopleHats
wdArtPeopleWaving
wdArtPoinsettias
wdArtPostageStamp
wdArtPumpkin1
wdArtPushPinNote1
wdArtPushPinNote2
wdArtPyramids
wdArtPyramidsAbove
wdArtQuadrants
wdArtRings
wdArtSafari
wdArtSawtooth

wdArtConfettiWhite
wdArtCornerTriangles
wdArtCouponCutoutDashes
wdArtCouponCutoutDots
wdArtCrazyMaze
wdArtCreaturesButterfly
wdArtCreaturesFish
wdArtCreaturesInsects
wdArtCreaturesLadyBug
wdArtCrossStitch
wdArtCup
wdArtDecoArch
wdArtDecoArchColor
wdArtDecoBlocks
wdArtDiamondsGray
wdArtDoubleD
wdArtDoubleDiamonds
wdArtEarth1
wdArtEarth2
wdArtEclipsingSquares1
wdArtEclipsingSquares2
wdArtEggsBlack
wdArtFans
wdArtFilm
wdArtFirecrackers
wdArtFlowersBlockPrint
wdArtFlowersDaisies
wdArtFlowersModern1
wdArtFlowersModern2
wdArtFlowersPansy
wdArtFlowersRedRose
wdArtFlowersRoses
wdArtFlowersTeacup
wdArtFlowersTiny
wdArtGems
wdArtGingerbreadMan
wdArtGradient
wdArtHandmade1
wdArtHandmade2
wdArtHeartBalloon
wdArtHeartGray
wdArtHearts
wdArtHeebieJeebies
wdArtSawtoothGray
wdArtScaredCat
wdArtSeattle
wdArtShadowedSquares
wdArtSharksTeeth
wdArtShorebirdTracks
wdArtSkyrocket
wdArtSnowflakeFancy
wdArtSnowflakes
wdArtSombrero
wdArtSouthwest
wdArtStars
wdArtStars3D
wdArtStarsBlack
wdArtStarsShadowed
wdArtStarsTop
wdArtSun
wdArtSwirligig
wdArtTornPaper
wdArtTornPaperBlack
wdArtTrees
wdArtTriangleParty
wdArtTriangles
wdArtTribal1
wdArtTribal2
wdArtTribal3
wdArtTribal4
wdArtTribal5
wdArtTribal6
wdArtTwistedLines1
wdArtTwistedLines2
wdArtVine
wdArtWaveline
wdArtWeavingAngles
wdArtWeavingBraid
wdArtWeavingRibbon
wdArtWeavingStrips
wdArtWhiteFlowers
wdArtWoodwork
wdArtXIllusions
wdArtZanyTriangles
wdArtZigZag
wdArtZigZagStitch

ArtStyle Property Example

This example adds a border of black dots around each page in first section in the selection.

```
For Each aBorder In Selection.Sections(1).Borders
    aBorder.ArtStyle = wdArtBasicBlackDots
    aBorder.ArtWidth = 6
Next aBorder
```

This example adds a picture border around each page in section one in the active document.

```
With ActiveDocument.Sections(1)
    .Borders.AlwaysInFront = True
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtPeople
        aBord.ArtWidth = 15
    Next aBord
End With
```

ArtWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproArtWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproArtWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproArtWidthA"}

Returns or sets the width (in points) of the specified graphical page border. Read/write **Long**.

ArtWidth Property Example

This example adds a 6-point dotted border around each page in the first section in the selection.

```
For Each aBorder In Selection.Sections(1).Borders
    aBorder.ArtStyle = wdArtBasicBlackDots
    aBorder.ArtWidth = 6
Next aBorder
```

DistanceFrom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromA"}

Returns or sets a value that indicates whether the specified page border is measured from the edge of the page or from the text it surrounds. Can be either of the following **WdBorderDistanceFrom** constants: **wdBorderDistanceFromPageEdge** or **wdBorderDistanceFromText**. Read/write **Long**.

DistanceFrom Property Example

This example adds a single border around each page in section one in the active document and then sets the distance between each border and the corresponding edge of the page.

```
With ActiveDocument.Sections(1)
    For Each aBord In .Borders
        aBord.LineStyle = wdLineStyleSingle
        aBord.LineWidth = wdLineWidth050pt
    Next aBord
    With .Borders
        .DistanceFrom = wdBorderDistanceFromPageEdge
        .DistanceFromTop = 20
        .DistanceFromLeft = 22
        .DistanceFromBottom = 20
        .DistanceFromRight = 22
    End With
End With
```

This example adds a border around each page in the first section in the selection, and then it sets the distance between the text and the page border to 6 points.

```
With Selection.Sections(1)
    For Each aBorder In .Borders
        aBorder.ArtStyle = wdArtSeattle
        aBorder.ArtWidth = 22
    Next aBorder
    With .Borders
        .DistanceFrom = wdBorderDistanceFromText
        .DistanceFromTop = 6
        .DistanceFromLeft = 6
        .DistanceFromBottom = 6
        .DistanceFromRight = 6
    End With
End With
```

EnableFirstPageInSection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnableFirstPageInSectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnableFirstPageInSectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnableFirstPageInSectionA"}

True if page borders are enabled for the first page in the section. Read/write **Boolean**.

EnableFirstPageInSection Property Example

This example adds a border around the first page in the first section in the selection.

```
With Selection.Sections(1)
    .Borders.EnableFirstPageInSection = True
    .Borders.EnableOtherPagesInSection = False
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtPeople
        aBord.ArtWidth = 15
    Next aBord
End With
```

EnableOtherPagesInSection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnableOtherPagesInSectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnableOtherPagesInSectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnableOtherPagesInSectionA"}

True if page borders are enabled for all pages in the section except for the first page. Read/write **Boolean**.

EnableOtherPagesInSection Property Example

This example adds a border around each page in the first section in the selection except for the first page.

```
With Selection.Sections(1)
    .Borders.EnableFirstPageInSection = False
    .Borders.EnableOtherPagesInSection = True
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtBabyRattle
        aBord.ArtWidth = 22
    Next aBord
End With
```

HasHorizontal Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasHorizontalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasHorizontalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasHorizontalA"}

True if a horizontal border can be applied to the object. Read-only **Boolean**.

Remarks

Horizontal borders can be applied to ranges that contain cells in two or more rows of a table or ranges that contain two or more paragraphs.

HasHorizontal Property Example

This example applies single-line horizontal borders, if the selection supports horizontal borders.

```
If Selection.Borders.HasHorizontal = True Then
    Selection.Borders(wdBorderHorizontal).LineStyle = wdLineStyleSingle
End If
```

HasVertical Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasVerticalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasVerticalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasVerticalA"}

True if a vertical border can be applied to the specified object. Read-only **Boolean**.

Remarks

Vertical borders can be applied to ranges that contain cells in two or more columns of a table.

HasVertical Property Example

If the selection supports vertical borders, this example applies a single vertical border.

```
If Selection.Borders.HasVertical = True Then
    Selection.Borders(wdBorderVertical).LineStyle = wdLineStyleSingle
End If
```

JoinBorders Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproJoinBordersC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproJoinBordersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproJoinBordersA"}

True if vertical borders at the edges of paragraphs and tables are removed so that the horizontal borders can connect to the page border. Read/write **Boolean**.

JoinBorders Property Example

This example add a border around each page in the first section in the selection. The example also removes the horizontal borders at the edges of tables and paragraphs, thus connecting the horizontal borders to the page border.

```
With Selection.Sections(1)
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtBalloonsHotAir
        aBord.ArtWidth = 15
    Next aBord
    With .Borders
        .DistanceFromLeft = 2
        .DistanceFromRight = 2
        .DistanceFrom = wdBorderDistanceFromText
        .JoinBorders = True
    End With
End With
```

SurroundFooter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSurroundFooterC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSurroundFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSurroundFooterA"}

True if a page border encompasses the document footer. Read/write **Boolean**.

SurroundFooter Property Example

This example formats the page border in section one in the active document so that it encompasses the header and footer on each page in the section.

```
With ActiveDocument.Sections(1).Borders
    .SurroundFooter = True
    .SurroundHeader = True
End With
```

This example adds a graphical page border around each page in section one. The page border doesn't encompass the header and footer areas.

```
With ActiveDocument.Sections(1)
    .Borders.SurroundFooter = False
    .Borders.SurroundHeader = False
    For Each aBord In .Borders
        aBord.ArtStyle = wdArtPeople
        aBord.ArtWidth = 15
    Next aBord
End With
```

SurroundHeader Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSurroundHeaderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSurroundHeaderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSurroundHeaderA"}

True if a page border encompasses the document header. Read/write **Boolean**.

SurroundHeader Property Example

This example formats the page border in section one in the active document to exclude the header and footer areas on each page.

```
With ActiveDocument.Sections(1).Borders
    .SurroundFooter = False
    .SurroundHeader = False
End With
```

DistanceFromRight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromRightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromRightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromRightA"}

Returns or sets the space (in points) between the right edge of the text and the right border. Read/write **Long**.

Note Using this property with a page border, you can set either the space between the text and the right border or the space between the right edge of the page and the right border. Where the distance is measured from depends on the value of the **DistanceFrom** property.

DistanceFromRight Property Example

This example adds a border around each paragraph in the selection and sets the distance between the text and the right border to 3 points.

```
With Selection.Paragraphs.Borders
    .Enable = True
    .DistanceFromRight = 3
End With
```

This example adds a single border around each page in section one in the active document. The example also sets the distance between the right and left border and the corresponding edges of the page to 22 points.

```
With ActiveDocument.Sections(1)
    For Each aBord In .Borders
        aBord.LineStyle = wdLineStyleSingle
        aBord.LineWidth = wdLineWidth050pt
    Next aBord
    With .Borders
        .DistanceFrom = wdBorderDistanceFromPageEdge
        .DistanceFromLeft = 22
        .DistanceFromRight = 22
    End With
End With
```

DistanceFromBottom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromBottomC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromBottomX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromBottomA"}

Returns or sets the space (in points) between the text and the bottom border. Read/write **Long**.

Note Using this property with a page border, you can set either the space between the text and the bottom page border or the space between the bottom edge of the page and the bottom page border. Where the distance is measured from depends on the value of the **DistanceFrom** property.

DistanceFromBottom Property Example

This example adds a border around the first paragraph in the active document and sets the distance between the text and the bottom border to 6 points.

```
With ActiveDocument.Paragraphs(1).Borders
    .Enable = True
    .DistanceFromBottom = 6
End With
```

This example adds a border around each table in Sales.doc. The example also sets the distance between the text and the border to 3 points for the top and bottom borders, and 6 points for the left and right borders.

```
For Each myTable In Documents("Sales.doc").Tables
    With myTable.Borders
        .OutsideLineStyle = wdLineStyleSingle
        .OutsideLineWidth = wdLineWidth150pt
        .DistanceFromBottom = 3
        .DistanceFromTop = 3
        .DistanceFromLeft = 6
        .DistanceFromRight = 6
    End With
Next myTable
```

DistanceFromLeft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromLeftC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromLeftA"}

Returns or sets the space (in points) between the text and the left border. Read/write **Long**.

Note Using this property with a page border, you can set either the space between the text and the left page border or the space between the left edge of the page and the left page border. Where the distance is measured from depends on the value of the **DistanceFrom** property.

DistanceFromLeft Property Example

This example adds a border around each frame in the active document and sets the distance between the frame and the border to 5 points.

```
For Each aFrame In ActiveDocument.Frames
    With aFrame.Borders
        .Enable = True
        .DistanceFromLeft = 5
        .DistanceFromRight = 5
        .DistanceFromTop = 5
        .DistanceFromBottom = 5
    End With
Next aFrame
```

This example adds a border around the first paragraph in the active document and sets the distance between the text and the left border to 3 points.

```
With ActiveDocument.Paragraphs(1).Borders
    .Enable = True
    .DistanceFromLeft = 3
End With
```

DistanceFromTop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceFromTopC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceFromTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceFromTopA"}

Returns or sets the space (in points) between the text and the top border. Read/write **Long**.

Note Using this property with a page border, you can set either the space between the text and the top page border or the space between the top edge of the page and the top page border. Where the distance is measured from depends on the value of the **DistanceFrom** property.

DistanceFromTop Property Example

This example adds a border around each paragraph in the selection and sets the distance between the text and the top border to 3 points.

```
With Selection.Borders
    .Enable = True
    .DistanceFromTop = 3
End With
```

This example adds a border around each page in the first section in the selection. The example also sets the distance between the text and the page border to 6 points.

```
With Selection.Sections(1)
    For Each aBorder In .Borders
        aBorder.ArtStyle = wdArtSeattle
        aBorder.ArtWidth = 22
    Next aBorder
    With .Borders
        .DistanceFrom = wdBorderDistanceFromText
        .DistanceFromTop = 6
        .DistanceFromLeft = 6
        .DistanceFromBottom = 6
        .DistanceFromRight = 6
    End With
End With
```

AcceptAllRevisions Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAcceptAllRevisionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAcceptAllRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAcceptAllRevisionsA"}

Accepts all tracked changes in the specified document.

Syntax

expression.**AcceptAllRevisions**

expression Required. An expression that returns a **Document** object.

AcceptAllRevisions Method Example

This example checks the main story in the active document for tracked changes, and if there are any, the example incorporates all revisions in all stories in the document.

```
If ActiveDocument.Revisions.Count >= 1 Then _
    ActiveDocument.AcceptAllRevisions
```

RejectAllRevisions Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRejectAllRevisionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRejectAllRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRejectAllRevisionsA"}

Rejects all tracked changes in the specified document.

Syntax

expression.**RejectAllRevisions**

expression Required. An expression that returns a **Document** object.

RejectAllRevisions Method Example

This example checks the main story in active document for tracked changes, and if there are any, the example rejects all revisions in all stories in the document.

```
If ActiveDocument.Revisions.Count >= 1 Then
ActiveDocument.RejectAllRevisions
```

ShowRevisions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowRevisionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowRevisionsA"}

True if tracked changes in the specified document are shown on the screen. Read/write **Boolean**.

ShowRevisions Property Example

This example sets the active document so that it tracks changes and makes them visible on the screen.

```
With ActiveDocument
    .TrackRevisions = True
    .ShowRevisions = True
End With
```

TrackRevisions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTrackRevisionsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTrackRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTrackRevisionsA"}

True if changes are tracked in the specified document. Read/write **Boolean**.

TrackRevisions Property Example

This example sets the active document so that it tracks changes and makes them visible on the screen.

```
With ActiveDocument
    .TrackRevisions = True
    .ShowRevisions = True
End With
```

CanOpen Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCanOpenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCanOpenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCanOpenA "}

True if the specified file converter is designed to open files. Read-only **Boolean**.

Note The **CanSave** property returns **True** if the specified file converter can be used to save (export) files.

CanOpen Property Example

This example determines whether the first file converter is able to open files.

```
If FileConverters(1).CanOpen = True Then
    MsgBox FileConverters(1).FormatName & " can open files"
End If
```

This example determines whether the WordPerfect6x file converter can be used to open files. If the **CanOpen** property returns **True**, a document named "Test.wp" is opened.

```
If FileConverters("WordPerfect6x").CanOpen = True Then
    Documents.Open FileName:="C:\Test.wp",
Format:=FileConverters("WordPerfect6x").OpenFormat
End If
```

CanSave Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCanSaveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCanSaveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCanSaveA "}

True if the specified file converter is designed to save files. Read-only **Boolean**.

Note The **CanOpen** property returns **True** if the specified file converter can be used to open (import) files.

CanSave Property Example

This example determines whether the WordPerfect converter can be used to save files. If the return value is **True**, the active document is saved in WordPerfect 6.x format.

```
If Application.FileConverters("WordPerfect6x").CanSave = True Then
    num = Application.FileConverters("WordPerfect6x").SaveFormat
    ActiveDocument.SaveAs FileName:="C:\Document.wp", FileFormat:=num
End If
```

This example displays a message that indicates whether the third converter in the **FileConverters** collection can save files.

```
If FileConverters(3).CanSave = True Then
    MsgBox FileConverters(3).FormatName & " can save files"
Else
    MsgBox FileConverters(3).FormatName & " cannot save files"
End If
```

ConvertMacWordChevrons Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproConvertMacWordChevronsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproConvertMacWordChevronsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproConvertMacWordChevronsA "}

Controls whether text enclosed in chevron characters (« ») is converted to merge fields. Read/write **Long**.

Can be one of the following **WdChevronConvertRule** constants.

Constant	Description
wdAlwaysConvert	The converter attempts to convert text enclosed in chevrons (« ») to mail merge fields.
wdNeverConvert	The converter passes the text through without attempting any interpretation.
wdAskToConvert, **wdAskToNotConvert**	The converter prompts the user to convert or not convert chevrons when a Word for the Macintosh document is opened.

Remarks

Word for the Macintosh version 4.0 and 5.x documents use chevron characters to denote mail merge fields.

ConvertMacWordChevrons Property Example

This example sets the **ConvertMacWordChevrons** property to convert the text enclosed in chevrons to mail merge fields, and then it opens the document named "Mac Word Document."

```
FileConverters.ConvertMacWordChevrons = wdAlwaysConvert
Documents.Open FileName:="C:\Documents\Mac Word Document"
```

FileConverters Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFileConvertersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFileConvertersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFileConvertersA "}

Returns a **FileConverters** collection that represents all the file converters available to Word. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

FileConverters Property Example

This example displays the path of the WordPerfect 5.0 file converter.

```
MsgBox FileConverters("WrdPrfctDOS50").Path
```

This example displays a message that indicates whether the third converter in the **FileConverters** collection can save files.

```
If FileConverters(3).CanSave = True Then
    MsgBox FileConverters(3).FormatName & " can save files"
Else
    MsgBox FileConverters(3).FormatName & " cannot save files"
End If
```

This example displays the name of the last file converter.

```
Set aConverter = FileConverters(FileConverters.Count)
MsgBox "The file extensions for " & aConverter.FormatName & _
    " files are: " & aConverter.Extensions
```

FormatName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFormatNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFormatNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFormatNameA "}

Returns the name of the specified file converter. The format names appear in the **Save as type** box in the **Save As** dialog box (**File** menu). Read-only **String**.

FormatName Property Example

This example displays the format name of the first converter in the **FileConverters** collection.

```
MsgBox FileConverters(1).FormatName
```

This example uses the `AvailableConv()` array to store the names of all the available file converters.

```
ReDim AvailableConv(FileConverters.Count - 1)
i = 0
For Each aConverter In FileConverters
    AvailableConv(i) = aConverter.FormatName
    i = i + 1
Next aConverter
```

DDEExecute Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDEExecuteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDEExecuteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDEExecuteA "}

Sends a command or series of commands to an application through the specified dynamic data exchange (DDE) channel.

Syntax

expression.**DDEExecute(*Channel, Command*)**

expression Optional. An expression that returns an **Application** object.

Channel Required **Long**. The channel number returned by the **DDEInitiate** method.

Command Required **String**. A command or series of commands recognized by the receiving application (the DDE server). If the receiving application cannot perform the specified command, an error occurs.

DDEExecute Method Example

This example creates a new worksheet in Microsoft Excel. The XLM macro instruction to create a new worksheet is `New(1)`.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[New(1)]"
DDETerminate Channel:=chan
```

This example runs the Microsoft Excel macro named "Macro1" in Personal.xls.

```
aChan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=aChan, Command:="[Run(" & Chr(34) & _
    "Personal.xls!Macro1" & Chr(34) & ")]"
DDETerminate Channel:=aChan
```

DDEInitiate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDEInitiateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDEInitiateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDEInitiateA "}

Opens a dynamic data exchange (DDE) channel to another application, and returns the channel number.

Syntax

expression.**DDEInitiate(*App, Topic*)**

expression Optional. An expression that returns an **Application** object.

App Required **String**. The name of the application.

Topic Required **String**. The name of a DDE topic – for example, the name of an open document – recognized by the application to which you're opening a channel.

Remarks

If it's successful, the **DDEInitiate** method returns the number of the open channel. All subsequent DDE functions use this number to specify the channel.

DDEInitiate Method Example

This example initiates a DDE conversation with the System topic and opens the Microsoft Excel workbook Sales.xls. The example terminates the DDE channel, initiates a channel to Sales.xls, and then inserts text into cell R1C1.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[OPEN(" & Chr(34) & "C:\Sales.xls" &
Chr(34) & ")]
DDETerminate Channel:=chan
chan = DDEInitiate(App:="Excel", Topic:="Sales.xls")
DDEPoke Channel:=chan, Item:="R1C1", Data:="1996 Sales"
DDETerminate Channel:=chan
```

DDEPoke Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDEPokeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDEPokeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDEPokeA "}

Uses an open dynamic data exchange (DDE) channel to send data to an application.

Syntax

expression.**DDEPoke(*Channel, Item, Data*)**

expression Optional. An expression that returns an **Application** object.

Channel Required **Long**. The channel number returned by the **DDEInitiate** method.

Item Required **String**. The item within a DDE topic to which the specified data is to be sent.

Data Required **String**. The data to be sent to the receiving application (the DDE server).

Remarks

If the **DDEPoke** method isn't successful, an error occurs.

DDEPoke Method Example

This example opens the Microsoft Excel workbook Sales.xls and inserts "1996 Sales" into cell R1C1.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[OPEN(" & Chr(34) & "C:\Sales.xls" &
Chr(34) & ")]
DDETerminate Channel:=chan
chan = DDEInitiate(App:="Excel", Topic:="Sales.xls")
DDEPoke Channel:=chan, Item:="R1C1", Data:="1996 Sales"
DDETerminate Channel:=chan
```

DDERequest Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDERequestC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDERequestX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDERequestA "}

Uses an open dynamic data exchange (DDE) channel to request information from the receiving application, and returns the information as a string.

Syntax

expression.**DDERequest**(***Channel, Item***)

expression Optional. An expression that returns an **Application** object.

Channel Required **Long**. The channel number returned by the **DDEInitiate** method.

Item Required **String**. The item to be requested.

Remarks

When you request information from the topic in the server application, you must specify the item in that topic whose contents you're requesting. In Microsoft Excel, for example, cells are valid items, and you refer to them by using either the "R1C1" format or named references.

Microsoft Excel and other applications that support DDE recognize a topic named "System." Three standard items in the System topic are described in the following table. Note that you can get a list of the other items in the System topic by using the SysItems item.

Item in System topic	Effect
SysItems	Returns a list of all the items in the System topic.
Topics	Returns a list of all the available topics.
Formats	Returns a list of all the Clipboard formats supported by Word.

DDERequest Method Example

This example opens the Microsoft Excel workbook Book1.xls and retrieves the contents of cell R1C1.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[OPEN(" & Chr(34) & "C:\My Documents\
Book1.xls" & Chr(34) & ")]"
DDETerminate Channel:=chan
chan = DDEInitiate(App:="Excel", Topic:="Book1.xls")
MsgBox DDERequest(Channel:=chan, Item:="R1C1")
DDETerminateAll
```

This example opens a channel to the System topic in Microsoft Excel and then uses the Topics item to return a list of available topics. The example inserts the topic list, which includes all open workbooks, after the selection.

```
aChan = DDEInitiate(App:="Excel", Topic:="System")
topicList = DDERequest(Channel:=aChan, Item:="Topics")
Selection.InsertAfter topicList
DDETerminate Channel:=aChan
```

DDETerminate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDETerminateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDETerminateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDETerminateA "}

Closes the specified dynamic data exchange (DDE) channel to another application.

Syntax

expression.**DDETerminate(*Channel*)**

expression Optional. An expression that returns an **Application** object.

Channel Required **Long**. The channel number returned by the **DDEInitiate** method.

DDETerminate Method Example

This example creates a new workbook in Microsoft Excel and then terminates the DDE conversation.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[New(1)]"
DDETerminate Channel:=chan
```

DDETerminateAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDDETerminateAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDDETerminateAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDDETerminateAllA "}

Closes all dynamic data exchange (DDE) channels opened by Word. This method doesn't close channels opened to Word by client applications. Using this method is the same as using the **DDETerminate** method for each open channel.

Syntax

expression.**DDETerminateAll**

expression Optional. An expression that returns an **Application** object.

Remarks

If you interrupt a macro that opens a DDE channel, you may inadvertently leave a channel open. Open channels aren't closed automatically when a macro ends, and each open channel uses system resources. For this reason, it's a good idea to use this method when you're debugging a macro that opens one or more DDE channels.

DDETerminateAll Method Example

This example opens the Microsoft Excel workbook Book1.xls, inserts text into cell R2C3, saves the workbook. and then terminates all DDE channels.

```
chan = DDEInitiate(App:="Excel", Topic:="System")
DDEExecute Channel:=chan, Command:="[OPEN(" & Chr(34) & _
    "C:\My Documents\Book1.xls" & Chr(34) & ")]"
DDETerminate Channel:=chan
chan = DDEInitiate(App:="Excel", Topic:="Book1.xls")
DDEPoke Channel:=chan, Item:="R2C3", Data:="Hello World"
DDEExecute Channel:=chan, Command:="[Save]"
DDETerminateAll
```

Kind Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKindC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKindX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKindA "}

Document object: Returns or sets the format type that Word uses when automatically formatting the specified document. Can be one of the following **WdDocumentKind** constants: **wdDocumentEmail**, **wdDocumentLetter**, or **wdDocumentNotSpecified**. Read/write **Long**.

Field object: Returns the type of link for a **Field** object. Read-only **Long**.

Can be one the following **WdFieldKind** constants.

Constant	Description
wdFieldKindHot	A field that's automatically updated each time it's displayed or each time the page is reformatted, but which can also be manually updated (for example, INCLUDEPICTURE or FORMDROPDOWN).
wdFieldKindWarm	A field that can be updated and has a result. This type includes fields that are automatically updated when the source changes as well as fields that can be manually updated (for example, DATE or INCLUDETEXT).
wdFieldKindNone	An invalid field (for example, a pair of field characters with nothing inside).
wdFieldKindCold	A field that doesn't have a result (for example, XE (Index Entry) fields, TC (Table of Contents Entry) fields, or Private fields).

Kind Property Example

This example asks the user whether the active document is an e-mail message. If the response is Yes, the document is formatted as an e-mail message.

```
response = MsgBox("Is this document an email message?", vbYesNo)
If response = vbYes Then
    ActiveDocument.Kind = wdDocumentEmail
    ActiveDocument.Content.AutoFormat
End If
```

This example updates all warm link fields in the active document.

```
For Each aField In ActiveDocument.Fields
    If aField.Kind = wdFieldKindWarm Then aField.Update
Next aField
```

AutoHyphenation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoHyphenationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoHyphenationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoHyphenationA "}

True if automatic hyphenation is turned on for the specified document. Read/write **Boolean**.

AutoHyphenation Property Example

This example turns on automatic hyphenation, with a hyphenation zone of 0.25 inch. Words in all capital letters aren't hyphenated.

```
With ActiveDocument
    .HyphenationZone = InchesToPoints(0.25)
    .HyphenateCaps = False
    .AutoHyphenation = True
End With
```

Comments Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCommentsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCommentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCommentsA "}

Returns a **Comments** collection that represents all the comments in the specified document, selection, or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Comments Property Example

This example adds a comment to the selected text.

```
ActiveWindow.View.ShowHiddenText = True
Selection.Comments.Add Range:=Selection.Range, Text:="Approved"
```

This example compares the author name of each comment in the active document with the user name on the **User Information** tab in the **Options** dialog box (**Tools** menu). If the names aren't the same, the comment reference mark is formatted to appear in red.

```
For Each comm In ActiveDocument.Comments
    If comm.Author <> Application.UserName Then _
        comm.Reference.Font.ColorIndex = wdRed
Next comm
```

ComputeStatistics Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthComputeStatisticsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthComputeStatisticsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthComputeStatisticsA "}

Syntax 1: Returns a statistic based on the contents of the specified document.

Syntax 2: Returns a statistic based on the contents of the specified range.

Syntax 1

expression.**ComputeStatistics(*Statistic, IncludeFootnotesAndEndnotes*)**

Syntax 2

expression.**ComputeStatistics(*Statistic*)**

expression Syntax 1: Required. An expression that returns a **Document** object.

Syntax 2: Required. An expression that returns a **Range** object.

Statistic Required **Long**. The statistic. Can be one of the following **WdStatistic** constants: **wdStatisticCharacters**, **wdStatisticCharactersWithSpaces**, **wdStatisticLines**, **wdStatisticPages**, **wdStatisticParagraphs**, or **wdStatisticWords**.

IncludeFootnotesAndEndnotes Optional **Variant**. **True** to include footnotes and endnotes when computing statistics. If this argument is omitted, the default value is **False**.

ComputeStatistics Method Example

This example displays the number of words and characters in the first paragraph of Report.doc.

```
Set myRange = Documents("Report.doc").Paragraphs(1).Range
wordCount = myRange.ComputeStatistics(Statistic:=wdStatisticWords)
charCount = myRange.ComputeStatistics(Statistic:=wdStatisticCharacters)
MsgBox "The first paragraph contains " & wordCount _
    & " words and a total of " & charCount & " characters."
```

This example displays the number of words in the active document, including footnotes.

```
MsgBox ActiveDocument.ComputeStatistics(Statistic:=wdStatisticWords, _
    IncludeFootnotesAndEndnotes:=True) & " words"
```

DataForm Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDataFormC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDataFormX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDataFormA "}

Displays the **Data Form** dialog box, in which you can add, delete, or modify data records.

Note You can use this method with a mail merge main document, a mail merge data source, or any document that contains data delimited by table cells or separator characters.

Syntax

expression.**DataForm**

expression Required. An expression that returns a **Document** object.

DataForm Method Example

This example displays the **Data Form** dialog box if the active document is a mail merge document.

```
If ActiveDocument.MailMerge.State <> wdNormalDocument Then
    ActiveDocument.DataForm
End If
```

This example creates a table in a new document and then displays the **Data Form** dialog box.

```
Set aDoc = Documents.Add
With aDoc
    .Tables.Add Range:=aDoc.Content, NumRows:=2, NumColumns:=2
    .Tables(1).Cell(1, 1).Range.Text = "Name"
    .Tables(1).Cell(1, 2).Range.Text = "Age"
    .DataForm
End With
```

Saved Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSavedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSavedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSavedA "}

True if the specified document or template hasn't changed since it was last saved. **False** if Word displays a prompt to save changes when the document is closed. Read/write **Boolean**.

Saved Property Example

This example saves the active document if it contains previously unsaved changes.

```
If ActiveDocument.Saved = False Then ActiveDocument.Save
```

This example changes the status of the Normal template to unchanged. If changes were made to the Normal template, the changes aren't saved when you quit Word.

```
NormalTemplate.Saved = True
Application.Quit
```

Document Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDocumentA "}

Returns a **Document** object associated with the specified pane, window, or selection. Read-only.

Document Property Example

This example displays the document name and path for the selection.

```
Msgbox Selection.Document.FullName
```

This example sets `myDoc` to the document associated with the active window. The focus is changed to the next window, and the window is split. The **Activate** method is used to switch back to the original document.

```
Set myDoc = ActiveWindow.Document
If Windows.Count >= 2 Then
    ActiveWindow.Next.Activate
    ActiveWindow.Split = True
    myDoc.Activate
End If
```

Documents Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDocumentsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDocumentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDocumentsA "}

Returns a **Documents** collection that represents all the open documents. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Documents Property Example

This example creates a new document based on the Normal template and then displays the **Save As** dialog box.

```
Documents.Add.Save
```

This example saves open documents that have changed since they were last saved.

```
For Each aDoc In Documents
   If aDoc.Saved = False Then aDoc.Save
Next aDoc
```

This example prints each open document after setting the left and right margins to 0.5 inch.

```
For Each openDoc In Documents
    With openDoc
        .PageSetup.LeftMargin = InchesToPoints(0.5)
        .PageSetup.RightMargin = InchesToPoints(0.5)
        .PrintOut
    End With
Next openDoc
```

This example opens MyDoc.doc as a read-only document.

```
Documents.Open FileName:="C:\MyFiles\MyDoc.doc", ReadOnly:=True
```

Format Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFormatA "}

Returns or sets the formatting for the specified object, as described in the following table.

Object	Description
Indexes	Returns or sets the formatting for the indexes in the specified document. Can be one of the following **WdIndexFormat** constants: **wdIndexBulleted**, **wdIndexClassic**, **wdIndexFancy**, **wdIndexFormal**, **wdIndexModern**, **wdIndexSimple** or **wdIndexTemplate**. Read/write **Long**
Find	**True** if formatting is included in the find operation. Read/write **Boolean**.
Paragraph, **Paragraphs**	Returns or sets a **ParagraphFormat** object that represents the formatting of the specified paragraph or paragraphs. Read/write.
TablesOfAuthorities	Returns or sets the formatting for the tables of authorities in the specified document. Can be one of the following **WdTOAFormat** constants: **wdTOAClassic**, **wdTOADistinctive**, **wdTOAFormal**, **wdTOASimple**, or **wdTOATemplate**. Read/write **Long**.
TablesOfContents	Returns or sets the formatting for the tables of contents in the specified document. Can be one of the following **WdTOCFormat** constants:, **wdTOCClassic**, **wdTOCDistinctive**, **wdTOCFancy**, **wdTOCFormal**, **wdTOCModern**, **wdTOCSimple** or **wdTOCTemplate**. Read/write **Long**.
TablesOfFigures	Returns or sets the formatting for the tables of figures in the specified document. Can be one of the following **WdTOFFormat** constants: **wdTOFCentered**, **wdTOFClassic**, **wdTOFDistinctive**, **wdTOFFormal**, **wdTOFSimple**, or **wdTOFTemplate**. Read/write **Long**.
TextInput	Returns the text formatting for the specified text box. Read-only **String**. Use the **EditType** method of the **TextInput** object to set the text formatting.

Format Property Example

The following example left aligns all the paragraphs in the active document.

```
ActiveDocument.Paragraphs.Format.Alignment = wdAlignParagraphLeft
```

This example applies Classic formatting to the tables of contents in Report.doc.

```
Documents("Report.doc").TablesOfContents.Format = wdTOCClassic
```

This example returns and displays the formatting of the text in the first field in the active document.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormTextInput Then
    MsgBox ActiveDocument.FormFields(1).TextInput.Format
Else
    MsgBox "First field is not a text form field"
End If
```

This example returns the formatting of the first paragraph in the active document and then applies the formatting to the selection.

```
Set paraFormat = ActiveDocument.Paragraphs(1).Format.Duplicate
Selection.Paragraphs.Format = paraFormat
```

This example removes all bold formatting in the active document.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Font.Bold = True
    .Format = True
    .Replacement.ClearFormatting
    .Replacement.Font.Bold = False
    .Execute Forward:=True, Replace:=wdReplaceAll, _
        FindText:="", ReplaceWith:=""
End With
```

HyphenateCaps Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHyphenateCapsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHyphenateCapsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHyphenateCapsA "}

True if words in all capital letters can be hyphenated. Read/write **Boolean**.

HyphenateCaps Property Example

This example enables automatic hyphenation for the active document and allows capitalized words to be hyphenated.

```
With ActiveDocument
    .AutoHyphenation = True
    .HyphenateCaps = True
End With
```

HyphenationZone Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHyphenationZoneC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHyphenationZoneX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHyphenationZoneA "}

Returns or sets the width of the hyphenation zone, in points. The hyphenation zone is the maximum amount of space that Word leaves between the end of the last word in a line and the right margin. Read/write **Long**.

HyphenationZone Property Example

This example enables automatic hyphenation for MyReport.doc. The hyphenation zone is set to 36 points (0.5 inch).

```
With Documents("MyReport.doc")
    .AutoHyphenation = True
    .HyphenationZone = 36
End With
```

This example sets the hyphenation zone to 0.25 inch (18 points) and then starts manual hyphenation of the active document.

```
With ActiveDocument
    .HyphenationZone = InchesToPoints(0.25)
    .ManualHyphenation
End With
```

Initial Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInitialC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInitialX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInitialA "}

Returns or sets the initials of the user associated with a specific comment. Read/write **String**.

Initial Property Example

This example displays the initials of the user who made the first comment in the selection.

```
If Selection.Comments.Count >= 1 Then
    MsgBox "Comment made by " & Selection.Comments(1).Initial
End If
```

This example checks the author initials associated with each comment in the first document section. If the author initials are "MSOffice," the example changes them to "KAE."

```
Set myRange = ActiveDocument.Sections(1).Range
For Each comm In myRange.Comments
    If comm.Initial = "MSOffice" Then comm.Initial = "KAE"
Next comm
```

ConsecutiveHyphensLimit Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLimitConsecutiveHyphensC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLimitConsecutiveHyphensX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLimitConsecutiveHyphensA "}

Returns or sets the maximum number of consecutive lines that can end with hyphens. Read/write. **Long**.

Note If this property is set to 0 (zero), any number of consecutive lines can end with hyphens.

ConsecutiveHyphensLimit Property Example

This example enables automatic hyphenation for MyReport.docand limits the number of consecutive lines that can end with hyphens to two.

```
With Documents("MyReport.doc")
    .AutoHyphenation = True
    .ConsecutiveHyphensLimit = 2
End With
```

This example sets no limit on the number of consecutive lines that can end with hyphens.

```
ActiveDocument.ConsecutiveHyphensLimit = 0
```

ManualHyphenation Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthManualHyphenationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthManualHyphenationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthManualHyphenationA "}

Initiates manual hyphenation of a document, one line at a time. The user is prompted to accept or decline suggested hyphenations.

Syntax

expression.**ManualHyphenation**

expression Required. An expression that returns a **Document** object.

ManualHyphenation Method Example

This example starts manual hyphenation of the active document.

```
ActiveDocument.ManualHyphenation
```

This example sets hyphenation options and then starts manual hyphenation of MyDoc.doc.

```
With Documents("MyDoc.doc")
    .HyphenationZone = InchesToPoints(0.25)
    .HyphenateCaps = False
    .ManualHyphenation
End With
```

PageSetup Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageSetupC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageSetupX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageSetupA "}

Returns a **PageSetup** object that's associated with the specified document, range, section, sections, or selection. Read-only.

PageSetup Property Example

This example sets the right margin of the active document to 72 points (1 inch).

```
ActiveDocument.PageSetup.RightMargin = InchesToPoints(1)
```

This example sets the gutter for the first section in Summary.doc to 36 points (0.5 inch).

```
Documents("Summary.doc").Sections(1).PageSetup.Gutter = 36
```

This example sets the header and footer distance to 18 points (0.25 inch) from the top and bottom of the page, respectively. This formatting change is applied to the section that contains the selection.

```
With Selection.PageSetup
    .FooterDistance = 18
    .HeaderDistance = 18
End With
```

This example displays the left margin setting, in inches.

```
MsgBox PointsToInches(ActiveDocument.PageSetup.LeftMargin) & " inches"
```

Path Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPathC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPathX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPathA "}

Returns the path to the specified **AddIn**, **Application**, **Dictionary**, **Document**, **FileConverter**, **RecentFile**, **Subdocument**, or **Template** object. Read-only **String**.

Note The path doesn't include a trailing character – for example,C:\MSOffice). Use the **PathSeparator** property to add the character that separates folders and drive letters (in Windows, a backslash (\); on the Macintosh, a colon (:)). Use the **Name** property to return the file name without the path and use the **FullName** property to return the file name and the path together.

Path Property Example

This example displays the path and file name of the active document.

```
MsgBox ActiveDocument.Path & Application.PathSeparator & _
    ActiveDocument.Name
```

This example changes the current folder to the path of the template attached to the active document.

```
ChDir ActiveDocument.AttachedTemplate.Path
```

This example displays the path of the first addin in the **AddIns** collection.

```
If AddIns.Count >= 1 Then MsgBox AddIns(1).Path
```

Position Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPositionA "}

CaptionLabel object: Returns or sets the position of caption label text. Can be either of the following **WdCaptionPosition** constants: **wdCaptionPositionAbove** or **wdCaptionPositionBelow**. Read/write **Long**.

DropCap object: Returns or sets the position of a dropped capital letter. Can be one of the following **WdDropPosition** constants: **wdDropMargin**, **wdDropNone**, or **wdDropNormal**. Read/write **Long**.

Font object: Returns or sets the position of text (in points) relative to the base line. A positive number raises the text, and a negative number lowers it. Read/write **Long**.

TabStop object: Returns or sets the position of a tab stop relative to the left margin. Read/write **Single**.

Position Property Example

This example lowers the selected text by 2 points.

```
Selection.Font.Position = -2
```

This example adds a right tab stop to the selected paragraphs 2 inches from the left margin. The position of the tab stop is then displayed in a message box.

```
With Selection.Paragraphs.TabStops
    .ClearAll
    .Add Position:=InchesToPoints(2), Alignment:=wdAlignTabRight
    MsgBox .Item(1).Position & " or " & _
        PointsToInches(.Item(1).Position) & " inches"
End With
```

This example sets the first paragraph in the active document to begin with a dropped capital letter. The position of the **DropCap** object is set to **wdDropNormal**.

```
With ActiveDocument.Paragraphs(1).DropCap
    .Enable
    .FontName= "Arial"
    .Position = wdDropNormal
End With
```

Exists Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExistsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExistsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExistsA "}

Determines whether the specified bookmark or task exists. Returns **True** if the bookmark or task exists.

Syntax

expression.**Exists(*Name*)**

expression An expression that returns a **Bookmarks** or **Tasks** object.

Name Required **String**. A bookmark or task name.

Exists Method Example

This example determines whether the bookmark named "start" exists in the active document. If the bookmark exists, it's deleted.

```
If ActiveDocument.Bookmarks.Exists("start") = True Then
    ActiveDocument.Bookmarks("start").Delete
End If
```

This example determines whether the Windows Calculator program is running (if the task exists). If Calculator isn't running, the **Shell** statement starts it. If Calculator is running, the application is activated.

```
If Tasks.Exists("Calculator") = False Then
    Shell "Calc.exe"
Else
    Tasks("Calculator").Activate
End If
Tasks("Calculator").WindowState = wdWindowStateNormal
```

PrintOut Method

Prints an object, as shown in the following table. For a detailed description of the **PrintOut** method for a particular object, select that object in the table.

Object	Description
Application, **Document**, **Window**	Prints all or part of the specified document
Envelope	Prints an envelope without adding the envelope to the active document
MailingLabel	Prints a label or a page of labels with the same address

PrintOut Method (MailingLabel Object)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPrintOutMailingLabelobjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPrintOutMailingLabelobjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPrintOutMailingLabelobjA "}

Prints a label or a page of labels with the same address.

Syntax

expression.**PrintOut(*Name, Address, ExtractAddress, LaserTray, SingleLabel, Row, Column*)**

expression Required. An expression that returns a **MailingLabel** object.

Name Optional **Variant**. The mailing label name.

Address Optional **Variant**. The text for the label address.

ExtractAddress Optional **Variant**. **True** to use the text marked by the "EnvelopeAddress" bookmark (a user-defined bookmark) as the label text. If this argument is specified, ***Address*** and ***AutoText*** are ignored.

LaserTray Optional **Variant**. The laser printer tray to be used. Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed	**wdPrinterManualFeed**
wdPrinterDefaultBin	**wdPrinterMiddleBin**
wdPrinterEnvelopeFeed	**wdPrinterOnlyBin**
wdPrinterFormSource	**wdPrinterPaperCassette**
wdPrinterLargeCapacityBin	**wdPrinterSmallFormatBin**
wdPrinterLargeFormatBin	**wdPrinterTractorFeed**
wdPrinterLowerBin	**wdPrinterUpperBin**
wdPrinterManualEnvelopeFeed	

SingleLabel Optional **Variant**. **True** to print a single label, **False** to print an entire page of the same label.

Row Optional **Variant**. The label row for a single label. Not valid if ***SingleLabel*** is **False**.

Column Optional **Variant**. The label column for a single label. Not valid if ***SingleLabel*** is **False**.

PrintOut Method (MailingLabel Object) Example

This example prints a page of Avery 5664 mailing labels, using the specified address.

```
addr = "Jane Doe" & vbCr & "123 Skye St." & vbCr & "OurTown, WA 98107"
Application.MailingLabel.PrintOut Name:="5664", Address:=addr
```

ProtectionType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProtectionTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProtectionTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProtectionTypeA "}

Returns the protection type for the specified document. Can be one of the following **WdProtectionType** constants: **wdAllowOnlyComments**, **wdAllowOnlyFormFields, wdAllowOnlyRevisions,** or **wdNoProtection**. Read-only **Long**.

ProtectionType Property Example

If the active document isn't already protected, this example protects the document for comments.

```
If ActiveDocument.ProtectionType = wdNoProtection Then
    ActiveDocument.Protect Type:=wdAllowOnlyComments
End If
```

This example unprotects the active document if it's protected.

```
Set Doc = ActiveDocument
If Doc.ProtectionType <> wdNoProtection Then Doc.Unprotect
```

Redo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRedoC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRedoX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRedoA "}

Redoes the last action that was undone (reverses the **Undo** method). Returns **True** if the actions were redone successfully.

Syntax

expression.**Redo(*Times*)**

expression Required. An expression that returns a **Document** object.

Times Optional **Variant**. The number of actions to be redone.

Redo Method Example

This example redoes the last two actions in the Sales.doc redo list.

```
Documents("Sales.doc").Redo 2
```

This example redoes the last action in the active document. If the action is successfully redone, a message is displayed in the status bar.

```
On Error Resume Next
If ActiveDocument.Redo = False Then StatusBar = "Redo was unsuccessful"
```

Repaginate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRepaginateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRepaginateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRepaginateA "}

Repaginates the entire document.

Syntax

expression.**Repaginate**

expression Required. An expression that returns a **Document** object.

Repaginate Method Example

This example repaginates the active document if it's changed since the last time it was saved.

```
If ActiveDocument.Saved = False Then ActiveDocument.Repaginate
```

This example repaginates all open documents.

```
For Each doc In Documents
    doc.Repaginate
Next doc
```

UpdateStylesOnOpen Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUpdateStylesOnOpenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUpdateStylesOnOpenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUpdateStylesOnOpenA "}

True if the styles in the specified document are updated to match the styles in the attached template each time the document is opened. Read/write **Boolean**.

UpdateStylesOnOpen Property Example

This example enables the option to update document styles for all open documents and then closes the documents. When any of these documents is reopened, changes to the styles in the attached template will automatically appear in the document.

```
For Each doc In Documents
    doc.UpdateStylesOnOpen = True
    doc.Close SaveChanges:=wdSaveChanges
Next doc
```

This example disables the option to update document styles so that changes made to the styles in the attached template aren't reflected in Report.doc.

```
Documents("Report.doc").UpdateStylesOnOpen = False
```

Save Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSaveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSaveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSaveA "}

Syntax 1: Saves the specified document or template. If the document or template hasn't been saved before, the **Save As** dialog box prompts the user for a file name.

Syntax 2: Saves all the documents in the **Documents** collection. If a document hasn't been saved before, the **Save As** dialog box prompts the user for a file name.

Syntax 3: Saves a version of the specified document with a comment.

Syntax 1

expression.**Save**

Syntax 2

expression.**Save(*NoPrompt, OriginalFormat*)**

Syntax 3

expression.**Save(*Comment*)**

expression Syntax 1: Required. An expression that returns a **Document** or **Template** object.

Syntax 2: Required. An expression that returns a **Documents** object.

Syntax 3: Required. An expression that returns a **Versions** object.

NoPrompt Optional **Variant**. **True** to have Word automatically save all documents. **False** to have Word prompt the user to save each document that has changed since it was last saved.

OriginalFormat Optional **Variant**. Specifies the way the documents are saved. Can be one of the following **WdOriginalFormat** constants: **wdOriginalDocumentFormat**, **wdPromptUser,** or **wdWordDocument**.

Comment Optional **Variant**. The comment string that's saved with the version.

Save Method Example

This example saves the active document if it's changed since it was last saved.

```
If ActiveDocument.Saved = False Then ActiveDocument.Save
```

This example saves each document in the **Documents** collection without first prompting the user.

```
Documents.Save NoPrompt:=True, OriginalFormat:=wdOriginalDocumentFormat
```

If Sales.doc is open, this example saves a version of Sales.doc, with a comment.

```
For Each doc in Documents
    If Instr(1, doc.Name, "Sales.doc", 1) > 0 Then
        doc.Versions.Save Comment:="Minor changes to intro"
    End If
Next doc
```

SaveAs Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSaveAsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSaveAsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSaveAsA "}

Saves the specfied document with a new name or format. The arguments for this method correspond to the options in the **Save As** dialog box (**File** menu).

Syntax

expression.**SaveAs(*FileName, FileFormat, LockComments, Password, AddToRecentFiles, WritePassword, ReadOnlyRecommended, EmbedTrueTypeFonts, SaveNativePictureFormat, SaveFormsData, SaveAsAOCELetter*)**

expression Required. An expression that returns a **Document** object.

FileName Optional **Variant**. The name for the document. The default is the current folder and file name. If the document has never been saved, the default name is used (for example, Doc1.doc). If a document with the specified ***FileName*** already exists, the document is overwritten without the user being prompted first.

FileFormat Optional **Variant**. The format in which the document is saved. Can be one of the following **WdSaveFormat** constants: **wdFormatDocument**, **wdFormatDOSText**, **wdFormatDOSTextLineBreaks**, **wdFormatRTF**, **wdFormatTemplate**, **wdFormatText**, **wdFormatTextLineBreaks**, or **wdFormatUnicodeText**. To save a document in another format, specify the appropriate value for the **SaveFormat** property of the **FileConverter** object.

LockComments Optional **Variant**. **True** to lock the document for comments.

Password Optional **Variant**. A password string for opening the document.

AddToRecentFiles Optional **Variant**. **True** to add the document to the list of recently used files on the **File** menu.

WritePassword Optional **Variant**. A password string for saving changes to the document.

ReadOnlyRecommended Optional **Variant**. **True** to have Word suggest read-only status whenever the document is opened.

EmbedTrueTypeFonts Optional **Variant**. **True** to save TrueType fonts with the document.

SaveNativePictureFormat Optional **Variant**. If graphics were imported from another platform (Macintosh), **True** to save only the Windows version of the imported graphics.

SaveFormsData Optional **Variant**. **True** to save the data entered by a user in a form as a data record.

SaveAsAOCELetter Optional **Variant**. If the document has an attached mailer, **True** to save the document as an AOCE letter (the mailer is saved).

SaveAs Method Example

This example saves the active document as Test.rtf in rich-text format (RTF).

```
ActiveDocument.SaveAs FileName:="Text.rtf", FileFormat:=wdFormatRTF
```

This example saves the active document in text-file format, with the file name extension ".txt".

```
myDocname = ActiveDocument.Name
pos = InStr(myDocname, ".")
If pos > 0 Then
    myDocname = Left(myDocname, pos - 1)
    myDocname = myDocname & ".txt"
    ActiveDocument.SaveAs FileName:=myDocname, FileFormat:=wdFormatText
End If
```

This example retrieves the **SaveFormat** property value for the WordPerfect 6.x converter, and then it uses this value with the **SaveAs** method.

```
formatNum = FileConverters("WordPerfect6x").SaveFormat
Documents(1).SaveAs FileName:="Hello.wp", FileFormat:=formatNum
```

This example saves MyDoc.doc with a write password and then closes the document.

```
With Documents("MyDoc.doc")
    .SaveAs WritePassword:="pass"
    .Close
End With
```

Scope Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproScopeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproScopeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproScopeA "}

Returns a **Range** object that represents the range of text marked by the specified comment. Read-only.

Scope Property Example

This example displays the text associated with the first comment in the selection.

```
If Selection.Comments.Count >= 1 Then
    Set myRange = Selection.Comments(1).Scope
    MsgBox myRange.Text
End If
```

This example copies the text associated with the last comment in the active document.

```
total = ActiveDocument.Comments.Count
If total >= 1 Then ActiveDocument.Comments(total).Scope.Copy
```

SendMail Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSendMailC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSendMailX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSendMailA "}

Opens a message window for sending the specified document through Microsoft Exchange.

Note Use the **SendMailAttach** property to control whether the document is sent as text in the message window or as an attachment.

Syntax

expression.**SendMail**

expression Required. An expression that returns a **Document** object.

SendMail Method Example

This example sends the active document as an attachment to a mail message.

```
Options.SendMailAttach = True
ActiveDocument.SendMail
```

ShowTip Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowTipC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowTipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowTipA "}

True if text associated with a comment is displayed in a screen tip. The screen tip remains displayed until you click the mouse or press a key. Read/write **Boolean**.

ShowTip Property Example

This example shows the screen tip for the first comment in the active document.

```
If ActiveDocument.Comments.Count >= 1 Then
    ActiveDocument.Comments(1).ShowTip = True
End If
```

This example shows the screen tip for the next comment in the active document.

```
If ActiveDocument.Comments.Count >= 1 Then
    With Selection
        .GoTo What:=wdGotoComment, Which:=wdGotoNext
        .MoveEnd Unit:=wdWord, Count:=1
        .Comments(1).ShowTip = True
    End With
End If
```

FitToPages Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFitToPagesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFitToPagesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFitToPagesA "}

Decreases the font size of text just enough so that the document will fit on one fewer pages. An error occurs if Word is unable to reduce the page count by one.

Syntax

expression.**FitToPages**

expression Required. An expression that returns a **Document** object.

FitToPages Method Example

This example attempts to reduce the page count of the active document by one page.

```
On Error GoTo errhandler
ActiveDocument.FitToPages
errhandler:
If Err = 5538 Then MsgBox "Fit to pages failed"
```

This example attempts to reduce the page count of each open document by one page.

```
For Each doc In Documents
    doc.FitToPages
Next doc
```

Undo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUndoC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUndoX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUndoA "}

Undoes the last action or a sequence of actions, which are displayed in the **Undo** list. Returns **True** if the actions were successfully undone.

Syntax

expression.**Undo(*Times*)**

expression Required. An expression that returns a **Document** object.

Times Optional **Variant**. The number of actions to be undone.

Undo Method Example

This example undoes the last two actions taken in Sales.doc.

```
Documents("Sales.doc").Undo 2
```

This example undoes the last action. If the action is successfully undone, a message is displayed in the status bar.

```
On Error Resume Next
If ActiveDocument.Undo = False Then StatusBar = "Undo was unsuccessful"
```

UnProtect Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUnProtectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUnProtectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUnProtectA "}

Removes protection from the specified document. If the document isn't protected, this method generates an error.

Syntax

expression.**UnProtect(*Password*)**

expression Required. An expression that returns a **Document** object.

Password Optional **Variant**. The password string used to protect the document. Passwords are case-sensitive. If the document is protected with a password and the correct password isn't supplied, a dialog box prompts the user for the password.

UnProtect Method Example

This example unprotects the active document, using "Blue" as the password. If the document has a password, a dialog box prompts the user for the password.

```
If ActiveDocument.ProtectionType <> wdNoProtection Then
    ActiveDocument.Unprotect Password:="Blue"
End If
```

This example unprotects the active document. Text is inserted, and the document is protected for revisions.

```
Set aDoc = ActiveDocument
If aDoc.ProtectionType <> wdNoProtection Then
    aDoc.Unprotect
    Selection.InsertBefore "department six"
    aDoc.Protect Type:=wdAllowOnlyRevisions, Password:="Blue"
End If
```

Variables Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVariablesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVariablesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVariablesA "}

Returns a **Variables** collection that represents the variables stored in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Variables Property Example

This example adds a document variable named "Value1" to the active document. The example then retrieves the value from the Value1 variable, adds 3 to the value, and displays the results.

```
ActiveDocument.Variables.Add Name:="Value1", Value:="1"
MsgBox ActiveDocument.Variables("Value1") + 3
```

This example displays the name and value of each document variable in the active document.

```
For Each myVar In ActiveDocument.Variables
    MsgBox "Name =" & myVar.Name & vbCr & "Value = " & myVar.Value
Next myVar
```

UpdateStyles Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateStylesA "}

Copies all styles from the attached template into the document, overwriting any existing styles in the document that have the same name.

Syntax

expression.**UpdateStyles**

expression Required. An expression that returns a **Document** object.

UpdateStyles Method Example

This example copies the styles from the attached template into each open document, and then it closes each document.

```
For Each aDoc In Documents
    aDoc.UpdateStyles
    aDoc.Close SaveChanges:=wdSaveChanges
Next aDoc
```

This example changes the formatting of the Heading 1 style in the template attached to the active document. The **UpdateStyles** method updates the styles in the active document, including the Heading 1 style.

```
Set aDoc = ActiveDocument.AttachedTemplate.OpenAsDocument
With aDoc.Styles(wdStyleHeading1).Font
    .Name = "Arial"
    .Bold = False
End With
aDoc.Close SaveChanges:=wdSaveChanges
ActiveDocument.UpdateStyles
```

CopyStylesFromTemplate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCopyStylesFromTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCopyStylesFromTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCopyStylesFromTemplateA "}

Copies styles from the specified template to a document.

Syntax

expression.**CopyStylesFromTemplate(*Template*)**

expression Required. An expression that returns a **Document** object.

Template Required **String**. The template file name.

Remarks

When styles are copied from a template to a document, like-named styles in the document are redefined to match the style descriptions in the template. Unique styles from the template are copied to the document. Unique styles in the document remain intact.

CopyStylesFromTemplate Method Example

This example copies the styles from the active document's template to the document.

```
ActiveDocument.CopyStylesFromTemplate _
    Template:=ActiveDocument.AttachedTemplate.FullName
```

This example copies the styles from the Sales96.dot template to Sales.doc.

```
Documents("Sales.doc").CopyStylesFromTemplate _
    Template:="C:\MSOffice\Templates\Sales96.dot"
```

SubAddress Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSubAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSubAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSubAddressA "}

Returns a named location in the destination of the specified hyperlink. Read-only **String**.

Note The named location can be a bookmark in a Microsoft Word document, a named cell or cell reference in a Microsoft Excel worksheet, a named object in a Microsoft Access database, or a slide number in a Microsoft PowerPoint presentation.

SubAddress Property Example

This example displays the subaddress of the selected hyperlink.

```
If Selection.Range.Hyperlinks.Count >= 1 Then
    MsgBox Selection.Range.Hyperlinks(1).SubAddress
End If
```

This example adds a hyperlink to the selection in the active document, sets the hyperlink destination and subaddress, and then displays them in a message box.

```
Set SCut = ActiveDocument.Hyperlinks.Add(Anchor:= Selection.Range, _
    Address:="C:\My Documents\Other.doc", SubAddress:= "temp")
MsgBox "The hyperlink goes to " & SCut.SubAddress
```

SummaryLength Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSummaryLengthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSummaryLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSummaryLengthA "}

Returns or sets the length of the summary as a percentage of the document length. The larger the number, the more detail that's included in the summary. Read/write **Long**.

Note This property takes effect immediately if the **AutoSummarize** toolbar is displayed; otherwise, it takes effect the next time the **AutoSummarize** method or the **SummaryViewMode** property is applied to the document.

SummaryLength Property Example

This example highlights the key points in the active document. The level of detail is set to 50 percent.

```
With ActiveDocument
    .AutoSummarize Mode:=wdSummaryModeHighlight
    .SummaryLength = 50
End With
```

This example displays the summary and sets the level of detail to 55 percent.

```
With ActiveDocument
    .ShowSummary = True
    .SummaryLength = 55
End With
```

SummaryViewMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSummaryViewModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSummaryViewModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSummaryViewModeA "}

Returns or sets the way a summary is displayed. This property corresponds to **Type of summary** in the **AutoSummarize** dialog box (**Tools** menu). Read/write **Long**.

Can be one of the following **WdSummaryMode** constants.

Constant	Description
wdSummaryModeHighlight	Highlights the key points in the specified document and displays the **AutoSummarize** toolbar.
wdSummaryModeInsert	Inserts a summary at the beginning of the specified document.
wdSummaryModeCreateNew	Creates a new document and inserts the specified summary.
wdSummaryModeHideAllButSummary	Hides everything except the specified summary and displays the **AutoSummarize** toolbar.

SummaryViewMode Property Example

This example hides everything in the active document except the summary text.

```
With ActiveDocument
    .SummaryViewMode = wdSummaryModeHideAllButSummary
    .SummaryLength = 60
    .ShowSummary = True
End With
```

Versions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVersionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVersionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVersionsA "}

Returns a **Versions** collection that represents all the versions of the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Versions Property Example

This example displays the user name and date of the most recent version of the document.

```
If ActiveDocument.Versions.Count >= 1 Then
    Set aVersion = ActiveDocument.Versions(ActiveDocument.Versions.Count)
    MsgBox "Saved by " & aVersion.SavedBy & " on " & aVersion.Date
End If
```

This example saves a version of Contract.doc with a short comment.

```
Documents("Contract.doc").Versions.Save Comment:="Added a single word"
```

RunAutoMacro Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRunAutoMacroC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRunAutoMacroX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRunAutoMacroA "}

Runs an auto macro that's stored in the specified document. If the specified auto macro doesn't exist, nothing happens.

Note Use the **Run** method to run any macro.

Syntax

expression.**RunAutoMacro(*Which*)**

expression Required. An expression that returns a **Document** object.

Which Required **Long**. The auto macro to run. Can be one of the following **WdAutoMacros** constants: **wdAutoClose**, **wdAutoExec**, **wdAutoExit**, **wdAutoNew**, or **wdAutoOpen**.

RunAutoMacro Method Example

This example runs the AutoOpen macro in the active document.

```
ActiveDocument.RunAutoMacro Which:=wdAutoOpen
```

Hyperlinks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHyperlinksC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHyperlinksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHyperlinksA "}

Returns a **Hyperlinks** collection that represents all the hyperlinks in the specified document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Hyperlinks Property Example

This example displays the target address of the second hyperlink in Home.doc.

```
If Documents("Home.doc").Hyperlinks.Count >= 2 Then
    MsgBox Documents("Home.doc").Hyperlinks(2).Name
End If
```

This example jumps to the address of the first hyperlink in the selection.

```
If Selection.Hyperlinks.Count >= 1 Then
    Selection.Hyperlinks(1).Follow
End If
```

This example displays the name of every hyperlink in the active document that includes the word "Microsoft" in its address.

```
For Each aHyperlink In ActiveDocument.Hyperlinks
    If InStr(LCase(aHyperlink.Address), "Microsoft") <> 0 Then
        MsgBox aHyperlink.Name
    End If
Next aHyperlink
```

SavedBy Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSavedByC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSavedByX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSavedByA "}

Returns the name of the user who saved the specified version of the document. Read-only **String**.

SavedBy Property Example

This example displays the name of the user who saved the first version of the active document.

```
If ActiveDocument.Versions.Count >= 1 Then
    MsgBox ActiveDocument.Versions(1).SavedBy
End If
```

This example saves a version of the document with a comment and then displays the user name.

```
ActiveDocument.Versions.Save Comment:="Added client information"
last = ActiveDocument.Versions.Count
MsgBox ActiveDocument.Versions(last).SavedBy
```

Compatibility Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCompatibilityC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCompatibilityX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCompatibilityA "}

True if the compatibility option specified by the ***Type*** argument is enabled. Compatibility options affect how a document is displayed in Word. These options are listed on the **Compatibility** tab in the **Options** dialog box (**Tools** menu). Read/write **Boolean**.

Syntax

expression.**Compatibility(*Type*)**

expression Required. An expression that returns a **Document** object.

Type Required **Long**. Can be one of the following **WdCompatibility** constants.

Constant	Description
wdConvMailMergeEsc	Word 97 correctly interprets characters preceded by backslashes (\) in Word version 2.x mail-merge data sources. For example, \" is interpreted as ".
wdExactOnTop	Word 97 places text at the top of the space allocated for the line when using exact line spacing.
wdLineWrapLikeWord6	Not used in U.S. English Word.
wdMWSmallCaps	Word 97 applies small capital letter ("small caps") formatting as in Word version 5.x for the Macintosh, which produces slightly larger small caps.
wdNoColumnBalance	Word 97 doesn't balance text columns above continuous section breaks.
wdNoExtraLineSpacing	Word 97 handles line spacing as in WordPerfect version 5.x. This argument is available only in Word version 7.0.
wdNoLeading	Word 97 displays lines of text without leading as in Word version 5.x for the Macintosh.
wdNoSpaceForUL	Word 97 doesn't add extra space for underlines.This argument is available only in Word version 7.0.
wdNoSpaceRaiseLower	Word 97 doesn't add extra line spacing for raised and lowered characters.
wdNoTabHangIndent	Word 97 doesn't automatically add a tab stop to a paragraph formatted with a hanging indent.
wdOrigWordTableRules	Word 97 combines table borders as in Word version 5.x for the Macintosh.
wdPrintBodyTextBeforeHeader	Word 97 prints the main text layer before the header and footer layer (the reverse of the default order). This allows Word to process PostScript codes in the main text layer the same as in Word version 5.x for the Macintosh.
wdPrintColBlack	Word 97 prints colors as black on printers that don't support color.
wdShowBreaksInFrames	Word 97 displays manual ("hard") page or column breaks in any frames that contain them.

wdSpacingInWholePoints	Word 97 rounds character-spacing measurements up or down to the nearest whole number.
wdSubFontBySize	Word 97 substitutes fonts based on the font size in WordPerfect version 6.0 documents.This argument is available only in Word version 7.0.
wdSuppressBottomSpacing	Word 97 removes extra line spacing at the bottom of the page.
wdSuppressSpBfAfterPgBrk	Word 97 removes space before or after hard page breaks and column breaks.
wdSuppressTopSpacing	Word 97 removes extra line spacing at the top of the page.
wdSuppressTopSpacingMac5	Word 97 handles extra line spacing at the top of the page the same as in Word version 5.x for the Macintosh.
wdSwapBordersFacingPages	Word 97 prints a left paragraph border (not a box) on the right side on odd-numbered pages if either the **Different odd and even** check box (**Layout** tab) or the **Mirror margins** check box (**Margins** tab) is selected in the **Page Setup** dialog box (**File** menu).
wdTransparentMetafiles	Word 97 doesn't "blank" the area behind metafile pictures.
wdTruncateFontHeight	Word 97 rounds the font size up or down as in WordPerfect version 6.x for Windows.This argument is available only in Word version 7.0.
wdUsePrinterMetrics	Word 97 uses printer metrics to lay out the document.
wdWPJustification	Word 97 lays out fully justified text the same as in WordPerfect version 6.x.
wdWPSpaceWidth	Word 97 sets the default width of spaces between words to be the same as in WordPerfect version 5.x for Windows and WordPerfect version 6.0 for DOS.
wdWrapTrailSpaces	Word 97 wraps spaces at the end of lines to the next line.
wdWW6BorderRules	Word 97 handles left and right borders of text lines affected by frames the same as in Word version 6.x.

Note The constants **wdLeaveBackslashAlone**, **wdExpandShiftReturn**, **wdDontULTrailSpace**, and **wdDontBalanceSingleByteDoubleByteWidth** are not used in U.S. English Word.

Compatibility Property Example

This example enables the **Suppress Space Before after a hard page or column break** option on the **Compatibility** tab in the **Options** dialog box (**Tools** menu) for the active document.

```
ActiveDocument.Compatibility(wdSuppressSpBfAfterPgBrk) = True
```

This example toggles the **Don't add automatic tab stop for hanging indent** option on or off.

```
ActiveDocument.Compatibility(wdNoTabHangIndent) = Not _
    ActiveDocument.Compatibility(wdNoTabHangIndent)
```

MakeCompatibilityDefault Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMakeCompatibilityDefaultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMakeCompatibilityDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMakeCompatibilityDefaultA "}

Sets the compatibility options on the **Compatibility** tab in the **Options** dialog box (**Tools** menu) as the default settings for new documents.

Syntax

expression.**MakeCompatibilityDefault**

expression Required. An expression that returns a **Document** object.

MakeCompatibilityDefault Method Example

This example sets a few compatibility options for the active document and then makes the current compatibility options the default settings.

```
With ActiveDocument
    .Compatibility(wdSuppressSpBfAfterPgBrk) = True
    .Compatibility(wdExpandShiftReturn) = True
    .Compatibility(wdUsePrinterMetrics) = True
    .Compatibility(wdNoLeading) = False
    .MakeCompatibilityDefault
End With
```

Address Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAddressA "}

Hyperlink object: Returns the address (for example, a file name or URL) of the specified hyperlink. Read-only **String**.

Envelope object: Returns the envelope delivery address as a **Range** object. Read-only.

Note An error occurs if you use this property when there hasn't been an envelope added to the specified document.

Address Property Example

This example adds a hyperlink to the selection in the active document, sets the address, and then displays the address in a message box.

```
Set aHLink = ActiveDocument.Hyperlinks.Add(Anchor:=Selection.Range, _
    Address:="http://forms")
MsgBox "The hyperlink goes to " & aHLink.Address
```

If the active document includes hyperlinks, this example inserts a list of the hyperlink destinations at the end of the document.

```
Set myRange = ActiveDocument.Range(Start:=ActiveDocument.Content.End - 1)
Count = 0
For Each aHyperlink In ActiveDocument.Hyperlinks
    Count = Count + 1
    With myRange
        .InsertAfter "Hyperlink #" & Count & vbTab
        .InsertAfter aHyperlink.Address
        .InsertParagraphAfter
    End With
Next aHyperlink
```

This example displays the delivery address if an envelope has been added to the document; otherwise, it displays a message.

```
On Error GoTo errhandler
addr = ActiveDocument.Envelope.Address.Text
MsgBox Prompt:=addr, Title:="Delivery Address"
errhandler:
If Err = 5852 Then MsgBox "Insert an envelope into the document"
```

PrintFractionalWidths Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintFractionalWidthsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintFractionalWidthsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintFractionalWidthsA"}

True if the specified document is formatted to use fractional point spacing to display and print characters on the Macintosh. Read/write **Boolean**.

Note In Windows, this property always returns **False**.

PrintFractionalWidths Property Example

This example enables fractional widths and then prints the active document.

```
With ActiveDocument
    .PrintFractionalWidths = True
    .PrintOut
End With
```

This example returns the fractional widths setting from the **Print** tab in the **Options** dialog box (**Tools** menu).

```
state = Documents("Report.doc").PrintFractionalWidths
```

WritePassword Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWritePasswordC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWritePasswordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWritePasswordA"}

Sets a password for saving changes to the specified document. Write-only **String**.

WritePassword Property Example

If the active document isn't already protected against saving changes, this example sets "secret" as the write password for the document.

```
Set myDoc = ActiveDocument
If myDoc.WriteReserved = False Then myDoc.WritePassword = "secret"
```

WriteReserved Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWriteReservedC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWriteReservedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWriteReservedA"}

True if the specified document is protected with a write password. Read-only **Boolean**.

WriteReserved Property Example

This example displays a message if the active document has a write password.

```
If ActiveDocument.WriteReserved = True Then
    MsgBox "Changes cannot be made to this document."
End If
```

HasPassword Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasPasswordC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasPasswordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasPasswordA"}

True if a password is required to open the specified document. Read-only **Boolean**.

HasPassword Property Example

This example sets the password "kittycat" for the active document and then displays a confirmation message.

```
ActiveDocument.Password = "kittycat"
If ActiveDocument.HasPassword = True Then MsgBox "The password is set."
```

Password Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPasswordC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPasswordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPasswordA"}

Sets a password that must be supplied to open the specified document. Write-only **String**.

Password Property Example

This example opens Earnings.doc, sets a password for it, and then closes the document.

```
Set myDoc = Documents.Open(FileName:="C:\My Documents\Earnings.doc")
myDoc.Password = "why"
myDoc.Close
```

ToggleFormsDesign Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproToggleFormsDesignC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproToggleFormsDesignX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproToggleFormsDesignA"}

Toggles form design mode on or off. When Word is in form design mode, the **Control Toolbox** toolbar is displayed. You can use this toolbar to insert ActiveX controls such as command buttons, scroll bars, and option buttons. In form design mode, event procedures don't run, and when you click an embedded control, the control's sizing handles appear.

Syntax

expression.**ToggleFormsDesign**

expression Required. An expression that returns a **Document** object.

ToggleFormsDesign Method Example

This example switches to form design mode if the **Control Toolbox** toolbar isn't currently displayed.

```
If CommandBars("Control Toolbox").Visible = False Then
    ActiveDocument.ToggleFormsDesign
End If
```

AutoVersion Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoVersionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoVersionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoVersionA"}

Returns or sets the state of the option for automatically saving document versions. Can be either of the following **WdAutoVersions** constants: **wdAutoVersionOff** or **wdAutoVersionOnClose**. Read/write **Long**.

Note When the **AutoVersion** property is set to **wdAutoVersionOnClose**, a document version is automatically saved when the document is closed.

AutoVersion Property Example

This example disables the option to save a document version automatically when the active document is closed.

```
ActiveDocument.Versions.AutoVersion = wdAutoVersionOff
```

This example displays a message in the status bar if the option to save a document version automatically is active for Report.doc.

```
If Documents("Report.doc").Versions.AutoVersion = wdAutoVersionOnClose Then
    StatusBar = "A version will be automatically saved"
End If
```

UndoClear Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUndoClearC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUndoClearX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUndoClearA"}

Clears the list of actions that can be undone for the specified document. Corresponds to the list of items that appears when you click the arrow beside the **Undo** button on the **Standard** toolbar.

Note Include this method at the end of a macro to keep Visual Basic actions from appearing in the **Undo** box (for example, "VBA-Selection.InsertAfter").

Syntax

expression.**UndoClear**

expression Required. An expression that returns a **Document** object.

UndoClear Method Example

This example clears the list of actions that can be undone for the active document.

```
ActiveDocument.UndoClear
```

AddToFavorites Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddToFavoritesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddToFavoritesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddToFavoritesA "}

Creates a shortcut to the document or hyperlink and adds it to the **Favorites** folder.

Syntax

expression.**AddToFavorites**

expression Required. An expression that returns a **Document** or **Hyperlink** object.

AddToFavorites Method Example

This example creates a shortcut for each hyperlink in the active document and adds it to the **Favorites** folder.

```
For Each myHyperlink In ActiveDocument.Hyperlinks
    myHyperlink.AddToFavorites
Next myHyperlink
```

This example creates a shortcut to Sales.doc and adds it to the **Favorites** folder. If Sales.doc isn't currently open, this example opens it from the C:\Documents folder.

```
For Each doc in Documents
    If LCase(doc.Name) = "sales.doc" Then isOpen = True
Next doc
If isOpen <> True Then Documents.Open _
    FileName:="C:\Documents\Sales.doc"
Documents("Sales.doc").AddToFavorites
```

Open Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenA "}

Syntax 1: Opens the specified document and adds it to the **Documents** collection. Returns a **Document** object.

Syntax 2: Opens the specified object. When applied to a **Subdocument** or **RecentFile** object, Syntax 2 returns a **Document** object.

Syntax 1

expression.**Open(*FileName, ConfirmConversions, ReadOnly, AddToRecentFiles, PasswordDocument, PasswordTemplate, Revert, WritePasswordDocument, WritePasswordTemplate, Format*)**

Syntax 2

expression.**Open**

expression Syntax 1: Required. An expression that returns a **Documents** object.

Syntax 2: Required. An expression that returns an **OLEFormat**, **RecentFile**, **Subdocument**, or **Version** object.

FileName Required **Variant**. The name of the document (paths are accepted). In Windows, you can specify multiple files by separating the file names with spaces; on the Macintosh, you can specify only one file.

ConfirmConversions Optional **Variant**. **True** to display the **Convert File** dialog box if the file isn't in Word format.

ReadOnly Optional **Variant**. **True** to open the document as read-only.

AddToRecentFiles Optional **Variant**. **True** to add the file name to the list of recently used files at the bottom of the **File** menu.

PasswordDocument Optional **Variant**. The password for opening the document.

PasswordTemplate Optional **Variant**. The password for opening the template.

Revert Optional **Variant**. Controls what happens if ***Name*** is the file name of an open document. **True** to discard any unsaved changes to the open document and reopen the file. **False** to activate the open document.

WritePasswordDocument Optional **Variant**. The password for saving changes to the document.

WritePasswordTemplate Optional **Variant**. The password for saving changes to the template.

Format Optional **Variant**. The file converter to be used to open the document. Can be one of the following **WdOpenFormat** constants: **wdOpenFormatAuto**, **wdOpenFormatDocument**, **wdOpenFormatRTF**, **wdOpenFormatTemplate**, **wdOpenFormatText**, or **wdOpenFormatUnicodeText**. The default value is **wdOpenFormatAuto**.

To specify an external file format, apply the **OpenFormat** property to a **FileConverter** object to determine the value to use with this argument.

Open Method Example

This example opens MyDoc.doc as a read-only document.

```
Documents.Open FileName:="C:\MyFiles\MyDoc.doc", ReadOnly:=True
```

This example opens each document in the **RecentFiles** collection.

```
For Each rFile In RecentFiles
    rFile.Open
Next rFile
```

This example opens the the most recent version of Report.doc.

```
Set myDoc = Documents("Report.doc")
myDoc.Versions(myDoc.Versions.Count).Open
```

This example opens Test.wp using the WordPerfect 6.x file converter.

```
num = FileConverters("WordPerfect6x").OpenFormat
Documents.Open FileName:="C:\Test.wp", Format:=num
```

This Macintosh example opens the document named Hello.

```
Documents.Open FileName:="Hard Drive:Files:Hello"
```

PageSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageSizeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageSizeA"}

Returns or sets the page size for the specified custom mailing label. Can be one of the following **WdCustomLabelPageSize** constants: **wdCustomLabelA4**, **wdCustomLabelA4LS**, **wdCustomLabelA5**, **wdCustomLabelA5LS**, **wdCustomLabelB5**, **wdCustomLabelFanfold**, **wdCustomLabelLetter**, **wdCustomLabelLetterLS**, or **wdCustomLabelMini**. Read/write **Long**.

PageSize Property Example

This example creates a new custom label named "Home Address" and then sets various properties for the label, including the page size.

```
Set myLabel = Application.MailingLabel _
    .CustomLabels.Add(Name:="Home Address", DotMatrix:=False)
With myLabel
    .Height = InchesToPoints(0.5)
    .HorizontalPitch = InchesToPoints(2.06)
    .NumberAcross = 4
    .NumberDown = 20
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.28)
    .TopMargin = InchesToPoints(0.5)
    .VerticalPitch = InchesToPoints(0.5)
    .Width = InchesToPoints(1.75)
End With
```

EnlargeFontsLessThan Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnlargeFontsLessThanC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnlargeFontsLessThanX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnlargeFontsLessThanA"}

Returns or sets the point size below which screen fonts are automatically scaled to the larger size. Read/write **Long**.

Note This property only affects the text as shown in online layout view. The point sizes that are displayed on the **Formatting** toolbar and used for printing aren't changed.

EnlargeFontsLessThan Property Example

This example switches the active window to online layout view and enlarges the fonts that are smaller than 10 points.

```
With ActiveWindow.View
    .Type = wdOnlineView
    .EnlargeFontsLessThan = 10
End With
```

This example switches the active window to online layout view and increases the current setting of the **Enlarge fonts less than** option on the **View** tab in the **Options** dialog box by 2 points.

```
With ActiveWindow.View
    .Type = wdOnlineView
    .EnlargeFontsLessThan = .EnlargeFontsLessThan + 2
End With
```

DefaultOpenFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultOpenFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultOpenFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultOpenFormatA"}

Returns or sets the default file converter used to open documents. Can be a number returned by the **OpenFormat** property, or one of the following **WdOpenFormat** constants: **wdOpenFormatAuto**, **wdOpenFormatDocument**, **wdOpenFormatRTF**, **wdOpenFormatTemplate**, **wdOpenFormatText**, or **wdOpenFormatUnicodeText**. Read/write **Long**.

Note Use the **Format** argument with the **Open** method to specify a file converter when you're opening a single document.

DefaultOpenFormat Property Example

This example sets the default converter for opening documents to the Word document format and then opens Forcast.doc.

```
Options.DefaultOpenFormat = wdOpenFormatDocument
Documents.Open FileName:="C:\Sales\Forcast.doc"
```

This example sets the default converter for opening documents to automatically determine the appropriate file converter to use when opening documents.

```
Options.DefaultOpenFormat = wdOpenFormatAuto
```

This example sets the default converter for opening documents to the WordPerfect 6.x format.

```
Options.DefaultOpenFormat = FileConverters("WordPerfect6x").OpenFormat
```

GetCrossReferenceItems Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGetCrossReferenceItemsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGetCrossReferenceItemsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGetCrossReferenceItemsA"}

Returns an array of items that can be cross-referenced based on the specified cross-reference type. The array corresponds to the items listed in the **For which** box in the **Cross-reference** dialog box (**Insert** menu).

Note An item returned by this method can be used as the ***ReferenceWhich*** argument for the **InsertCrossReference** method.

Syntax

expression.**GetCrossReferenceItems(*ReferenceType*)**

expression Required. An expression that returns a **Document** object.

ReferenceType Required **Variant**. The type of item you want to insert a cross-reference to. Can be one of the following **WdReferenceType** constants: **wdRefTypeBookmark**, **wdRefTypeEndnote**, **wdRefTypeFootnote**, **wdRefTypeHeading**, or **wdRefTypeNumberedItem**.

GetCrossReferenceItems Method Example

This example displays the name of the first bookmark in the active document that can be cross-referenced.

```
If ActiveDocument.Bookmarks.Count >= 1 Then
    myBookmarks = ActiveDocument.GetCrossReferenceItems(wdRefTypeBookmark)
    MsgBox myBookmarks(1)
End If
```

This example uses the **GetCrossReferenceItems** method to retrieve a list of headings that can be cross-referenced and then inserts a cross-reference to the page that includes the heading "Introduction."

```
myHeadings = ActiveDocument.GetCrossReferenceItems(wdRefTypeHeading)
For i = 1 To Ubound(myHeadings)
    If Instr(LCase$(myHeadings(i)), "introduction") Then
        Selection.InsertCrossReference ReferenceType:=wdRefTypeHeading, _
            ReferenceKind:=wdPageNumber, ReferenceItem:=i
        Selection.InsertParagraphAfter
    End If
Next i
```

SaveFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSaveFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSaveFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSaveFormatA"}

Returns the file format of the specified document or file converter. Can be a unique number that specifies an external file convertor, or one of the following **WdSaveFormat** constants: **wdFormatDocument**, **wdFormatDOSText**, **wdFormatDOSTextLineBreaks**, **wdFormatRTF**, **wdFormatTemplate**, **wdFormatText**, **wdFormatTextLineBreaks**, or **wdFormatUnicodeText**. Read-only **Long**.

SaveFormat Property Example

If the active document is a Rich Text Format (RTF) document, this example saves it as a Word document.

```
If ActiveDocument.SaveFormat = wdFormatRTF Then
    ActiveDocument.SaveAs FileFormat:=wdFormatDocument
End If
```

This example displays the converters that can be used to save documents. The message box displays the unique save format value and the format name.

```
For Each fc In FileConverters
    If fc.CanSave = True Then MsgBox fc.SaveFormat & vbCr & fc.FormatName
Next fc
```

This example saves the active document in the WordPerfect 5.1 or 5.2 secondary file format.

```
ActiveDocument.SaveAs FileFormat:=FileConverters("WrdPrfctDat").SaveFormat
```

UpdateSummaryProperties Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateSummaryPropertiesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateSummaryPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateSummaryPropertiesA"}

Updates the keyword and comment text in the **Properties** dialog box (**File** menu) to reflect the AutoSummary content for the specified document.

Syntax

expression.**UpdateSummaryProperties**

expression Required. An expression that returns a **Document** object.

UpdateSummaryProperties Method Example

This example highlights key points in the active document and updates the summary information in the **Properties** dialog box (**File** menu).

```
With ActiveDocument
    .AutoSummarize Length:=wd25Percent, Mode:=wdSummaryModeHighlight
    .UpdateSummaryProperties
End With
```

New Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNewA "}

Inserts an empty, 1-inch-square Word picture object surrounded by a border. This method returns the new graphic as an **InlineShape** object.

Syntax

expression.**New(*Range*)**

expression Required. An expression that returns an **InlineShapes** object.

Range Required **Range** object. The location of the new graphic.

New Method Example

This example inserts a new, empty picture in the active document and applies a shadow border around the picture.

```
Set myPic = ActiveDocument.InlineShapes.New(Range:=Selection.Range)
myPic.Borders.Shadow = True
ActiveWindow.View.ShowFieldCodes = False
```

Background Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackgroundC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackgroundX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackgroundA "}

Returns a **Shape** object that represents the background image for the specified document. Read-only.

Note Backgrounds are visible only in online layout view.

Background Property Example

This example sets the background color for online layout view to light gray for the active window.

```
ActiveWindow.View.Type = wdOnlineView
With ActiveDocument.Background.Fill
    .Visible = True
    .ForeColor.RGB = RGB(192, 192, 192)
End With
```

This example sets the background bitmap image of online layout view to Bubbles.bmp.

```
ActiveWindow.View.Type = wdOnlineView
ActiveDocument.Background.Fill.UserPicture PictureFile:="C:\Windows\
Bubbles.bmp"
```

Reload Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthReloadC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthReloadX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthReloadA "}

Reloads a cached document by resolving the hyperlink to the document and downloading it. This method is available only with cached documents (it cannot be used with FAT files).

Note If the document has been accessed before, this method ensures that the cached version of the specified document isn't used. If the specified document wasn't opened from the World Wide Web or a local intranet, an error occurs.

Syntax

expression.**Reload**

expression Required. An expression that returns a **Document** object.

Reload Method Example

This example opens and reloads the hyperlink to the address "main" on a local intranet.

```
With ActiveDocument
    .FollowHyperlink Address:="http://main
    .Reload
End With
```

Follow Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFollowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFollowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFollowA "}

Displays a cached document associated with the specified **Hyperlink** object, if it's already been downloaded. Otherwise, this method resolves the hyperlink, downloads the target document, and displays the document in the appropriate application.

Note If the hyperlink uses the file protocol, this method opens the document instead of downloading it.

Syntax

expression.**Follow(*NewWindow, AddHistory, ExtraInfo, Method, HeaderInfo*)**

expression Required. An expression that returns a **Hyperlink** object.

NewWindow Optional **Variant**. **True** to display the target document in a new window. The default value is **False**.

AddHistory Optional **Variant**. This argument is reserved for future use.

ExtraInfo Optional **Variant**. A string or byte array that specifies additional information for HTTP to use to resolve the hyperlink. For example, you can use ***ExtraInfo*** to specify the coordinates of an image map, the contents of a form, or a FAT file name. The string is either posted or appended, depending on the value of ***Method***. Use the **ExtraInfoRequired** property to determine whether extra information is required.

Method Optional **Variant**. Specifies the way additional information for HTTP is handled. Can be one of the following **MsoExtraInfoMethod** constants.

Constant	Description
msoMethodGet	***ExtraInfo*** is a string that's appended to the address.
msoMethodPost	***ExtraInfo*** is posted as a string or a byte array.

HeaderInfo Optional **Variant**. A string that specifies header information for the HTTP request. The defaut 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.

Follow Method Example

This example follows the first hyperlink in Home.doc.

```
Documents("Home.doc").HyperLinks(1).Follow
```

This example inserts a hyperlink to www.msn.com and then follows the hyperlink.

```
With Selection
    .Collapse Direction:=wdCollapseEnd
    .InsertAfter "MSN "
    .Previous
End With
Set myLink = ActiveDocument.Hyperlinks.Add(Address:="http://www.msn.com", _
    Anchor:=Selection.Range)
myLink.Follow NewWindow:=False, AddHistory:=True
```

FollowHyperlink Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFollowHyperlinkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFollowHyperlinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFollowHyperlinkA "}

Displays a cached document, if it's already been downloaded. Otherwise, this method resolves the hyperlink, downloads the target document, and displays the document in the appropriate application.

Note If the hyperlink uses the file protocol, this method opens the document instead of downloading it.

Syntax

expression.**FollowHyperlink(*Address, SubAddress, NewWindow, AddHistory, ExtraInfo, Method, HeaderInfo*)**

expression Required. An expression that returns a **Document** object.

Address Required **String**. The address of the target document.

SubAddress Optional **Variant**. The location within the target document. The default value is an empty string.

NewWindow Optional **Variant**. **True** to display the target location in a new window. The default value is **False**.

AddHistory Optional **Variant**. This argument is reserved for future use.

ExtraInfo Optional **Variant**. A string or a byte array that specifies additional information for HTTP to use to resolve the hyperlink. For example, you can use ***ExtraInfo*** to specify the coordinates of an image map, the contents of a form, or a FAT file name. The string is either posted or appended, depending on the value of ***Method***. Use the **ExtraInfoRequired** property to determine whether extra information is required.

Method Optional **Variant**. Specifies the way additional information for HTTP is handled. Can be one of the following **MsoExtraInfoMethod** constants.

Constant	Description
msoMethodGet	***ExtraInfo*** is a string that's appended to the address.
msoMethodPost	***ExtraInfo*** is posted as a string or a byte array.

HeaderInfo Optional **Variant**. A string that specifies header information for the HTTP request. The defaut 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.

FollowHyperlink Method Example

This example follows the specified URL address and displays the Microsoft home page in a new window.

```
ActiveDocument.FollowHyperlink Address:="http://www.Microsoft.com", _
    NewWindow:=True, AddHistory:=True
```

This example displays the HTML document named "Default.htm."

```
ActiveDocument.FollowHyperlink Address:="file:C:\Pages\Default.htm"
```

ExtraInfoRequired Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExtraInfoRequiredC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExtraInfoRequiredX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExtraInfoRequiredA"}

True if extra information is required to resolve the specified hyperlink. Read-only **Boolean**.

Note You can specify extra information by using the ***ExtraInfo*** argument with the **Follow** or **FollowHyperlink** method. For example, you can use ***ExtraInfo*** to specify the coordinates of an image map, the contents of a form, or a FAT file name.

ExtraInfoRequired Property Example

This example inserts a hyperlink to www.msn.com and then follows the hyperlink if extra information isn't required.

```
With Selection
    .Collapse Direction:=wdCollapseEnd
    .InsertAfter "MSN "
    .Previous
End With
Set myLink = ActiveDocument.Hyperlinks.Add(Address:="http://www.msn.com", _
    Anchor:=Selection.Range)
If myLink.ExtraInfoRequired = False Then
    myLink.Follow
End If
```

CodeName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCodeNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCodeNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCodeNameA"}

Returns the code name for the specified document. Read-only **String**.

Remarks

The code name is the name for the module that houses event macros for a document. The default name for the module is "ThisDocument"; you can view it in the **Project** window. For information about using events with the **Document** object, see Using Events with the Document Object.

CodeName Property Example

This example returns the name of the code window for the active document.

```
Msgbox ActiveDocument.CodeName
```

Container Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContainerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContainerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContainerA"}

Returns the object that represents the container application for the specified OLE object. Read-only.

Remarks

This property provides access to the specified document's container application if the document is embedded in another application as an OLE object.

The **Container** property also provides a pathway into the object model of the container application if a Word document is opened as an ActiveX document − for example, when a Word document is opened in Microsoft Office Binder or Internet Explorer 3.0.

Container Property Example

This example displays the name of the container application for the first shape in the active document. For the example to work, this shape must be an OLE object.

```
Msgbox ActiveDocument.Shapes(1).OLEFormat.Object.Container.Name
```

FormsDesign Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFormsDesignC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFormsDesignX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFormsDesignA"}

True if the specified document is in form design mode. Read-only **Boolean**.

Note This property will always return **False** when it's run from a macro in Word.

Remarks

When Word is in form design mode, the **Control Toolbox** is displayed. You can use this toolbar to insert ActiveX controls such as command buttons, scroll bars, and option buttons. In form design mode, event procedures don't run, and when you click an embedded control, the control's sizing handles appear.

FormsDesign Property Example

This example displays a message box that indicates whether the active document is in form design mode. This example will always return **False**.

```
Msgbox ActiveDocument.FormsDesign
```

PresentIt Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPresentItC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPresentItX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPresentItA"}

Opens PowerPoint with the specified Word document loaded.

Syntax

expression.**PresentIt**

expression Required. An expression that returns a **Document** object.

PresentIt Method Example

This example sends a copy of the document named "MyPresentation.doc" to PowerPoint.

```
Documents("MyPresentation.doc").PresentIt
```

SendFax Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSendFaxC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSendFaxX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSendFaxA"}

Application object: Starts the Fax Wizard.

Document object: Sends the specified document as a fax, without any user interaction.

Syntax 1

expression.**SendFax**

Syntax 2

expression.**SendFax(*Address, Subject*)**

expression Required. An expression that returns an **Application** object (Syntax 1) or a Document object (Syntax 2).

Address Required **String**. The recipient's fax number.

Subject Optional **Variant**. The text for the subject line. The character limit is 255.

SendFax Method Example

This example sends the active document as a fax.

```
ActiveDocument.SendFax Address:="12065551234", _
   Subject:="Important Fax"
```

This example starts the Fax Wizard.

```
Application.SendFax
```

VBProject Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVBProjectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVBProjectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVBProjectA"}

Returns the **VBProject** object for the specified template or document. Read-only.

Remarks

Use this property to gain access to code modules and user forms.

To view the **VBProject** object in the object browser, you must select the **Microsoft Visual Basic for Applications Extensibility** check box in the **References** dialog box (**Tools** menu) in the Visual Basic Editor.

VBProject Property Example

This example displays the name of the Visual Basic project for the Normal template.

```
Set normProj = NormalTemplate.VBProject
MsgBox normProj.Name
```

This example displays the name of the Visual Basic project for the active document.

```
Set currProj = ActiveDocument.VBProject
MsgBox currProj.Name
```

This example adds a standard code module to the active document and renames it "MyModule."

```
Set newModule =
ActiveDocument.VBProject.VBComponents.Add(vbext_ct_StdModule)
NewModule.Name = "MyModule"
```

ViewCode Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthViewCodeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthViewCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthViewCodeA"}

Displays the code window for the selected ActiveX control in the specified document.

Note This method is available only from outside of Word.

Syntax

expression.**ViewCode**

expression Required. An expression that returns a **Document** object.

ViewPropertyBrowser Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthViewPropertyBrowserC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthViewPropertyBrowserX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthViewPropertyBrowserA"}

Displays the property window for the selected ActiveX control in the specified document.

Note This method is available only from outside of Word.

Syntax

expression.**ViewPropertyBrowser**

expression Required. An expression that returns a **Document** object.

See also

ViewCode method

LanguageSpecific Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguageSpecificC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguageSpecificX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguageSpecificA"}

True if the custom dictionary is to be used only with text formatted for a specific language. Read/write **Boolean**.

LanguageSpecific Property Example

This example checks to see whether any custom dictionaries are language specific. If any of them are, the example removes them from the list of active custom dictionaries.

```
For each d in CustomDictionaries
   If d.LanguageSpecific = True Then d.Delete
Next d
```

This example adds a custom dictionary that will check only text that's formatted as German.

```
Set myD = CustomDictionaries.Add("MyGerman.dic")
myD.LanguageSpecific = True
myD.LanguageID = wdGerman
```

SpellingDictionaryType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpellingDictionaryTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpellingDictionaryTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpellingDictionaryTypeA"}

Returns or sets the proofing tool type. Can be one of the following **WdDictionaryType** constants: **wdGrammar**, **wdHyphenation**, **wdSpelling**, **wdSpellingComplete**, **wdSpellingCustom**, **wdSpellingLegal**, **wdSpellingMedical**, or **wdThesaurus**. Read/write **Long**.

Remarks

You can use this property to change the active spelling dictionary to one of the available add-on dictionaries that work with Word. For example, there are legal, medical, and complete spelling dictionaries you can used instead of the standard dictionary.

SpellingDictionaryType Property Example

This example returns the type of spelling dictionary used for U.S. English.

```
myType = Languages(wdEnglishUS).SpellingDictionaryType
```

This example makes the legal dictionary the active spelling dictionary.

```
Languages(wdEnglishUS).SpellingDictionaryType = wdSpellingLegal
```

DefaultLabelName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultLabelNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultLabelNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultLabelNameA"}

Returns or sets the name for the default mailing label. Read/write **String**.

Note To find the string for the specified built-in label, select the label in the **Label Options** dialog box (**Tools** menu, **Envelopes and Labels** dialog box, **Labels** tab, **Options** button). Then click **Details** and lookat the **Label name** box, which contains the correct string to use for this property. To set a custom label as the default mailing label, use the label name that appears in the **Details** dialog box, or use the **Name** property with a **CustomLabel** object.

DefaultLabelName Property Example

This example returns the name of the current default mailing label.

```
Msgbox Application.MailingLabel.DefaultLabelName
```

This example sets the Avery Standard, 5160 Address label as the default mailing label.

```
Application.MailingLabel.DefaultLabelName = "5160"
```

AppendToSpike Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAppendToSpikeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAppendToSpikeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAppendToSpikeA "}

Deletes the specified range and adds the contents of the range to the Spike (a built-in AutoText entry). This method returns the Spike as an **AutoTextEntry** object.

Syntax

expression.**AppendToSpike(*Range*)**

expression Required. An expression that returns an **AutoTextEntries** object.

Range Required **Range** object. The range that's deleted and appended to the Spike.

AppendToSpike Method Example

This example deletes the selection and adds its contents to the Spike in the Normal template.

```
If Len(Selection.Range.Text) > 1 Then
    NormalTemplate.AutoTextEntries.AppendToSpike _
        Range:=Selection.Range
End If
```

This example clears the Spike and adds the first and third words in the active document to the Spike in the attached template. The contents of the Spike are then inserted at the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
Set myTemplate = ActiveDocument.AttachedTemplate
For Each entry In myTemplate.AutoTextEntries
    If entry.Name = "Spike" Then entry.Delete
Next entry
With myTemplate.AutoTextEntries
    .AppendToSpike Range:=ActiveDocument.Words(3)
    .AppendToSpike Range:=ActiveDocument.Words(1)
    .Item("Spike").Insert Where:=Selection.Range
End With
```

AttachedTemplate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAttachedTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAttachedTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAttachedTemplateA "}

Returns a **Template** object that represents the template attached to the specified document. To set this property, specify either the name of the template or an expression that returns a **Template** object. Read/write **Variant**.

AttachedTemplate Property Example

This example displays the name and path of the template attached to the active document.

```
Set myTemplate = ActiveDocument.AttachedTemplate
MsgBox myTemplate.Path & Application.PathSeparator & _
    myTemplate.Name
```

This example inserts the contents of the Spike (a built-in AutoText entry) at the beginning of document one.

```
Set myRange = Documents(1).Range(0, 0)
Documents(1).AttachedTemplate.AutoTextEntries("Spike").Insert myRange
```

This example attaches the template "Letter.dot" to the active document.

```
ActiveDocument.AttachedTemplate = "C:\Templates\Letter.dot"
```

AutoTextEntries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoTextEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoTextEntriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoTextEntriesA "}

Returns an **AutoTextEntries** collection that represents all the AutoText entries in the specified template. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

AutoTextEntries Property Example

This example deletes the AutoText entry named "Hello" if the entry exists in the attached template.

```
For Each entry In ActiveDocument.AttachedTemplate.AutoTextEntries
    If entry.Name = "Hello" Then entry.Delete
Next entry
```

This example adds an AutoText entry named "Temp" to the Normal template. The contents of the AutoText entry (the first word in the document) are then displayed in a message box.

```
Set myEntry = NormalTemplate.AutoTextEntries.Add(Name:="Temp", _
    Range:=ActiveDocument.Words(1))
MsgBox myEntry.Value
```

This example stores the contents of the selection as an AutoText entry named "Address" in the attached template.

```
If Len(Selection.Text) > 1 Then
    ActiveDocument.AttachedTemplate.AutoTextEntries.Add _
        Range:=Selection.Range, Name:="Address"
End If
```

OpenAsDocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenAsDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenAsDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenAsDocumentA "}

Opens the specified template as a document and returns a **Document** object.

Note Opening a template as a document allows the user to edit the contents of the template. This may be necessary if a property or method (the **Styles** property, for example) isn't available from the **Template** object.

Syntax

expression.**OpenAsDocument()**

expression Required. An expression that returns a **Template** object.

OpenAsDocument Method Example

This example opens the template attached to the active document, displays a message box if the template contains anything more than a single paragraph mark, and then closes the template.

```
Set aDoc = ActiveDocument.AttachedTemplate.OpenAsDocument
If aDoc.Content.Text <> Chr(13) Then
    MsgBox "Template is not empty"
Else
    MsgBox "Template is empty"
End If
aDoc.Close SaveChanges:=wdDoNotSaveChanges
```

This example saves a copy of the Normal template as "Backup.dot."

```
Set aDoc = NormalTemplate.OpenAsDocument
With aDoc
    .SaveAs FileName:="Backup.dot"
    .Close SaveChanges:=wdDoNotSaveChanges
End With
```

This example changes the formatting of the Heading 1 style in the template attached to the active document. The **UpdateStyles** method updates the styles in the active document.

```
Set aDoc = ActiveDocument.AttachedTemplate.OpenAsDocument
With aDoc.Styles(wdStyleHeading1).Font
    .Name = "Arial"
    .Size = 16
    .Bold = False
End With
aDoc.Close SaveChanges:=wdSaveChanges
ActiveDocument.UpdateStyles
```

Templates Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTemplatesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTemplatesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTemplatesA "}

Returns a **Templates** collection that represents all the available templates – global templates as well as those attached to open documents. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Templates Property Example

This example displays the name of each template in the **Templates** collection.

```
Count = 1
For Each aTemplate In Templates
    MsgBox aTemplate.Name & " is template number " & Count
    Count = Count + 1
Next aTemplate
```

In this example, if template one is a global template, its path is stored in `thePath`. The **ChDir** statement is used to make the folder with the path stored in `thePath` the current folder. When this change is made, the **Open** dialog box is displayed.

```
If Templates(1).Type = wdGlobalTemplate Then
    thePath = Templates(1).Path
    If thePath <> "" Then ChDir thePath
    Dialogs(wdDialogFileOpen).Show
End If
```

NormalTemplate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNormalTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNormalTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNormalTemplateA "}

Returns a **Template** object that represents the Normal template. Read-only.

NormalTemplate Property Example

This example inserts the AutoText entry named "Test" from the Normal template, if this entry is contained in the **AutoTextEntries** collection.

```
For Each entry In NormalTemplate.AutoTextEntries
    If entry.Name = "Test" Then entry.Insert Where:=Selection.Range
Next entry
```

This example saves the Normal template if changes have been made to it.

```
If NormalTemplate.Saved = False Then NormalTemplate.Save
```

Duplicate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDuplicateC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDuplicateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDuplicateA"}

Font object: Returns a **Font** object that represents the character formatting of the specified font. Read-only.

LetterContent object: Returns a **LetterContent** object that represents the contents of the specified letter created by the Letter Wizard. Read-only.

ParagraphFormat object: Returns a **ParagraphFormat** object that represents the paragraph formatting of the specified paragraph. Read-only.

Range object: Returns a **Range** object that represents all the properties of the specified range. Read-only.

TextRetrievalMode object: Returns a **TextRetrievalMode** object that represents options related to retrieving text from the specified **Range** object. Read-only.

Remarks

You can use the **Duplicate** property to pick up the settings of all the properties of a duplicated **Font**, **LetterContent**, or **ParagraphFormat** object. You can assign the object returned by the **Duplicate** property to another object of the same type to apply those settings all at once. Before assigning the duplicate object to another object, you can change any of the properties of the duplicate object without affecting the original.

By duplicating a **Range** object, you can change the starting or ending character position of the duplicate range without changing the original range.

Duplicate Property Example

This example duplicates the **Range** object assigned to the variable `myRange`. The example collapses the duplicate range to its end point, expands it by one character, and makes this character uppercase. The example then applies italic formatting to the original **Range** object (`myRange`).

```
Set myRange = Selection.Range
With myRange.Duplicate
    .Collapse Direction:=wdCollapseEnd
    .Expand Unit:=wdCharacter
    .Case = wdUpperCase
End With
myRange.Font.Italic = True
```

This example sets the variable `MyDupFont` to the character formatting of the selection, removes bold formatting from `MyDupFont`, and adds italic formatting to it instead. The example also creates a new document, inserts text into it, and then applies the formatting stored in `MyDupFont` to the text.

```
Set myDupFont = Selection.Font.Duplicate
With myDupFont
    .Bold = False
    .Italic = True
End With
Documents.Add
Selection.InsertAfter "This is some text."
Selection.Font = myDupFont
```

This example duplicates the paragraph formatting of the first paragraph in the active document and stores the formatting in the variable `myDup`, and then it changes the left indent for `myDup` to 1 inch. The example also creates a new document, inserts text into it, and then applies the paragraph formatting stored in `myDup` to the text.

```
ActiveDocument.Range(Start:=0, End:=0).InsertAfter "Paragraph Number 1"
Set myDup = ActiveDocument.Paragraphs(1).Format.Duplicate
myDup.LeftIndent = InchesToPoints(1)
Documents.Add
Selection.InsertAfter "This is a new paragraph."
Selection.Paragraphs.Format = myDup
```

Document, **HeaderFooter**, **HyperLink**

Adjustments, **Range**, **CalloutFormat**, **Frame**, **InlineShape**, **ShapeRange**, **FillFormat**, **Shape**
GroupShapes, **Hyperlink**, **LineFormat**, **LinkFormat**, **OLEFormat**, **ShapeNodes**, **PictureFormat**,
ShadowFormat, **TextEffectFormat**, **TextFrame**, **ThreeDFormat**, **WrapFormat**

Shape Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woobjShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapeP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapeM"}

Multiple Objects

Shapes (Shape)

Multiple Objects

Represents an object in the drawing layer, such as an AutoShape, freeform, OLE object, ActiveX control, or picture. The **Shape** object is a member of the **Shapes** collection, which includes all the shapes in the main story of a document or in all the headers and footers of a document.

A shape is always attached to an anchoring range. You can position the shape anywhere on the page that contains the anchor.

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.

Using the Shape Object

This section describes how to:

- Return an existing shape on a document, indexed by name or number.
- Return a shape or shapes within a selection.
- Return a newly created shape.
- Return a single shape from within a group.
- Return a newly formed group of shapes.

Returning an existing shape on a document

Use **Shapes**(*index*), where *index* is the name or the index number, to return a single **Shape** object. The following example horizontally flips shape one on the active document.

```
ActiveDocument.Shapes(1).Flip msoFlipHorizontal
```

The following example horizontally flips the shape named "Rectangle 1" on the active document.

```
ActiveDocument.Shapes("Rectangle 1").Flip msoFlipHorizontal
```

Each shape is assigned a default name when it is created. For example, if you add three different shapes to a document, they might be named "Rectangle 2," "TextBox 3," and "Oval 4." To give a shape a more meaningful name, set the **Name** property.

Returning a Shape or Shapes Within a Selection

Use **Selection.ShapeRange**(*index*), where *index* is the name or the index number, to return a **Shape** object that represents a shape within a selection. The following example sets the fill for the first shape

in the selection, assuming that the selection contains at least one shape.

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

The following example sets the fill for all the shapes in the selection, assuming that the selection contains at least one shape.

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

Returning a Newly Created Shape

To add a **Shape** object to the collection of shapes for the specified document and return a **Shape** object that represents the newly created shape, use one of the following methods of the **Shapes** collection: **AddCallout**, **AddCurve**, **AddLabel**, **AddLine**, **AddOleControl**, **AddOleObject**, **AddPolyline**, **AddShape**, **AddTextbox**, **AddTextEffect**, or **BuildFreeForm**. The following example adds a rectangle to the active document.

```
ActiveDocument.Shapes.AddShape msoShapeRectangle, 50, 50, 100, 200
```

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.

Anchoring and Positioning a Shape

Every **Shape** object is anchored to a range of text. A shape is anchored to the beginning of the first paragraph that contains the anchoring range. The shape will always remain on the same page as its anchor.

You can view the anchor itself by setting the **ShowObjectAnchors** property to **True**. The shape's **Top** and **Left** properties determine its vertical and horizontal positions. The shape's **RelativeHorizontalPosition** and **RelativeVerticalPosition** properties determine whether the position is measured from the anchoring paragraph, the column that contains the anchoring paragraph, the margin, or the edge of the page.

If the **LockAnchor** property for the shape is set to **True**, you cannot drag the anchor from its position on the page.

Formatting a Shape

Use the **Fill** property to return the **FillFormat** object, which contains all the properties and methods for formatting the fill of a closed shape. The **Shadow** property returns the **ShadowFormat** object, which you use to format a shadow. Use the **Line** property to return the **LineFormat** object, which contains properties and methods for formatting lines and arrows. The **TextEffect** property returns the **TextEffectFormat** object, which you use to format WordArt. The **Callout** property returns the **CalloutFormat** object, which you use to format line callouts. The **WrapFormat** property returns the **WrapFormat** object, which you use to define how text wraps around shapes. The **ThreeD** property returns the **ThreeDFormat** object, which you use to create 3-D shapes. You can use the **PickUp** and **Apply** methods to transfer formatting from one shape to another.

Use the **SetShapesDefaultProperties** method for a **Shape** object to set the formatting for the default shape for the document. New shapes inherit many of their attributes from the default shape.

Other Important Shape Properties

Use the **Type** property to specify the type of shape: freeform, AutoShape, OLE object, callout, or linked picture, for instance. Use the **AutoShapeType** property to specify the type of AutoShape: oval, rectangle, or balloon, for instance.

Use the **Width** and **Height** properties to specify the size of the shape.

The **TextFrame** property returns the **TextFrame** object, which contains all the properties and methods for attaching text to shapes and linking the text between text frames.

Remarks

Shape objects are anchored to a range of text but are free-floating and can be positioned anywhere on the page. **InlineShape** objects are treated like characters and are positioned as characters within a line of text. You can use the **ConvertToInlineShape** method and the **ConvertToShape** method to convert shapes from one type to the other. You can convert only pictures, OLE objects, and ActiveX controls to inline shapes.

Range, **Selection**

Adjustments, **Range**, **CalloutFormat**, **Frame**, **InlineShape**, **ShapeRange**, **FillFormat**, **Shape**
GroupShapes, **Hyperlink**, **LineFormat**, **ShapeNodes**, **PictureFormat**, **ShadowFormat**,
TextEffectFormat, **TextFrame**, **ThreeDFormat**, **WrapFormat**

ShapeRange Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapeRangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woobjShapeRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapeRangeP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapeRangeM"}

Multiple Objects

└

ShapeRange (Shape)

└

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.

Note Most operations that you can do with a **Shape** object, you can also do with a **ShapeRange** object that contains only one shape. Some operations, when performed on a **ShapeRange** object that contains more than one shape, will cause an error.

Using the ShapeRange Collection

This section describes how to:

- Return a set of shapes you specify by name or index number.
- Return a shaperange object within a selection or range.

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 the active document.

```
ActiveDocument.Shapes.Range(Array(1, 3)).Fill.Patterned
msoPatternHorizontalBrick
```

The following example selects the shapes named "Oval 4" and "Rectangle 5" on the active document.

```
ActiveDocument.Shapes.Range(Array("Oval 4", "Rectangle 5")).Select
```

Although you can use the **Range** method to return any number of shapes, 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 a ShapeRange Object Within a Selection or Range

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

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

This example selects all the shapes in the first section of the active document.

```
Set myRange = ActiveDocument.Sections(1).Range
myRange.ShapeRange.Select
```

Aligning, Distributing, and Grouping Shapes in a ShapeRange Object

Use the **Align**, **Distribute**, and **ZOrder** methods to position a set of shapes relative to each other or relative to the document.

Use the **Group**, **Regroup** , and **UnGroup** methods to create and work with a single shape formed from a shape range. The **GroupItems** property for a **Shape** object returns the **GroupShapes** object, which represents all the shapes that were grouped to form one shape.

Remarks

The recorder always uses the **ShapeRange** property when recording shapes.

A **ShapeRange** object doesn't include **InlineShape** objects.

Shapes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woobjShapesX":1} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapesP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapesM"}

Multiple Objects

└

Shapes (Shape)

└

Multiple Objects

A collection of **Shape** objects that represent all the shapes in a document or all the shapes in all the headers and footers in a document. 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.

Using the Shapes Collection

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

```
ActiveDocument.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 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 one of the following methods of the **Shapes** collection: **AddCallout**, **AddCurve**, **AddLabel**, **AddLine**, **AddOleControl**, **AddOleObject**, **AddPolyline**, **AddShape**, **AddTextbox**, **AddTextEffect**, or **BuildFreeForm** to add a shape to a document return a **Shape** object that represents the newly created shape The following example adds a rectangle to the active document.

```
ActiveDocument.Shapes.AddShape msoShapeRectangle, 50, 50, 100, 200
```

Use **Shapes**(*index*), where *index* is the name or the index number, to return a single **Shape** object. The following example horizontally flips shape one on the active document.

```
ActiveDocument.Shapes(1).Flip msoFlipHorizontal
```

This example horizontally flips the shape named "Rectangle 1" on the active document.

```
ActiveDocument.Shapes("Rectangle 1").Flip msoFlipHorizontal
```

Each shape is assigned a default name when it is created. For example, if you add three different shapes to a document, they might be named "Rectangle 2," "TextBox 3," and "Oval 4." To give a shape a more meaningful name, set the **Name** property.

Remarks

The **Shapes** collection does not include **InlineShape** objects. **InlineShape** objects are treated like characters and are positioned as characters within a line of text. **Shape** objects are anchored to a range of text but are free-floating and can be positioned anywhere on the page. You can use the **ConvertToInlineShape** method and the **ConvertToShape** method to convert shapes from one type

to the other. You can convert only pictures, OLE objects, and ActiveX controls to inline shapes.

The **Count** property for this collection in a document returns the number of items in the main story only. To count the shapes in all the headers and footers, use the **Shapes** collection with any **HeaderFooter** object.

Shape, **ShapeRange**

Range, **TextFrame**

TextFrame Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextFrameC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextFrameP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextFrameM"}

Represents the text frame in a **Shape** object. Contains the text in the text frame as well as the properties that control the margins and orientation of the text frame.

Using the TextFrame Object

Use the **TextFrame** property to return the **TextFrame** object for a shape. The **TextRange** property returns a **Range** object that represents the range of text inside the specified text frame. The following example adds text to the text frame of shape one in the active document.

```
ActiveDocument.Shapes(1).TextFrame.TextRange.Text = "My Text"
```

Note Some shapes don't support attached text (lines, freeforms, pictures, and OLE objects, for example). If you attempt to return or set properties that control text in a text frame for those objects, an error occurs.

Use the **HasText** property to determine whether the text frame contains text, as shown in the following example.

```
For Each s In ActiveDocument.Shapes
    With s.TextFrame
        If .HasText Then MsgBox .TextRange.Text
    End With
Next
```

Text frames can be linked together so that the text flows from the text frame of one shape into the text frame of another shape. Use the **Next** and **Previous** properties to link text frames. The following example creates a text box (a rectangle with a text frame) and adds some text to it. It then creates another text box and links the two text frames together so that the text flows from the first text frame into the second one.

```
Set myTB1 = ActiveDocument.Shapes.AddTextbox _
    (msoTextOrientationHorizontal, 72, 72, 72, 36)
myTB1.TextFrame.TextRange = "This is some text. This is some more text."
Set myTB2 = ActiveDocument.Shapes.AddTextbox _
    (msoTextOrientationHorizontal, 72, 144, 72, 36)
myTB1.TextFrame.Next = myTB2.TextFrame
```

Use the **ContainingRange** property to return a **Range** object that represents the entire story that flows between linked text frames. The following example checks the spelling in the text in TextBox 3 and any other text that's linked to TextBox 3.

```
Set myStory = ActiveDocument.Shapes("TextBox 3") _
    .TextFrame.ContainingRange
myStory.CheckSpelling
```

text frame

The area within a shape that can contain text.

Shape, **ShapeRange**

WrapFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjWrapFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjWrapFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjWrapFormatM"}

Multiple Objects

└

WrapFormat

Represents all the properties for wrapping text around a shape or shape range.

Using the WrapFormat Object

Use the **WrapFormat** property to return the **WrapFormat** object. The following example adds an oval to the active document and specifies that document text wrap around the left and right sides of the square that circumscribes the oval. There will be a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 36, 36, 100, 35)
With myOval.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

InlineShape Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjInlineShapeC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjInlineShapeP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjInlineShapeM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjInlineShapeE "}

Multiple Objects

└

InlineShapes (InlineShape)

└

Multiple Objects

Represents an object in the text layer of a document. An inline shape can only be a picture, an OLE object, or an ActiveX control. **InlineShape** objects are treated like characters and are positioned as characters within a line of text. The **InlineShape** object is a member of the **InlineShapes** collection. The **InlineShapes** collection contains all the shapes in a document, range, or selection.

Using the InlineShape Object

Use **InlineShapes**(*index*), where *index* is the index number, to return a single **InlineShape** object. Inline shapes don't have names. The following example activates the first inline shape in the active document.

```
ActiveDocument.InlineShapes(1).Activate
```

Remarks

Shape objects are anchored to a range of text but are free-floating and can be positioned anywhere on the page. You can use the **ConvertToInlineShape** method and the **ConvertToShape** method to convert shapes from one type to the other. You can convert only pictures, OLE objects, and ActiveX controls to inline shapes. Use the **Type** property to return the type of inline shape: picture, linked picture, embedded OLE object, linked OLE object, or ActiveX control.

When you open a document created in an earlier version of Word, pictures are converted to inline shapes.

InlineShapes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjInlineShapesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjInlineShapesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjInlineShapesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjInlineShapesE "}

Multiple Objects

└

InlineShapes (InlineShape)

└

Multiple Objects

A collection of **InlineShape** objects that represent all the inline shapes in a document, range, or selection.

Using the InlineShapes Collection

Use the **InlineShapes** property to return the **InlineShapes** collection. The following example converts each inline shape in the active document to a **Shape** object.

```
For Each iShape In ActiveDocument.InlineShapes
    iShape.ConvertToShape
Next iShape
```

Use the **New** method to create a new picture as an inline shape. You can use the **AddPicture** and **AddOLEObject** methods to add pictures or OLE objects and link them to a source file. Use the **AddOLEControl** method to add an ActiveX control.

Remarks

Shape objects are anchored to a range of text but are free-floating and can be positioned anywhere on the page. You can use the **ConvertToInlineShape** method and the **ConvertToShape** method to convert shapes from one type to the other. You can convert only pictures, OLE objects, and ActiveX controls to inline shapes.

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

When you open a document created in an earlier version of Word, pictures are converted to inline shapes.

Document, **Range**, **Selection**, **Field**

Borders, **Shape**, **Field**, **FillFormat**, **Hyperlink**, **LineFormat**, **LinkFormat**, **OLEFormat**, **PictureFormat**, **Range**

Adjustments Object

Shapes (Shape)

L

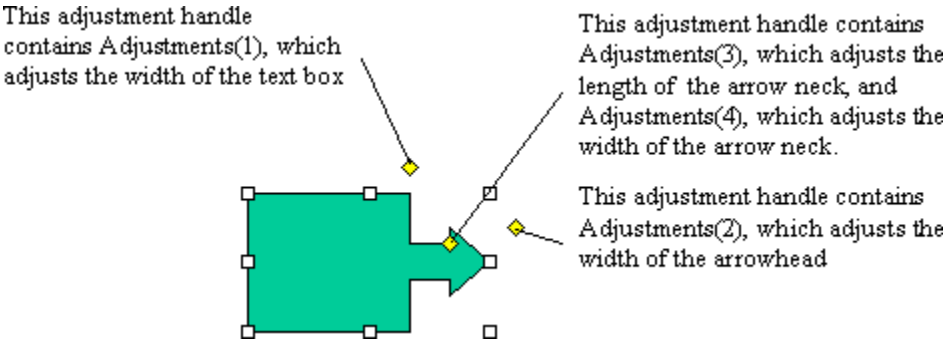
Adjustments

Contains a collection of adjustment values for the specified AutoShape or WordArt object. 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 or vertical)	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 the corresponding adjustment will be 0.5. For shapes such as callouts, where the values 0.0 and 1.0 represent the limits of the rectangle defined by the starting and ending points of the 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 the active document 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 rac = ActiveDocument.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
```

CalloutFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCalloutFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCalloutFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCalloutFormatM"}

Shapes (Shape)

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 the active document: 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; and the callout line will contain two segments. For this example to work, shape three must be a callout.

```
With ActiveDocument.Shapes(3).Callout
    .Accent = True
    .Angle = msoCalloutAngle30
    .Border = False
    .PresetDrop msoCalloutDropTop
    .Type = msoCalloutThree
End With
```

ColorFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjColorFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjColorFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjColorFormatM"}

Multiple Objects

ColorFormat

Represents the color of a one-color object or the foreground or background color of an object with a gradient or patterned fill. You can set colors to an explicit red-green-blue value by using the **RGB** property.

Using the ColorFormat Object

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

To return a ColorFormat object that represents this	Use this property	With this object
Background fill color (used in a shaded or patterned fill)	**BackColor**	**FillFormat**
Foreground fill color (or simply the fill color for a solid fill)	**ForeColor**	**FillFormat**
Background line color (used in a patterned line)	**BackColor**	**LineFormat**
Foreground line color (or just the line color for a solid line)	**ForeColor**	**LineFormat**
Shadow color	**ForeColor**	**ShadowFormat**
Color of the sides of an extruded object	**ExtrusionColor**	**ThreeDFormat**

Use the **RGB** property to set a color to an explicit red-green-blue value. The following example adds a rectangle to the active document and then sets the foreground color, background color, and gradient for the rectangle's fill.

```
With ActiveDocument.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
```

FillFormat, **LineFormat**, **ShadowFormat**, **ThreeDFormat**

FillFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFillFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFillFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFillFormatM"}

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 the active document and then sets the gradient and color for the rectangle's fill.

```
With ActiveDocument.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.

FreeformBuilder Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFreeformBuilderC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFreeformBuilderP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFreeformBuilderM"}

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

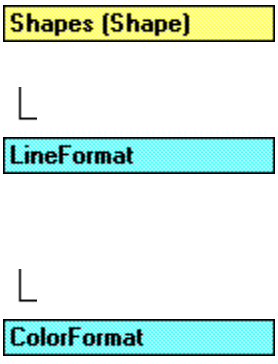
Using the FreeformBuilder Object

Use the **BuildFreeform** method to return a **FreeformBuilder** object. Use the **AddNodes** method to add nodes to the freefrom. 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 the active document.

```
With ActiveDocument.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
```

LineFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLineFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLineFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLineFormatM"}



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 a blue, dashed line to the active document. There's a short, narrow oval at the line's starting point and a long, wide triangle at its end point.

```
With ActiveDocument.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
```

PictureFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapeP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapeM"}

Shapes (Shape)

L

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 the active document 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.

```
With ActiveDocument.Shapes(1).PictureFormat
    .Brightness = 0.3
    .Contrast = 0.7
    .ColorType = msoPictureGrayScale
    .CropBottom = 18
End With
```

ShadowFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShadowFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShadowFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShadowFormatM"}

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 the active document. The semitransparent, blue shadow is offset 5 points to the right of the rectangle and 3 points above it.

```
With ActiveDocument.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
```

ShapeNode Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapeNodeC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapeNodeP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapeNodeM"}

Shapes (Shape)

└

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 the active document is a corner point, the following example makes it a smooth point. For this example to work, shape three must be a freeform.

```
With ActiveDocument.Shapes(3)
    If .Nodes(1).EditingType = msoEditingCorner Then
        .Nodes.SetEditingType 1, msoEditingSmooth
    End If
End With
```

ShapeNodes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShapeNodesC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShapeNodesP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShapeNodesM"}

Shapes (Shape)

└

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 the active document. For this example to work, shape three must be a freeform with at least four nodes.

```
ActiveDocument.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 the active document. For this example to work, shape three must be a freeform with at least four nodes.

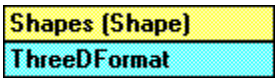
```
With ActiveDocument.Shapes(3).Nodes
    .Insert 4, msoSegmentCurve, msoEditingSmooth, 210, 100
End With
```

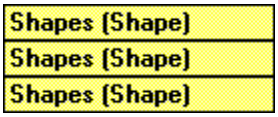
Use **Nodes**(*index*), where *index* is the node index number, to return a single **ShapeNode** object. If node one in shape three on the active document is a corner point, the following example makes it a smooth point. For this example to work, shape three must be a freeform.

```
With ActiveDocument.Shapes(3)
    If .Nodes(1).EditingType = msoEditingCorner Then
        .Nodes.SetEditingType 1, msoEditingSmooth
    End If
End With
```

ThreeDFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjThreeDFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjThreeDFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjThreeDFormatM"}





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 the active document and then specifies that the oval be extruded to a depth of 50 points and that the extrusion be purple.

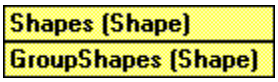
```
Set myShape = ActiveDocument.Shapes.AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    .ExtrusionColor.RGB = RGB(255, 100, 255)    ' RGB value for purple
End With
```

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.

GroupShapes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjGroupShapesC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjGroupShapesP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjGroupShapesM"}



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.

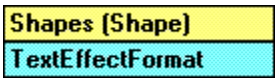
Using The Groupshapes Collection

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 the **GroupShapes** collection. The following example adds three triangles to the active document, groups them, sets a color for the entire group, and then changes the color for the second triangle only.

```
With ActiveDocument.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
```

TextEffectFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextEffectFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextEffectFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextEffectFormatM"}



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 the active document. For this example to work, shape one must be a WordArt object.

```
With ActiveDocument.Shapes(1).TextEffect
    .FontName = "Courier New"
    .FontBold = True
    .FontItalic = True
End With
```

AddCallout Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCalloutC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCalloutX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCalloutA"}

Creates a borderless line callout. Returns a **Shape** object that represents the new callout.

Syntax

expression.**AddCallout(*Type, Left, Top, Width, Height, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

Type Required **Long**. Can be one of the following **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), or **msoCalloutFour** (a three-segment line).

Left, ***Top*** Required **Single**. The position (in points) of the upper-left corner of the callout's bounding box, relative to the anchor.

Width, ***Height*** Required **Single**. The width and height of the callout's bounding box, in points.

Anchor Optional **Variant**. A **Range** object that represents the text that the callout is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the callout is positioned relative to the top and left edges of the page.

Remarks

You can insert a greater variety of callouts by using the **AddShape** method.

AddCallout Method Example

This example adds a callout to a newly created document and then sets the callout angle to 45 degrees.

```
Set myDocument = Documents.Add
Set myCall = myDocument.Shapes.AddCallout(msoCalloutTwo, 72, 36, 25, 25)
myCall.Callout.Angle = msoCalloutAngle45
```

AddCurve Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddCurveC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddCurveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddCurveA"}

Creates a Bézier curve. Returns a **Shape** object that represents the new curve.

Syntax

expression.**AddCurve(*SafeArrayOfPoints, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

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.

Anchor Optional **Variant**. A **Range** object that represents the text that the curve is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the curve is positioned relative to the top and left edges of the page.

coordinate pair

A pair of values representing the x- and y-coordinates of a point that are stored in a two-dimensional array that can contain coordinates for many points.

The following example creates a zero-based array that contains the coordinate pairs 50, 150 and 100, 200. Note that when you dimension the array, the size of the first dimension is the number of points you want to include, and the size of the second dimension is two (the first position within the dimension is for the x-coordinate, the second is for the y-coordinate).

```
Dim cpArray(1, 1)
cpArray(0, 0) = 50     ' The x-coordinate of the first point
cpArray(0, 1) = 150    ' The y-coordinate of the first point
cpArray(1, 0) = 100    ' The x-coordinate of the second point
cpArray(1, 1) = 200    ' The y-coordinate of the second point
```

AddCurve Method Example

This example adds a two-segment Bézier curve to `myDoc` and anchors the curve to the second paragraph.

```
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 myDoc = ActiveDocument
myDoc.Shapes.AddCurve pts, myDoc.Paragraphs(2).Range
```

AddLabel Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddLabelC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddLabelA"}

Creates a label. Returns a **Shape** object that represents the new label.

Syntax

expression.**AddLabel(*Orientation, Left, Top, Width, Height, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

Orientation Required **Long**. The orientation of the text. Can be one of the following **MsoTextOrientation** constants: **msoTextOrientationDownward**, **msoTextOrientationHorizontal**, **msoTextOrientationMixed**, or **msoTextOrientationUpward**. (Don't use any **MsoTextOrientation** constants other than these in U.S. English versions of Word.)

Left, ***Top*** Required **Single**. The position (in points) of the upper-left corner of the label, relative to the anchor.

Width, ***Height*** Required **Single**. The width and height of the label, in points.

Anchor Optional **Variant**. A **Range** object that represents the text that the label is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the label is positioned relative to the top and left edges of the page.

AddLabel Method Example

This example adds a label that contains the text "Test Label" to a new document.

```
Set myDocument = Documents.Add
Set myLabel =
myDocument.Shapes.AddLabel(Orientation:=msoTextOrientationHorizontal, _
    Left:=100, Top:=100, Width:=300, Height:=200)
myLabel.TextFrame.TextRange = "Test Label"
```

AddLine Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddLineC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddLineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddLineA"}

Creates a line. Returns a **Shape** object that represents the new line.

Syntax

expression.**AddLine(*BeginX, BeginY, EndX, EndY, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

BeginX, ***BeginY*** Required **Single**. The position (in points) of the line's starting point, relative to the anchor.

EndX, ***EndY*** Required **Single**. The position (in points) of the line's end point, relative to the anchor.

Anchor Optional **Variant**. A **Range** object that represents the text that the line is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the line is positioned relative to the top and left edges of the page.

Remarks

To create an arrow, use the **Line** property to format a line.

AddLine Method Example

This example adds a dashed, green line to a new document.

```
Set myDocument = Documents.Add
Set myLine = myDocument.Shapes.AddLine(100, 100, 60, 20)
With myLine.Line
    .DashStyle = msoLineDash
    .ForeColor.RGB = RGB(0, 128, 0)
End With
```

This example adds a line to the active document and then formats the line as a red arrow.

```
Set myLine = ActiveDocument.Shapes.AddLine(100, 100, 60, 20)
With myLine.Line
    .BeginArrowheadStyle = msoArrowheadNone
    .EndArrowheadStyle = msoArrowheadTriangle
    .ForeColor.RGB = RGB(128, 0, 0)
End With
```

AddPolyline Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddPolylineC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddPolylineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddPolylineA"}

Creates an open polyline or a closed polygon drawing. Returns a **Shape** object that represents the new polyline or polygon.

Syntax

expression.**AddPolyline(*SafeArrayOfPoints*)**

expression Required. An expression that returns a **Shapes** collection.

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.

AddPolyline Method Example

This example adds a triangle to `myDocument`. Because the first and last points of the triangle have the same coordinates, the polygon is closed and filled.

```
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 = Documents.Add
myDocument.Shapes.AddPolyline triArray
```

AddShape Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddShapeA"}

Creates an AutoShape. Returns a **Shape** object that represents the new AutoShape.

Syntax

expression.**AddShape(*Type, Left, Top, Width, Height, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

Type Required **Long**. The type of AutoShape to be created. Can be any of the **MsoAutoShapeType** constants.

Left, ***Top*** Required **Single**. The position (in points) of the upper-left corner of the AutoShape's bounding box, relative to the anchor.

Width, ***Height*** Required **Single**. The width and height of the AutoShape's bounding box, in points.

Anchor Optional **Variant**. A **Range** object that represents the text that the AutoShape is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the AutoShape is positioned relative to the top and left edges of the page.

Remarks

To change the type of an AutoShape that you've added, set the **AutoShapeType** property.

AddShape Method Example

This example adds a rectangle to a new document.

```
Set myDocument = Documents.Add
myDocument.Shapes.AddShape msoShapeRectangle, 50, 50, 100, 200
```

AddTextbox Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTextboxC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTextboxX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTextboxA"}

Creates a text box. Returns a **Shape** object that represents the new text box.

Syntax

expression.**AddTextbox(Orientation*, Left, Top, Width, Height, Anchor*)**

expression Required. An expression that returns a **Shapes** collection.

Orientation Required **Long**. The orientation of the text. Can be one of the following **MsoTextOrientation** constants: **msoTextOrientationDownward**, **msoTextOrientationHorizontal**, **msoTextOrientationMixed**, or **msoTextOrientationUpward**. (Don't use any **MsoTextOrientation** constants other than these in U.S. English versions of Word.)

Left, ***Top*** Required **Single**. The position (in points) of the upper-left corner of the text box, relative to the anchor.

Width, ***Height*** Required **Single**. The width and height of the text box, in points.

Anchor Optional **Variant**. A **Range** object that represents the text that the text box is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the text box is positioned relative to the top and left edges of the page.

AddTextbox Method Example

This example adds a text box that contains the text "Test Box" to a new document.

```
Set myDocument = Documents.Add
Set myTBox =
myDocument.Shapes.AddTextBox(Orientation:=msoTextOrientationHorizontal, _
    Left:=100, Top:=100, Width:=300, Height:=200)
myTBox.TextFrame.TextRange = "Test Box"
```

AddTextEffect Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddTextEffectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddTextEffectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddTextEffectA"}

Creates a WordArt object. Returns a **Shape** object that represents the new WordArt object.

Syntax

expression.**AddTextEffect(*PresetTextEffect, Text, FontName, FontSize, FontBold, FontItalic, Left, Top, Anchor***

expression Required. An expression that returns a **Shapes** collection.

PresetTextEffect Required **Long**. A preset text effect. Can be one of the following **MsoPresetTextEffect** constants.

msoTextEffect1	**msoTextEffect16**
msoTextEffect2	**msoTextEffect17**
msoTextEffect3	**msoTextEffect18**
msoTextEffect4	**msoTextEffect19**
msoTextEffect5	**msoTextEffect20**
msoTextEffect6	**msoTextEffect21**
msoTextEffect7	**msoTextEffect22**
msoTextEffect8	**msoTextEffect23**
msoTextEffect9	**msoTextEffect24**
msoTextEffect10	**msoTextEffect25**
msoTextEffect11	**msoTextEffect26**
msoTextEffect12	**msoTextEffect27**
msoTextEffect13	**msoTextEffect28**
msoTextEffect14	**msoTextEffect29**
msoTextEffect15	**msoTextEffect30**

The values of these constants correspond to the formats listed in the **WordArt Gallery** dialog box (numbered from left to right and from top to bottom).

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 **Long**. **True** to set the font used in the WordArt to bold.

FontItalic Required **Long**. **True** to set the font used in the WordArt to italic.

Left, ***Top*** Required **Single**. The position (in points) of the upper-left corner of the WordArt's bounding box, relative to the anchor

Anchor Optional **Variant**. A **Range** object that represents the text that the WordArt is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the WordArt is positioned relative to the top and left edges of the page.

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.

AddTextEffect Method Example

This example adds WordArt that contains the text "Test Text" to the active document and anchors the WordArt to the second paragraph.

```
ActiveDocument.Shapes.AddTextEffect _
    PresetTextEffect:=msoTextEffect11, _
    Text:="Test Text", _
    FontName:="Arial Black", _
    FontSize:=36, _
    FontBold:=True, _
    FontItalic:=False, _
    Left:=InchesToPoints(1), _
    Top:=InchesToPoints(1), _
    Anchor:=ActiveDocument.Paragraphs(2).Range
```

Hyperlink Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHyperlinkC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHyperlinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHyperlinkA"}

Returns a **Hyperlink** object that represents the hyperlink associated with the specified **Shape**, **InlineShape**, or **ShapeRange** object. Read-only.

Note If there's no hyperlink associated with the specified shape, an error occurs.

Hyperlink Property Example

This example displays the address for the hyperlink for the first shape in the active document.

```
MsgBox ActiveDocument.Shapes(1).Hyperlink.Address
```

SelectAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectAllC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectAllA"}

Selection object: Selects all the text in the selection.

Shapes object: Selects all the shapes in the main story of the document or in all the headers and footers in the document.

Syntax

expression.**SelectAll**

expression Required. An expression that returns a **Shapes** or **Selection** object.

Remarks

This method doesn't select **InlineShape** objects.

SelectAll Method Example

This example selects all the text in the selection.

```
Selection.SelectAll
```

This example selects all the shapes in the active document.

```
ActiveDocument.Shapes.SelectAll
```

This example selects all the shapes in the headers and footers in the active document and adds a shadow to each shape.

```
With ActiveWindow
    .View.Type = wdPageView
    .ActivePane.View.SeekView = wdSeekCurrentPageHeader
End With
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).Shapes.SelectAll
Selection.ShapeRange.Shadow.Type = msoShadow10
```

ConvertToShape Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToShapeA"}

InlineShape object: Converts an inline shape to a free-floating shape. Returns a **Shape** object that represents the new shape.

FreeformBuilder object: Creates a shape that has the geometric characteristics of the specified object. Returns a **Shape** object that represents the new shape.

Syntax 1

expression.**ConvertToShape**

Syntax 2

expression.**ConvertToShape(*Anchor*)**

expression Required. An expression that returns an InlineShape object (Syntax 1) or a **FreeformBuilder** object (Syntax 2).

Anchor Optional **Variant**. A **Range** object that represents the text that the shape is bound to. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, the anchoring range is selected automatically and the shape is positioned relative to the top and left edges of the page.

Remarks

You must apply the **AddNodes** method to a **FreeformBuilder** object at least once before you use the **ConvertToShape** method.

ConvertToShape Method Example

This example converts the first inline shape in the active document to a floating shape.

```
ActiveDocument.InlineShapes(1).ConvertToShape
```

This example adds a freeform with five vertices to `myDocument`.

```
Set myDocument = ActiveDocument
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
```

RGB Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRGBC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRGBX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRGBA"}

Returns or sets the red-green-blue (RGB) value of the specified color. Read/write **Long**.

RGB value

Value returned by the RGB function; specifies a color as a combination of red, green, and blue values. The RGB function has the following syntax:

RGB(*red, green, blue*)

The arguments specify the red, green, and blue components of the color as integers from 0 (zero) through 255.

RGB Property Example

This example sets the color of the second shape in the active document to gray.

```
ActiveDocument.Shapes(2).Fill.ForeColor.RGB = RGB(128, 128, 128)
```

This example sets the color of the shadow for Rectangle 1 in the active document to blue.

```
ActiveDocument.Shapes("Rectangle 1").Shadow.ForeColor.RGB = RGB(0, 0, 255)
```

This example returns the value of the foreground color of the first shape in the active document.

```
MsgBox ActiveDocument.Shapes(1).Fill.ForeColor.RGB
```

TextRange Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextRangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextRangeA"}

Returns a **Range** object that represents the text in the specified text frame. Read-only.

TextRange Property Example

This example adds a text box to the active document and then adds text to the text box.

```
Set myTBox =
ActiveDocument.Shapes.AddTextBox(Orientation:=msoTextOrientationHorizontal,
_
    Left:=100, Top:=100, Width:=300, Height:=200)
myTBox.TextFrame.TextRange = "Test Box"
```

This example adds text to TextBox 1 in the active document.

```
ActiveDocument.Shapes("TextBox 1").TextFrame.TextRange.InsertAfter("New
Text")
```

This example returns the text from TextBox 1 in the active document and displays it in a message box.

```
MsgBox ActiveDocument.Shapes("TextBox 1").TextFrame.TextRange.Text
```

InlineShape Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInlineShapeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInlineShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInlineShapeA "}

Returns an **InlineShape** object that represents the picture, OLE object, or ActiveX control that is the result of an INCLUDEPICTURE or EMBED field. Read-only.

Remarks

An **InlineShape** object is treated like a character and is positioned as a character within a line of text.

InlineShape Property Example

This example returns the width of the inline shape associated with the first field in the active document. For this example to work, the field must be an INCLUDEPICTURE field.

```
If ActiveDocument.Fields(1).Type = wdFieldIncludePicture Then
    MsgBox ActiveDocument.Fields(1).InlineShape.Width
End If
```

InlineShapes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInlineShapesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInlineShapesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInlineShapesA "}

Returns an **InlineShapes** collection that represents all the **InlineShape** objects in a document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

InlineShapes Property Example

This example displays the number of shapes and inline shapes in the active document.

```
Set doc = ActiveDocument
Msgbox "InlineShape = " & doc.InlineShapes.Count & _
    vbCr & "Shapes = " & doc.Shapes.Count
```

Item Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproItemC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproItemX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproItemA"}

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**.

Syntax

expression.**Item(*Index*)**

expression Required. An expression that returns an **Adjustments** object.

Index Required **Long**. The index number of the adjustment.

Remarks

AutoShapes and WordArt objects have up to eight adjustments.

Item Property 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 = Documents(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 = Documents(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
```

Line Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineA"}

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.

Line Property Example

This example adds a blue dashed line to `myDocument`.

```
Set myDocument = ActiveDocument
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 `myDocument` and then sets its border to be 8 points thick and red.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeCross, 10, 10, 50, 70).Line
    .Weight = 8
    .ForeColor.RGB = RGB(255, 0, 0)
End With
```

LockAspectRatio Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLockAspectRatioC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLockAspectRatioX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLockAspectRatioA"}

True if the specified shape retains its original proportions when you resize it. **False** if you can change the height and width of the shape independently of one another when you resize it. Read/write **Long**.

LockAspectRatio Property Example

This example adds a cube to `myDocument`. The cube can be moved and resized, but not reproportioned.

```
Set myDocument = ActiveDocument
myDocument.Shapes.AddShape(msoShapeCube, 50, 50, 100, 200).LockAspectRatio
= True
```

Nodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNodesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNodesA"}

Returns a **ShapeNodes** collection that represents the geometric description of the specified shape. Applies to **Shape** or **ShapeRange** objects that represent freeform drawings.

Nodes Property 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 = ActiveDocument
With myDocument.Shapes(3).Nodes
    .Insert 4, msoSegmentCurve, msoEditingSmooth, 210, 100
End With
```

PickUp Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPickUpC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPickUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPickUpA"}

Copies the formatting of the specified shape. Use the **Apply** method to apply the copied formatting to another shape.

Syntax

expression.**PickUp**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

PickUp Method Example

This example copies the formatting of shape one on `myDocument` and then applies the copied formatting to shape two.

```
Set myDocument = ActiveDocument
With myDocument
    .Shapes(1).PickUp
    .Shapes(2).Apply
End With
```

PictureFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPictureFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPictureFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPictureFormatA"}

Returns a **PictureFormat** object that contains picture formatting properties for the specified object. Applies to **Shape**, **ShapeRange**, or **InlineShape** objects that represent pictures or OLE objects. Read-only.

PictureFormat Property 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 = ActiveDocument
With myDocument.Shapes(1).PictureFormat
    .Brightness = 0.3
    .Contrast = .75
End With
```

Rotation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRotationC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRotationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRotationA"}

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.

Rotation Property Example

This example matches the rotation of all shapes on `myDocument` to the rotation of shape one.

```
Set myDocument = ActiveDocument
With myDocument.Shapes
    sh1Rotation = .Item(1).Rotation
    For o = 1 To .Count
        .Item(o).Rotation = sh1Rotation
    Next
End With
```

TextEffect Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextEffectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextEffectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextEffectA"}

Returns a **TextEffectFormat** object that contains text-effect formatting properties for the specified shape. Applies to **Shape** or **ShapeRange** objects that represent WordArt. Read-only.

TextEffect Property Example

This example sets the font style to bold for shape three on `myDocument` if the shape is WordArt.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.FontBold = True
    End If
End With
```

TextFrame Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextFrameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextFrameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextFrameA"}

Returns a **TextFrame** object that contains the text for the specified shape.

TextFrame Property Example

This example adds a rectangle to `myDocument`, adds text to the rectangle, and sets the margins for the text frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250,
140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 100
    .MarginRight = 0
    .MarginTop = 20
End With
```

ThreeD Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproThreeDC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproThreeDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproThreeDA"}

Returns a **ThreeDFormat** object that contains 3-D – effect formatting properties for the specified shape. Read-only.

ThreeD Property 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 = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .Depth = 50
    .ExtrusionColor.RGB = RGB(255, 100, 255)    ' RGB value for purple
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```

Ungroup Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUngroupC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUngroupX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUngroupA"}

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.

Syntax

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.

Ungroup Method Example

This example ungroups any grouped shapes and disassembles any pictures or OLE objects on `myDocument`.

```
Set myDocument = ActiveDocument
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 document.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    If s.Type = msoGroup Then s.Ungroup
Next
```

VerticalFlip Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalFlipC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalFlipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalFlipA"}

True if the specified shape is flipped around the vertical axis. Read-only **Long**.

VerticalFlip Property Example

This example restores each shape on `myDocument` to its original state if it's been flipped horizontally or vertically.

```
For Each s In ActiveDocument.Shapes
    If s.HorizontalFlip Then s.Flip msoFlipHorizontal
    If s.VerticalFlip Then s.Flip msoFlipVertical
Next
```

Vertices Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticesA"}

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
`vertArray(1, 1)`	The horizontal distance from the first vertex to the left side of the document
`vertArray(1, 2)`	The vertical distance from the first vertex to the top of the document
`vertArray(2, 1)`	The horizontal distance from the second vertex to the left side of the document
`vertArray(2, 2)`	The vertical distance from the second vertex to the top of the document
`vertArray(3, 1)`	The horizontal distance from the third vertex to the left side of the document
`vertArray(3, 2)`	The vertical distance from the third vertex to the top of the document

Vertices Property 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. Shape one must be a freeform drawing.

```
Set myDocument = ActiveDocument
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 be a Bézier curve containing 3*n*+1 vertices for this example to succeed.

```
Set myDocument = ActiveDocument
With myDocument.Shapes
    .AddCurve .Item(1).Vertices, Selection.Range
End With
```

ZOrder Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthZOrderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthZOrderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthZOrderA"}

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).

Syntax

expression.**ZOrder(*ZOrderCmd*)**

expression Required. An expression that returns a **Shape** object.

ZOrderCmd Required **Long**. Specifies where to move the specified shape relative to the other shapes. Can be one of the following **MsoZOrderCmd** constants: **msoBringForward**, **msoBringInFrontOfText**, **msoBringToFront**, **msoSendBackward**, **msoSendBehindText**, or **msoSendToBack**.

Remarks

Use the **ZOrderPosition** property to determine a shape's current position in the z-order.

ZOrder Method 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 document.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeOval, 100, 100, 100, 300)
    While .ZOrderPosition > 2
        .ZOrder msoSendBackward
    Wend
End With
```

ZOrderPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproZOrderPositionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproZOrderPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproZOrderPositionA"}

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 `myDocument`, 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.

ZOrderPosition Property 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 document.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeOval, 100, 100, 100, 300)
    While .ZOrderPosition > 2
        .ZOrder msoSendBackward
    Wend
End With
```

Adjustments Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAdjustmentsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAdjustmentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAdjustmentsA"}

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 or WordArt. Read-only.

Adjustments Property Example

This example sets to 0.25 the value of adjustment one on shape three on `myDocument`.

```
Set myDocument = ActiveDocument
myDocument.Shapes(3).Adjustments(1) = 0.25
```

Align Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAlignC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAlignX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAlignA"}

Aligns the shapes in the specified range of shapes.

Syntax

expression.**Align(*AlignCmd, RelativeTo*)**

expression Required. An expression that returns a **ShapeRange** object.

AlignCmd Required **Long**. Specifies the way the shapes in the specified shape range are to be aligned. Can be one of the following **MsoAlignCmd** constants: **msoAlignBottoms**, **msoAlignCenters**, **msoAlignLefts**, **msoAlignMiddles**, **msoAlignRights**, or **msoAlignTops**.

RelativeTo Required **Long**. **True** to align shapes relative to the edge of the document. **False** to align shapes relative to one another.

Align Method Example

This example aligns the left edges of all the shapes in the selection of shapes in `myDocument` with the left edge of the leftmost shape in the range.

```
Set myShapeRange = Selection.ShapeRange
myShapeRange.Align msoAlignLefts, False
```

AutoShapeType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoShapeTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoShapeTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoShapeTypeA"}

Returns or sets the shape type for the specified **Shape** or **ShapeRange** object, which must represent an AutoShape other than a line or freeform drawing. Read/write **Long**.

Note When you change the type of a shape, the shape retains its size, color, and other attributes.

Can be one of the following **MsoAutoShapeType** constants:

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
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

msoShapeDonut
msoShapeDoubleBrace
msoShapeDoubleBracket
msoShapeDoubleWave
msoShapeDownArrow
msoShapeDownArrowCallout
msoShapeDownRibbon
msoShapeExplosion1
msoShapeExplosion2
msoShapeFlowchartAlternateProcess
msoShapeFlowchartCard
msoShapeFlowchartCollate
msoShapeFlowchartConnector
msoShapeFlowchartData
msoShapeFlowchartDecision
msoShapeFlowchartDelay
msoShapeFlowchartDirectAccessStorage
msoShapeFlowchartDisplay
msoShapeFlowchartDocument
msoShapeFlowchartExtract
msoShapeFlowchartInternalStorage
msoShapeFlowchartMagneticDisk
msoShapeFlowchartManualInput
msoShapeFlowchartManualOperation
msoShapeFlowchartMerge
msoShapeFlowchartMultidocument
msoShapeFlowchartOffpageConnector
msoShapeFlowchartOr
msoShapeFlowchartPredefinedProcess
msoShapeFlowchartPreparation
msoShapeFlowchartProcess
msoShapeFlowchartPunchedTape
msoShapeFlowchartSequentialAccessStorage
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

AutoShapeType Property Example

This example replaces all 16-point stars with 32-point stars in `myDocument`.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    If s.AutoShapeType = msoShape16pointStar Then
        s.AutoShapeType = msoShape32pointStar
    End If
Next
```

BuildFreeform Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthBuildFreeformC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthBuildFreeformX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthBuildFreeformA"}

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.

Syntax

expression.**BuildFreeform(*EditingType, X1, Y1*)**

expression Required. An expression that returns a **Shapes** object.

EditingType Required **Long**. The editing property of the first node. Can be either of the following **MsoEditingType** constants: **msoEditingAuto** or **msoEditingCorner** (cannot be **msoEditingSmooth** or **msoEditingSymmetric**).

X1, ***Y1*** Required **Single**. The position (in points) of the first node in the freeform drawing relative to the upper-left corner of the document.

BuildFreeform Method Example

This example adds a freeform with five vertices to `myDocument`.

```
Set myDocument = ActiveDocument
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
```

Callout Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCalloutC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCalloutX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCalloutA"}

Returns a **CalloutFormat** object that contains callout formatting properties for the specified shape. Applies to **Shape** or **ShapeRange** objects that represent callouts. Read-only.

Callout Property 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 = ActiveDocument
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
```

Fill Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFillC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFillX":1}
{ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFillA"}

Returns a **FillFormat** object that contains fill formatting properties for the specified shape. Read-only.

Fill Property 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 = Documents(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
```

Flip Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFlipC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFlipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFlipA"}

Flips the specified shape around it's horizontal or vertical axis.

Syntax

expression.**Flip(*FlipCmd*)**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

FlipCmd Required **Long**. Specifies whether the shape is to be flipped horizontally or vertically. Can be either of the following **MsoFlipCmd** constants: **msoFlipHorizontal** or **msoFlipVertical**.

Flip Method Example

This example adds a triangle to `myDocument`, duplicates the triangle, and then flips the duplicate triangle vertically and makes it red.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRightTriangle, 10, 10, 50,
50).Duplicate
    .Fill.ForeColor.RGB = RGB(255, 0, 0)
    .Flip msoFlipVertical
End With
```

Group Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGroupC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGroupX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGroupA"}

Groups the shapes in the specified range. Returns the grouped shapes as a single **Shape** object.

Syntax

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.

Group Method 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 = ActiveDocument
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
```

GroupItems Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGroupItemsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGroupItemsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGroupItemsA"}

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.

GroupItems Property 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 = ActiveDocument
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
```

HorizontalFlip Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalFlipC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalFlipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalFlipA"}

True if the specified shape is flipped around the horizontal axis. Read-only **Long**.

HorizontalFlip Property Example

This example restores each shape on `myDocument` to its original state if it's been flipped horizontally or vertically.

```
For Each s In ActiveDocument.Shapes
    If s.HorizontalFlip Then s.Flip msoFlipHorizontal
    If s.VerticalFlip Then s.Flip msoFlipVertical
Next
```

Accent Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAccentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAccentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAccentA"}

True if a vertical accent bar separates the callout text from the callout line. Read/write **Long**.

Accent Property 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 = ActiveDocument
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
```

AddNodes Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddNodesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddNodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddNodesA"}

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.

Syntax

expression.**AddNodes(*SegmentType, EditingType, X1, Y1, X2, Y2, X3, Y3*)**

expression Required. An expression that returns a **FreeformBuilder** object.

SegmentType Required **Long**. The type of segment to be added. Can be either of the following **MsoSegmentType** constants: **msoSegmentCurve** or **msoSegmentLine**.

EditingType Required **Long**. The editing property of the vertex. Can be either of the following **MsoEditingType** constants: **msoEditingAuto** or **msoEditingCorner** (cannot be **msoEditingSmooth** or **msoEditingSymmetric**). If ***SegmentType*** is **msoSegmentLine**, ***EditingType*** must be **msoEditingAuto**.

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.

AddNodes Method Example

This example adds a freeform with five vertices to `myDocument`.

```
Set myDocument = ActiveDocument
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
```

Angle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAngleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAngleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAngleA"}

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. Can be one of the following **MsoCalloutAngleType** constants: **msoCalloutAngle30**, **msoCalloutAngle45**, **msoCalloutAngle60**, **msoCalloutAngle90**, **msoCalloutAngleAutomatic**, or **msoCalloutAngleMixed**. Read/write **Long**.

Remarks

If you set the value of this property to anything other than **msoCalloutAngleAutomatic**, the callout line maintains a fixed angle as you drag the callout.

Angle Property Example

This example sets to 90 degrees the callout angle for a callout named "co1" on `myDocument`.

```
Set myDocument = Documents(1)
myDocument.Shapes("co1").Callout.Angle = msoCalloutAngle90
```

AutoAttach Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoAttachC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoAttachX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoAttachA"}

True if 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 **Long**.

Remarks

When the value of this property is **True**, 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 **False**, 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.

AutoAttach Property Example

This example adds two callouts to `myDocument`. If you drag the text box for each of these callouts to the left of the callout line origin, the place on the text box where the callout line attaches will change for the automatically attached callout.

```
Set myDocument = ActiveDocument
With myDocument.Shapes
    With .AddCallout(msoCalloutTwo, 100, 170, 200, 50)
        .TextFrame.TextRange.Text = "auto-attached"
        .Callout.AutoAttach = True
    End With
    With .AddCallout(msoCalloutTwo, 100, 350, 200, 50)
        .TextFrame.TextRange.Text = "not auto-attached"
        .Callout.AutoAttach = False
    End With
End With
```

AutoLength Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoLengthA"}

True if the first segment of the callout line (the segment attached to the text callout box) is scaled automatically whenever the callout is moved. **False** if the first segment of the callout retains the fixed length specified 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**). Read-only **Long**.

Remarks

This property is read-only. Use the **AutomaticLength** method to set this property to **True**, and use the **CustomLength** method to set this property to **False**.

AutoLength Property 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 = ActiveDocument
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```

AutomaticLength Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutomaticLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutomaticLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutomaticLengthA"}

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**).

Syntax

expression.**AutomaticLength**

expression Required. An expression that returns a **CalloutFormat** object.

Remarks

Applying this method sets the **AutoLength** property to **True**.

AutomaticLength Method 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 = ActiveDocument
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```

BackColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackColorA"}

Returns or sets a **ColorFormat** object that represents the background color for the specified fill or patterned line. Read/write.

BackColor Property 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 = ActiveDocument
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 = ActiveDocument
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
```

BeginArrowheadLength Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBeginArrowheadLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBeginArrowheadLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBeginArrowheadLengthA"}

Returns or sets the length of the arrowhead at the beginning of the specified line. Can be one of the following **MsoArrowheadLength** constants: **msoArrowheadLengthMedium**, **msoArrowheadLengthMixed**, **msoArrowheadLong**, or **msoArrowheadShort**. Read/write **Long**.

BeginArrowheadLength Property 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 = ActiveDocument
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

BeginArrowheadStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBeginArrowheadStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBeginArrowheadStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBeginArrowheadStyleA"}

Returns or sets the style of the arrowhead at the beginning of the specified line. Can be one of the following **MsoArrowheadStyle** constants: **msoArrowheadDiamond**, **msoArrowheadNone**, **msoArrowheadOpen**, **msoArrowheadOval**, **msoArrowheadStealth**, **msoArrowheadStyleMixed**, or **msoArrowheadTriangle**. Read/write **Long**.

BeginArrowheadStyle Property 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 = ActiveDocument
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

BeginArrowheadWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBeginArrowheadWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBeginArrowheadWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBeginArrowheadWidthA"}

Returns or sets the width of the arrowhead at the beginning of the specified line. Can be one of the following **MsoArrowheadWidth** constants: **msoArrowheadNarrow**, **msoArrowheadWide**, **msoArrowheadWidthMedium**, or **msoArrowheadWidthMixed**. Read/write **Long**.

BeginArrowheadWidth Property 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 = Documents(1)
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

Border Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBorderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBorderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBorderA"}

True if the text in the specified callout is surrounded by a border. Read/write **Long**.

Border Property 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 = ActiveDocument
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
```

Brightness Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBrightnessC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBrightnessX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBrightnessA"}

Returns or sets the brightness of the specified picture or OLE object. The value for this property must be a number from 0.0 (dimmest) to 1.0 (brightest). Read/write **Single**.

Brightness Property Example

This example sets the brightness for shape one on `myDocument`. Shape one must be either a picture or an OLE object.

```
Set myDocument = ActiveDocument
myDocument.Shapes(1).PictureFormat.Brightness = 0.3
```

ColorType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColorTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColorTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColorTypeA"}

Returns or sets the type of color transformation applied to the specified picture or OLE object. Can be one of the following **MsoPictureColorType** constants: **msoPictureAutomatic**, **msoPictureBlackAndWhite**, **msoPictureGrayscale**, **msoPictureMixed**, or **msoPictureWatermark**. Read/write **Long**.

ColorType Property 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 = ActiveDocument
myDocument.Shapes(1).PictureFormat.ColorType = msoPictureGrayScale
```

Contrast Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContrastC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContrastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContrastA"}

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**.

Contrast Property Example

This example sets the contrast for shape one on `myDocument`. Shape one must be either a picture or an OLE object.

```
Set myDocument = Documents(1)
myDocument.Shapes(1).PictureFormat.Contrast = 0.8
```

CustomDrop Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCustomDropC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCustomDropX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCustomDropA"}

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 which case the drop distance is measured from the bottom of the text box.

Syntax

expression.**CustomDrop(*Drop*)**

expression Required. An expression that returns a **CalloutFormat** object.

Drop Required **Single**. The drop distance, in points.

Remarks

If the **PresetDrop** method was previously used to set the drop for the specified callout, use the statement `PresetDrop msoCalloutDropCustom` before using the **CustomDrop** method so that the custom drop setting takes effect.

CustomDrop Method Example

This example cancels any preset drop that's been set for shape one in `myDocument`, 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 one must be a callout.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(1).Callout
    .PresetDrop msoCalloutDropCustom
    .CustomDrop 14
    .AutoAttach = False
End With
```

CustomLength Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCustomLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCustomLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCustomLengthA"}

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**).

Syntax

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.

CustomLength Method 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 = ActiveDocument
With myDocument.Shapes(1).Callout
    If .AutoLength Then
        .CustomLength 50
    Else
        .AutomaticLength
    End If
End With
```

DashStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDashStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDashStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDashStyleA"}

Returns or sets the dash style for the specified line. Can be one of the following **MsoLineDashStyle** constants: **msoLineDash**, **msoLineDashDot**, **msoLineDashDotDot**, **msoLineDashStyleMixed**, **msoLineLongDash**, **msoLineLongDashDot**, **msoLineRoundDot**, **msoLineSolid**, or **msoLineSquareDot**. Read/write **Long**.

DashStyle Property Example

This example adds a blue dashed line to `myDocument`.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(50, 0, 128)
End With
```

Depth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDepthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDepthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDepthA"}

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**.

Depth Property 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 = ActiveDocument
Set myShape = myDocument.Shapes.AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    .ExtrusionColor.RGB = RGB(255, 100, 255)    ' RGB value for purple
End With
```

Drop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDropC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDropX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDropA"}

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 which 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**. Use the statement `PresetDrop msoCalloutCustomDrop` to set the **DropType** property to **msoCalloutDropCustom**.

Drop Property 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 = ActiveDocument
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
```

DropType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDropTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDropTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDropTypeA"}

Returns a value that indicates where the callout line attaches to the callout text box. Can be one of the following **MsoCalloutDropType** constants: **msoCalloutDropBottom**, **msoCalloutDropCenter**, **msoCalloutDropCustom**, **msoCalloutDropMixed**, or **msoCalloutDropTop**. Read-only **Long**.

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.

DropType 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 = ActiveDocument
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
```

EditingType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEditingTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEditingTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEditingTypeA"}

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. Can be one of the following **MsoEditingType** constants: **msoEditingAuto**, **msoEditingCorner**, **msoEditingSmooth**, or **msoEditingSymmetric**. If the node is a control point for a curved segment, this property returns the editing type of the adjacent vertex. Read-only **Long**.

Remarks

This property is read-only. Use the **SetEditingType** method to set the value of this property.

EditingType 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 = ActiveDocument
With myDocument.Shapes(3).Nodes
    For n = 1 to .Count
        If .Item(n).EditingType = msoEditingCorner Then
            .SetEditingType n, msoEditingSmooth
        End If
    Next
End With
```

EndArrowheadLength Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEndArrowheadLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEndArrowheadLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEndArrowheadLengthA"}

Returns or sets the length of the arrowhead at the end of the specified line. Can be one of the following **MsoArrowheadLength** constants: **msoArrowheadLengthMedium**, **msoArrowheadLengthMixed**, **msoArrowheadLong**, or **msoArrowheadShort**. Read/write **Long**.

EndArrowheadLength Property 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 = ActiveDocument
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

EndArrowheadStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEndArrowheadStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEndArrowheadStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEndArrowheadStyleA"}

Returns or sets the style of the arrowhead at the end of the specified line. Can be one of the following **MsoArrowheadStyle** constants: **msoArrowheadDiamond**, **msoArrowheadNone**, **msoArrowheadOpen**, **msoArrowheadOval**, **msoArrowheadStealth**, **msoArrowheadStyleMixed**, or **msoArrowheadTriangle**. Read/write **Long**.

EndArrowheadStyle Property 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 = ActiveDocument
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

EndArrowheadWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEndArrowheadWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEndArrowheadWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEndArrowheadWidthA"}

Returns or sets the width of the arrowhead at the end of the specified line. Can be one of the following **MsoArrowheadWidth** constants: **msoArrowheadNarrow**, **msoArrowheadWide**, **msoArrowheadWidthMedium**, or **msoArrowheadWidthMixed**. Read/write **Long**.

EndArrowheadWidth Property 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 = ActiveDocument
With myDocument.Shapes.AddLine(100, 100, 200, 300).Line
    .BeginArrowheadLength = msoArrowheadShort
    .BeginArrowheadStyle = msoArrowheadOval
    .BeginArrowheadWidth = msoArrowheadNarrow
    .EndArrowheadLength = msoArrowheadLong
    .EndArrowheadStyle = msoArrowheadTriangle
    .EndArrowheadWidth = msoArrowheadWide
End With
```

ExtrusionColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExtrusionColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExtrusionColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExtrusionColorA"}

Returns a **ColorFormat** object that represents the color of the shape's extrusion. Read-only.

ExtrusionColor Property 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 = ActiveDocument
Set myShape = myDocument.Shapes.AddShape(msoShapeOval, 90, 90, 90, 40)
With myShape.ThreeD
    .Visible = True
    .Depth = 50
    .ExtrusionColor.RGB = RGB(255, 100, 255)    ' RGB value for purple
End With
```

ExtrusionColorType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExtrusionColorTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExtrusionColorTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExtrusionColorTypeA"}

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. Can be one of the following **MsoExtrusionColorType** constants: **msoExtrusionColorAutomatic** (extrusion color based on shape fill), **msoExtrusionColorCustom** (extrusion color independent of shape fill), or **msoExtrusionColorTypeMixed**. Read/write **Long**.

ExtrusionColorType Property Example

If shape one on `myDocument` has an automatic extrusion color, this example gives the extrusion a custom yellow color.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(1).ThreeD
    If .ExtrusionColorType = msoExtrusionColorAutomatic Then
        .ExtrusionColor.RGB = RGB(240, 235, 16)
    End If
End With
```

FontBold Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontBoldC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontBoldX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontBoldA"}

True if the font in the specified WordArt is bold. Read/write **Long**.

FontBold Property Example

This example sets the font to bold for shape three on `myDocument` if the shape is WordArt.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.FontBold = True
    End If
End With
```

FontItalic Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontItalicC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontItalicX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontItalicA"}

True if the font in the specified WordArt is italic. Read/write **Long**.

FontItalic Property Example

This example sets the font to italic for the shape named "WordArt 4" in `myDocument`.

```
Set myDocument = ActiveDocument
myDocument.Shapes("WordArt 4").TextEffect.FontItalic = True
```

FontSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontSizeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontSizeA"}

Returns or sets the font size for the specified WordArt, in points. Read/write **Single**.

FontSize Property Example

This example sets the font size to 16 points for the shape named "WordArt 2" in `myDocument`.

```
Set myDocument = ActiveDocument
myDocument.Shapes("WordArt 2").TextEffect.FontSize = 16
```

ForeColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproForeColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproForeColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproForeColorA"}

Returns or sets a **ColorFormat** object that represents the foreground color for the fill, line, or shadow. Read/write.

ForeColor Property 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 = ActiveDocument
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 = ActiveDocument
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
```

Gap Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGapC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGapX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGapA"}

Returns or sets the horizontal distance (in points) between the end of the callout line and the text bounding box. Read/write **Single**.

Gap Property 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 = ActiveDocument
myDocument.Shapes(1).Callout.Gap = 3
```

GradientColorType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGradientColorTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGradientColorTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGradientColorTypeA"}

Returns the gradient color type for the specified fill. Can be one of the following **MsoGradientColorType** constants: **msoGradientColorMixed**, **msoGradientOneColor**, **msoGradientPresetColors**, or **msoGradientTwoColors**. Read-only **Long**.

This property is read-only. Use the **OneColorGradient**, **PresetGradient**, or **TwoColorGradient** method to set the gradient type for the fill.

GradientColorType Property 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 = Documents(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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGradientDegreeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGradientDegreeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGradientDegreeA"}

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 coloris mixed in. Read-only **Single**.

This property is read-only. Use the **OneColorGradient** method to set the gradient degree for the fill.

GradientDegree Property 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 = ActiveDocument
With myDocument.Shapes
    gradDegree1 = .Item("Rectangle 2").Fill.GradientDegree
    With .AddShape(msoShapeRectangle, 0, 0, 40, 80).Fill
        .ForeColor.RGB = RGB(0, 128, 128)
        .OneColorGradient msoGradientHorizontal, 1, gradDegree1
    End With
End With
```

GradientStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGradientStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGradientStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGradientStyleA"}

Returns the gradient style for the specified fill. Can be one of the following **MsoGradientStyle** constants: **msoGradientDiagonalDown**, **msoGradientDiagonalUp**, **msoGradientFromCenter**, **msoGradientFromCorner**, **msoGradientHorizontal**, **msoGradientMixed**, or **msoGradientVertical**. (The constant **msoGradientFromTitle** is used only in PowerPoint.) Read-only **Long**.

This property is read-only. Use the **OneColorGradient** or **TwoColorGradient** method to set the gradient style for the fill.

Note 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.

GradientStyle Property 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 = ActiveDocument
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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGradientVariantC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGradientVariantX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGradientVariantA"}

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 **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** or **TwoColorGradient** method to set the gradient variant for the fill.

GradientVariant Property 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.

```
With ActiveDocument.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
```

HasText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasTextC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasTextA"}

True if the specified shape has text associated with it. Read-only **Boolean**.

HasText Property Example

If shape two on `myDocument` contains text, this example displays a message if the text overflows its frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(2).TextFrame
    If .HasText = True Then
        If .Overflowing = True Then
            Msgbox "Text overflows the frame."
        End If
    End If
End With
```

IncrementBrightness Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementBrightnessC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementBrightnessX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementBrightnessA"}

Changes the brightness of the picture by the specified amount. Use the **Brightness** property to set the absolute brightness of the picture.

Syntax

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.

IncrementBrightness Method 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 = ActiveDocument
With myDocument.Shapes(1).Duplicate
    .PictureFormat.IncrementBrightness -0.2
    .IncrementLeft 50
    .IncrementTop 50
End With
```

IncrementContrast Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementContrastC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementContrastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementContrastA"}

Changes the contrast of the picture by the specified amount. Use the **Contrast** property to set the absolute contrast for the picture.

Syntax

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.

IncrementContrast Method Example

This example increases the contrast for all embedded OLE objects on `myDocument` that aren't already set to maximum contrast.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    If s.Type = msoEmbeddedOLEObject Then
        s.PictureFormat.IncrementContrast 0.1
    End If
Next
```

IncrementLeft Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementLeftC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementLeftA"}

Moves the specified shape horizontally by the specified number of points.

Syntax

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.

IncrementLeft 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 = ActiveDocument
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```

IncrementOffsetX Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementOffsetXC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementOffsetXX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementOffsetXA"}

Changes the horizontal offset of the shadow by the specified number of points. Use the **OffsetX** property to set the absolute horizontal shadow offset.

Syntax

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.

IncrementOffsetX Method Example

This example moves the shadow on shape three on `myDocument` to the left by 3 points.

```
Set myDocument = ActiveDocument
myDocument.Shapes(3).Shadow.IncrementOffsetX -3
```

IncrementOffsetY Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementOffsetYC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementOffsetYX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementOffsetYA"}

Changes the vertical offset of the shadow by the specified number of points. Use the **OffsetY** property to set the absolute vertical shadow offset.

Syntax

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.

IncrementOffsetY Method Example

This example moves the shadow on shape three on `myDocument` up by 3 points.

```
Set myDocument = ActiveDocument
myDocument.Shapes(3).Shadow.IncrementOffsetY -3
```

IncrementRotation Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementRotationC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementRotationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementRotationA"}

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.

Syntax

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.

IncrementRotation 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 = ActiveDocument
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```

IncrementRotationX Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementRotationXC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementRotationXX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementRotationXA"}

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.

Syntax

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.

IncrementRotationX 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 = ActiveDocument
myDocument.Shapes(1).ThreeD.IncrementRotationX 10
```

IncrementRotationY Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementRotationYC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementRotationYX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementRotationYA"}

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.

Syntax

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.

IncrementRotationY Method 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 = ActiveDocument
myDocument.Shapes(1).ThreeD.IncrementRotationY -10
```

IncrementTop Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIncrementTopC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIncrementTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIncrementTopA"}

Moves the specified shape vertically by the specified number of points.

Syntax

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.

IncrementTop 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 = ActiveDocument
With myDocument.Shapes(1).Duplicate
    .Fill.PresetTextured msoTextureGranite
    .IncrementLeft 70
    .IncrementTop -50
    .IncrementRotation 30
End With
```

KernedPairs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKernedPairsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKernedPairsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKernedPairsA"}

True if character pairs in the specified WordArt are kerned. Read/write **Long**.

KernedPairs Property Example

This example turns on character pair kerning for shape three on `myDocument` if the shape is WordArt.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(3)
    If .Type = msoTextEffect Then
        .TextEffect.KernedPairs =True
    End If
End With
```

MarginBottom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMarginBottomC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMarginBottomX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMarginBottomA"}

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**.

MarginBottom Property Example

This example adds a rectangle to `myDocument`, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250,
140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 100
    .MarginRight = 0
    .MarginTop = 20
End With
```

MarginLeft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMarginLeftC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMarginLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMarginLeftA"}

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**.

MarginLeft Property Example

This example adds a rectangle to `myDocument`, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250,
140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 100
    .MarginRight = 0
    .MarginTop = 20
End With
```

MarginRight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMarginRightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMarginRightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMarginRightA"}

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**.

MarginRight Property Example

This example adds a rectangle to `myDocument`, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250,
140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 100
    .MarginRight = 0
    .MarginTop = 20
End With
```

MarginTop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMarginTopC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMarginTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMarginTopA"}

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**.

MarginTop Property Example

This example adds a rectangle to `myDocument`, adds text to the rectangle, and then sets the margins for the text frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 250,
140).TextFrame
    .TextRange.Text = "Here is some test text"
    .MarginBottom = 0
    .MarginLeft = 100
    .MarginRight = 0
    .MarginTop = 20
End With
```

NormalizedHeight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNormalizedHeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNormalizedHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNormalizedHeightA"}

True if all characters (both uppercase and lowercase) in the specified WordArt are the same height. Read/write **Long**.

NormalizedHeight Property 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 = ActiveDocument
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 = True
```

Obscured Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproObscuredC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproObscuredX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproObscuredA"}

True if the shadow of the specified shape appears filled in and is obscured by the shape, even if the shape has no fill. **False** if the shadow has no fill and the outline of the shadow is visible through the shape if the shape has no fill. Read/write **Long**.

Obscured Property Example

This example sets the horizontal and vertical offsets for the shadow of 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 = ActiveDocument
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
    .Obscured = True
End With
```

OffsetX Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOffsetXC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOffsetXX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOffsetXA"}

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.

OffsetX Property Example

This example sets the horizontal and vertical offsets for the shadow of 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 = ActiveDocument
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
End With
```

OffsetY Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOffsetYC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOffsetYX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOffsetYA"}

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.

OffsetY Property Example

This example sets the horizontal and vertical offsets for the shadow of 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 = ActiveDocument
With myDocument.Shapes(3).Shadow
    .Visible = True
    .OffsetX = 5
    .OffsetY = -3
End With
```

OneColorGradient Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOneColorGradientC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOneColorGradientX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOneColorGradientA"}

Sets the specified fill to a one-color gradient.

Syntax

expression.**OneColorGradient(*Style, Variant, Degree*)**

expression Required. An expression that returns a **FillFormat** object.

Style Required **Long**. The gradient style. Can be one of the following **MsoGradientStyle** constants: **msoGradientDiagonalDown**, **msoGradientDiagonalUp**, **msoGradientFromCenter**, **msoGradientFromCorner**, **msoGradientHorizontal**, or **msoGradientVertical**. (The constant **msoGradientFromTitle** is used only in PowerPoint.)

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 **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).

OneColorGradient Method Example

This example adds a rectangle with a one-color gradient fill to `myDocument`.

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

Pattern Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPatternC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPatternX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPatternA"}

Returns a value that represents the pattern applied to the specified fill or line. Read-only **Long**.

Can be one of the following **MsoPatternType** constants:

msoPattern10Percent
msoPattern20Percent
msoPattern25Percent
msoPattern30Percent
msoPattern40Percent
msoPattern50Percent
msoPattern5Percent
msoPattern60Percent
msoPattern70Percent
msoPattern75Percent
msoPattern80Percent
msoPattern90Percent
msoPatternDarkDownwardDiagonal
msoPatternDarkHorizontal
msoPatternDarkUpwardDiagonal
msoPatternDarkVertical
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

This property is read-only. Use the **Patterned** method to set the pattern for the fill or line.

Use the **BackColor** and **ForeColor** properties to set the colors used in the pattern.

Pattern Property Example

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 = ActiveDocument
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
```

This example adds a patterned line to `myDocument`.

```
Set myDocument = ActiveDocument
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
```

Patterned Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPatternedC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPatternedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPatternedA"}

Sets the specified fill to a pattern.

Syntax

expression.**Patterned(*Pattern*)**

expression Required. An expression that returns a **FillFormat** object.

Pattern Required **Long**. The pattern to be used for the specified fill. Can be any of the **MsoPatternType** constants.

Remarks

Use the **BackColor** and **ForeColor** properties to set the colors used in the pattern.

Patterned Method Example

This example adds an oval with a patterned fill to `myDocument`.

```
Set myDocument = ActiveDocument
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
```

Perspective Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPerspectiveC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPerspectiveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPerspectiveA"}

True if the extrusion appears in perspective—that is, if the walls of the extrusion narrow toward a vanishing point. **False** if the extrusion is a parallel, or orthographic, projection—that is, if the walls don't narrow toward a vanishing point. Read/write **Long**.

Perspective Property 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 = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .Depth = 100
    .Perspective = False
End With
```

Points Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPointsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPointsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPointsA"}

Returns the position of the specified node as a coordinate pair. Each coordinate is expressed in points. Read-only **Variant**.

Remarks

This property is read-only. Use the **SetPosition** method to set the location of the node.

Points Property 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 = ActiveDocument
With myDocument.Shapes(3).Nodes
    pointsArray = .Item(2).Points
    currXvalue = pointsArray(0, 0)
    currYvalue = pointsArray(0, 1)
    .SetPosition 2, currXvalue + 200, currYvalue + 300
End With
```

PresetDrop Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPresetDropC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPresetDropX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPresetDropA"}

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.

Syntax

expression.**PresetDrop(*DropType*)**

expression Required. An expression that returns a **CalloutFormat** object.

DropType Required **Long**. The starting position of the callout line relative to the text bounding box. Can be one of the following **MsoCalloutDropType** constants: **msoCalloutDropBottom**, **msoCalloutDropCenter**, **msoCalloutDropCustom**, or **msoCalloutDropTop**. If you specify **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.

PresetDrop Method 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 = ActiveDocument
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 = ActiveDocument
With myDocument.Shapes(1).Callout
    If .DropType = msoCalloutDropTop Then
        .PresetDrop msoCalloutDropBottom
    ElseIf .DropType = msoCalloutDropBottom Then
        .PresetDrop msoCalloutDropTop
    End If
End With
```

PresetExtrusionDirection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetExtrusionDirectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetExtrusionDirectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetExtrusionDirectionA"}

Returns the direction that the extrusion's sweep path takes away from the extruded shape (the front face of the extrusion). Can be one of the following **MsoPresetExtrusionDirection** constants: **msoExtrusionBottom**, **msoExtrusionBottomLeft**, **msoExtrusionBottomRight**, **msoExtrusionLeft**, **msoExtrusionNone**, **msoExtrusionRight**, **msoExtrusionTop**, **msoExtrusionTopLeft**, **msoExtrusionTopRight**, or **msoPresetExtrusionDirectionMixed**. Read-only **Long**.

Remarks

This property is read-only. To set the value of this property, use the **SetExtrusionDirection** method.

PresetExtrusionDirection Property 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 = ActiveDocument
For Each s In myDocument.Shapes
    With s.ThreeD
        If .PresetExtrusionDirection = msoExtrusionTopLeft Then
            .SetExtrusionDirection msoExtrusionBottomRight
        End If
    End With
Next
```

PresetGradient Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPresetGradientC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPresetGradientX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPresetGradientA"}

Sets the specified fill to a preset gradient.

Syntax

expression.**PresetGradient(*Style, Variant, PresetGradientType*)**

expression Required. An expression that returns a **FillFormat** object.

Style Required **Long**. The gradient style. Can be one of the following **MsoGradientStyle** constants: **msoGradientDiagonalDown**, **msoGradientDiagonalUp**, **msoGradientFromCenter**, **msoGradientFromCorner**, **msoGradientHorizontal**, or **msoGradientVertical**. (The constant **msoGradientFromTitle** is used only in PowerPoint.)

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 **msoGradientFromCenter**, this argument can be either 1 or 2.

PresetGradientType Required **Long**. The gradient type. Can be one of the following **MsoPresetGradientType** constants:

msoGradientBrass	**msoGradientLateSunset**
msoGradientCalmWater	**msoGradientMahogany**
msoGradientChrome	**msoGradientMoss**
msoGradientChromeII	**msoGradientNightfall**
msoGradientDaybreak	**msoGradientOcean**
msoGradientDesert	**msoGradientParchment**
msoGradientEarlySunset	**msoGradientPeacock**
msoGradientFire	**msoGradientRainbow**
msoGradientFog	**msoGradientRainbowII**
msoGradientGold	**msoGradientSapphire**
msoGradientGoldII	**msoGradientSilver**
msoGradientHorizon	**msoGradientWheat**

PresetGradient Method Example

This example adds a rectangle with a preset gradient fill to `myDocument`.

```
Set myDocument = ActiveDocument
myDocument.Shapes.AddShape(msoShapeRectangle, 90, 90, 140, 80) _
    .Fill.PresetGradient msoGradientHorizontal, 1, msoGradientBrass
```

PresetGradientType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetGradientTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetGradientTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetGradientTypeA"}

Returns the preset gradient type for the specified fill. Read-only **Long**.

Can be one of the following **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

This property is read-only. Use the **PresetGradient** method to set the preset gradient type for the fill.

PresetGradientType Property 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 = ActiveDocument
For Each s In myDocument.Shapes
    With s.Fill
        If .PresetGradientType = msoGradientMoss Then
            .PresetGradient msoGradientHorizontal, 1, msoGradientFog
        End If
    End With
Next
```

PresetLightingDirection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetLightingDirectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetLightingDirectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetLightingDirectionA"}

Returns or sets the position of the light source relative to the extrusion. Can be one of the following **MsoPresetLightingDirection** constants: **msoLightingBottom**, **msoLightingBottomLeft**, **msoLightingBottomRight**, **msoLightingLeft**, **msoLightingNone**, **msoLightingRight**, **msoLightingTop**, **msoLightingTopLeft**, **msoLightingTopRight**, or **msoPresetLightingDirectionMixed**. Read/write **Long**.

Note You won't see the lighting effects you set if the extrusion has a wire frame surface.

PresetLightingDirection Property 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 = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```

PresetLightingSoftness Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetLightingSoftnessC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetLightingSoftnessX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetLightingSoftnessA"}

Returns or sets the intensity of the extrusion lighting. Can be one of the following **MsoPresetLightingSoftness** constants: **msoLightingBright**, **msoLightingDim**, **msoLightingNormal**, or **msoPresetLightingSoftnessMixed**. Read/write **Long**.

PresetLightingSoftness Property Example

This example specifies that the extrusion for shape one on `myDocument` extend be lit brightly from the left.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .PresetLightingSoftness = msoLightingBright
    .PresetLightingDirection = msoLightingLeft
End With
```

PresetMaterial Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetMaterialC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetMaterialX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetMaterialA"}

Returns or sets the extrusion surface material. Can be one of the following **MsoPresetMaterial** constants: **msoMaterialMatte**, **msoMaterialMetal**, **msoMaterialPlastic**, **msoMaterialWireFrame**, or **msoPresetMaterialMixed**. Read/write **Long**.

PresetMaterial Property Example

This example specifies that the extrusion surface for shape one in `myDocument` be wire frame.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .PresetMaterial = msoMaterialWireFrame
End With
```

PresetShape Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetShapeA"}

Returns or sets the shape of the specified WordArt. Read/write **Long**.

Can be one of the following **MsoPresetTextEffectShape** constants:

msoTextEffectShapeArchDownCurve
msoTextEffectShapeArchDownPour
msoTextEffectShapeArchUpCurve
msoTextEffectShapeArchUpPour
msoTextEffectShapeButtonCurve
msoTextEffectShapeButtonPour
msoTextEffectShapeCanDown
msoTextEffectShapeCanUp
msoTextEffectShapeCascadeDown
msoTextEffectShapeCascadeUp
msoTextEffectShapeChevronDown
msoTextEffectShapeChevronUp
msoTextEffectShapeCircleCurve
msoTextEffectShapeCirclePour
msoTextEffectShapeCurveDown
msoTextEffectShapeCurveUp
msoTextEffectShapeDeflate
msoTextEffectShapeDeflateBottom
msoTextEffectShapeDeflateInflate
msoTextEffectShapeDeflateInflateDeflate
msoTextEffectShapeDeflateTop
msoTextEffectShapeDoubleWave1
msoTextEffectShapeDoubleWave2
msoTextEffectShapeFadeDown
msoTextEffectShapeFadeLeft
msoTextEffectShapeFadeRight
msoTextEffectShapeFadeUp
msoTextEffectShapeInflate
msoTextEffectShapeInflateBottom
msoTextEffectShapeInflateTop
msoTextEffectShapeMixed
msoTextEffectShapePlainText
msoTextEffectShapeRingInside
msoTextEffectShapeRingOutside
msoTextEffectShapeSlantDown
msoTextEffectShapeSlantUp
msoTextEffectShapeStop
msoTextEffectShapeTriangleDown
msoTextEffectShapeTriangleUp
msoTextEffectShapeWave1
msoTextEffectShapeWave2

Remarks

Setting the **PresetTextEffect** property automatically sets the **PresetShape** property.

PresetShape Property Example

This example sets the shape of all WordArt on `myDocument` to a chevron whose center points down.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    If s.Type = msoTextEffect Then
        s.TextEffect.PresetShape = msoTextEffectShapeChevronDown
    End If
Next
```

PresetTexture Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetTextureC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetTextureX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetTextureA"}

Returns the preset texture for the specified fill. Read-only **Long**.

Can be one of the following **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

This property is read-only. Use the **PresetTextured** method to set the preset texture for the fill.

PresetTexture Property Example

This example adds a rectangle to `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 = ActiveDocument
With myDocument.Shapes
    presetTexture2 = .Item(2).Fill.PresetTexture
    .AddShape(msoShapeRectangle, 100, 0, 40, 80).Fill _
        .PresetTextured presetTexture2
End With
```

PresetTextured Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPresetTexturedC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPresetTexturedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPresetTexturedA"}

Sets the specified fill to a preset texture.

Syntax

expression.**PresetTextured(*PresetTexture*)**

expression Required. An expression that returns a **FillFormat** object.

PresetTexture Required **Long**. The preset texture. Can be one of the following **MsoPresetTexture** constants:

msoTextureBlueTissuePaper	**msoTexturePaperBag**
msoTextureBouquet	**msoTexturePapyrus**
msoTextureBrownMarble	**msoTextureParchment**
msoTextureCanvas	**msoTexturePinkTissuePaper**
msoTextureCork	**msoTexturePurpleMesh**
msoTextureDenim	**msoTextureRecycledPaper**
msoTextureFishFossil	**msoTextureSand**
msoTextureGranite	**msoTextureStationery**
msoTextureGreenMarble	**msoTextureWalnut**
msoTextureMediumWood	**msoTextureWaterDroplets**
msoTextureNewsprint	**msoTextureWhiteMarble**
msoTextureOak	**msoTextureWovenMat**

PresetTextured Method Example

This example adds a rectangle with a green-marble textured fill to `myDocument`.

```
Set myDocument = ActiveDocument
myDocument.Shapes.AddShape(msoShapeCan, 90, 90, 40, 80) _
    .Fill.PresetTextured msoTextureGreenMarble
```

PresetThreeDFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetThreeDFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetThreeDFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetThreeDFormatA"}

Returns the preset extrusion format. Each preset extrusion format contains a set of preset values for the various properties of the extrusion. If the extrusion has a custom format rather than a preset format, this property returns **msoPresetThreeDFormatMixed**. Can be one of the following **MsoPresetThreeDFormat** constants: **msoPresetThreeDFormatMixed**, **msoThreeD1**, **msoThreeD10**, **msoThreeD11**, **msoThreeD12**, **msoThreeD13**, **msoThreeD14**, **msoThreeD15**, **msoThreeD16**, **msoThreeD17**, **msoThreeD18**, **msoThreeD19**, **msoThreeD2**, **msoThreeD20**, **msoThreeD3**, **msoThreeD4**, **msoThreeD5**, **msoThreeD6**, **msoThreeD7**, **msoThreeD8**, or **msoThreeD9**. 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 **Long**.

Remarks

This property is read-only. To set the preset extrusion format, use the **SetThreeDFormat** method.

PresetThreeDFormat Property 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 = ActiveDocument
With myDocument.Shapes(1).ThreeD
    If .PresetThreeDFormat = msoPresetThreeDFormatMixed Then
        .SetThreeDFormat msoThreeD12
    End If
End With
```

ResetRotation Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetRotationC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetRotationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetRotationA"}

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.

Syntax

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.

ResetRotation Method 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 = ActiveDocument
myDocument.Shapes(1).ThreeD.ResetRotation
```

RotatedChars Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRotatedCharsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRotatedCharsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRotatedCharsA"}

True if characters in the specified WordArt are rotated 90 degrees relative to the WordArt's bounding shape. **False** if characters in the specified WordArt retain their original orientation relative to the bounding shape. Read/write **Long**.

Remarks

If the WordArt has horizontal text, setting the **RotatedChars** property to **True** rotates the characters 90 degrees counterclockwise. If the WordArt has vertical text, setting the **RotatedChars** property to **False** 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.

RotatedChars Property Example

This example adds WordArt that contains the text "Test" to `myDocument` and rotates the characters 90 degrees counterclockwise.

```
Set myDocument = ActiveDocument
Set newWordArt =
myDocument.Shapes.AddTextEffect(PresetTextEffect:=msoTextEffect1,
Text:="Test", _
    FontName:="Arial Black", FontSize:=36, FontBold:=False,
FontItalic:=False, Left:=10, Top:=10)
newWordArt.TextEffect.RotatedChars = True
```

RotationY Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRotationYC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRotationYX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRotationYA"}

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.

RotationY Property 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 = ActiveDocument
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
```

SegmentType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSegmentTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSegmentTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSegmentTypeA"}

Returns a value that indicates whether the segment associated with the specified node is straight or curved. Can be either of the following **MsoSegmentType** constants: **msoSegmentCurve** or **msoSegmentLine**. If the specified node is a control point for a curved segment, this property returns **msoSegmentCurve**. Read-only **Long**.

Remarks

This property is read-only. Use the **SetSegmentType** method to set the value of this property.

SegmentType 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 = ActiveDocument
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
```

SetEditingType Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetEditingTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetEditingTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetEditingTypeA"}

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.

Syntax

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 **Long**. The editing property of the vertex. Can be one of the following **MsoEditingType** constants: **msoEditingAuto**, **msoEditingCorner**, **msoEditingSmooth**, or **msoEditingSymmetric**.

SetEditingType Method Example

This example changes all corner nodes to smooth nodes in shape three on `myDocument`. Shape three must be a freeform drawing.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(3).Nodes
    For n = 1 to .Count
        If .Item(n).EditingType = msoEditingCorner Then
            .SetEditingType n, msoEditingSmooth
        End If
    Next
End With
```

SetExtrusionDirection Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetExtrusionDirectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetExtrusionDirectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetExtrusionDirectionA"}

Sets the direction that the extrusion's sweep path takes away from the extruded shape.

Syntax

expression.**SetExtrusionDirection(*PresetExtrusionDirection*)**

expression Required. An expression that returns a **ThreeDFormat** object.

PresetExtrusionDirection Required **Long**. Specifies the extrusion direction. Can be one of the following **MsoPresetExtrusionDirection** constants: **msoExtrusionBottom**, **msoExtrusionBottomLeft**, **msoExtrusionBottomRight**, **msoExtrusionLeft**, **msoExtrusionNone**, **msoExtrusionRight**, **msoExtrusionTop**, **msoExtrusionTopLeft**, or **msoExtrusionTopRight**.

Remarks

This method sets the **PresetExtrusionDirection** property to the direction specified by the ***PresetExtrusionDirection*** argument.

SetExtrusionDirection Method 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 = ActiveDocument
With myDocument.Shapes(1).ThreeD
    .Visible = True
    .SetExtrusionDirection msoExtrusionTop
    .PresetLightingDirection = msoLightingLeft
End With
```

SetPosition Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetPositionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetPositionA"}

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.

Syntax

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.

SetPosition Method 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 = ActiveDocument
With myDocument.Shapes(3).Nodes
    pointsArray = .Item(2).Points
    currXvalue = pointsArray(0, 0)
    currYvalue = pointsArray(0, 1)
    .SetPosition 2, currXvalue + 200, currYvalue + 300
End With
```

SetSegmentType Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetSegmentTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetSegmentTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetSegmentTypeA"}

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.

Syntax

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 **Long**. Specifies if the segment is straight or curved. Can be either of the following **MsoSegmentType** constants: **msoSegmentCurve** or **msoSegmentLine**.

SetSegmentType Method Example

This example changes all straight segments to curved segments in shape three on `myDocument`. Shape three must be a freeform drawing.

```
Set myDocument = ActiveDocument
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
```

SetThreeDFormat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetThreeDFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetThreeDFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetThreeDFormatA"}

Sets the preset extrusion format. Each preset extrusion format contains a set of preset values for the various properties of the extrusion.

Syntax

expression.**SetThreeDFormat(*PresetThreeDFormat*)**

expression Required. An expression that returns a **ThreeDFormat** object.

PresetThreeDFormat Required **Long**. Specifies a preset extrusion format that corresponds to one of the options (numbered from left to right, top to bottom) displayed when you click the **3-D** button on the **Drawing** toolbar. Can be one of the following **MsoPresetThreeDFormat** constants: **msoThreeD1**, **msoThreeD10**, **msoThreeD11**, **msoThreeD12**, **msoThreeD13**, **msoThreeD14**, **msoThreeD15**, **msoThreeD16**, **msoThreeD17**, **msoThreeD18**, **msoThreeD19**, **msoThreeD2**, **msoThreeD20**, **msoThreeD3**, **msoThreeD4**, **msoThreeD5**, **msoThreeD6**, **msoThreeD7**, **msoThreeD8**, or **msoThreeD9**. Note that specifying **msoPresetThreeDFormatMixed** for this argument causes an error.

Remarks

This method sets the **PresetThreeDFormat** property to the format specified by the ***PresetThreeDFormat*** argument.

SetThreeDFormat Method Example

This example adds an oval to `myDocument` and sets its extrusion format to 3D Style 12.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeOval, 30, 30, 50, 25).ThreeD
    .Visible = True
    .SetThreeDFormat msoThreeD12
End With
```

Solid Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSolidC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSolidX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSolidA"}

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.

Syntax

expression.**Solid**

expression Required. An expression that returns a **FillFormat** object.

Solid Method Example

This example converts all fills on `myDocument` to uniform red fills.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    With s.Fill
        .Solid
        .ForeColor.RGB = RGB(255, 0, 0)
    End With
Next
```

TextureName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextureNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextureNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextureNameA"}

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.

TextureName Property Example

This example adds an oval to `myDocument`. If shape two on `myDocument` has a user-defined textured fill, the new oval will have the same fill as shape two. If shape two has any other type of fill, the new oval will have a green marble fill.

```
Set myDocument = ActiveDocument
With myDocument.Shapes
    Set newFill = .AddShape(msoShapeOval, 0, 0, 200, 90).Fill
    With .Item(2).Fill
        If.TextureType = msoTextureUserDefined Then
            newFill.UserTextured .TextureName
        Else
            newFill.PresetTextured msoTextureGreenMarble
        End If
    End With
End With
```

TextureType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextureTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextureTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextureTypeA"}

Returns the texture type for the specified fill. Can be one of the following **MsoTextureType** constants: **msoTexturePreset**, **msoTextureTypeMixed**, or **msoTextureUserDefined**. Read-only **Long**.

This property is read-only. Use the **PresetTextured**, **UserPicture**, or **UserTextured** method to set the texture type for the fill.

TextureType Property Example

This example changes the fill for all shapes on `myDocument` with a custom textured fill to a canvas fill.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    With s.Fill
        If .TextureType = msoTextureUserDefined Then
            .PresetTextured msoTextureCanvas
        End If
    End With
Next
```

Tracking Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTrackingC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTrackingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTrackingA"}

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) thorugh 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

Tracking Property Example

This example adds WordArt that contains the text "Test" to `myDocument` and specifies that the characters be very tightly spaced.

```
Set myDocument = ActiveDocument
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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTransparencyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTransparencyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTransparencyA"}

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.

Transparency Property Example

This example sets the shadow of shape three on `myDocument` to semitransparent red. If the shape doesn't already have a shadow, this example adds one to it.

```
Set myDocument = ActiveDocument
With myDocument.Shapes(3).Shadow
    .Visible = True
    .ForeColor.RGB = RGB(255, 0, 0)
    .Transparency = 0.5
End With
```

TransparencyColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTransparencyColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTransparencyColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTransparencyColorA"}

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.

TransparencyColor Property 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 = ActiveDocument
With myDocument.Shapes(1)
    With .PictureFormat
        .TransparentBackground = True
        .TransparencyColor = blueScreen
    End With
    .Fill.Visible = False
End With
```

TransparentBackground Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTransparentBackgroundC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTransparentBackgroundX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTransparentBackgroundA"}

True if the parts of the picture that are the color defined as the transparent color appear transparent. Use the **TransparencyColor** property to set the transparent color. 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.

TransparentBackground Property 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 = ActiveDocument
With myDocument.Shapes(1)
    With .PictureFormat
        .TransparentBackground = True
        .TransparencyColor = blueScreen
    End With
    .Fill.Visible = False
End With
```

TwoColorGradient Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTwoColorGradientC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTwoColorGradientX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTwoColorGradientA"}

Sets the specified fill to a two-color gradient.

Syntax

expression.**TwoColorGradient(*Style, Variant*)**

expression Required. An expression that returns a **FillFormat** object.

Style Required **Long**. The gradient style. Can be one of the following **MsoGradientStyle** constants: **msoGradientDiagonalDown**, **msoGradientDiagonalUp**, **msoGradientFromCenter**, **msoGradientFromCorner**, **msoGradientHorizontal**, or **msoGradientVertical**. (The constant **msoGradientFromTitle** is used only in PowerPoint.)

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 **msoGradientFromCenter**, this argument can be either 1 or 2.

TwoColorGradient Method 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 = ActiveDocument
With myDocument.Shapes.AddShape(msoShapeRectangle, 0, 0, 40, 80).Fill
    .ForeColor.RGB = RGB(128, 0, 0)
    .BackColor.RGB = RGB(0, 170, 170)
    .TwoColorGradient msoGradientHorizontal, 1
End With
```

UserPicture Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUserPictureC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUserPictureX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUserPictureA"}

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.

Syntax

expression.**UserPicture(*PictureFile*)**

expression Required. An expression that returns a **FillFormat** object.

PictureFile Required **String**. The name of the picture file.

UserPicture Method 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 = ActiveDocument
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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUserTexturedC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUserTexturedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUserTexturedA"}

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.

Syntax

expression.**UserTextured(*TextureFile*)**

expression Required. An expression that returns a **FillFormat** object.

TextureFile Required **String**. The name of the picture file.

UserTextured Method 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 = ActiveDocument
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
```

Weight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWeightA"}

Returns or sets the thickness of the specified line, in points. Read/write **Single**.

Weight Property Example

This example adds a green dashed line two points thick to `myDocument`.

```
Set myDocument = ActiveDocument
With myDocument.Shapes.AddLine(10, 10, 250, 250).Line
    .DashStyle = msoLineDashDotDot
    .ForeColor.RGB = RGB(0, 255, 255)
    .Weight = 2
End With
```

PresetTextEffect Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPresetTextEffectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPresetTextEffectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPresetTextEffectA"}

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 **Long**.

Can be one of the following **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

Remarks

Setting the **PresetTextEffect** property automatically sets many other formatting properties of the specified shape.

PresetTextEffect Property Example

This example sets the style for all WordArt on `myDocument` to the first style listed in the **WordArt Gallery** dialog box.

```
Set myDocument = ActiveDocument
For Each s In myDocument.Shapes
    If s.Type = msoTextEffect Then
        s.TextEffect.PresetTextEffect = msoTextEffect1
    End If
Next
```

ScaleHeight Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthScaleHeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthScaleHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthScaleHeightA"}

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.

Syntax

expression.**ScaleHeight(*Factor, RelativeToOriginalSize, Scale*)**

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 **Long**. **True** to scale the shape relative to its original size. **False** to scale it relative to its current size. You can specify **True** for this argument only if the specified shape is a picture or an OLE object.

Scale **Optional** Long**. The part of the shape that retains its position when the shape is scaled. Can be one of the following** MsoScaleFrom **constants:** msoScaleFromBottomRight**,** msoScaleFromMiddle**, or** msoScaleFromTopLeft**. The default value is** msoScaleFromTopLeft**.**

ScaleHeight Method 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 = ActiveDocument
For Each s In myDocument.Shapes
    Select Case s.Type
    Case msoEmbeddedOLEObject, msoLinkedOLEObject, msoOLEControlObject, _
            msoLinkedPicture, msoPicture
        s.ScaleHeight 1.75, True
        s.ScaleWidth 1.75, True
    Case Else
        s.ScaleHeight 1.75, False
        s.ScaleWidth 1.75, False
    End Select
Next
```

ScaleWidth Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthScaleWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthScaleWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthScaleWidthA"}

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.

Syntax

expression.**ScaleWidth(*Factor, RelativeToOriginalSize, Scale*)**

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 **Long**. **True** to scale the shape relative to its original size. **False** to scale it relative to its current size. You can specify **True** for this argument only if the specified shape is a picture or an OLE object.

Scale **Optional** Long**. The part of the shape that retains its position when the shape is scaled. Can be one of the following** MsoScaleFrom **constants:** msoScaleFromBottomRight**,** msoScaleFromMiddle**, or** msoScaleFromTopLeft**. The default value is** msoScaleFromTopLeft**.**

ScaleWidth Method 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 = ActiveDocument
For Each s In myDocument.Shapes
    Select Case s.Type
    Case msoEmbeddedOLEObject, msoLinkedOLEObject, msoOLEControlObject, _
            msoLinkedPicture, msoPicture
        s.ScaleHeight 1.75, True
        s.ScaleWidth 1.75, True
    Case Else
        s.ScaleHeight 1.75, False
        s.ScaleWidth 1.75, False
    End Select
Next
```

ToggleVerticalText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthToggleVerticalTextC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthToggleVerticalTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthToggleVerticalTextA"}

Switches the text flow in the specified WordArt from horizontal to vertical, or vice versa.

Syntax

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.

ToggleVerticalText Method 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 = ActiveDocument
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
```

Regroup Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRegroupC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRegroupX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRegroupA"}

Regroups the group that the specified shape range belonged to previously. Returns the regrouped shapes as a single **Shape** object.

Syntax

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.

Regroup Method 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
```

CropBottom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCropBottomC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCropBottomX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCropBottomA"}

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.

CropBottom Property Example

This example crops 20 points off the bottom of shape three on the active document. For the example to work, shape three must be either a picture or an OLE object.

```
ActiveDocument.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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCropLeftC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCropLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCropLeftA"}

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.

CropLeft Property Example

This example crops 20 points off the left side of shape three on the active document. For the example to work, shape three must be either a picture or an OLE object.

```
ActiveDocument.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
    origWidth = .Width
    .Delete
End With
cropPoints = origWidth * percentToCrop / 100
shapeToCrop.PictureFormat.CropLeft = cropPoints
```

CropRight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCropRightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCropRightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCropRightA"}

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.

CropRight Property Example

This example crops 20 points off the right side of shape three on the active document. For this example to work, shape three must be either a picture or an OLE object.

```
ActiveDocument.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

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCropTopC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCropTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCropTopA"}

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.

CropTop Property Example

This example crops 20 points off the top of shape three on the active document. For the example to work, shape three must be either a picture or an OLE object.

```
ActiveDocument.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
```

Distribute Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDistributeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDistributeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDistributeA"}

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 page or just over the space they originally occupy.

Syntax

expression.**Distribute(*DistributeCmd, RelativeTo*)**

expression Required. An expression that returns a **ShapeRange** object.

DistributeCmd Required **Long**. Specifies whether shapes in the range are to be distributed horizontally or vertically. Can be either of the following **MsoDistributeCmd** constants: **msoDistributeHorizontally** or **msoDistributeVertically**.

RelativeTo Required **Long**. **True** to distribute the shapes evenly over the entire horizontal or vertical space on the page. **False** to distribute them within the horizontal or vertical space that the range of shapes originally occupies.

Distribute Method Example

This example defines a shape range that contains all the AutoShapes on the active document and then horizontally distributes the shapes in this range.

```
With ActiveDocument.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, False
        End If
    End If
End With
```

Length Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLengthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLengthA"}

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 **Single**.

Remarks

This property is read-only. Use the **CustomLength** method to set the value of this property for the **CalloutFormat** object.

Length Property Example

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.

```
With ActiveDocument.Shapes
    With .Item("co1").Callout
        If Not .AutoLength Then len1 = .Length
    End With
    If len1 Then .Item("co2").Callout.CustomLength len1
End With
```

RotationX Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRotationXC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRotationXX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRotationXA"}

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.

RotationX Property Example

This example adds three identical extruded ovals to the active document and sets their rotation around the x-axis to – 30, 0, and 30 degrees, respectively.

```
With ActiveDocument.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
```

AddressStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAddressStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAddressStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAddressStyleA "}

Returns a **Style** object that represents the delivery address style for the envelope. Read-only

Note If an envelope is added to the document, text formatted with the Envelope Address style is automatically updated.

AddressStyle Property Example

This example modifies the font formatting associated with the Envelope Address style.

```
With ActiveDocument.Envelope.AddressStyle.Font
    .Bold = False
    .Name = "Times New Roman"
    .Size = 16
End With
```

ReturnAddressStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnAddressStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnAddressStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnAddressStyleA "}

Returns a **Style** object that represents the return address style for the envelope. Read-only

Note If an envelope is added to the document, text formatted with the Envelope Return style is automatically updated.

ReturnAddressStyle Property Example

This example displays the style name and description of the envelope return address.

```
Set myStyle = ActiveDocument.Envelope.ReturnAddressStyle
MsgBox Prompt:=myStyle.Description, Title:=myStyle.NameLocal
```

This example sets the line spacing and space-after formatting for the envelope return address style.

```
With ActiveDocument.Envelope.ReturnAddressStyle.ParagraphFormat
    .LineSpacingRule = wdLineSpaceExactly
    .LineSpacing = 13
    .SpaceAfter = 6
End With
```

CustomLabels Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCustomLabelsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCustomLabelsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCustomLabelsA "}

Returns a **CustomLabels** collection that represents the available custom mailing labels. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

CustomLabels Property Example

This example creates a new custom label named "AdminAddress" and then creates a page of mailing labels using a predefined return address.

```
addr = "Administration" & vbCr & "Mail Stop 22-16"
Set myLabel = Application.MailingLabel _
    .CustomLabels.Add(Name:="AdminAddress", DotMatrix:= False)
With myLabel
    .Height = InchesToPoints(0.5)
    .Width = InchesToPoints(1)
    .HorizontalPitch = InchesToPoints(2.06)
    .VerticalPitch = InchesToPoints(0.5)
    .NumberAcross = 4
    .NumberDown = 20
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.28)
    .TopMargin = InchesToPoints(0.5)
End With
Application.MailingLabel.CreateNewDocument Name:="AdminAddress",
Address:=addr
```

DefaultFaceUp Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultFaceUpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultFaceUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultFaceUpA "}

True if envelopes are fed face up by default. Read/write **Boolean**.

DefaultFaceUp Property Example

This example sets envelopes to be fed face up by default. The **UpdateDocument** method updates the envelope in the active document.

```
With ActiveDocument.Envelope
    .DefaultFaceUp = True
    .DefaultOrientation = wdCenterPortrait
    .UpdateDocument
End With
```

This example displays a message telling the user how to feed the envelopes into the printer based on the default setting.

```
If ActiveDocument.Envelope.DefaultFaceUp = True Then
    MsgBox "Feed envelopes face up."
Else
    MsgBox "Feed envelopes face down."
End If
```

DefaultOrientation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultOrientationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultOrientationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultOrientationA "}

Returns or sets the default orientation for feeding envelopes. Can be one of the following **WdEnvelopeOrientation** constants: **wdCenterClockwise**, **wdCenterLandscape**, **wdCenterPortrait**, **wdLeftClockwise**, **wdLeftLandscape**, **wdLeftPortrait**, **wdRightClockwise**, **wdRightLandscape**, or **wdRightPortrait**. Read/write **Long**

DefaultOrientation Property Example

This example sets envelopes to be fed face up, centered, and in portrait orientation.

```
With ActiveDocument.Envelope
    .DefaultFaceUp = True
    .DefaultOrientation = wdCenterPortrait
    MsgBox "Feed envelopes centered, face up, in portrait orientation"
End With
```

DefaultWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultWidthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultWidthA "}

Returns or sets the default envelope width, in points. Read/write **Single**

Note If you set the **DefaultHeight** or **DefaultWidth** property, the envelope size is automatically changed to Custom Size in the **Envelope Options** dialog box (**Tools** menu). Use the **DefaultSize** property to set the default size to a predefined size.

DefaultWidth Property Example

This example sets the default custom envelope width and height and adds an envelope to the active document.

```
addr = "Tim O' Brien " & vbCr & "123 Skye St." & vbCr & "Bellevue, WA
98004"
ret = "Dave Edson" & vbCr & "123 West Main" & vbCr & "Seattle, WA  98004"
With ActiveDocument.Envelope
    .DefaultWidth = InchesToPoints(9)
    .DefaultHeight = InchesToPoints(3.85)
End With
ActiveDocument.Envelope.Insert Address:=addr, ReturnAddress:=ret
```

UpdateDocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateDocumentA "}

Updates the envelope in the document with the current envelope settings.

Note If you use this property before an envelope has been added to the document, an error occurs.

Syntax

expression.**UpdateDocument**

expression Required. An expression that returns an **Envelope** object.

UpdateDocument Method Example

This example formats the envelope in Report.doc to use a custom envelope size (4.5 inches by 7.5 inches).

```
On Error GoTo errhandler
With Documents("Report.doc").Envelope
    .DefaultHeight = InchesToPoints(4.5)
    .DefaultWidth = InchesToPoints(7.5)
    .UpdateDocument
End With
errhandler:
If Err = 5852 Then MsgBox "Report.doc doesn't include an envelope"
```

This example adds an envelope to the active document, using predefined addresses. The default envelope bar code and Facing Identification Mark (FIM-A) settings are set to **True**, and the envelope in the active document is updated.

```
addr = "Darlene Rudd" & vbCr & "1234 E. Main St." & vbCr & "Our Town, WA
98004"
ret = "Patricia Reed" & vbCr & "N. 33rd St." & vbCr & "Other Town, WA
98040"
ActiveDocument.Envelope.Insert Address:=addr, ReturnAddress:=ret
With ActiveDocument.Envelope
    .DefaultPrintBarCode = True
    .DefaultPrintFIMA = True
    .UpdateDocument
End With
```

DefaultLaserTray Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultLaserTrayC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultLaserTrayX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultLaserTrayA"}

Returns or sets the default paper tray that contains sheets of mailing labels. Read/write **Long**.

Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed
wdPrinterDefaultBin
wdPrinterEnvelopeFeed
wdPrinterFormSource
wdPrinterLargeCapacityBin
wdPrinterLargeFormatBin
wdPrinterLowerBin
wdPrinterManualEnvelopeFeed
wdPrinterManualFeed
wdPrinterMiddleBin
wdPrinterOnlyBin
wdPrinterPaperCassette
wdPrinterSmallFormatBin
wdPrinterTractorFeed
wdPrinterUpperBin

DefaultLaserTray Property Example

This example checks to determine whether the mailing label printer is set for feed labels manually, and then it displays a message on the status bar.

```
If Application.MailingLabel.DefaultLaserTray = wdPrinterManualEnvelopeFeed
Then
    StatusBar = "Printer set for feeding labels manually"
Else
    StatusBar = "Check printer paper tray setting"
End If
```

This example sets the mailing-label paper tray to the upper bin.

```
Application.MailingLabel.DefaultLaserTray = wdPrinterUpperBin
```

AddressFromLeft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAddressFromLeftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAddressFromLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAddressFromLeftA "}

Returns or sets the distance (in points) between the left edge of the envelope and the delivery address. Read/write **Single**.

Note If you use this property before an envelope has been added to the document, an error occurs.

AddressFromLeft Property Example

This example creates a new document and adds an envelope with a predefined delivery address and return address. The example then sets the distance between the left edge of the envelope and the delivery address to 3.75 inches.

```
addr = "James Allard" & vbCr & "123 Skye St." & vbCr & "Our Town, WA
98004"
retaddr = "Rich Andrews" & vbCr & "123 Main" & vbCr & "Other Town, WA
98004"
With Documents.Add.Envelope
    .Insert Address:=addr, ReturnAddress:=retaddr
    .AddressFromLeft = InchesToPoints(3.75)
End With
ActiveWindow.View.Type = wdPageView
```

AddressFromTop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAddressFromTopC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAddressFromTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAddressFromTopA "}

Returns or sets the distance (in points) between the top edge of the envelope and the delivery address. Read/write **Single**.

Note If you use this property before an envelope has been added to the document, an error occurs.

AddressFromTop Property Example

This example creates a new document and adds an envelope with a predefined delivery address and return address. The example then sets the distance between the top edge of the envelope and the delivery address to 1.75 inches and sets the distance between the left edge of the envelope and the delivery address is set to 3.75 inches.

```
addr = "Michael Bunney" & vbCr & "123 Skye St." & vbCr & "Our Town, WA
98040"
retaddr = "Kate Dresen" & vbCr & "123 Main" & vbCr & "Other Town, WA
98040"
With Documents.Add.Envelope
    .Insert Address:=addr, ReturnAddress:=retaddr
    .AddressFromTop = InchesToPoints(1.75)
    .AddressFromLeft = InchesToPoints(3.75)
End With
ActiveWindow.View.Type = wdPageView
```

CreateNewDocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreateNewDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreateNewDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreateNewDocumentA "}

Creates a new label document using either the default label options or ones that you specify. Returns a **Document** object.

Syntax

expression.**CreateNewDocument(*Name, Address, AutoText*, *ExtractAddress, LaserTray*)**

expression Required. An expression that returns a **MailingLabel** object.

Name Optional **Variant**. The mailing label name.

Address Optional **Variant**. The text for the mailing label.

AutoText Optional **Variant**. The name of the AutoText entry that includes the mailing label text.

ExtractAddress Optional **Variant**. **True** to use the address text marked by the user-defined bookmark named "EnvelopeAddress" instead of using the ***Address*** argument.

LaserTray Optional **Variant**. The laser printer tray. Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed	**wdPrinterManualFeed**
wdPrinterDefaultBin	**wdPrinterMiddleBin**
wdPrinterEnvelopeFeed	**wdPrinterOnlyBin**
wdPrinterFormSource	**wdPrinterPaperCassette**
wdPrinterLargeCapacityBin	**wdPrinterSmallFormatBin**
wdPrinterLargeFormatBin	**wdPrinterTractorFeed**
wdPrinterLowerBin	**wdPrinterUpperBin**
wdPrinterManualEnvelopeFeed	

CreateNewDocument Method Example

This example creates a new Avery 2160 minilabel document using a predefined address.

```
addr = "Dave Edson" & vbCr & "123 Skye St." & vbCr & "Our Town, WA  98004"
Application.MailingLabel.CreateNewDocument _
    Name:="2160 mini", Address:=addr, ExtractAddress:=False
```

This example creates a new Avery 5664 shipping-label document using the selected text as the address.

```
addr = Selection.Text
Application.MailingLabel.CreateNewDocument Name:="5664", Address:=addr, _
    LaserTray:=wdPrinterUpperBin
```

This example creates a new self-adhesive-label document using the EnvelopeAddress bookmark text as the address.

```
If ActiveDocument.Bookmarks.Exists("EnvelopeAddress") = True Then
    Application.MailingLabel.CreateNewDocument _
        Name:="Self Adhesive Tab 1 1/2""", ExtractAddress:=True
End If
```

DefaultHeight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultHeightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultHeightA "}

Returns or sets the default envelope height, in points. Read/write **Single**.

Note If you set either the **DefaultHeight** or **DefaultWidth** property, the envelope size is automatically changed to Custom Size in the **Envelope Options** dialog box (**Tools** menu). Use the **DefaultSize** property to set the default size to a predefined size.

DefaultHeight Property Example

This example sets the default envelope size to 4.5 inches by 7.5 inches.

```
With ActiveDocument.Envelope
    .DefaultHeight = InchesToPoints(4.5)
    .DefaultWidth = InchesToPoints(7.5)
End With
```

DefaultOmitReturnAddress Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultOmitReturnAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultOmitReturnAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultOmitReturnAddressA "}

True if the return address is omitted from envelopes by default. Read/write **Boolean**.

DefaultOmitReturnAddress Property Example

This example omits return addresses from new envelopes by default.

```
ActiveDocument.Envelope.DefaultOmitReturnAddress = True
```

This example displays the return address status in a message box.

```
If ActiveDocument.Envelope.DefaultOmitReturnAddress = True Then
    MsgBox "A return address is not included by default."
Else
    MsgBox "A return address is included by default."
End If
```

DefaultPrintBarCode Property

This item is not available for this version of Office.

DefaultPrintBarCode Property Example

This item is not available for this version of Office.

DefaultPrintFIMA Property

This item is not available for this version of Office.

DefaultPrintFIMA Property Example

This item is not available for this version of Office.

DotMatrix Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDotMatrixC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDotMatrixX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDotMatrixA "}

True if the printer type for the specified custom label is dot matrix. **False** if the printer type is either laser or ink jet. Read-only **Boolean**.

DotMatrix Property Example

This example displays the name and printer type of the first custom mailing label.

```
Set myLabel = Application.MailingLabel
If myLabel.CustomLabels.Count >= 1 Then
    If myLabel.CustomLabels(1).DotMatrix = True Then
        MsgBox myLabel.CustomLabels(1).Name & " is dot matrix"
    Else
        MsgBox myLabel.CustomLabels(1).Name & " is laser or ink jet"
    End If
End If
```

Envelope Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnvelopeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnvelopeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnvelopeA "}

Returns an **Envelope** object that represents envelope functionality and the envelope in the specified document. Read-only.

Envelope Property Example

This example sets the default envelope size to C4 (229 x 324 mm).

```
ActiveDocument.Envelope.DefaultSize = "C4"
```

This example displays the delivery address if an envelope has been added to the document; otherwise, a message box is displayed.

```
On Error GoTo errhandler
addr = ActiveDocument.Envelope.Address.Text
MsgBox Prompt:=addr, Title:="Delivery Address"
errhandler:
If Err = 5852 Then MsgBox "Add an envelope to the document"
```

This example creates a new document and adds an envelope with a predefined delivery address and return address.

```
addr = "Don Funk" & vbCr & "123 Skye St." & vbCr & "Our Town, WA  98040"
retaddr = "Karin Gallagher" & vbCr & "123 Main" & vbCr & "Other Town, WA
98004"
Documents.Add.Envelope.Insert Address:=addr, ReturnAddress:=retaddr
ActiveWindow.View.Type = wdPageView
```

FeedSource Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFeedSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFeedSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFeedSourceA "}

Returns or sets the paper tray for the envelope. Read/write **Long**.

Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed
wdPrinterManualFeed
wdPrinterDefaultBin
wdPrinterMiddleBin
wdPrinterEnvelopeFeed
wdPrinterOnlyBin
wdPrinterFormSource
wdPrinterPaperCassette
wdPrinterLargeCapacityBin
wdPrinterSmallFormatBin
wdPrinterLargeFormatBin
wdPrinterTractorFeed
wdPrinterLowerBin
wdPrinterUpperBin
wdPrinterManualEnvelopeFeed

Note If you use this property before an envelope has been added to the document, an error occurs.

FeedSource Property Example

This example asks the user whether envelopes are fed into the printer manually. If the answer is yes, the example sets the paper tray to manual envelope feed.

```
response = MsgBox("Are the envelopes manually fed?", vbYesNo)
If response = vbYes then
    On Error GoTo errhandler
    ActiveDocument.Envelope.FeedSource = wdPrinterManualEnvelopeFeed
    errhandler:
    If Err = 5852 Then MsgBox "Envelope not part of the active document"
End If
```

HorizontalPitch Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalPitchC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalPitchX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalPitchA "}

Returns or sets the horizontal distance (in points) between the left edge of one custom mailing label and the left edge of the next mailing label. Read/write **Single**.

Note If this property is changed to a value that isn't valid for the specified mailing label layout, an error occurs.

HorizontalPitch Property Example

This example defines the layout of the custom label named "Laser labels." The horizontal distance between the left edge of one label and the left edge of the next label is set to 4.19 inches.

```
With Application.MailingLabel.CustomLabels("Laser labels")
    .Height = InchesToPoints(2)
    .HorizontalPitch = InchesToPoints(4.19)
    .NumberAcross = 2
    .NumberDown = 5
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.16)
    .TopMargin = InchesToPoints(0.5)
    .VerticalPitch = InchesToPoints(2)
    .Width = InchesToPoints(4)
End With
```

NumberAcross Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberAcrossC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberAcrossX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberAcrossA "}

Returns or sets the number of custom mailing labels across a page. Read/write **Long**.

Note If this property is changed to a value that isn't valid for the specified mailing label layout, an error occurs.

NumberAcross Property Example

This example creates a new custom label named "Dept. Labels" and defines the layout, including the number of labels across the page.

```
Set myLabel = Application.MailingLabel.CustomLabels.Add(Name:="Dept.
Labels", _
    DotMatrix:=False)
With myLabel
    .Height = InchesToPoints(0.5)
    .HorizontalPitch = InchesToPoints(2.06)
    .NumberAcross = 4
    .NumberDown = 20
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.28)
    .TopMargin = InchesToPoints(0.5)
    .VerticalPitch = InchesToPoints(2)
    .Width = InchesToPoints(1.75)
End With
```

NumberDown Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberDownC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberDownX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberDownA "}

Returns or sets the number of custom mailing labels down the length of a page. Read/write **Long**.

Note If this property is changed to a value that isn't valid for the specified mailing label layout, an error occurs.

NumberDown Property Example

This example displays the number of labels across and down the page for the first custom label in the **CustomLabels** collection.

```
numAcr = Application.MailingLabel.CustomLabels(1).NumberAcross
numDwn = Application.MailingLabel.CustomLabels(1).Numberdown
MsgBox  Prompt:= "Number of labels across " & numAcr & vbCr & _
    "Number of labels down " & numDwn & vbCr , _
    Title:="Label Page Configuration"
```

ReturnAddress Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnAddressA "}

Envelope object: Returns a **Range** object that represents the envelope return address. Read-only.

LetterContent object: Returns or sets the return address for a letter created with the Letter Wizard. Read/write **String**.

Note If you use this property before an envelope has been added to the document, an error occurs.

ReturnAddress Property Example

This example displays the return address if an envelope has been added to the active document; otherwise, a message box is displayed.

```
On Error GoTo errhandler
addr = ActiveDocument.Envelope.ReturnAddress.Text
MsgBox Prompt:=addr, Title:="Return Address"
errhandler:
If Err = 5852 Then MsgBox "The active document doesn't include an envelope"
```

This example creates a new **LetterContent** object, sets the return address and several other properties, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Dim oLC as New LetterContent
With oLC
    .LetterStyle = wdFullBlock
    .Salutation ="Hello"
    .SalutationType = wdSalutationOther
    .ReturnAddress = Application.UserAddress
End With
Documents.Add.RunLetterWizard LetterContent:=oLC
```

ReturnAddressFromLeft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnAddressFromLeftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnAddressFromLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnAddressFromLeftA "}

Returns or sets the distance (in points) between the left edge of the envelope and the return address. Read/write **Single**.

Note If you use this property before an envelope has been added to the document, an error occurs.

ReturnAddressFromLeft Property Example

This example creates a new document and adds an envelope with a predefined delivery address and return address. The example then sets the distance between the left edge of the envelope and the return address to 0.75 inch.

```
addr = "Karin Gallagher" & vbCr & "123 Skye St." & vbCr & "Our Town, WA
98004"
retaddr = "Don Funk" & vbCr & "123 Main" & vbCr & "Other Town, WA  98040"
With Documents.Add.Envelope
    .Insert Address:=addr, ReturnAddress:=retaddr
    .ReturnAddressFromLeft = InchesToPoints(0.75)
End With
ActiveWindow.View.Type = wdPageView
```

ReturnAddressFromTop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnAddressFromTopC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnAddressFromTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnAddressFromTopA "}

Returns or sets the distance (in points) between the top edge of the envelope and the return address. Read/write **Single**.

Note If you use this property before an envelope has been added to the document, an error occurs.

ReturnAddressFromTop Property Example

This example creates a new document and adds an envelope with a predefined delivery address and return address. The example then sets the distance between the top edge of the envelope and the return address to 0.5 inch and sets the distance between the left edge of the envelope and the return address to 0.75 inch.

```
addr = "Eric Lang" & vbCr & "123 Main" & vbCr & "Seattle, WA  98040"
retaddr = "Nate Sun" & vbCr & "123 Main" & vbCr & "Bellevue, WA  98004"
With Documents.Add.Envelope
    .Insert Address:=addr, ReturnAddress:=retaddr
    .ReturnAddressFromTop = InchesToPoints(0.5)
    .ReturnAddressFromLeft = InchesToPoints(0.75)
End With
```

SideMargin Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSideMarginC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSideMarginX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSideMarginA "}

Returns or sets the side margin widths (in points) for the specified custom mailing label. Read/write **Single**.

Note If this property is changed to a value that isn't valid for the specified mailing label layout, an error occurs.

SideMargin Property Example

This example creates a custom label named "VisitorPass" and defines its layout. The left and right margins for each label are 0.75 inch.

```
Set myLabel =
Application.MailingLabel.CustomLabels.Add(Name:="VisitorPass", _
    DotMatrix:=False)
With myLabel
    .Height = InchesToPoints(2.17)
    .HorizontalPitch = InchesToPoints(3.5)
    .NumberAcross = 2
    .NumberDown = 4
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.75)
    .TopMargin = InchesToPoints(0.17)
    .VerticalPitch = InchesToPoints(2.17)
    .Width = InchesToPoints(3.5)
End With
```

VerticalPitch Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalPitchC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalPitchX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalPitchA "}

Returns or sets the vertical distance between the top of one mailing label and the top of the next mailing label. Read/write **Single**.

Note If this property is changed to a value that isn't valid for the specified mailing label layout, an error occurs.

VerticalPitch Property Example

This example creates a custom label named "VisitorPass" and defines its layout. The distance between the top edge of one label to the top edge of the next label is 2.17 inches.

```
Set myLabel =
Application.MailingLabel.CustomLabels.Add(Name:="VisitorPass", _
    DotMatrix:=False)
With myLabel
    .Height = InchesToPoints(2.17)
    .HorizontalPitch = InchesToPoints(3.5)
    .NumberAcross = 2
    .NumberDown = 4
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0.75)
    .TopMargin = InchesToPoints(0.17)
    .VerticalPitch = InchesToPoints(2.17)
    .Width = InchesToPoints(3.5)
End With
```

Close Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtCloseC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtCloseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtCloseA "}

Occurs when a document is closed.

Syntax

Private Sub Document_Close()

Remarks

If the event procedure is stored in a template, the procedure will run when a new document based on that template is closed and when the template itself is closed (after being opened as a document).

For information about using events with a **Document** object, see Using Events with the Document Object.

Close Event Example

This example makes a backup copy of the document on a file server when the document is closed. (The procedure is stored in a document, not a template.)

```
Private Sub Document_Close()
    ThisDocument.Save
    ThisDocument.SaveAs "\\network\backup\" & ThisDocument.Name
End Sub
```

DocumentChange Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtDocumentChangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtDocumentChangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtDocumentChangeA "}

Occurs when a new document is created, when an existing document is opened, or when another document is made the active document.

Syntax

Private Sub *object*_**DocumentChange()**

object 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.

DocumentChange Event Example

This example arranges all open windows when the document focus changes.

```
Private Sub myApp_DocumentChange()
    Application.Windows.Arrange wdTiled
End Sub
```

New Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtNewC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtNewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtNewA "}

Occurs when a new document based on the template is created. A procedure for the **New** event will run only if it is stored in a template.

Syntax

Private Sub Document_New()

Remarks

For information about using events with the **Document** object, see Using Events with the Document Object.

New Event Example

This example arranges all open windows when a new document based on the tempate is created. (This procedure is stored in a template, not a document.)

```
Private Sub Document_New()
    Application.Windows.Arrange wdTiled
End Sub
```

Quit Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtQuitC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtQuitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtQuitA "}

Occurs when the user quits Word.

Syntax

Private Sub *object*_**Quit()**

object 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.

Quit Event Example

This example ensures that the **Standard** and **Formatting** toolbars are visible before the user quits Word. As a result, when Word is started again, the **Standard** and **Formatting** toolbars will be visible.

```
Private Sub myApp_Quit()
    CommandBars("Standard").Visible = True
    CommandBars("Formatting").Visible = True
End Sub
```

Open Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtOpenC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtOpenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtOpenA "}

Occurs when a document is opened.

Syntax

Private Sub Document_Open()

Remarks

If the event procedure is stored in a template, the procedure will run when a new document based on that template is opened and when the template itself is opened as a document.

For information about using events with the **Document** object, see Using Events with the Document Object.

Open Event Example

This example displays a message when a document is opened.

```
Private Sub Document_Open()
    Msgbox "This document is copywrited"
End Sub
```

GotFocus Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtGotFocusC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtGotFocusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtGotFocusA "}

Occurs when the focus is moved to an embedded ActiveX control.

For information about using events with ActiveX controls, see Using Events with ActiveX Controls.

Syntax

Private Sub *object*_**GotFocus()**

object The name of an ActiveX control.

GotFocus Event Example

This example displays a message when the focus moves to CheckBox1.

```
Private Sub CheckBox1_GotFocus()
    Msgbox "Check this box if you are a permanent employee."
End Sub
```

LostFocus Event

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woevtLostFocusC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woevtLostFocusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woevtLostFocusA "}

Occurs when the focus is moved from an embedded ActiveX control.

For information about using events with ActiveX controls, see Using Events with ActiveX Controls.

Syntax

Private Sub *object*_**LostFocus()**

object The name of an ActiveX control.

LostFocus Event Example

This example leaves CommandButton1 disabled until the user enters a value in TextBox1.

```
Private Sub TextBox1_LostFocus()
    If TextBox1.Value = "" Then
        CommandButton1.Enabled = False
    Else
        CommandButton1.Enabled = True
    End If
End Sub
```

IncludeHiddenText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludeHiddenTextC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludeHiddenTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludeHiddenTextA"}

True if the text retrieved from the specified range includes hidden text. Read/write **Boolean**.

Note The default value is the same as the current setting of the **Hidden text** option on the **View** tab in the **Options** dialog box (**Tools** menu) until this property has been set. Use the **Text** property with a **Range** object to retrieve text from the specified range.

IncludeHiddenText Property Example

This example displays the text of the first sentence in the active document in a message box. The example uses the **IncludeHiddenText** property to include hidden text.

```
Set myRange = ActiveDocument.Sentences(1)
myRange.TextRetrievalMode.IncludeHiddenText = True
MsgBox myRange.Text
```

This example posts a message if the entire selection is formatted as hidden text.

```
If Selection.Type = wdSelectionNormal Then
    Set myRange = Selection.Range
    myRange.TextRetrievalMode.IncludeHiddenText = False
    If myRange.Text = "" Then MsgBox "Selection is hidden"
End If
```

IncludeFieldCodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludeFieldCodesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludeFieldCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludeFieldCodesA"}

True if the text retrieved from the specified range includes field codes. Read/write **Boolean**.

Note The default value is the same as the setting of the **Field codes** option on the **View** tab in the **Options** dialog box (**Tools** menu) until this property has been set. Use the **Text** property with a **Range** object to retrieve text from the specified range.

IncludeFieldCodes Property Example

This example displays the text of the first paragraph in the active document in a message box. The example uses the **IncludeFieldCodes** property to exclude field codes.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.TextRetrievalMode.IncludeFieldCodes = False
MsgBox myRange.Text
```

This example excludes fields codes and hidden text from the range that refers to the selected text, and then it displays the text in a message box.

```
If Selection.Type = wdSelectionNormal Then
    Set myRange = Selection.Range
    With myRange.TextRetrievalMode
        .IncludeHiddenText = False
        .IncludeFieldCodes = False
    End With
    MsgBox myRange.Text
End If
```

Code Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCodeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCodeA "}

Returns a **Range** object that represents a field's code. A field's code is everything that's enclosed by the field characters (**{ }**) including the leading space and trailing space characters. You can access a field's code without changing the view from field results. Read/write.

Code Property Example

This example displays the field code for each field in the active document.

```
For Each aField In ActiveDocument.Fields
    MsgBox Chr(34) & aField.Code.Text & Chr(34)
Next aField
```

This example changes the field code for the first field in the active document to CREATEDATE.

```
Set myRange = ActiveDocument.Fields(1).Code
myRange.Text = " CREATEDATE "
ActiveDocument.Fields(1).Update
```

This example determines whether the active document includes a mail merge field named "Title."

```
For Each aField In ActiveDocument.MailMerge.Fields
    If InStr(1, aField.Code.Text, "Title", 1) Then
        MsgBox "A Title merge field is in this document"
    End If
Next aField
```

Fields Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFieldsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFieldsA "}

Document, **Range**, or **Selection** object: Returns a **Fields** collection that represents all the fields in the document, range, or selection. Read-only.

Note When applied to the **Document** object, the **Fields** property returns a **Fields** collection that contains only the fields in the main text story.

MailMerge object: Returns a **MailMergeFields** collection that represents all the mail merge related fields in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Fields Property Example

This example updates all the fields in the active document.

```
ActiveDocument.Fields.Update
```

This example removes all the fields from the main text story and the footer in the active document.

```
For Each aField in ActiveDocument.Fields
    aField.Delete
Next aField
Set myRange = ActiveDocument.Sections(1).Footers _
    (wdHeaderFooterPrimary).Range
For Each aField In myRange.Fields
    aField.Delete
Next aField
```

This example adds a DATE field at the insertion point.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .Fields.Add Range:=Selection.Range, Type:=wdFieldDate
End With
```

This example adds a mail merge field named "Title" at the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.MailMerge.Fields.Add Range:= Selection.Range, _
    Name:= "Title"
```

Result Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproResultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproResultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproResultA "}

Field object: Returns a **Range** object that represents a field's result. You can access a field result without changing the view from field codes. Read/write.

Note Use the **Text** property to return text from a **Range** object.

FormField object: Returns the result of the specified form field. Read/write **String**.

Result Property Example

This example displays the result of each form field in the active document.

```
For Each aField In ActiveDocument.FormFields
    MsgBox aField.Result
Next aField
```

This example applies bold formatting to the first field in the selection.

```
If Selection.Fields.Count >= 1 Then
    Set myRange = Selection.Fields(1).Result
    myRange.Bold = True
End If
```

Unlink Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUnlinkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUnlinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUnlinkA "}

Field object: Replaces the specified field with its most recent result.

Fields object: Replaces all the fields in the **Fields** collection with their most recent results.

Syntax

expression.**Unlink**

expression Required. An expression that returns a **Field** or **Fields** object.

Remarks

When you unlink a field, it's current result is converted to text or a graphic and can no longer be updated automatically. Note that some fields − such as XE (Index Entry) fields and SEQ (Sequence) fields − cannot be unlinked.

Unlink Method Example

This example unlinks the first field in "Sales.doc."

```
Documents("Sales.doc").Fields(1).Unlink
```

This example updates and unlinks all the fields in the first section in the active document.

```
With ActiveDocument.Sections(1).Range.Fields
    .Update
    .Unlink
End With
```

DoClick Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDoClickC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDoClickX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDoClickA "}

Clicks the specified field. If the field is a GOTOBUTTON field, this method moves the insertion point to the specified location or selects the specified bookmark. If the field is a MACROBUTTON field, this method runs the specified macro. If the field is a HYPERLINK field, this method jumps to the target location.

Syntax

expression.**DoClick**

expression Required. An expression that returns a **Field** object.

DoClick Method Example

If the first field in the selection is a GOTOBUTTON field, this example clicks it (the insertion point is moved to the specified location, or the specified bookmark is selected).

```
Set myField = Selection.Fields(1)
If myField.Type = wdFieldGoToButton Then myField.DoClick
```

UpdateSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateSourceA "}

Saves the changes made to the results of an INCLUDETEXT field back to the source document.

Note The source document must be formatted as a Word document.

Syntax

expression.**UpdateSource**

expression Required. An expression that returns a **Field** or **Fields** object.

UpdateSource Method Example

This example updates the INCLUDETEXT fields in the active document.

```
For Each aField In ActiveDocument.Fields
    If aField.Type = wdFieldIncludeText Then aField.UpdateSource
Next aField
```

Field Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFieldC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFieldX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFieldA"}

Returns a **Field** object that represents the field associated with the specified shape. Read-only.

Note Use the **Fields** property to return the **Fields** collection.

Field Property Example

This example inserts a graphic as an inline shape (using an INCLUDEPICTURE field) and then displays the shape's field code.

```
Set iShape = ActiveDocument.InlineShapes.AddPicture(FileName:="C:\Windows\
Tiles.bmp", _
    LinkToFile:=True, SaveWithDocument:=False, Range:=Selection.Range)
MsgBox iShape.Field.Code.Text
```

LinkFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLinkFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLinkFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLinkFormatA"}

Returns a **LinkFormat** object that represents the link options of the specified field, inline shape, or shape that's linked to a file. Read/only.

LinkFormat Property Example

This example inserts a graphic as an inline shape (using an INCLUDEPICTURE field) and then displays the source name (Tiles.bmp).

```
Set iShape = ActiveDocument.InlineShapes.AddPicture(FileName:="C:\windows\
Tiles.bmp", _
    LinkToFile:=True, SaveWithDocument:=False, Range:=Selection.Range)
MsgBox iShape.LinkFormat.SourceName
```

This example updates any fields in the active document that aren't updated automatically.

```
For Each afield In ActiveDocument.Fields
    If afield.LinkFormat.AutoUpdate = False Then afield.LinkFormat.Update
Next afield
```

ShowCodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowCodesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowCodesA"}

True if field codes are displayed for the specified field instead of field results. Read/write **Boolean**.

ShowCodes Property Example

This example selects the next field and displays the field codes.

```
With Selection
    .GoTo What:=wdGoToField
    .Expand Unit:=wdWord
    If .Fields.Count = 1 Then .Fields(1).ShowCodes = True
End With
```

This example updates and displays the result of the first field in the active document.

```
If ActiveDocument.Fields.Count >= 1 Then
    With ActiveDocument.Fields(1)
        .Update
        .ShowCodes = False
    End With
End If
```

ToggleShowCodes Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthToggleShowCodesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthToggleShowCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthToggleShowCodesA"}

Toggles the display of the fields between field codes and field results.

Note Use the **ShowCodes** property to control the display of an individual field.

Syntax

expression.**ToggleShowCodes**

expression Required. An expression that returns a **Fields** object.

ToggleShowCodes Method Example

This example toggles the display of fields in the selection (the equivalent of pressing SHIFT+F9).

```
Selection.Fields.ToggleShowCodes
```

This example toggles the display of fields in the active document (the equivalent of pressing ALT+F9).

```
ActiveDocument.Fields.ToggleShowCodes
```

ClearFormatting Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthClearFormattingC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthClearFormattingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthClearFormattingA"}

Removes any formatting specified as part of a find or replace operation. Corresponds to the **No Formatting** button in the **Find and Replace** dialog box (**Edit** menu).

Note To ensure that unwanted formats aren't included as criteria in a find or replace operation, use this method before carrying out the operation.

Syntax

expression.**ClearFormatting**

expression Required. An expression that returns a **Find** or **Replacement** object.

ClearFormatting Method Example

This example clears formatting from the find or replace criteria before replacing the word "Inc." with "incorporated" throughout the active document.

```
Set myRange = ActiveDocument.Content
With myRange.Find
    .ClearFormatting
    .Replacement.ClearFormatting
    .MatchWholeWord = True
    .Execute FindText:="Inc.", ReplaceWith:="incorporated",
Replace:=wdReplaceAll
End With
```

This example removes formatting from the find criteria before searching through the selection. If the word "Hello" with bold formatting is found, the entire paragraph is selected and copied to the Clipboard.

```
With Selection.Find
    .ClearFormatting
    .Font.Bold = True
    .Execute FindText:="Hello", Format:=True, Forward:=True
    If .Found = True Then
        .Parent.Expand Unit:=wdParagraph
        .Parent.Copy
    End If
End With
```

Find Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFindC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFindX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFindA"}

Returns a **Find** object that contains the criteria for a find operation. Read-only.

Note When this property is used with a **Selection** object, the selection is changed if the find operation is successful. If this property is used with a **Range** object, the selection isn't changed unless the **Select** method is applied.

Find Property Example

The following example searches forward through the document for the word "Microsoft." If the word is found, it's automatically selected.

```
With Selection.Find
    .Forward = True
    .ClearFormatting
    .MatchWholeWord = True
    .MatchCase = False
    .Wrap = wdFindContinue
    .Execute FindText:="Microsoft"
End With
```

This example inserts "Tip: " at the beginning of every paragraph formatted with the Heading 3 style in the active document. The **Do…Loop** statement is used to repeat a series of actions each time this style is found.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Style = wdStyleHeading3
    Do While .Execute(FindText:="", Forward:=True, Format:=True) = True
        With .Parent
            .StartOf Unit:=wdParagraph, Extend:=wdMove
            .InsertAfter "Tip: "
            .Move Unit:=wdParagraph, Count:=1
        End With
    Loop
End With
```

Highlight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHighlightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHighlightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHighlightA"}

Find object: **True** if highlight formatting is included in the find criteria. Can return or be set to **True**, **False**, or **wdUndefined**. Read/write **Long**.

Note The **wdUndefined** value can be used with the **Find** object to ignore the state of highlight formatting in the selection or range that is searched.

Replacement object: **True** if highlight formatting is applied to the replacement text. Can return or be set to **True**, **False**, or **wdUndefined**. Read/write **Long**.

Highlight Property Example

This example finds all instances of highlighted text in the active document and removes the highlight formatting by setting the **Highlight** property of the **Replacement** object to **False**.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
With myRange.Find
    .ClearFormatting
    .Highlight = True
    With .Replacement
        .ClearFormatting
        .Highlight = False
    End With
    .Execute Replace:=wdReplaceAll, Forward:=True, FindText:="", _
         ReplaceWith:="", Format:=True
End With
```

This example applies highlight formatting to the next instance of bold text in the active document.

```
With Selection.Find
    .ClearFormatting
    .Font.Bold = True
    With .Replacement
        .ClearFormatting
        .Highlight = True
    End With
    .Execute Forward:=True, FindText:="", ReplaceWith:="", _
        Format:=True
End With
```

MatchAllWordForms Property

This item is not available for this version on Office.

MatchAllWordForms Property Example

This item is not available for this version of Office

MatchCase Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMatchCaseC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMatchCaseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMatchCaseA"}

True if the find operation is case sensitive. Read/write **Boolean**.

Note Use the **Text** property of the **Find** object or use the ***FindText*** argument with the **Execute** method to specify the text to be located in a document.

MatchCase Property Example

This example selects the next occurrence of the word "library" in the selection, regardless of the case.

```
With Selection.Find
    .ClearFormatting
    .MatchWholeWord = True
    .MatchCase = False
    .Execute FindText:="library"
End With
```

MatchWildcards Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMatchWildcardsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMatchWildcardsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMatchWildcardsA"}

True if the text to find contains search wildcards. Corresponds to the **Use wildcards** check box in the **Find and Replace** dialog box (**Edit** menu). Read/write **Boolean**.

Note Use the **Text** property of the **Find** object or use the ***FindText*** argument with the **Execute** method to specify the text to be located in a document.

MatchWildcards Property Example

This example finds and selects the next three-letter word that begins with "s" and ends with "t."

```
With Selection.Find
    .ClearFormatting
    .Text = "s?t"
    .MatchWildcards = True
    .MatchSoundsLike = False
    .MatchAllWordForms = False
    .Execute Format:=False, Forward:=True
End With
```

MatchSoundsLike Property

This item is not available for this version of Office.

MatchSoundsLike Property Example

This item is not available for this version of Office.

MatchWholeWord Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMatchWholeWordC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMatchWholeWordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMatchWholeWordA"}

True if the find operation locates only entire words and not text that's part of a larger word. Read/write **Boolean**.

Note Use the **Text** property of the **Find** object or use the ***FindText*** argument with the **Execute** method to specify the text to be located in a document.

MatchWholeWord Property Example

This example clears all formatting from the find and replace criteria before replacing the word "Inc." with "incorporated" throughout the active document.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Replacement.ClearFormatting
    .MatchWholeWord = True
    .Execute FindText:="Inc.", ReplaceWith:="incorporated",
Replace:=wdReplaceAll
End With
```

Replacement Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReplacementC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReplacementX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReplacementA"}

Returns a **Replacement** object that contains the criteria for a replace operation. Read-only.

Replacement Property Example

This example removes bold formatting from the active document. The **Bold** property is **True** for the **Find** object and **False** for the **Replacement** object.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Font.Bold = True
    With .Replacement
        .ClearFormatting
        .Font.Bold = False
    End With
    .Execute FindText:="", ReplaceWith:="", Format:=True, _
        Replace:=wdReplaceAll
End With
```

This example finds every instance of the word "Start" in the active document and replaces it with "End." The find operation ignores formatting and matches the case of the text to find ("Start").

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
With myRange.Find
    .ClearFormatting
    .Text = "Start"
    With .Replacement
        .ClearFormatting
        .Text = "End"
    End With
    .Execute Replace:=wdReplaceAll, Format:=True, MatchCase:=True, _
        MatchWholeWord:=True
End With
```

Forward Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproForwardC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproForwardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproForwardA"}

True if the find operation searches forward through the document. **False** if it searches backward through the document. Read/write **Boolean**.

Forward Property Example

This example replaces the next occurrence of the word "hi" in the selection with "hello."

```
With Selection.Find
    .Forward = True
    .Text = "hi"
    .ClearFormatting
    .Replacement.Text = "hello"
    .Execute Replace:=wdReplaceOne
End With
```

The following example searches backward through the document for the word "Microsoft." If the word is found, it's automatically selected.

```
Selection.Collapse Direction:=wdCollapseStart
With Selection.Find
    .Forward = False
    .ClearFormatting
    .MatchWholeWord = True
    .MatchCase = False
    .Wrap = wdFindContinue
    .Execute FindText:="Microsoft"
End With
```

AllCaps Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAllCapsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAllCapsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAllCapsA "}

True if the font is formatted as all capital letters. Returns **True**, **False**, or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **AllCaps** to **True** sets **SmallCaps** to **False**, and vice versa.

AllCaps Property Example

This example checks the third paragraph in the active document for text formatted as all capital letters.

```
If ActiveDocument.Paragraphs(3).Range.Font.AllCaps = True Then
    MsgBox "Text is all caps."
Else
    MsgBox "Text is not all caps."
End if
```

This example formats the selected text as all capital letters.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.AllCaps = True
Else
    MsgBox "You need to select some text."
End If
```

Animation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAnimationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAnimationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAnimationA "}

Returns or sets the type of animation applied to the font. Read/write **Long**.

Can be one of the following **WdAnimation** constants:

wdAnimationBlinkingBackground
wdAnimationLasVegasLights
wdAnimationMarchingBlackAnts
wdAnimationMarchingRedAnts
wdAnimationNone
wdAnimationShimmer
wdAnimationSparkleText

Animation Property Example

This example animates the text in a new document.

```
Set newDoc = Documents.Add
With newDoc.Content
    .InsertAfter "This is a test of animation."
    .Font.Animation = wdAnimationLasVegasLights
End With
```

This example animates the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Animation = wdAnimationShimmer
Else
    MsgBox "You need to select some text."
End If
```

Bold Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBoldC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBoldX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBoldA "}

True if the font or range is formatted as bold. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Bold Property Example

This example formats the sixth word in a new document as bold.

```
Set newDoc = Documents.Add
Set myRange = newDoc.Content
myRange.InsertAfter "This is a test of bold."
myRange.Words(6).Bold = True
```

This example makes the entire selection bold if part of the selection is formatted as bold.

```
If Selection.Type = wdSelectionNormal Then
    If Selection.Font.Bold = wdUndefined Then Selection.Font.Bold = True
Else
    MsgBox "You need to select some text."
End If
```

This example toggles the bold format for the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Range.Bold = wdToggle
End If
```

This example makes the first paragraph in the active document bold.

```
ActiveDocument.Paragraphs(1).Range.Bold = True
```

Grow Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGrowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGrowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGrowA "}

Increases the font size to the next available size. If the selection or range contains more than one font size, each size is increased to the next available setting.

Syntax

expression.**Grow**

expression Required. An expression that returns a **Font** object.

Grow Method Example

This example increases the font size of the fourth word in a new document.

```
Set myRange = Documents.Add.Content
myRange.InsertAfter "This is a test of the Grow method."
MsgBox "Click OK to increase the font size of the fourth word."
myRange.Words(4).Font.Grow
```

This example increases the font size of the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Grow
Else
    MsgBox "You need to select some text."
End If
```

Hidden Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHiddenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHiddenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHiddenA "}

True if the font is formatted as hidden text. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

To control the display of hidden text, use the **ShowHiddenText** property of the **View** object.

To control whether properties and methods that return **Range** objects include or exclude hidden text when hidden text isn't displayed, use the **IncludeHiddenText** property of the **TextRetrievalMode** object.

Hidden Property Example

This example inserts text and formats the password number as hidden text.

```
Selection.Collapse Direction:=wdCollapseEnd
With Selection.Range
    .InsertAfter "Smith account password: 8116"
    .Words(5).Font.Hidden = True
End With
```

This example checks the selection for hidden text.

```
If Selection.Type = wdSelectionNormal Then
    If Selection.Font.Hidden = wdUndefined or _
            Selection.Font.Hidden = True Then
        MsgBox "There's hidden text in the selection."
    Else
        MsgBox "No hidden text in the selection."
    End If
Else
    MsgBox "You need to select some text."
End If
```

This example makes all hidden text in the active window visible and then formats the selection as hidden text.

```
ActiveWindow.View.ShowHiddenText = True
If Selection.Type = wdSelectionNormal Then Selection.Font.Hidden = True
```

Engrave Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEngraveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEngraveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEngraveA "}

True if the font is formatted as engraved. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **Engrave** to **True** sets **Emboss** to **False**, and vice versa.

Engrave Property Example

This example formats the first letter in the active document as engraved.

```
Set myRange = ActiveDocument.Characters(1)
With myRange.Font
    .Size = 20
    .Engrave = True
End With
```

This example formats the selection as engraved.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Engrave = True
Else
    MsgBox "You need to select some text."
End If
```

Italic Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproItalicC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproItalicX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproItalicA "}

True if the font or range is formatted as italic. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Italic Property Example

This example formats the first word in the active document as italic.

```
ActiveDocument.Words(1).Italic = True
```

This example checks the selection for italic formatting and removes any that it finds.

```
If Selection.Type = wdSelectionNormal Then
    mySel = Selection.Font.Italic
    If mySel = wdUndefined or mySel = True Then
        MsgBox "There's italic text in selection. Click OK to remove."
        Selection.Font.Italic = False
    Else
        MsgBox "No italic text in the selection."
    End If
Else
    MsgBox "You need to select some text."
End If
```

Kerning Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKerningC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKerningX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKerningA "}

Returns or sets the minimum font size for which Word will adjust kerning automatically. Read/write **Single**.

Kerning Property Example

This example adjusts the kerning of all text with font size 12 points or larger in the active document.

```
ActiveDocument.Content.Font.Kerning = 12
```

This example determines the minimum font size for which Word will automatically adjust kerning in the selected text.

```
If Selection.Type = wdSelectionNormal Then
    MsgBox Selection.Font.Kerning
Else
    MsgBox "You need to select some text."
End If
```

Shrink Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShrinkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShrinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShrinkA "}

Font object: Decreases the font size to the next available size. If the selection or range contains more than one font size, each size is decreased to the next available setting.

Selection object: Shrinks the selection to the next smaller unit of text. The progression is as follows: entire document, section, paragraph, sentence, word, insertion point.

Syntax

expression.**Shrink**

expression Required. An expression that returns a **Font** or **Selection** object.

Shrink Method Example

This example inserts a line of increasingly smaller Z's in a new document.

```
Set myRange = Documents.Add.Content
myRange.Font.Size = 45
For Count = 1 To 5
    myRange.InsertAfter "Z"
    For Count2 = 1 to 3
        myRange.Characters(Count).Font.Shrink
    Next Count2
Next Count
```

This example reduces the font size of the selected text by one size.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Shrink
Else
    MsgBox "You need to select some text."
End If
```

Size Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSizeA "}

Font object: Returns or sets the font size, in points. Read/write **Single**.

CheckBox object: Returns or sets the size of the specified check box, in points. Read/write **Single**.

Size Property Example

This example inserts text and then sets the font size of the seventh word of the inserted text to 20 points.

```
Selection.Collapse Direction:=wdCollapseEnd
With Selection.Range
    .Font.Reset
    .InsertBefore "This is a demonstration of font size."
    .Words(7).Font.Size = 20
End With
```

This example determines the font size of the selected text.

```
mySel = Selection.Font.Size
If mySel = wdUndefined Then
    MsgBox "There's a mix of font sizes in the selection."
Else
    MsgBox mySel & " points"
End If
```

This example sets the size of the check box named "Check1" in the active document to 12 points and then sets the check box as selected.

```
With ActiveDocument.FormFields("Check1").CheckBox
    .AutoSize = False
    .Size = 12
    .Value = True
End With
```

SmallCaps Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSmallCapsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSmallCapsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSmallCapsA "}

True if the font is formatted as small capital letters. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **SmallCaps** to **True** sets **AllCaps** to **False**, and vice versa.

SmallCaps Property Example

This example demonstrates the difference between small capital letters and all capital letters in a new document.

```
Set myRange = Documents.Add.Content
With myRange
    .InsertAfter "This is a demonstration of SmallCaps."
    .Words(6).Font.SmallCaps = True
    .InsertParagraphAfter
    .InsertAfter "This is a demonstration of AllCaps."
    .Words(14).Font.AllCaps = True
End With
```

This example formats the entire selection as small capital letters if part of the selection is already formatted as small capital letters.

```
If Selection.Type = wdSelectionNormal Then
    mySel = Selection.Font.SmallCaps
    If mySel = wdUndefined Then Selection.Font.SmallCaps = True
Else
    MsgBox "You need to select some text."
End If
```

Spacing Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpacingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpacingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpacingA "}

Font object: Returns or sets the spacing between characters, in points. Read/write **Single**.

TextColumns object: Returns or sets the spacing between the columns in the collection, in points. After this property has been set, the **EvenlySpaced** property is **True**. Read/write **Single**.

Note To return or set the spacing for a single text column when **EvenlySpaced** is **False**, use the **SpaceAfter** property of the **TextColumn** object.

Spacing Property Example

This example demonstrates two different character spacings at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
With myRange
    .InsertAfter "Demonstration of no character spacing. "
    .InsertParagraphAfter
    .InsertAfter "Demonstration of character spacing (1.5pt)."
    .InsertParagraphAfter
End With
ActiveDocument.Paragraphs(2).Range.Font.Spacing = 1.5
```

This example sets the character spacing of the selected text to 2 points.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Spacing = 2
Else
    MsgBox "You need to select some text."
End If
```

This example formats the active document to display text in two columns with 0.5 inch (36 points) spacing between the columns.

```
With ActiveDocument.PageSetup.TextColumns
    .SetCount NumColumns:=2
    .LineBetween = False
    .EvenlySpaced = True
    .Spacing = 36
End With
```

StrikeThrough Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStrikeThroughC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStrikeThroughX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStrikeThroughA "}

True if the font is formatted as strikethrough text. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Note To set or return double strikethrough formatting, use the **DoubleStrikeThrough** property.

StrikeThrough Property Example

This example applies strikethrough formatting to the first three words in the active document.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Words(1).Start, _
    End:=myDoc.Words(3).End)
myRange.Font.StrikeThrough = True
```

This example applies strikethrough formatting to the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.StrikeThrough = True
Else
    MsgBox "You need to select some text."
End If
```

Subscript Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSubscriptC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSubscriptX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSubscriptA "}

True if the font is formatted as subscript. Returns **True**, **False** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **Subscript** to **True** sets **Superscript** to **False**, and vice versa.

Subscript Property Example

This example inserts text at the beginning of the active document and formats the tenth character as subscript.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myRange.InsertAfter "Water = H20"
myRange.Characters(10).Font.Subscript = True
```

This example checks the selected text for subscript formatting.

```
If Selection.Type = wdSelectionNormal Then
    mySel = Selection.Font.Subscript
    If mySel = wdUndefined or mySel = True Then
        MsgBox "Subscript text exists in the selection."
    Else
        MsgBox "No subscript text in the selection."
    End If
Else
    MsgBox "You need to select some text."
End If
```

Superscript Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSuperscriptC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSuperscriptX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSuperscriptA "}

True if the font is formatted as superscript. Returns **True**, **False,** or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **Superscript** to **True** sets **Subscript** to **False**, and vice versa.

Superscript Property Example

This example inserts text at the beginning of the active document and formats two characters in the fourth word as superscript.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myRange.InsertAfter "Superscript in the 4th word."
ActiveDocument.Range(Start:=20, End:=22).Font.Superscript = True
```

This example formats the selected text as superscript.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Superscript = True
Else
    MsgBox "You need to select some text."
End If
```

Underline Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUnderlineC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUnderlineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUnderlineA "}

Returns or sets the type of underline applied to the font or range. Can be one of the following **WdUnderline** constants: **wdUnderlineNone**, **wdUnderlineDash**, **wdUnderlineDotDash**, **wdUnderlineDotDotDash**, **wdUnderlineThick**, **wdUnderlineDotted**, **wdUnderlineDouble**, **wdUnderlineSingle**, **wdUnderlineWords**, or **wdUnderlineWavy**. Read/write **Long**.

Underline Property Example

This example applies a double underline to the fourth word in the active document.

```
ActiveDocument.Words(4).Underline = wdUnderlineDouble
```

This example applies a single underline to the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Underline = wdUnderlineSingle
Else
    MsgBox "You need to select some text."
End If
```

SetAsTemplateDefault Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetAsTemplateDefaultC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetAsTemplateDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetAsTemplateDefaultA"}

Font object: Sets the specified font formatting as the default for the active document and all new documents based on the active template. The default font formatting is stored in the Normal style.

PageSetup object: Sets the specified page setup formatting as the default for the active document and all new documents based on the active template.

Syntax

expression.**SetAsTemplateDefault**

expression Required. An expression that returns a **Font** or **PageSetup** object.

SetAsTemplateDefault Method Example

This example sets the character formatting in the selection as the default.

```
Selection.Font.SetAsTemplateDefault
```

This example changes the left and right margin settings for the active document and then sets the page setup formatting as the default.

```
With ActiveDocument.PageSetup
    .LeftMargin = InchesToPoints(1)
    .RightMargin = InchesToPoints(1)
    .SetAsTemplateDefault
End With
```

Shadow Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShadowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShadowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShadowA "}

Font object: **True** if the specified font is formatted as shadowed. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Borders object: **True** if the specified border is formatted as shadowed. Read/write **Boolean**.

Shape and **ShapeRange** objects: Returns a **ShadowFormat** object that represents the shadow formatting for the specified shape. Read-only **Long**.

Shadow Property Example

This example demonstrates two different border styles in a new document.

```
Set myRange = Documents.Add.Content
With myRange
   .InsertAfter "Demonstration of border with shadow. "
   .InsertParagraphAfter
   .InsertParagraphAfter
   .InsertAfter "Demonstration of border without shadow."
End With
With ActiveDocument
    .Paragraphs(1).Borders.Shadow = True
    .Paragraphs(3).Borders.Enable = True
End With
```

This example applies shadow and bold formatting to the selection.

```
If Selection.Type = wdSelectionNormal Then
    With Selection.Font
        .Shadow = True
        .Bold = True
    End With
Else
    MsgBox "You need to select some text."
End If
```

This example adds an arrow with shadow formatting to the active document.

```
Set myShape = ActiveDocument.Shapes.AddShape(Type:=msoShapeRightArrow, _
    Left:=90, Top:=79, Width:=64, Height:=43)
myShape.Shadow.Type = msoShadow5
```

Emboss Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEmbossC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEmbossX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEmbossA "}

True if the specified font is formatted as embossed. Returns **True**, **False**, or **wdUndefined**. Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Remarks

Setting **Emboss** to **True** sets **Engrave** to **False**, and vice versa.

Emboss Property Example

This example embosses the second sentence in a new document.

```
With Documents.Add.Content
    .InsertAfter "This is the first sentence. "
    .InsertAfter "This is the second sentence. "
    .Sentences(2).Font.Emboss = True
End With
```

This example embosses the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.Emboss = True
Else
    MsgBox "You need to select some text."
End If
```

Font Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontA "}

Returns or sets a **Font** object that represents the character formatting of the specified object. To set this property, specify an expression that returns a **Font** object. Read/write **Font**.

Font Property Example

This example removes bold formatting from the Heading 1 style in the active document.

```
ActiveDocument.Styles(wdStyleHeading1).Font.Bold = False
```

This example toggles the font of the second paragraph in the active document between Arial and Times New Roman.

```
Set myRange = ActiveDocument.Paragraphs(2).Range
If myRange.Font.Name = "Times New Roman" Then
    myRange.Font.Name = "Arial"
Else
    myRange.Font.Name = "Times New Roman"
End If
```

This example displays the font of the selected text.

```
MsgBox Selection.Font.Name
```

This example applies the character formatting of the selected text to the first paragraph in the active document.

```
Set myFont = Selection.Font.Duplicate
ActiveDocument.Paragraphs(1).Range.Font = myFont
```

This example finds the next range of text that's formatted with the Times New Roman font.

```
With Selection.Find
    .ClearFormatting
    .Font.Name = "Times New Roman"
    .Execute FindText:="", ReplaceWith:="", Format:=True, _
        Forward:=True
End With
```

ContinuationNotice Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContinuationNoticeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContinuationNoticeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContinuationNoticeA "}

Returns a **Range** object that represents the footnote or endnote continuation notice. Read-only.

ContinuationNotice Property Example

This example replaces the current footnote continuation notice with the text "Continued...".

```
With ActiveDocument.Footnotes.ContinuationNotice
    .Delete
    .InsertBefore "Continued..."
End With
```

ContinuationSeparator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContinuationSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContinuationSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContinuationSeparatorA "}

Returns a **Range** object that represents the footnote or endnote continuation separator. Read-only.

ContinuationSeparator Property Example

This example replaces the current endnote continuation separator with a series of underscore characters.

```
With ActiveDocument.Endnotes.ContinuationSeparator
    .Delete
    .InsertBefore "____"
End With
```

Convert Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertA "}

Footnotes or **Endnotes** object: Converts endnotes to footnotes, or vice versa.

ListTemplate object: Converts a multiple-level list to a single-level list, or vice versa.

Syntax 1

expression.**Convert**

Syntax 2

expression.**Convert(*Level*)**

expression Required. An expression that returns a **ListTemplate** object (Syntax 1) or an **Endnotes** or **Footnotes** object (Syntax 2).

Level Optional **Long**. The level to use for formatting the new list. When converting a multiple-level list to a single-level list, this argument can be a number from 1 through 9. When converting a single-level list to a multiple-level list, 1 is the only valid value. If this argument is omitted, 1 is used.

Remarks

You cannot use the **Convert** method on a list template that is derived from the **ListGalleries** collection.

Convert Method Example

This example converts all endnotes in the active document to footnotes.

```
Set myEndnotes = ActiveDocument.Endnotes
If myEndnotes.Count > 0 Then myEndnotes.Convert
```

This example converts the footnotes in the selection to endnotes.

```
If Selection.Footnotes.Count > 0 Then Selection.Footnotes.Convert
```

This example converts the first list template in the active document. If the list template is multiple-level, it becomes single-level, or vice versa.

```
ActiveDocument.ListTemplates(1).Convert
```

Endnotes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEndnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEndnotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEndnotesA "}

Returns an **Endnotes** collection that represents all the endnotes in a range, selection, or document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Endnotes Property Example

This example positions the endnotes in the active document at the end of the document and formats the endnote reference marks as lowercase roman numerals.

```
With ActiveDocument.Endnotes
    .Location = wdEndOfDocument
    .NumberStyle = wdNoteNumberStyleLowercaseRoman
End With
```

Footnotes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFootnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFootnotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFootnotesA "}

Returns a **Footnotes** collection that represents all the footnotes in a range, selection, or document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Footnotes Property Example

This example changes the footnote reference marks for the footnotes in the active document to lowercase letters, starting with the letter "c".

```
With ActiveDocument.Footnotes
    .StartingNumber = 3
    .NumberStyle = wdNoteNumberStyleLowercaseLetter
End With
```

This example inserts an automatically numbered footnote at the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.Footnotes.Add Range:=Selection.Range, _
    Text:="(Lone Creek Press, 1995)"
```

Location Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLocationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLocationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLocationA "}

Footnotes object: Returns or sets the position of all footnotes. Can be either of the following **WdFootnoteLocation** constants: **wdBottomOfPage** or **wdBeneathText**. Read/write **Long**.

Endnotes object: Returns or sets the position of all endnotes. Can be either of the following **WdEndnoteLocation** constants: **wdEndOfSection** or **wdEndOfDocument**. Read/write **Long**.

Location Property Example

This example positions footnotes at the bottom of each page.

```
ActiveDocument.Footnotes.Location = wdBottomOfPage
```

This example positions all endnotes at the end of sections.

```
ActiveDocument.Endnotes.Location = wdEndOfSection
```

NumberingRule Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberingRuleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberingRuleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberingRuleA "}

Returns or sets the way footnotes or endnotes are numbered after page breaks or section breaks. Can be one of the following **WdNumberingRule** constants: **wdRestartContinuous**, **wdRestartSection**, or **wdRestartPage** (**Footnotes** only). Read/write **Long**.

NumberingRule Property Example

This example restarts endnote numbering after each section break in the active document.

```
ActiveDocument.Endnotes.NumberingRule = wdRestartSection
```

If the footnote numbering in section one is set to restart after each section break, this example sets the numbering to restart on each page.

```
Set myRange = ActiveDocument.Sections(1).Range
If myRange.Footnotes.NumberingRule = wdRestartSection Then
    myRange.Footnotes.NumberingRule = wdRestartPage
End If
```

Reference Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReferenceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReferenceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReferenceA "}

Returns a **Range** object that represents a footnote, endnote, or comment reference mark. Read-only.

Reference Property Example

This example sets `myRange` to the first footnote reference mark in the active document and then copies the reference mark.

```
Set myRange = ActiveDocument.Footnotes(1).Reference
myRange.Copy
```

This example formats the comment reference marks in the selection to be red.

```
For Each comm In Selection.Comments
    comm.Reference.Font.ColorIndex = wdRed
Next comm
```

ResetContinuationNotice Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetContinuationNoticeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetContinuationNoticeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetContinuationNoticeA "}

Resets the footnote or endnote continuation notice to the default notice. The default notice is blank (no text).

Syntax

expression.**ResetContinuationNotice**

expression Required. An expression that returns an **Endnotes** or **Footnotes** object.

ResetContinuationNotice Method Example

This example resets the endnote continuation notice for the active document.

```
ActiveDocument.Endnotes.ResetContinuationNotice
```

This example resets the footnote continuation notice and sets the starting number for footnote reference marks to 2 in Sales.doc.

```
With Documents("Sales.doc").Sections(1).Range.Footnotes
    .ResetContinuationNotice
    .NumberingRule = wdRestartContinuous
    .StartingNumber = 2
End With
```

ResetContinuationSeparator Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetContinuationSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetContinuationSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetContinuationSeparatorA "}

Resets the footnote or endnote continuation separator to the default separator. The default separator is a long horizontal line that separates document text from notes continued from the previous page.

Syntax

expression.**ResetContinuationSeparator**

expression Required. An expression that returns an **Endnotes** or **Footnotes** object.

ResetContinuationSeparator Method Example

This example resets the footnote continuation separator to the default separator line.

```
ActiveDocument.Footnotes.ResetContinuationSeparator
```

This example resets the endnote continuation separator for the first section in each open document.

```
For Each doc In Documents
    doc.Sections(1).Range.Endnotes.ResetContinuationSeparator
Next doc
```

ResetSeparator Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthResetSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthResetSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthResetSeparatorA "}

Resets the footnote or endnote separator to the default separator. The default separator is a short horizontal line that separates document text from notes.

Syntax

expression.**ResetSeparator**

expression Required. An expression that returns an **Endnotes** or **Footnotes** object.

ResetSeparator Method Example

This example resets the footnote separator to the default separator line.

```
ActiveDocument.Footnotes.ResetSeparator
```

This example resets the endnote separator for the notes in the document where the selection is located.

```
Selection.Endnotes.ResetSeparator
```

Separator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSeparatorA "}

Endnotes or **Footnotes** object: Returns a **Range** object that represents the endnote or footnote separator. Read-only.

CaptionLabel object: Returns or sets the character between the chapter number and the sequence number. Can be one of the following **WdSeparatorType** constants: **wdSeparatorColon**, **wdSeparatorEmDash**, **wdSeparatorEnDash**, **wdSeparatorHyphen**, or **wdSeparatorPeriod**. Read/write **Long**.

TableOfAuthorities object: Returns or sets the characters (up to five) between the sequence number and the page number. A hyphen (-) is the default character. This property corresponds to the \d switch for a TOA field. Read/write **String**.

Separator Property Example

This example inserts a Figure caption, which has a colon (:) between the chapter number and the sequence number.

```
With CaptionLabels("Figure")
    .Separator = wdSeparatorColon
    .IncludeChapterNumber = True
End With
Selection.InsertCaption "Figure"
```

This example changes the footnote separator to a single border indented 3 inches from the right margin.

```
With ActiveDocument.Footnotes.Separator
    .Delete
    .Borders(wdBorderTop).LineStyle = wdLineStyleSingle
    .ParagraphFormat.RightIndent = InchesToPoints(3)
End With
```

This example inserts a table of authorities at the beginning of the active document, and then it formats the table to include a sequence number and a page number, separated by a hyphen (-).

```
Set myRange = ActiveDocument.Range(0, 0)
With ActiveDocument.TablesOfAuthorities.Add(Range:=myRange)
    .IncludeSequenceName = "Chapter"
    .Separator = "-"
End With
```

SuppressEndnotes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSuppressEndnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSuppressEndnotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSuppressEndnotesA "}

True if endnotes are printed at the end of the next section that doesn't suppress endnotes. Suppressed endnotes are printed before the endnotes in that section. Read/write **Boolean**.

Note This property takes effect only if the **Location** property is set to **wdEndOfSection**.

SuppressEndnotes Property Example

This example suppresses endnotes in the first section in the active document.

```
If ActiveDocument.Endnotes.Location = wdEndOfSection Then
    ActiveDocument.Sections(1).PageSetup.SuppressEndnotes = True
End If
```

SwapWithEndnotes Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSwapWithEndnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSwapWithEndnotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSwapWithEndnotesA "}

Converts all footnotes in a document to endnotes and vice versa.

Note To convert a range of footnotes to endnotes, use the **Convert** method.

Syntax

expression.**SwapWithEndnotes**

expression Required. An expression that returns a **Footnotes** object.

SwapWithEndnotes Method Example

This example converts the footnotes in the active document to endnotes and converts the endnotes to footnotes.

```
ActiveDocument.Footnotes.SwapWithEndnotes
```

SwapWithFootnotes Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSwapWithFootnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSwapWithFootnotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSwapWithFootnotesA "}

Converts all endnotes in a document to footnotes and vice versa.

Note To convert a range of endnotes to footnotes, use the **Convert** method.

Syntax

expression.**SwapWithFootnotes**

expression Required. An expression that returns an **Endnotes** object.

SwapWithFootnotes Method Example

This example converts the endnotes in the active document to footnotes and converts the footnotes to endnotes.

```
ActiveDocument.Endnotes.SwapWithFootnotes
```

AutoSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoSizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoSizeA "}

True if the check box size is determined by the font size of the surrounding text. **False** if the check box size is determined by the **Size** property. Read/write **Boolean**.

AutoSize Property Example

This example sets the size of the check box named "Check1" to Auto and then selects the check box.

```
With ActiveDocument.FormFields("Check1").CheckBox
    .AutoSize = True
    .Value = True
End With
```

CheckBox Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCheckBoxC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCheckBoxX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCheckBoxA "}

Returns a **CheckBox** object that represents a check box form field. Read-only.

Remarks

If the **CheckBox** property is applied to a **FormField** object that isn't a check box form field, the property won't fail, but the **Valid** property for the returned object will be **False**.

CheckBox Property Example

This example clears the check box named "Blue."

```
ActiveDocument.FormFields("Blue").CheckBox.Value = False
```

This example compares the current value with the default value of the check box named "Check1." If the values are equal, the `same` variable is set to **True**.

```
Set myField = ActiveDocument.FormFields("Check1").CheckBox
If myField.Default = myField.Value Then same = True
```

Default Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultA "}

CheckBox object: Returns or sets the default check box value. **True** if the default value is checked. Read/write **Boolean**.

DropDown object: Returns or sets the default drop-down item. The first item in a drop-down form field is 1, the second item is 2, and so on. Read/write **Long**.

TextInput object: Returns or sets the text that represents the default text box contents. Read/write **String**.

Default Property Example

This example sets the default text for the text form field named "Name."

```
ActiveDocument.FormFields("Name").TextInput.Default = "your name"
```

If the first form field in the active document is a check box, this example retrieves the default value.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormCheckBox Then _
    defvalue = ActiveDocument.FormFields(1).CheckBox.Default
```

This example sets the default item for the drop-down form field named "Colors" in Sales.doc.

```
Documents("Sales.doc").FormFields("Colors").DropDown.Default = 2
```

DropDown Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDropDownC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDropDownX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDropDownA "}

Returns a **DropDown** object that represents a drop-down form field. Read-only.

Remarks

If the **DropDown** property is applied to a **FormField** object that isn't a drop-down form field, the property won't fail, but the **Valid** property for the returned object will be **False**.

DropDown Property Example

This example displays the text of the item selected in the drop-down form field named "Colors."

```
Set myDrop = ActiveDocument.FormFields("Colors").DropDown
MsgBox myDrop.ListEntries(myDrop.Value).Name
```

This example adds "Seattle" to the drop-down form field named "Places" in Form.doc.

```
With Documents("Form.doc").FormFields("Places").DropDown.ListEntries
    .Add Name:="Seattle"
End With
```

Enabled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnabledC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnabledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnabledA "}

True if a form field is enabled. If a form field is enabled, its contents can be changed as the form is filled in. Read/write **Boolean**.

Enabled Property Example

If the first form field in the active document is an enabled check box, this example selects the check box.

```
Set myFField = ActiveDocument.FormFields(1)
If myFField.Enabled = True And myFField.Type = wdFieldFormCheckBox Then
    myFField.CheckBox.Value = True
End If
```

EntryMacro Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEntryMacroC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEntryMacroX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEntryMacroA "}

Returns or sets an entry macro name for the specified form field (**CheckBox**, **DropDown**, or **TextInput**). The entry macro runs when the form field gets the focus. Read/write **String**.

EntryMacro Property Example

This example assigns the macro named "Blue" to the first form field in "Form.doc."

```
Documents("Form.doc").FormFields(1).EntryMacro = "Blue"
```

This example assigns the macro named "Breadth" to the form field named "Text1" in the active document.

```
ActiveDocument.FormFields("Text1").EntryMacro = "Breadth"
```

ExitMacro Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExitMacroC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExitMacroX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExitMacroA "}

Returns or sets an exit macro name for the specified form field (**CheckBox**, **DropDown**, or **TextInput**). The exit macro runs when the form field loses the focus. Read/write **String**.

ExitMacro Property Example

This example assigns the macro named "Reformat" to the first form field in the selection.

```
If Selection.FormFields.Count > 0 Then _
    Selection.FormFields(1).ExitMacro = "Reformat"
```

This example assigns the macro named "Blue" to the last form field in "Form.doc."

```
iMax = Documents("Form.doc").FormFields.Count
Documents("Form.doc").FormFields(iMax).ExitMacro = "Blue"
```

FormFields Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFormFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFormFieldsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFormFieldsA "}

Returns a **FormFields** collection that represents all the form fields in the document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

FormFields Property Example

This example sets the content of the form field named "Text1" to "Name."

```
ActiveDocument.FormFields("Text1").Result = "Name"
```

This example retrieves the type of the first form field in section two.

```
myType = ActiveDocument.Sections(2).Range.FormFields(1).Type
Select Case myType
    Case wdFieldFormTextInput
        thetype = "TextBox"
    Case wdFieldFormDropDown
        thetype = "DropDown"
    Case wdFieldFormCheckBox
        thetype = "CheckBox"
End Select
```

This example displays the name of the first form field in the selection.

```
If Selection.FormFields.Count > 0 Then
    MsgBox Selection.FormFields(1).Name
End If
```

HelpText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHelpTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHelpTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHelpTextA "}

Returns or sets the text that's displayed in a message box when the form field has the focus and the user presses F1 (Windows) or COMMAND+/ or HELP (Macintosh). If the **OwnHelp** property is set to **True**, **HelpText** specifies the text string value. If **OwnHelp** is set to **False**, **HelpText** specifies the name of an AutoText entry that contains help text for the form field. Read/write **String**.

HelpText Property Example

This example sets the help text for the form field named "Name."

```
With ActiveDocument.FormFields("Name")
    .OwnHelp = True
    .HelpText = "Type your full legal name."
End With
```

ListEntries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListEntriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListEntriesA "}

Returns a **ListEntries** collection that represents all the items in a **DropDown** object. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ListEntries Property Example

This example retrieves the text of the active item from the drop-down form field named "DropDown1."

```
Set myField = ActiveDocument.FormFields("DropDown1").DropDown
num = myField.Value
myName = myField.ListEntries(num).Name
```

This example retrieves the total number of items in the active drop-down form field (the document should be protected for forms). If there are two or more items, this example sets the second item as the active item.

```
Set myField = Selection.FormFields(1)
If myfield.Type = wdFieldFormDropDown Then
    num = myField.DropDown.ListEntries.Count
    If num >= 2 Then myField.DropDown.Value = 2
End If
```

OwnHelp Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOwnHelpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOwnHelpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOwnHelpA "}

Specifies the source of the text that's displayed in a message box when a form field has the focus and the user presses F1 (Windows) or COMMAND+/ or HELP (Macintosh). If **True**, the text specified by the **HelpText** property is displayed. If **False**, the text in the AutoText entry specified by the **HelpText** property is displayed. Read/write **Boolean**.

OwnHelp Property Example

This example sets the help text for first form field in current section to the contents of the AutoText entry named "Sample."

```
With Selection.Sections(1).Range.FormFields(1)
    .OwnHelp = False
    .HelpText = "Sample"
End With
```

OwnStatus Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOwnStatusC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOwnStatusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOwnStatusA "}

Specifies the source of the text that's displayed in the status bar when a form field has the focus. If **True**, the text specified by the **StatusText** property is displayed. If **False**, the text of the AutoText entry specified by the **StatusText** property is displayed. Read/write **Boolean**.

OwnStatus Property Example

This example sets the status bar help text for the form field named "Account" to the contents of the AutoText entry named "Acct."

```
With ActiveDocument.FormFields("Account")
    .OwnStatus = False
    .StatusText = "Acct"
End With
```

StatusText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStatusTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStatusTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStatusTextA "}

Returns or sets the text that's displayed in the status bar when a form field has the focus. If the **OwnStatus** property is set to **True**, **StatusText** specifies the status bar text. If the **OwnStatus** property is set to **False**, **StatusText** specifies the name of an AutoText entry that contains status bar text for the form field. Read/write **String**.

StatusText Property Example

This example sets the status bar help text for the form field named "Age."

```
With ActiveDocument.FormFields("Age")
    .OwnStatus = True
    .StatusText = "Type your current age."
End With
```

TextInput Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextInputC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextInputX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextInputA "}

Returns a **TextInput** object that represents a text form field. Read-only.

Remarks

If the **TextInput** property is applied to a **FormField** object that isn't a drop-down form field, the property won't fail, but the **Valid** property for the returned object will be **False**.

Use the **Result** property with the **FormField** object to return or set the contents of a **TextInput** object.

```
ActiveDocument.FormFields("Text1").Result = "John Doe"
```

TextInput Property Example

This example protects the active document for forms and deletes the contents of the form field named "Text1."

```
ActiveDocument.Protect Type:=wdAllowOnlyFormFields
ActiveDocument.FormFields("Text1").TextInput.Clear
```

If the first form field in the active document is a text form field that accepts regular text, this example sets the contents of the form field.

```
Set myField = ActiveDocument.FormFields(1)
If myField.Type = wdFieldFormTextInput And _
    myField.TextInput.Type = wdRegularText Then
    myField.Result = "Hello"
End If
```

CalculateOnExit Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCalculateOnExitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCalculateOnExitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCalculateOnExitA "}

True if references to the specified form field are automatically updated whenever the field is exited. Read/write **Boolean**.

Remarks

A REF field can be used to reference the contents of a form field. For example, **{**REF SubTotal**}** references the form field marked by the SubTotal bookmark.

CalculateOnExit Property Example

This example keeps references to form fields in Form.doc from being automatically updated whenever the form field is exited.

```
For Each aFField In Documents("Form.doc").FormFields
    aFField.CalculateOnExit = False
Next aFField
```

This example adds a text form field and a REF field in a new document. Whenever text is typed and the Text1 field is exited, the REF field is automatically updated.

```
With Documents.Add
    .FormFields.Add Range:=Selection.Range, Type:=wdFieldFormTextInput
    .Fields.Add Range:=Selection.Range, Type:=wdFieldRef, Text:="Text1"
    .FormFields("Text1").CalculateOnExit = True
    .Protect Type:=wdAllowOnlyFormFields
End With
```

EditType Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditTypeA "}

Sets options for the specified text form field.

Syntax

expression.**EditType(*Type, Default, Format, Enabled*)**

expression Required. An expression that returns a **TextInput** object.

Type Required **Long**. The text box type. Can be one of the following **WdTextFormFieldType** constants: **wdCalculationText**, **wdCurrentDateText**, **wdCurrentTimeText**, **wdDateText**, **wdNumberText**, or **wdRegularText**.

Default Optional **Variant**. The default text that appears in the text box.

Format Optional **Variant**. The formatting string used to format the text, number, or date (for example, "0.00," "Title Case," or "M/d/yy"). For more examples of formats, see the list of formats for the specified text form field type in the **Text Form Field Options** dialog box.

Enabled Optional **Variant**. **True** to enable the form field for text entry.

EditType Method Example

This example adds a text form field named "Today" at the beginning of the active document. The **EditType** method is used to set the type to **wdCurrentDateText** and set the date format to "M/d/yy."

```
With ActiveDocument.FormFields.Add(Range:=ActiveDocument.Range(0, 0), _
    Type:=wdFieldFormTextInput)
    .Name = "Today"
    .TextInput.EditType Type:=wdCurrentDateText, Format:="M/d/yy",
Enabled:=False
End With
```

Shaded Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShadedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShadedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShadedA "}

True if shading is applied to form fields. Read/write **Boolean**.

Remarks

Shading makes form fields easier to locate in a document and doesn't affect the printed output.

Shaded Property Example

This example removes shading from form fields in Employment Form.doc.

```
Documents("Employment Form.doc").FormFields.Shaded = False
```

This example adds shading to the form fields in the active document and protects the document for forms.

```
With ActiveDocument
    .FormFields.Shaded = True
    .Protect Type:=wdAllowOnlyFormFields, NoReset:=True
End With
```

ProtectedForForms Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProtectedForFormsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProtectedForFormsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProtectedForFormsA "}

True if the specified section is protected for forms. When a section is protected for forms, you can select and modify text only in form fields. Read/write **Boolean**.

Note To protect an entire document, use the **Protect** method of the **Document** object.

ProtectedForForms Property Example

This example protects the second section in the active document for forms.

```
If ActiveDocument.Sections.Count >= 2 Then _
    ActiveDocument.Sections(2).ProtectedForForms = True
```

This example unprotects the first section in the selection.

```
Selection.Sections(1).ProtectedForForms = False
```

This example toggles the protection for the first section in the selection.

```
Selection.Sections(1).ProtectedForForms = Not _
    Selection.Sections(1).ProtectedForForms
```

Valid Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproValidC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproValidX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproValidA "}

CheckBox, **DropDown**, or **TextInput** object: **True** if the specified form field object is a valid check box form field, drop down form field, or text form field. **False** if it isn't valid. Read-only **Boolean**.

CustomLabel object: **True** if the various properties (for example, **Height**, **Width**, and **NumberDown**) for the specified custom label work together to produce a valid mailing label.

Remarks

Use the **Type** property of the **FormField** object to determine the type of form field (returns **wdFieldFormCheckBox**, **wdFieldFormDropDown**, or **wdFieldFormTextInput**) before applying the **CheckBox**, **DropDown**, or **TextInput** property. This precaution ensures that the **FormField** object is the expected type. If the first form field in the active document is a check box, the following example selects the check box.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormCheckBox Then
    ActiveDocument.FormFields(1).CheckBox.Value = True
End If
```

Valid Property Example

This example determines whether the first form field in the active document is a text form field. If the **Valid** property is **True**, the contents of the text form field are changed to "Hello."

```
If ActiveDocument.FormFields(1).TextInput.Valid = True Then
    ActiveDocument.FormFields(1).Result = "Hello"
End If
```

This example adds a text form field at the insertion point. Because `myFormField` isn't a valid check box form field, the message box displays "False."

```
Selection.Collapse Direction:=wdCollapseStart
Set myFormField = ActiveDocument.FormFields.Add(Range:= _
    Selection.Range, Type:=wdFieldFormTextInput)
MsgBox myFormField.CheckBox.Valid
```

If the settings for the custom label named "My Labels" are valid, this example creates a new document of labels using the My Labels settings.

```
addr = "James Allard" & vbCr & "123 Main St." & vbCr _
    & "Seattle, WA  98040"
If Application.MailingLabel.CustomLabels("My Labels").Valid = True Then
    Application.MailingLabel.CreateNewDocument _
        Name:="My Labels", Address:=addr
End If
```

Frame Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFrameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFrameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFrameA "}

Returns a **Frame** object that represents the frame formatting for the specified style or find-and-replace operation. Read-only.

Frame Property Example

This example creates a style with frame formatting and then applies the style to the first paragraph in the selection.

```
Set myStyle = ActiveDocument.Styles.Add(Name:="frame", Type:=wdParagraph)
With myStyle.Frame
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionMargin
    .HeightRule = wdFrameAuto
    .WidthRule = wdFrameAuto
    .TextWrap = True
End With
Selection.Paragraphs(1).Range.Style = "frame"
```

This example finds the first frame with wrap around formatting. If such a frame is found, a message is displayed on the status bar.

```
With ActiveDocument.Content.Find
    .Text = ""
    .Frame.TextWrap = True
    .Execute Forward:=True, Wrap:=wdFindContinue, Format:=True
    If .Found = True Then StatusBar = "Frame was found"
    .Parent.Select
End With
```

Frames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFramesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFramesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFramesA "}

Returns a **Frames** collection that represents all the frames in a document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Frames Property Example

This example causes text to wrap around frames in the first section in the active document.

```
For Each aFrame In ActiveDocument.Sections(1).Range.Frames
    aFrame.TextWrap = True
Next aFrame
```

This example adds a frame around the selection and returns a frame object to the `myFrame` variable.

```
Set myFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
```

HorizontalDistanceFromText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalDistanceFromTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalDistanceFromTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalDistanceFromTextA "}

Returns or sets the horizontal distance between a frame and the surrounding text, in points. Read/write **Single**.

HorizontalDistanceFromText Property Example

This example adds a frame around the selection and sets the horizontal distance between the frame and the text to 12 points.

```
Set myFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
myFrame.HorizontalDistanceFromText = 12
```

This example adds a frame around the first paragraph and sets several properties of the frame.

```
Set aFrame =
ActiveDocument.Frames.Add(Range:=ActiveDocument.Paragraphs(1).Range)
With aFrame
    .HorizontalDistanceFromText = InchesToPoints(0.25)
    .VerticalDistanceFromText = InchesToPoints(0.25)
    .HeightRule = wdFrameAuto
    .WidthRule = wdFrameAuto
    .Borders.Enable = False
End With
```

HorizontalPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalPositionA "}

Returns or sets the horizontal distance between the edge of the frame and the item specified by the **RelativeHorizontalPosition** property. Can be a number that indicates a measurement in points, or can be one of the following **WdFramePosition** constants: **wdFrameLeft**, **wdFrameRight**, **wdFrameCenter**, **wdFrameInside**, or **wdFrameOutside**. Read/write **Single**.

HorizontalPosition Property Example

This example aligns the first frame in the active document horizontally with the right margin.

```
If ActiveDocument.Frames.Count >= 1 Then
    With ActiveDocument.Frames(1)
        .RelativeHorizontalPosition = wdRelativeHorizontalPositionMargin
        .HorizontalPosition = wdFrameRight
    End With
End If
```

RelativeHorizontalPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRelativeHorizontalPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRelativeHorizontalPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRelativeHorizontalPositionA "}

Specifies what the horizontal position of a frame or shape is relative to. Can be one of the following **WdRelativeHorizontalPosition** constants: **wdRelativeHorizontalPositionColumn**, **wdRelativeHorizontalPositionMargin**, or **wdRelativeHorizontalPositionPage**. Read/write **Long**.

RelativeHorizontalPosition Property Example

This example adds a frame around the selection and aligns the frame horizontally with the right margin.

```
Set myFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
With myFrame
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionMargin
    .HorizontalPosition = wdFrameRight
End With
```

This example repositions the selected shape object.

```
With Selection.ShapeRange
    .Left = InchesToPoints(0.6)
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
    .Top = InchesToPoints(1)
    .RelativeVerticalPosition = wdRelativeVerticalPositionParagraph
End With
```

RelativeVerticalPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRelativeVerticalPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRelativeVerticalPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRelativeVerticalPositionA "}

Specifies what the vertical position of a frame or shape is relative to. Can be one of the following **WdRelativeVerticalPosition** constants: **wdRelativeVerticalPositionMargin**, **wdRelativeVerticalPositionPage**, or **wdRelativeVerticalPositionParagraph**. Read/write **Long**.

RelativeVerticalPosition Property Example

This example adds a frame around the selection and aligns the frame vertically with the top of the page.

```
Set myFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
With myFrame
    .RelativeVerticalPosition = wdRelativeVerticalPositionPage
    .VerticalPosition = wdFrameTop
End With
```

This example repositions the first shape object in the active document.

```
With ActiveDocument.Shapes(1)
    .Left = InchesToPoints(0.6)
    .RelativeHorizontalPosition = wdRelativeHorizontalPositionPage
    .Top = InchesToPoints(1)
    .RelativeVerticalPosition = wdRelativeVerticalPositionParagraph
End With
```

TextWrap Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextWrapC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextWrapX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextWrapA "}

True if document text wraps around the specified frame. Read/write **Boolean**.

TextWrap Property Example

This example causes text to not wrap around the first frame in the active document.

```
If ActiveDocument.Frames.Count >= 1 Then
    ActiveDocument.Frames(1).TextWrap = False
End If
```

This example causes text to wrap around all frames in the active document.

```
For Each aFrame In ActiveDocument.Frames
    aFrame.TextWrap = True
Next aFrame
```

VerticalDistanceFromText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalDistanceFromTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalDistanceFromTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalDistanceFromTextA "}

Returns or sets the vertical distance between a frame and the surrounding text, in points. Read/write **Single**.

VerticalDistanceFromText Property Example

This example sets the vertical distance between the selected frame and the surrounding text to 12 points.

```
If Selection.Frames.Count = 1 Then
    Selection.Frames(1).VerticalDistanceFromText = 12
End If
```

This example adds a frame around the selection and sets several properties of the frame.

```
Set aFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
With aFrame
    .HorizontalDistanceFromText = InchesToPoints(0.13)
    .VerticalDistanceFromText = InchesToPoints(0.13)
    .HeightRule = wdFrameAuto
    .WidthRule = wdFrameAuto
End With
```

VerticalPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalPositionA "}

Returns or sets the vertical distance between the edge of the frame and the item specified by the **RelativeVerticalPosition** property. Can be a number that indicates a measurement in points, or can be one of the following **WdFramePosition** constants: **wdFrameTop**, **wdFrameBottom**, or **wdFrameCenter**. Read/write **Single**.

VerticalPosition Property Example

This example aligns the first frame in the active document vertically with the top of the page.

```
Set myFrame = ActiveDocument.Frames(1)
With myFrame
    .RelativeVerticalPosition = wdRelativeVerticalPositionPage
    .VerticalPosition = wdFrameTop
End With
```

This example adds a frame around the first shape in the active document and positions the frame 1 inch from the top margin.

```
If ActiveDocument.Shapes.Count >= 1 Then
    ActiveDocument.Shapes(1).Select
    Set aFrame = ActiveDocument.Frames.Add(Range:=Selection.Range)
    With aFrame
        .RelativeVerticalPosition = wdRelativeVerticalPositionMargin
        .VerticalPosition = InchesToPoints(1)
    End With
End If
```

WidthRule Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWidthRuleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWidthRuleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWidthRuleA "}

Returns or sets the rule used to determine the width of a frame. Read/write **Long**.

Can be one of the following **WdFrameSizeRule** constants.

Constant	Description
wdFrameAuto	Sets the width according to the width of the item in the frame.
wdFrameExact	Sets the width to an exact value specified by the **Width** property.
wdFrameAtLeast	Sets the width to a value equal to or greater than the value specified by the **Width** property.

WidthRule Property Example

This example sets the width of the last frame in the active document to exactly 72 points (1 inch).

```
If ActiveDocument.Frames.Count >= 1 Then
    With ActiveDocument.Frames(ActiveDocument.Frames.Count)
        .WidthRule = wdFrameExact
        .Width = 72
    End With
End If
```

DistanceBottom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceBottomC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceBottomX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceBottomA"}

Returns or sets the distance (in points) between the document text and the bottom edge of the text-free area surrounding the specified shape. The size and shape of the specified shape, together with the values of the **Type** and **Side** properties of the **WrapFormat** object, determine the size and shape of this text-free area. Read/write **Single**.

DistanceBottom Property Example

This example adds an oval to the active document and specifies that the document text wrap around the left and right sides of the square that circumscribes the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 36, 36, 100, 35)
With myOval.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

DistanceLeft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceLeftC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceLeftA"}

Returns or sets the distance (in points) between the document text and the left edge of the text-free area surrounding the specified shape. The size and shape of the specified shape, together with the values of the **Type** and **Side** properties of the **WrapFormat** object, determine the size and shape of this text-free area. Read/write **Single**.

DistanceLeft Property Example

This example adds an oval to the active document and specifies that the document text wrap tightly around the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the oval.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 0, 0, 200, 50)
With myEll.WrapFormat
    .Type = wdWrapTight
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

DistanceRight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceRightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceRightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceRightA"}

Returns or sets the distance (in points) between the document text and the right edge of the text-free area surrounding the specified shape. The size and shape of the specified shape, together with the values of the **Type** and **Side** properties of the **WrapFormat** object, determine the size and shape of this text-free area. Read/write **Single**.

DistanceRight Property Example

This example adds an oval to the active document and specifies that the document text wrap around the left and right sides of the square that circumscribes the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 0, 0, 200, 50)
With myOval.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

DistanceTop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDistanceTopC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDistanceTopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDistanceTopA"}

Returns or sets the distance (in points) between the document text and the top edge of the text-free area surrounding the specified shape. The size and shape of the specified shape, together with the values of the **Type** and **Side** properties of the **WrapFormat** object, determine the size and shape of this text-free area. Read/write **Single**.

DistanceTop Property Example

This example adds an oval to the active document and specifies that the document text wrap around the left and right sides of the square that circumscribes the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 0, 0, 200, 50)
With myOval.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

Side Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSideC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSideX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSideA"}

Returns or sets a value that indicates whether the document text should wrap on both sides of the specified shape, on either the left or right side only, or on the side of the shape that's farthest from the page margin. If the text wraps on only one side of the shape, there's a text-free area between the other side of the shape and the page margin. Can be one of the following **WdWrapSideType** constants: **wdWrapBoth**, **wdWrapLargest**, **wdWrapLeft**, or **wdWrapRight**. Read/write **Long**.

Side Property Example

This example adds an oval to the active document and specifies that the document text wrap around the left and right sides of the square that circumscribes the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 0, 0, 200, 50)
With myEll.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

ActiveGrammarDictionary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveGrammarDictionaryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveGrammarDictionaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveGrammarDictionaryA "}

Returns a **Dictionary** object that represents the active grammar dictionary for the specifed language. Read-only.

Remarks

If there's no grammar dictionary installed for the specified language, this property returns **Nothing**.

ActiveGrammarDictionary Property Example

This example displays the full path and file name of the active grammar dictionary.

```
myLang = Selection.LanguageID
Set myGram = Languages(myLang).ActiveGrammarDictionary
MsgBox myGram.Path & Application.PathSeparator & myGram.Name
```

ActiveHyphenationDictionary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveHyphenationDictionaryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveHyphenationDictionaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveHyphenationDictionaryA "}

Returns a **Dictionary** object that represents the active hyphenation dictionary for the specifed language. Read-only.

Remarks

If there's no hyphenation dictionary installed for the specified language, this property returns **Nothing**.

ActiveHyphenationDictionary Property Example

This example displays the full path and file name of the active hyphenation dictionary.

```
myLang = Selection.LanguageID
Set myHyph = Languages(myLang).ActiveHyphenationDictionary
MsgBox myHyph.Path & Application.PathSeparator & myHyph.Name
```

CheckGrammar Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCheckGrammarC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCheckGrammarX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCheckGrammarA "}

Syntax 1: Begins a spelling and grammar check for the specified document or range. If the document or range contains errors, this method displays the **Spelling and Grammar** dialog box (**Tools** menu), with the **Check grammar** option selected. When applied to a document, this method checks all available stories (such as headers, footers, and text boxes).

Syntax 2: Checks a string for grammatical errors. Returns **True** if the string contains no errors.

Syntax 1

expression.**CheckGrammar()**

Syntax 2

expression.**CheckGrammar(*String*)**

expression Syntax 1: Required. An expression that returns a **Document** or **Range** object.
Syntax 2: Required. An expression that returns an **Application** object.

String Required **String**. The string you want to check for grammatical errors.

CheckGrammar Method Example

This example begins a spelling and grammar check for all stories in the active document.

```
ActiveDocument.CheckGrammar
```

This example begins a spelling and grammar check on section two in MyDocument.doc.

```
Set Range2 = Documents("MyDocument.doc").Sections(2).Range
Range2.CheckGrammar
```

This example begins a spelling and grammar check on the selection.

```
Selection.Range.CheckGrammar
```

This example displays the result of a grammar check on the selection.

```
pass = Application.CheckGrammar(String:=Selection.Text)
MsgBox "Selection is grammatically correct: " & pass
```

CheckGrammarAsYouType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCheckGrammarAsYouTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCheckGrammarAsYouTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCheckGrammarAsYouTypeA "}

True if Word checks grammar and marks errors automatically as you type. Read/write **Boolean**.

Remarks

This property marks grammatical errors, but to see them on screen, you must set the **ShowGrammaticalErrors** property to **True**.

CheckGrammarAsYouType Property Example

This example sets Word to check for grammatical errors as you type and to display any errors found in the active document.

```
Options.CheckGrammarAsYouType = True
ActiveDocument.ShowGrammaticalErrors = True
```

This example returns the status of the **Check grammar as you type** option on the **Spelling & Grammar** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.CheckGrammarAsYouType
```

GrammarChecked Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGrammarCheckedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGrammarCheckedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGrammarCheckedA "}

True if a grammar check has been run on the specified range or document. **False** if some of the specified range or document hasn't been checked for grammar. Read/write **Boolean**.

Remarks

To recheck the grammar in a range or document, set the **GrammarChecked** property to **False**.

GrammarChecked Property Example

This example determines whether grammar has been checked in the active document. If it has, the word count is displayed. If grammar hasn't been checked, a spelling and grammar check is started.

```
Set myStat = ActiveDocument.ReadabilityStatistics
passGram = ActiveDocument.GrammarChecked
If passGram = True Then
    Msgbox myStat(1).Name & " - " & myStat(1).Value
Else
    ActiveDocument.CheckGrammar
End If
```

This example sets the **GrammarChecked** property to **False** for the active document, and then it runs a grammar check again.

```
ActiveDocument.GrammarChecked = False
ActiveDocument.CheckGrammar
```

ShowGrammaticalErrors Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowGrammaticalErrorsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowGrammaticalErrorsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowGrammaticalErrorsA "}

True if grammatical errors are marked by a wavy green line in the specified document. Read/write **Boolean**.

Note To view grammatical errors in your document, you must set the **CheckGrammarAsYouType** property to **True**.

ShowGrammaticalErrors Property Example

This example sets Word to check for grammatical errors as you type and to display any errors found in the active document.

```
Options.CheckGrammarAsYouType = True
ActiveDocument.ShowGrammaticalErrors = True
```

This example returns the opposite value of the **Hide grammatical errors in this document** option on the **Spelling & Grammar** tab in the **Options** dialog box (**Tools** menu).

```
temp = ActiveDocument.ShowGrammaticalErrors
```

DefaultWritingStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultWritingStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultWritingStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveDefaultStyleA"}

Returns or sets the default writing style used by the grammar checker for the specified language. The name of the writing style is the localized name for the specified language. Read/write **String**.

Remarks

This property controls the global setting for the writing style. When setting this property, you must use the exact name found in the **Writing style** box on the **Spelling & Grammar** tab in the **Options** dialog box (**Tools** menu).

The **ActiveWritingStyle** property sets the writing style for each language in a document. The **ActiveWritingStyle** setting overrides the **DefaultWritingStyle** setting.

DefaultWritingStyle Property Example

This example returns the default writing style in a message box.

```
myLang = Selection.LanguageID
Msgbox Languages(myLang).DefaultWritingStyle
```

This example sets the writing style for U.S. English to Casual, and then it checks spelling and grammar in the active document.

```
Languages(wdEnglishUS).DefaultWritingStyle = "Casual"
ActiveDocument.CheckGrammar
```

CustomDictionaries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCustomDictionariesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCustomDictionariesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCustomDictionariesA"}

Returns a **Dictionaries** object that represents the collection of active custom dictionaries. Active custom dictionaries are marked with a check in the **Custom Dictionaries** dialog box. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

CustomDictionaries Property Example

This example adds a new, blank custom dictionary to the collection. The path and file name of the new custom dictionary are then displayed in a message box.

```
Set myHome = CustomDictionaries.Add(Filename:="Home.dic")
Msgbox myHome.Path & Application.PathSeparator & myHome.Name
```

This example removes all custom dictionaries so that no custom dictionaries are active. The custom dictionary files aren't deleted, though.

```
CustomDictionaries.ClearAll
```

This example displays the name of each custom dictionary in the collection.

```
For Each di In CustomDictionaries
    MsgBox di.Name
Next di
```

ActiveCustomDictionary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveCustomDictionaryC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveCustomDictionaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveCustomDictionaryA"}

Returns or sets a **Dictionary** object that represents the custom dictionary to which words will be added. Read/write.

ActiveCustomDictionary Property Example

This example displays the full path and file name of the active custom dictionary.

```
Set myCust = Application.CustomDictionaries.ActiveCustomDictionary
MsgBox myCust.Path & Application.PathSeparator & myCust.Name
```

This example clears all existing custom dictionaries, adds a custom dictionary named "Home.dic," and then loads the new dictionary.

```
Application.CustomDictionaries.ClearAll
Set myCust = Application.CustomDictionaries.Add(FileName:="C:\Program
Files\Microsoft Office\WinWord\Home.dic")
CustomDictionaries.ActiveCustomDictionary = myCust
```

GrammaticalErrors Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGrammaticalErrorsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGrammaticalErrorsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGrammaticalErrorsA"}

Returns a **ProofreadingErrors** collection that represents the sentences that failed the grammar check on the specified document or range. There can be more than one error per sentence. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Remarks

If there are no grammatical errors, the **Count** property for the **ProofreadingErrors** object returned by the **GrammaticalErrors** property returns 0 (zero).

GrammaticalErrors Property Example

This example checks the third paragraph in the active document for grammatical errors and displays each sentence that contains one or more errors.

```
Set myErrors = ActiveDocument.Paragraphs(3).Range.GrammaticalErrors
For Each myerr In myErrors
    MsgBox myerr.Text
Next myerr
```

This example checks the active document for grammatical errors. If any errors are found, a new spelling and grammar check is started.

```
If ActiveDocument.GrammaticalErrors.Count = 0 Then
    Msgbox "There are no grammatical errors."
Else
    ActiveDocument.CheckGrammar
End If
```

ReadabilityStatistics Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReadabilityStatisticsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReadabilityStatisticsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReadabilityStatisticsA"}

Returns a **ReadabilityStatistics** collection that represents the readability statistics for the specified document or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ReadabilityStatistics Property Example

This example displays each readability statistic, along with its value, for document one.

```
For Each rs In Documents(1).ReadabilityStatistics
    Msgbox rs.Name & " - " & rs.Value
Next rs
```

WritingStyleList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWritingStyleListC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWritingStyleListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWritingStyleListA"}

Returns a string array that contains the names of all writing styles available for the specified language. Read-only **Variant**.

WritingStyleList Property Example

This example displays each writing style available for U.S. English. Each writing style and its number in the array are also displayed in the **Immediate** window in the Visual Basic editor.

```
WrStyles = Languages(wdEnglishUS).WritingStyleList
For i = 1 To UBound(WrStyles)
    MsgBox WrStyles(i)
    Debug.Print WrStyles(i) & " [" & Trim(Str$(i)) & "]"
Next i
```

ActiveWritingStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveWritingStyleC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveWritingStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveWritingStyleA"}

Returns or sets the writing style for a specified language in the specified document. Read/write **String**.

Note The **WritingStyleList** property returns an array of the names of the available writing styles.

Syntax

expression.**ActiveWritingStyle(*LanguageID*)**

expression Required. An expression that returns a **Document** object.

LanguageID Required **Variant**. The language you want to set the writing style for in the specified document. Can be either a string or one of the following **WdLanguageID** constants:

wdAfrikaans
wdArabic
wdBasque
wdBelgianDutch
wdBelgianFrench
wdBrazilianPortuguese
wdBulgarian
wdByelorussian
wdCatalan
wdCroation
wdCzech
wdDanish
wdDutch
wdEnglishAUS
wdEnglishCanadian
wdEnglishNewZealand
wdEnglishSouthAfrica
wdEnglishUK
wdEnglishUS
wdEstonian
wdFarsi
wdFinnish
wdFrench
wdFrenchCanadian
wdGerman
wdGreek
wdHebrew
wdHungarian
wdItalian
wdIcelandic
wdJapanese
wdKorean
wdLatvian
wdMacedonian
wdMalaysian
wdMexicanSpanish
wdNorwegianBokmol
wdNorwegianNynorsk
wdPolish
wdPortuguese
wdRomanian
wdRussian
wdSerbianCyrillic
wdSerbianLatin
wdSesotho
wdSimplifiedChinese
wdSlovak
wdSlovenian
wdSpanish
wdSpanishModernSort
wdSwedish
wdSwissFrench
wdSwissGerman
wdSwissItalian
wdTraditionalChinese
wdTsonga
wdTswana
wdTurkish
wdUkrainian
wdVenda
wdXhosa
wdZulu

ActiveWritingStyle Property Example

This example sets the writing style used for French, German, and U.S. English for the active document. You must have the grammar files installed for French, German, and US English to run this example.

```
With ActiveDocument
    .ActiveWritingStyle(wdFrench) = "Commercial"
    .ActiveWritingStyle(wdGerman) = "Technisch/Wiss"
    .ActiveWritingStyle(wdEnglishUS) = "Technical"
End With
```

This example returns the writing style for the language of the selection.

```
myLang = Selection.LanguageID
MsgBox ActiveDocument.ActiveWritingStyle(myLang)
```

BuildKeyCode Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthBuildKeyCodeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthBuildKeyCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthBuildKeyCodeA "}

Returns a unique number for the specified key combination.

Syntax

expression.**BuildKeyCode(*Arg1, Arg2, Arg3, Arg4*)**

expression Optional. An expression that returns an **Application** object.

Arg1 Required **Long**. A key you specify by using one of the **WdKey** constants.

Arg2 – ***Arg4*** Optional **Variant**. A key you specify by using one of the **WdKey** constants.

BuildKeyCode Method Example

This example assigns the ALT + F1 key combination to the **Organizer** command.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCode:=BuildKeyCode(Arg1:=wdKeyAlt, Arg2:=wdKeyF1), _
    KeyCategory:=wdKeyCategoryCommand, Command:="Organizer"
```

This example removes the ALT+F1 key assignment from the Normal template.

```
CustomizationContext = NormalTemplate
FindKey(BuildKeyCode(Arg1:=wdKeyAlt, Arg2:=wdKeyF1)).Clear
```

This example displays the command assigned to the F1 key.

```
CustomizationContext = NormalTemplate
MsgBox FindKey(BuildKeyCode(Arg1:=wdKeyF1)).Command
```

Disable Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDisableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDisableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDisableA "}

Removes the specified key combination if it's currently assigned to a command. After you use this method, the key combination has no effect. Using this method is the equivalent to clicking the **Remove** button in the **Customize Keyboard** dialog box (**Tools** menu).

Note Use the **Clear** method with a **KeyBinding** object to reset a built-in command to its default key assignment. You don't need to remove or rebind a **KeyBinding** object before adding it elsewhere.

Syntax

expression.**Disable**

expression Required. An expression that returns a **KeyBinding** object.

Disable Method Example

This example removes the CTRL+SHIFT+B key assignment. This key combination is assigned to the **Bold** command by default.

```
CustomizationContext = NormalTemplate
FindKey(BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyB)).Disable
```

This example assigns the CTRL+SHIFT+O key combination to the **Organizer** command. The example then uses the **Disable** method to remove the CTRL+SHIFT+O key combination and displays a message.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyO, wdKeyControl, wdKeyShift), _
    KeyCategory:=wdKeyCategoryCommand, Command:="Organizer"
With FindKey(BuildKeyCode(wdKeyO, wdKeyControl, wdKeyShift))
    MsgBox .Command & " is assigned to CTRL+Shift+O"
    .Disable
    If .Command = "" Then MsgBox "Nothing is assigned to CTRL+Shift+O"
End With
```

This example removes all key assignments for the global macro named "Macro1."

```
CustomizationContext = NormalTemplate
For Each aKey In KeysBoundTo(KeyCategory:=wdKeyCategoryMacro,
Command:="Macro1")
    aKey.Disable
Next aKey
```

Execute Method (Dialog, KeyBinding, and MailMerge Objects)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExecuteDialogKeyBindingMailMergeObjC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExecuteDialogKeyBindingMailMergeObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExecuteDialogKeyBindingMailMergeObjA"}

Dialog object: Applies the current settings of a Word dialog box.

KeyBinding object: Runs the command associated with the specified key combination.

MailMerge object: Performs the specified mail merge operation.

Syntax 1

expression.**Execute(*Pause*)**

Syntax 2

expression.**Execute**

expression Required. An expression that returns a **MailMerge** object (Syntax 1), or a **Dialog** object or **KeyBinding** object (Syntax 2).

Pause Optional **Variant**. **True** to have Word pause and display a troubleshooting dialog box if a mail merge error is found. **False** to report errors in a new document.

Execute Method (Dialog, KeyBinding, and MailMerge Objects) Example

The following example enables the **Keep with next** check box on the **Line and Page Breaks** tab in the **Paragraph** dialog box.

```
With Dialogs(wdDialogFormatParagraph)
    .KeepWithNext = 1
    .Execute
End With
```

This example assigns the CTRL+SHIFT+C key combination to the **FileClose** command and then executes the key combination (the document is closed).

```
CustomizationContext = ActiveDocument.AttachedTemplate
Keybindings.Add KeyCode:=BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyC), _
    KeyCategory:=wdKeyCategoryCommand, Command:="FileClose"
FindKey(BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyC)).Execute
```

This example executes a mail merge if the active document is a main document with an attached data source.

```
Set myMerge = ActiveDocument.MailMerge
If myMerge.State = wdMainAndDataSource Then MyMerge.Execute
```

Execute Method (Find Object)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExecuteFindObjC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExecuteFindObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExecuteFindObjA"}

Runs the specified find operation. Returns **True** if the find operation is successful.

Syntax

expression.**Execute(*FindText, MatchCase, MatchWholeWord, MatchWildcards, MatchSoundsLike, MatchAllWordForms, Forward, Wrap, Format, ReplaceWith, Replace*)**

expression Required. An expression that returns a **Find** object.

FindText Optional **Variant**. The text to be searched for. Use an empty string ("") to search for formatting only. You can search for special characters by specifying appropriate character codes. For example, "^p" corresponds to a paragraph mark and "^t" corresponds to a tab character. For a list of special characters you can use, see Examples of special characters and document elements you can find and replace.

If ***MatchWildcards*** is **True**, you can specify wildcard characters and other advanced search criteria. For example, "*(ing)" finds any word that ends in "ing." For more information, see Examples of search wildcards.

To search for a symbol character, type a caret (^),a zero (0), and then the symbol's character code. In Windows, for example, "^0151" corresponds to an em dash (—).

MatchCase Optional **Variant**. **True** to specify that the find text be case sensitive. Corresponds to the **Match case** check box in the **Find and Replace** dialog box (**Edit** menu).

MatchWholeWord Optional **Variant**. **True** to have the find operation locate only entire words, not text that's part of a larger word. Corresponds to the **Find whole words only** check box in the **Find and Replace** dialog box.

MatchWildcards Optional **Variant**. **True** to have the find text be a special search operator. Corresponds to the **Use wildcards** check box in the **Find and Replace** dialog box.

MatchSoundsLike Optional **Variant**. **True** to have the find operation locate words that sound similar to the find text be. Corresponds to the **Sounds like** check box in the **Find and Replace** dialog box.

MatchAllWordForms Optional **Variant**. **True** to have the find operation locate all forms of the find text (for example, "sit" locates "sitting" and "sat"). Corresponds to the **Find all word forms** check box in the **Find and Replace** dialog box.

Forward Optional **Variant**. **True** to search forward (toward the end of the document).

Wrap Optional **Variant**. Controls what happens if the search begins at a point other than the beginning of the document and the end of the document is reached (or vice versa if ***Forward*** is set to **False**). This argument also controls what happens if there's a selection or range and the search text isn't found in the selection or range. Can be one of the following **WdFindWrap** constants.

Constant	Description
wdFindAsk	After searching the selection or range, Word displays a message asking whether to search the remainder of the document.
wdFindContinue	The find operation continues if the beginning or end of the search range is reached.
wdFindStop	The find operation ends if the beginning or end of the search range is reached.

Format Optional **Variant**. **True** to have the find operation locate formatting in addition to or instead of the find text.

ReplaceWith Optional **Variant**. The replacement text. To delete the text specified by the ***Find*** argument, use an empty string (""). You specify special characters and advanced search criteria

just as you do for the ***Find*** argument.To specify a graphic object or other nontext item as the replacement, put the item on the Clipboard and specify "^c" for ***ReplaceWith***.

Replace Optional **Variant**. Specifies how many replacements are to be made: one, all, or none. Can be one of the following **WdReplace** constants: **wdReplaceAll**, **wdReplaceNone**, or **wdReplaceOne**.

Execute Method (Find Object) Example

This example finds and selects the next occurrence of the word "library."

```
With Selection.Find
    .ClearFormatting
    .MatchWholeWord = True
    .MatchCase = False
    .Execute FindText:="library"
End With
```

This example finds all occurrences of the word "hi" in the active document and replaces each occurrence with "hello."

```
Set myRange = ActiveDocument.Content
myRange.Find.Execute FindText:="hi", ReplaceWith:="hello",
Replace:=wdReplaceAll
```

FindKey Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFindKeyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFindKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFindKeyA "}

Returns a **KeyBinding** object that represents the specified key combination. Read-only.

Syntax

expression.**FindKey(*KeyCode, KeyCode2*)**

expression Optional. An expression that returns an **Application** object.

KeyCode Required **Long**. A key you specify by using one of the **WdKey** constants.

KeyCode2 Optional **Variant**. A second key you specify by using one of the **WdKey** constants.

Remarks

You can use the **BuildKeyCode** method to create the ***KeyCode*** or ***KeyCode2*** argument.

FindKey Property Example

This example disables the ALT+SHIFT+F12 key combination in the template attached to the active document. To return a **KeyBinding** object that includes more than two keys, use the **BuildKeyCode** method, as shown in the example.

```
CustomizationContext = ActiveDocument.AttachedTemplate
FindKey(KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyShift, wdKeyF12)).Disable
```

This example displays the command assigned to the F1 key.

```
CustomizationContext = NormalTemplate
MsgBox FindKey(KeyCode:=wdKeyF1).Command
```

Key Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthKeyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthKeyA "}

Returns a **KeyBinding** object that represents the specified custom key combination. If the key combination doesn't exist, this method returns **Nothing**.

Syntax

expression.**Key(*KeyCode, KeyCode2*)**

expression Required. An expression that returns a **KeyBindings** or **KeysBoundTo** object.

KeyCode Required **Long**. A key you specify by using one of the **WdKey** constants.

KeyCode2 Optional **Variant**. A second key you specify by using one of the **WdKey** constants.

Remarks

You can use the **BuildKeyCode** method to create the ***KeyCode*** or ***KeyCode2*** argument.

Key Method Example

This example assigns the ALT+F4 key combination to the Arial font and then displays the number of items in the **KeyBindings** collection. The example then clears the key combinations (returns it to its default setting) and redisplays the number of items in the **KeyBindings** collection.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyF4), _
    KeyCategory:=wdKeyCategoryFont, Command:="Arial"
MsgBox KeyBindings.Count & " keys in KeyBindings collection"
KeyBindings.Key(KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyF4)).Clear
MsgBox KeyBindings.Count & " keys in KeyBindings collection"
```

This example assigns the CTRL+SHIFT+U key combination to the macro named "Macro1" in the active document. The example uses the **Key** property to return a **KeyBinding** object so that Word can retrieve and display the command name.

```
CustomizationContext = ActiveDocument
KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyU), _
    KeyCategory:=wdKeyCategoryMacro, Command:="Macro1"
MsgBox KeyBindings.Key(BuildKeyCode(wdKeyControl, wdKeyShift,
wdKeyU)).Command
```

This example determines whether the CTRL+SHIFT+A key combination is part of the **KeyBindings** collection.

```
CustomizationContext = NormalTemplate
Set myKey = KeyBindings.Key(BuildKeyCode(wdKeyControl, wdKeyShift,wdKeyA))
If (myKey Is Nothing) Then MsgBox "Key is not in the KeyBindings
collection"
```

KeyBindings Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeyBindingsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeyBindingsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeyBindingsA "}

Returns a **KeyBindings** collection that represents customized key assignments, which include a key code, a key category, and a command. Read-only.

KeyBindings Property Example

This example assigns the CTRL+ALT+W key combination to the **FileClose** command. This keyboard customization is saved in the Normal template.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyW), _
    KeyCategory:=wdKeyCategoryCommand, Command:="FileClose"
```

This example inserts the command name and key combination string for each item in the **KeyBindings** collection.

```
CustomizationContext = NormalTemplate
For Each aKey In KeyBindings
    Selection.InsertAfter aKey.Command & vbTab & aKey.KeyString & vbCr
    Selection.Collapse Direction:=wdCollapseEnd
Next aKey
```

KeyCode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeyCodeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeyCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeyCodeA "}

Returns a unique number for the first key in the specified key binding. Read-only **Long**.

Note You create this number by using the **BuildKeyCode** method when you're adding key bindings by using the **Add** method of the **KeyBindings** object.

KeyCode Property Example

This example displays a message if the **KeyBindings** collection includes the ALT+CTRL+W key combination.

```
CustomizationContext = NormalTemplate
aCode = BuildKeyCode(wdKeyAlt, wdKeyControl, wdKeyW)
For Each aKey In KeyBindings
    If aCode = aKey.KeyCode Then
        MsgBox aKey.KeyString & " is already in use"
    End If
Next aKey
```

KeyCode2 Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeyCode2C "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeyCode2X":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeyCode2A "}

Returns a unique number for the second key in the specified key binding. Read-only **Long**.

KeyCode2 Property Example

This example displays the key codes of each key in the **KeyBindings** collection (the collection of all the customized keys in the active document).

```
CustomizationContext = ActiveDocument
For Each aKey In KeyBindings
    If aKey.KeyCode2 <> wdNoKey Then
        MsgBox aKey.KeyString & vbCr & "KeyCode1 = " & _
            aKey.KeyCode & vbCr & "KeyCode2 = " & aKey.KeyCode2
    Else
        MsgBox aKey.KeyString & vbCr & "KeyCode1 = " & aKey.KeyCode
    End If
Next aKey
```

KeyString Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthKeyStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthKeyStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthKeyStringA "}

Returns the key combination string for the specified keys (for example, CTRL+SHIFT+A).

Syntax

expression.**KeyString**(***KeyCode, KeyCode2***)

expression Optional. An expression that returns an **Application** object.

KeyCode Required **Long**. A key you specify by using one of the **WdKey** constants.

KeyCode2 Optional **Variant**. A second key you specify by using one of the **WdKey** constants.

Remarks

You can use the **BuildKeyCode** method to create the ***KeyCode*** or ***KeyCode2*** argument.

KeyString Method Example

This example displays the key combination string (CTRL+SHIFT+A) for the following **WdKey** constants: **wdKeyControl**, **wdKeyShift**, and **wdKeyA**.

```
CustomizationContext = ActiveDocument.AttachedTemplate
MsgBox KeyString(KeyCode:=BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyA))
```

KeyString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeyStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeyStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeyStringA "}

Returns the key combination string for the specified keys (for example, CTRL+SHIFT+A). Read-only **String**.

KeyString Property Example

This example displays the key combination string for the first customized key combination in the Normal template.

```
CustomizationContext = NormalTemplate
If KeyBindings.Count >= 1 Then
    MsgBox KeyBindings(1).KeyString
End If
```

This example displays a message if the **KeyBindings** collection includes the ALT+CTRL+W key combination.

```
CustomizationContext = NormalTemplate
aCode = BuildKeyCode(wdKeyAlt, wdKeyControl, wdKeyW)
For Each aKey In KeyBindings
    If aCode = aKey.KeyCode Then
        MsgBox aKey.KeyString & " is already in use"
    End If
Next aKey
```

Protected Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProtectedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProtectedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProtectedA "}

True if you cannot change the specified key binding in the **Customize Keyboard** dialog box (**Tools** menu). Read-only **Boolean**.

Note The **Add** method of the **KeyBindings** object adds a key binding regardless of the protected status.

Protected Property Example

This example displays the protection status for the CTRL+S key binding.

```
CustomizationContext = ActiveDocument.AttachedTemplate
MsgBox FindKey(BuildKeyCode(wdKeyControl, wdKeyS)).Protected
```

This example displays a message if the A key binding is protected.

```
CustomizationContext = NormalTemplate
If FindKey(BuildKeyCode(wdKeyA)).Protected = True Then
    MsgBox "The A key is protected"
End If
```

Rebind Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRebindC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRebindX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRebindA "}

Changes the command assigned to the specified key binding.

Syntax

expression.**Rebind(Key*Category, Command, CommandParameter*)**

expression Required. An expression that returns a **KeyBinding** object.

KeyCategory Required **Long**. The key category of the specified key binding. Can be one of the **WdKeyCategory** constants:

WdKeyCategoryAutoText	**WdKeyCategoryNil**
WdKeyCategoryCommand	**WdKeyCategoryPrefix**
WdKeyCategoryDisable	**WdKeyCategoryStyle**
WdKeyCategoryFont	**WdKeyCategorySymbol**
WdKeyCategoryMacro	

Command Required **String**. The name of the specified command.

CommandParameter Optional **Variant**. Additional text, if any, required for the command specified by ***Command***. For information about values for this argument, see the **Add** method for the **KeyBindings** object.

Rebind Method Example

This example reassigns the CTRL+SHIFT+S key binding to the **FileSaveAs** command.

```
CustomizationContext = NormalTemplate
Set myKey = FindKey(BuildKeyCode(wdKeyControl, wdKeyShift, wdKeyS))
myKey.Rebind KeyCategory:=wdKeyCategoryCommand, Command:="FileSaveAs"
```

This example rebinds all keys assigned to the macro named "Macro1" to the macro named "ReportMacro."

```
CustomizationContext = ActiveDocument.AttachedTemplate
For Each myKey In KeysBoundTo(KeyCategory:=wdKeyCategoryMacro, _
    Command:="Macro1")
    myKey.Rebind KeyCategory:=wdKeyCategoryMacro, Command:="ReportMacro"
Next myKey
```

Command Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCommandC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCommandX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCommandA"}

Returns the command assigned to the specified key combination. Read-only **String**.

Command Property Example

This example displays the keys assigned to font names. A message is displayed if no keys have been assigned to fonts.

```
For Each aKey In KeyBindings
    If aKey.KeyCategory = wdKeyCategoryFont Then
        Count = Count + 1
        MsgBox aKey.Command & vbCr & aKey.KeyString
    End If
Next aKey
If Count = 0 Then MsgBox "Keys haven't been assigned to fonts"
```

KeyCategory Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeyCategoryC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeyCategoryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeyCategoryA"}

Returns the type of item assigned to the specified key binding. Read-only **Long**.

Can be one of the following **WdKeyCategory** constants:

wdKeyCategoryAutoText
wdKeyCategoryCommand
wdKeyCategoryDisable
wdKeyCategoryFont
wdKeyCategoryMacro
wdKeyCategoryNil
wdKeyCategoryPrefix
wdKeyCategoryStyle
wdKeyCategorySymbol

KeyCategory Property Example

This example displays the keys assigned to font names. A message is displayed if no keys have been assigned to fonts.

```
For Each aKey In KeyBindings
    If aKey.KeyCategory = wdKeyCategoryFont Then
        Count = Count + 1
        MsgBox aKey.Command & vbCr & aKey.KeyString
    End If
Next aKey
If Count = 0 Then MsgBox "Keys haven't been assigned to fonts"
```

Add Method (KeyBindings Collection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddKeyBindingsObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddKeyBindingsObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddKeyBindingsObjA "}

Adds a new **KeyBinding** object to the **KeyBindings** collection. A **KeyBinding** object represents a custom key assignment, which includes a key code, a key category, and a command.

Syntax

expression.**Add(*KeyCategory, Command, KeyCode, KeyCode2, CommandParameter*)**

expression Required. An expression that returns a **KeyBindings** object.

KeyCategory Required **Long**. The category of the key assignment. Can be one of the following **WdKeyCategory** constants:

wdKeyCategoryAutoText	**wdKeyCategoryNil**
wdKeyCategoryCommand	**wdKeyCategoryPrefix**
wdKeyCategoryDisable	**wdKeyCategoryStyle**
wdKeyCategoryFont	**wdKeyCategorySymbol**
wdKeyCategoryMacro	

Command Required **String**. The command that the specified key combination executes.

KeyCode Required **Long**. A key you specify by using one of the **WdKey** constants.

KeyCode2 Optional **Variant**. A second key you specify by using one of the **WdKey** constants.

CommandParameter Optional **Variant**. Additional text, if any, required for the command specified by ***Command***. For details, see the "Remarks" section.

Remarks

You can use the **BuildKeyCode** method to create the ***KeyCode*** or ***KeyCode2*** argument.

In the following table, the left-hand column contains commands that require a command value, and the right-hand column describes what you must do to specify ***CommandParameter*** for each of these commands. (The equivalent action in the **Customize Keyboard** dialog box (**Tools** menu) to specifying ***CommandParameter*** is selecting an item in the list box that appears when you select one of the following commands in the **Commands** box.)

If *Command* is set to	CommandParameter must be
Borders, **Color**, or **Shading**	A number – specified as text – corresponding to the position of the setting selected in the list box that contains values, where 0 (zero) is the first item, 1 is the second item, and so on
Columns	A number between 1 and 45 – specified as text – corresponding to the number of columns you want to apply
Condensed	A text measurement between 0.1 point and 12.75 points specified in 0.05-point increments (72 points = 1 inch)
Expanded	A text measurement between 0.1 point and 12.75 points specified in 0.05-point increments (72 points = 1 inch)
FileOpenFile	The path and file name of the file to be opened. If the path isn't specified, the current folder is used.
Font Size	A positive text measurement, specified in 0.5-point increments (72 points = 1 inch)

Lowered, Raised	A text measurement between 1 point and 64 points, specified in 0.5-point increments (72 points = 1 inch)
Symbol	A string created by concatenating a **Chr()** instruction and the name of a symbol font (for example, `Chr(167) & "Symbol"`)

Add Method (KeyBindings Collection) Example

This example adds the CTRL+ALT+W key combination to the **FileClose** command. The keyboard customization is saved in the Normal template.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCategory:=wdKeyCategoryCommand, Command:="FileClose",
KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyW)
```

This example adds the ALT+F4 key combination to the Arial font and then displays the number of items in the **KeyBindings** collection. The example then clears the ALT+F4 key combination (returned it to its default setting) and redisplays the number of items in the **KeyBindings** collection.

```
CustomizationContext = ActiveDocument.AttachedTemplate
Set myKey = KeyBindings.Add(KeyCategory:=wdKeyCategoryFont,
Command:="Arial", _
    KeyCode:=BuildKeyCode(wdKeyAlt, wdKeyF4))
MsgBox KeyBindings.Count & " keys in KeyBindings collection"
myKey.Clear
MsgBox KeyBindings.Count & " keys in KeyBindings collection"
```

This example adds the CTRL+ALT+S key combination to the **Font** command with 8 points specified for the font size.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCategory:=wdKeyCategoryCommand, Command:="FontSize", _
    KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyS),
CommandParameter:="8"
```

This example adds the CTRL+ALT+H key combination to the Heading 1 style in the active document.

```
CustomizationContext = ActiveDocument
KeyBindings.Add KeyCategory:=wdKeyCategoryStyle, Command:="Heading 1", _
    KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyH)
```

This example adds the CTRL+ALT+O key combination to the AutoText entry named "Hello."

```
CustomizationContext = ActiveDocument
KeyBindings.Add KeyCategory:=wdKeyCategoryAutoText, Command:="Hello", _
    KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyO)
```

Context Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContextC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContextA"}

Returns an object that represents the storage location of the specified key binding. This property can return a **Document**, **Template**, or **Application** object. Read-only.

Note Built-in key assignments (for example, CTRL+I for **Italic**) return the **Application** object as the context. Any key bindings you add will return a **Document** or **Template** object, depending on the customization context in effect when the **KeyBinding** object was added.

Context Property Example

This example displays the name of the document or template where the macro named "Macro1" is stored.

```
Set kb = KeysBoundTo(KeyCategory:=wdKeyCategoryMacro, Command:="Macro1")
MsgBox kb.Context.Name
```

This example adds the F2 key to the **Italic** command and then uses the **For Each...Next** loop to display the keys assigned to the **Italic** command along with the context.

```
CustomizationContext = NormalTemplate
KeyBindings.Add KeyCategory:=wdKeyCategoryCommand, _
    Command:="Italic", KeyCode:=wdKeyF2
For Each oKey In KeysBoundTo(KeyCategory:=wdKeyCategoryCommand,
Command:="Italic")
    MsgBox oKey.KeyString & vbCr & oKey.Context.Name
Next
```

KeysBoundTo Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeysBoundToC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeysBoundToX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeysBoundToA "}

Returns a **KeysBoundTo** object that represents all the key combinations assigned to the specified item. Read-only.

Syntax

expression.**KeysBoundTo(*KeyCategory, Command, CommandParameter*)**

expression Optional. An expression that returns an **Application** object.

KeyCategory Required **Long**. The category of the key combination. Can be one of the following **WdKeyCategory** constants: **wdKeyCategoryAutoText**, **wdKeyCategoryCommand**, **wdKeyCategoryFont**, **wdKeyCategoryMacro**, **wdKeyCategoryStyle**, or **wdKeyCategorySymbol**.

Command Required **String**. The name of the command.

CommandParameter Optional **Variant**. Additional text, if any, required for the command specified by ***Command***. For more information, see the "Remarks" section in **Add Method (KeyBindings Object).**

KeysBoundTo Property Example

This example displays all the key combinations assigned to the **FileOpen** command in the template attached to the active document.

```
CustomizationContext = ActiveDocument.AttachedTemplate
For Each myKey In KeysBoundTo(KeyCategory:=wdKeyCategoryCommand,
Command:="FileOpen")
    myStr = myStr & myKey.KeyString & vbCr
Next myKey
MsgBox myStr
```

This example removes all key assignments from Macro1 in the Normal template.

```
CustomizationContext = NormalTemplate
For Each aKey In KeysBoundTo(KeyCategory:=wdKeyCategoryMacro,
Command:="Macro1")
    aKey.Disable
Next aKey
```

AttentionLine Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAttentionLineC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAttentionLineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAttentionLineA "}

Returns or sets the attention line text for a letter created by the Letter Wizard. Read/write **String**.

AttentionLine Property Example

This example retrieves the Letter Wizard elements from the active document. If the attention line isn't blank, the example displays the text in a message box.

```
If ActiveDocument.GetLetterContent.AttentionLine <> "" Then
    MsgBox ActiveDocument.GetLetterContent.AttentionLine
End If
```

This example retrieves the Letter Wizard elements from the active document, changes the attention line text, and then uses the **SetLetterContent** method to update the document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.AttentionLine = "Greetings"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

CCList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCCListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCCListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCCListA "}

Returns or sets the carbon copy (CC) recipients for a letter created by the Letter Wizard. Read/write **String**.

CCList Property Example

This example displays the CC list text for the active document.

```
MsgBox ActiveDocument.GetLetterContent.CCList
```

This example creates a new **LetterContent** object, sets the courtesy copies by setting the **CClist** property, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myLetterContent = New LetterContent
myLetterContent.CCList = "K. Jordan, D. Funk, D. Morrison"
ActiveDocument.RunLetterWizard LetterContent:=myLetterContent
```

Closing Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproClosingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproClosingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproClosingA "}

Returns or sets the closing text for a letter created by the Letter Wizard (for example, "Sincerely yours"). Read/write **String**.

Closing Property Example

This example displays the closing text from the active document.

```
MsgBox ActiveDocument.GetLetterContent.Closing
```

This example retrieves letter elements from the active document, changes the closing text by setting the **Closing** property, and then uses the **SetLetterContent** method to update the document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.Closing = "Sincerely yours,"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

DateFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDateFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDateFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDateFormatA "}

Returns or sets the date for a letter created by the Letter Wizard. Read/write **String**.

DateFormat Property Example

This example displays the date from the letter that appears in the active document.

```
MsgBox ActiveDocument.GetLetterContent.DateFormat
```

This example creates a new **LetterContent** object, sets the date line to the current date, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set aLetterContent = New LetterContent
aLetterContent.DateFormat = Date$
ActiveDocument.RunLetterWizard LetterContent:=aLetterContent
```

EnclosureNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnclosureNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnclosureNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnclosureNumberA "}

Returns or sets the number of enclosures for a letter created by the Letter Wizard. Read/write **String**.

EnclosureNumber Property Example

This example displays the number of enclosures specified in the active document.

```
MsgBox ActiveDocument.GetLetterContent.EnclosureNumber
```

This example retrieves letter elements from the active document, changes the number of enclosures by setting the **EnclosureNumber** property, and then uses the **SetLetterContent** method to update the active document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.EnclosureNumber = "5"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

GetLetterContent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGetLetterContentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGetLetterContentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGetLetterContentA "}

Retrieves letter elements from the specified document and returns a **LetterContent** object.

Syntax

expression.**GetLetterContent**

expression Required. An expression that returns a **Document** object.

GetLetterContent Method Example

This example displays the salutation and recipient name from the letter in the active document.

```
MsgBox ActiveDocument.GetLetterContent.Salutation &
ActiveDocument.GetLetterContent.RecipientName
```

This example retrieves letter elements from the active document, changes the list of carbon copy (CC) recipients by setting the **CCIist** property, and then uses the **SetLetterContent** method to update the active document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
With myLetterContent
    .CCList = "J. Burns, L. Scarpaczyk, K. Wong"
    .RecipientName = "Amy Anderson"
    .RecipientAddress = "123 Main" & vbCr & "Bellevue, WA  98004"
    .LetterStyle = wdFullBlock
End With
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

Letterhead Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLetterheadC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLetterheadX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLetterheadA "}

True if space is reserved for a preprinted letterhead in a letter created by the Letter Wizard. Read/write **Boolean**.

Note The **LetterheadSize** property controls the size of the reserved letterhead space.

Letterhead Property Example

This example creates a new **LetterContent** object, reserves an inch of space at the top of the page for a preprinted letterhead, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set aLetterContent = New LetterContent
With aLetterContent
    .Letterhead = True
    .LetterheadLocation = wdLetterTop
    .LetterheadSize = InchesToPoints(1)
End With
ActiveDocument.RunLetterWizard LetterContent:=aLetterContent,
WizardMode:=True
```

LetterheadLocation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLetterheadLocationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLetterheadLocationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLetterheadLocationA "}

Returns or sets the location of the preprinted letterhead in a letter created by the Letter Wizard. Can be one of the following **WdLetterheadLocation** constants: **wdLetterBottom**, **wdLetterLeft**, **wdLetterRight**, or **wdLetterTop**. Read/write **Long**.

LetterheadLocation Property Example

This example creates a new **LetterContent** object, reserves an inch of space at the top of the page for a preprinted letterhead, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set aLetterContent = New LetterContent
With aLetterContent
    .Letterhead = True
    .LetterheadLocation = wdLetterTop
    .LetterheadSize = InchesToPoints(1)
End With
ActiveDocument.RunLetterWizard LetterContent:=aLetterContent
```

LetterheadSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLetterheadSizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLetterheadSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLetterheadSizeA "}

Returns or sets the amount of space (in points) to be reserved for a preprinted letterhead in a letter created by the Letter Wizard. Read/write **Single**.

LetterheadSize Property Example

This example retrieves the Letter Wizard elements from the active document, changes the preprinted letterhead settings, and then uses the **SetLetterContent** method to update the active document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
With myLetterContent
    .Letterhead = True
    .LetterheadLocation = wdLetterTop
    .LetterheadSize = InchesToPoints(1.5)
End With
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

LetterStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLetterStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLetterStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLetterStyleA "}

Returns or sets the layout of a letter created by the Letter Wizard. Can be one of the following **WdLetterStyle** constants: **wdFullBlock**, **wdModifiedBlock**, or **wdSemiBlock**. Read/write **Long**.

LetterStyle Property Example

This example creates a new **LetterContent** object, selects a letter style, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set aLetterContent = New LetterContent
aLetterContent.LetterStyle = wdFullBlock
ActiveDocument.RunLetterWizard LetterContent:=aLetterContent,
WizardMode:=True
```

MailingInstructions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailingInstructionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailingInstructionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailingInstructionsA "}

Returns or sets the mailing instruction text for a letter created by the Letter Wizard (for example, "Certified Mail"). Read/write **String**.

MailingInstructions Property Example

This example retrieves the Letter Wizard elements from the active document, changes the text of the mailing instructions, and then uses the **SetLetterContent** method to update the active document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.MailingInstructions = "Air Mail"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

This example creates a new **LetterContent** object, sets several properties (including the mailing instruction text), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .RecipientReference = "In reply to:"
    .Salutation ="Hello"
    .MailingInstructions = "Certified Mail"
End With
Documents.Add.RunLetterWizard LetterContent:=myContent
```

PageDesign Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageDesignC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageDesignX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageDesignA "}

Returns or sets the name of the template attached to the document created by the Letter Wizard. Read/write **String**.

PageDesign Property Example

This example creates a new **LetterContent** object, includes the header and footer from the Contemporary Letter template, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .PageDesign = "C:\MSOffice\Templates\Letters & Faxes\Contemporary
Letter.dot"
    .IncludeHeaderFooter = True
End With
Documents.Add.RunLetterWizard LetterContent:=myContent
```

RecipientAddress Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientAddressA "}

Returns or sets the mailing address of the person who'll be receiving the letter created by the Letter Wizard. Read/write **String**.

RecipientAddress Property Example

This example creates a new **LetterContent** object, sets several properties (including the recipient address), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Dim oLC as New LetterContent
With oLC
    .ReturnAddress = Application.UserAddress
    .RecipientName = "Amy Anderson"
    .RecipientAddress = "123 Main" & vbCr & "Bellevue, WA  98004"
End With
Documents.Add.RunLetterWizard LetterContent:=oLC, WizardMode:=True
```

RecipientName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientNameA "}

Returns or sets the name of the person who'll be receiving the letter created by the Letter Wizard. Read/write **String**.

RecipientName Property Example

This example displays the salutation and recipient name for the active document.

```
MsgBox ActiveDocument.GetLetterContent.Salutation & _
    Space(1) & ActiveDocument.GetLetterContent.RecipientName
```

This example creates a new **LetterContent** object, sets several properties (including the recipient name), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Dim oLC as New LetterContent
With oLC
    .LetterStyle = wdFullBlock
    .ReturnAddress = Application.UserAddress
    .RecipientName = "Amy Anderson"
    .RecipientAddress = "123 Main" & vbCr & "Bellevue, WA  98004"
End With
Documents.Add.RunLetterWizard LetterContent:=oLC, WizardMode:=True
```

Salutation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSalutationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSalutationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSalutationA "}

Returns or sets the salutation text for a letter created by the Letter Wizard. Read/write **String**.

Salutation Property Example

This example creates a new **LetterContent** object, sets several properties (including the salutation text), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
myContent.Salutation ="Hello,"
Documents.Add.RunLetterWizard LetterContent:=myContent
```

SalutationType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSalutationTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSalutationTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSalutationTypeA "}

Returns or sets type of salutation style for a letter created by the Letter Wizard. Can be one of the following **WdSalutationType** constants: **wdSalutationBusiness**, **wdSalutationFormat**, **wdSalutationInformal**, or **wdSalutationOther**. Read/write **Long**.

SalutationType Property Example

This example creates a new **LetterContent** object, sets several properties (including the salutation text), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
myContent.SalutationType = wdSalutationBusiness
Documents.Add.RunLetterWizard LetterContent:=myContent, WizardMode:=True
```

SenderCompany Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderCompanyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderCompanyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderCompanyA "}

Returns or sets the company name of the person creating a letter with the Letter Wizard. Read/write **String**.

SenderCompany Property Example

This example retrieves the Letter Wizard elements from the active document. If the sender's company name isn't blank, the example displays the text in a message box.

```
If ActiveDocument.GetLetterContent.SenderCompany <> "" Then
    MsgBox ActiveDocument.GetLetterContent.SenderCompany
End If
```

SenderJobTitle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderJobTitleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderJobTitleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderJobTitleA "}

Returns or sets the job title of the person creating a letter with the Letter Wizard. Read/write **String**.

SenderJobTitle Property Example

This example retrieves the Letter Wizard elements from the active document and displays the sender's job title.

```
Set myLetterContent = ActiveDocument.GetLetterContent
MsgBox myLetterContent.SenderJobTitle
```

SenderName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderNameA "}

Returns or sets the name of the person creating a letter with the Letter Wizard. Read/write **String**.

SenderName Property Example

This example creates a new **LetterContent** object, with the sender name and initials from the **User Information** tab in the **Options** dialog box (**Tools** menu). The example creates a new document and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .SenderName = Application.UserName
    .SenderInitials =Application.UserInitials
End With
Documents.Add.RunLetterWizard LetterContent:=myContent, WizardMode:=True
```

SetLetterContent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetLetterContentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetLetterContentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetLetterContentA "}

Inserts the contents of the specified **LetterContent** object into a document.

Syntax

expression.**SetLetterContent(*LetterContent*)**

expression Required. An expression that returns a **Document** object.

LetterContent Required **LetterContent** object. The **LetterContent** object that includes the various elements of the letter.

Remarks

This method is similar to the **RunLetterWizard** method except that it doesn't display the Letter Wizard dialog box. The method adds, deletes, or restyles letter elements in the specifed document based on the contents of the **LetterContent** object.

SetLetterContent Method Example

This example retrieves the Letter Wizard elements from the active document, changes the attention line text, and then uses the **SetLetterContent** method to update the active document to reflect the changes.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.AttentionLine = "Greetings"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

SenderInitials Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderInitialsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderInitialsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderInitialsA "}

Returns or sets the initials of the person creating a letter with the Letter Wizard. Read/write **String**.

SenderInitials Property Example

This example creates a new **LetterContent** object, with the sender name and initials from the **User Information** tab in the **Options** dialog box (**Tools** menu). The example creates a new document and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .SenderName = Application.UserName
    .SenderInitials =Application.UserInitials
End With
Documents.Add.RunLetterWizard LetterContent:=myContent, WizardMode:=True
```

RunLetterWizard Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRunLetterWizardC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRunLetterWizardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRunLetterWizardA "}

Runs the Letter Wizard on the specified document.

Syntax

expression.**RunLetterWizard(*LetterContent, WizardMode*)**

expression Required. An expression that returns a **Document** object.

LetterContent Optional **Variant**. A **LetterContent** object. Any filled properties in the **LetterContent** object show up as prefilled elements in the Letter Wizard dialog boxes. If this argument is omitted, the **GetLetterContent** method is automatically used to get a **LetterContent** object from the specified document.

WizardMode Optional **Variant**. **True** to display the **Letter Wizard** dialog box as a series of steps with a **Next**, **Back**, and **Finish** button. **False** to display the **Letter Wizard** dialog box as if it were opened from the **Tools** menu (a properties dialog box with an **OK** button and a **Cancel** button). The default value is **False**.

Remarks

Use the **CreateLetterContent** method to return a **LetterContent** object, given various letter element properties. Use the **GetLetterContent** method to return a **LetterContent** object based on the contents of the specified document. You can use the resulting **LetterContent** object with the **RunLetterWizard** method to preset elements in the **Letter Wizard** dialog box.

RunLetterWizard Method Example

This example creates a new **LetterContent** object, sets several properties for it, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .Salutation ="Hello"
    .SalutationType = wdSalutationOther
    .SenderName = Application.UserName
    .SenderInitials =Application.UserInitials
End With
Documents.Add.RunLetterWizard LetterContent:=myContent, WizardMode:=True
```

The following example uses the **CreateLetterContent** method to create a new **LetterContent** object in the active document, and then it uses this object with the **RunLetterWizard** method.

```
Set myLetter = ActiveDocument.CreateLetterContent(DateFormat:="July 31,
1996", _
    IncludeHeaderFooter:=False, _
    PageDesign:="C:\MSOffice\Templates\Letters & Faxes\Contemporary
Letter.dot", _
    LetterStyle:=wdFullBlock, Letterhead:=True,
LetterheadLocation:=wdLetterTop, _
    LetterheadSize:=InchesToPoints(1.5), RecipientName:="Dave Edson", _
    RecipientAddress:="436 SE Main St." & vbCr & "Bellevue, WA 98004", _
    Salutation:="Dear Dave,", SalutationType:=wdSalutationInformal, _
    RecipientReference:="", MailingInstructions:="", AttentionLine:="", _
    Subject:="End of year report", CCList:="", ReturnAddress:="", _
    SenderName:="", Closing:="Sincerely yours,", SenderCompany:="", _
    SenderJobTitle:="", SenderInitials:="", EnclosureNumber:=0)
ActiveDocument.RunLetterWizard LetterContent:=myLetter
```

CreateLetterContent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreateLetterContentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreateLetterContentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreateLetterContentA"}

Creates and returns a **LetterContent** object based on the specified letter elements.

Syntax

expression.**CreateLetterContent(*DateFormat, IncludeHeaderFooter, PageDesign, LetterStyle, Letterhead, LetterheadLocation, LetterheadSize, RecipientName, RecipientAddress, Salutation, SalutationType, RecipientReference, MailingInstructions, AttentionLine, Subject, CCList, ReturnAddress, SenderName, Closing, SenderCompany, SenderJobTitle, SenderInitials, EnclosureNumber, InfoBlock, RecipientCode, RecipientGender, ReturnAddressShortForm, SenderCity, SenderCode, SenderGender, SenderReference*)**

expression Required. An expression that returns a **Document** object.

DateFormat Required **String**. The date for the letter.

IncludeHeaderFooter Required **Boolean**. **True** to include the header and footer from the page design template.

PageDesign Required **String**. The name of the template attached to the document.

LetterStyle Required **Long**. The document layout. Can be one of the following **WdLetterStyle** constants: **wdFullBlock**, **wdModifiedBlock**, or **wdSemiBlock**.

Letterhead Required **Boolean**. **True** to reserve space for a preprinted letterhead.

LetterheadLocation Required **Long**. The location of the preprinted letterhead. Can be one of the following **WdLetterheadLocation** constants: **wdLetterBottom**, **wdLetterLeft**, **wdLetterRight**, or **wdLetterTop**.

LetterheadSize Required **Single**. The amount of space (in points) to be reserved for a preprinted letterhead.

RecipientName Required **String**. The name of the person who'll be receiving the letter.

RecipientAddress Required **String**. The mailing address of the person who'll be receiving the letter.

Salutation Required **String**. The salutation text for the letter.

SalutationType Required **Long**. The salutation type for the letter. Can be one of the following **WdSalutationType** constants: **wdSalutationBusiness**, **wdSalutationFormat**, **wdSalutationInformal**, or **wdSalutationOther**.

RecipientReference Required **String**. The reference line text for the letter (for example, "In reply to:").

MailingInstructions Required **String**. The mailing instruction text for the letter (for example, "Certified Mail").

AttentionLine Required **String**. The attention line text for the letter (for example, "Attention:").

Subject Required **String**. The subject text for the specified letter.

CCList Required **String**. The names of the carbon copy (CC) recipients for the letter.

ReturnAddress Required **String**. The text of the return mailing address for the letter.

SenderName Required **String**. The name of the person sending the letter.

Closing Required **String**. The closing text for the letter.

SenderCompany Required **String**. The company name of the person creating the letter.

SenderJobTitle Required **String**. The job title of the person creating the letter.

SenderInitials Required **String**. The initials of the person creating the letter.

EnclosureNumber Required **Long**. The number of enclosures for the letter.

InfoBlock Optional **Variant**. Not used in the U.S. English version of Word.

RecipientCode Optional **Variant**. Not used in the U.S. English version of Word.
RecipientGender Optional **Variant**. Not used in the U.S. English version of Word.
ReturnAddressShortForm Optional **Variant**. Not used in the U.S. English version of Word.
SenderCity Optional **Variant**. Not used in the U.S. English version of Word.
SenderCode Optional **Variant**. Not used in the U.S. English version of Word.
SenderGender Optional **Variant**. Not used in the U.S. English version of Word.
SenderReference Optional **Variant**. Not used in the U.S. English version of Word.

CreateLetterContent Method Example

The following example uses the **CreateLetterContent** method to create a new **LetterContent** object in the active document and then uses this object with the **RunLetterWizard** method.

```
Set myLetter = ActiveDocument.CreateLetterContent(DateFormat:="July 31,
1996", _
    IncludeHeaderFooter:=False, PageDesign:="", _
    LetterStyle:=wdFullBlock, Letterhead:=True,
LetterheadLocation:=wdLetterTop, _
    LetterheadSize:=InchesToPoints(1.5), RecipientName:="Dave Edson", _
    RecipientAddress:="436 SE Main St." & vbCr & "Bellevue, WA 98004", _
    Salutation:="Dear Dave,", SalutationType:=wdSalutationInformal, _
    RecipientReference:="", MailingInstructions:="", AttentionLine:="", _
    Subject:="End of year report", CCList:="", ReturnAddress:="", _
    SenderName:="", Closing:="Sincerely yours,", SenderCompany:="", _
    SenderJobTitle:="", SenderInitials:="", EnclosureNumber:=0)
ActiveDocument.RunLetterWizard LetterContent:=myLetter
```

IncludeHeaderFooter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludeHeaderFooterC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludeHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludeHeaderFooterA"}

True if the header and footer from the page design template are included in a letter created by the Letter Wizard. Read/write **Boolean**.

Note Use the **PageDesign** property to set the name of the template attached to a document created by the Letter Wizard.

IncludeHeaderFooter Property Example

This example creates a new **LetterContent** object, includes the header and footer from the Contemporary Letter template, and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .PageDesign = "C:\MSOffice\Templates\Letters & Faxes\Contemporary
Letter.dot"
    .IncludeHeaderFooter = True
End With
Documents.Add.RunLetterWizard LetterContent:=myContent
```

RecipientReference Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientReferenceC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientReferenceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientReferenceA"}

Returns or sets the reference line (for example, "In reply to:") for a letter created by the Letter Wizard. Read/write **String**.

RecipientReference Property Example

This example creates a new **LetterContent** object, sets several properties (including the reference line), and then runs the Letter Wizard by using the **RunLetterWizard** method.

```
Set myContent = New LetterContent
With myContent
    .RecipientReference = "In reply to:"
    .Salutation ="Hello"
    .MailingInstructions = "Certified Mail"
End With
Documents.Add.RunLetterWizard LetterContent:=myContent
```

InfoBlock Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInfoBlockC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInfoBlockX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInfoBlockA"}

Not used in the U.S. English version of Microsoft Word.

ReturnAddressShortForm Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnAddressShortFormC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnAddressShortFormX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnAddressShortFormA"}

Not used in the U.S. English version of Microsoft Word.

SenderReference Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderReferenceC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderReferenceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderReferenceA"}

Not used in the U.S. English version of Microsoft Word.

RecipientCode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientCodeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientCodeA"}

Not used in the U.S. English version of Microsoft Word.

SenderCode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderCodeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderCodeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderCodeA"}

Not used in the U.S. English version of Microsoft Word.

SenderCity Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderCityC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderCityX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderCityA"}

Not used in the U.S. English version of Microsoft Word.

RecipientGender Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientGenderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientGenderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientGenderA"}

Not used in the U.S. English version of Microsoft Word.

SenderGender Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderGenderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderGenderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderGenderA"}

Not used in the U.S. English version of Microsoft Word.

ApplyBulletDefault Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyBulletDefaultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyBulletDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyBulletDefaultA "}

Adds bullets and formatting to the paragraphs in the range for the specified **ListFormat** object. If the paragraphs are already formatted with bullets, this method removes the bullets and formatting.

Syntax

expression.**ApplyBulletDefault**

expression Required. An expression that returns a **ListFormat** object.

ApplyBulletDefault Method Example

This example adds bullets and formatting to the paragraphs in the selection. If there are already bullets in the selection, the example removes the bullets and formatting.

```
Selection.Range.ListFormat.ApplyBulletDefault
```

This example adds a bullet and formatting to, or removes them from, the second paragraph in MyDoc.doc.

```
Documents("MyDoc.doc").Paragraphs(2).Range.ListFormat.ApplyBulletDefault
```

This example sets the variable `myRange` to a range that includes paragraphs three through six of the active document, and then it checks to see whether the range contains list formatting. If there's no list formatting, default bullets are added.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(3).Range.Start, _
    End:=myDoc.Paragraphs(6).Range.End)
If myRange.ListFormat.ListType = wdListNoNumbering Then
    myRange.ListFormat.ApplyBulletDefault
End If
```

ApplyListTemplate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyListTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyListTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyListTemplateA "}

Applies a set of list-formatting characteristics to the specified **List** or **ListFormat** object.

Syntax 1

expression.**ApplyListTemplate(*ListTemplate, ContinuePreviousList*)**

Syntax 2

expression.**ApplyListTemplate(*ListTemplate, ContinuePreviousList, ApplyTo*)**

expression Syntax 1: Required. An expression that returns a **List** object.

Syntax 2: Required. An expression that returns a **ListFormat** object.

ListTemplate Required **ListTemplate** object. The list template to be applied.

ContinuePreviousList Optional **Variant**. **True** to continue the numbering from the previous list; **False** to start a new list.

ApplyTo Optional **Variant**. The portion of the list that the list template is to be applied to. Can be one of the following **WdListApplyTo** constants: **wdListApplyToSelection**, **wdListApplyToWholeList**, or **wdListApplyToThisPointForward**.

ApplyListTemplate Method Example

This example sets the variable `myRange` to a range in the active document, and then it checks to see whether the range has list formatting. If no list formatting has been applied, the fourth outline-numbered list template is applied to the range.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(3).Range.Start, _
    End:=myDoc.Paragraphs(6).Range.End)
If myRange.ListFormat.ListType = wdListNoNumbering Then
    myRange.ListFormat.ApplyListTemplate _

ListTemplate:=ListGalleries(wdOutlineNumberGallery).ListTemplates(4)
End If
```

This example sets the variable `myList` to the fourth list in MyDocument.doc, and then it applies the third bulleted list template to the list.

```
Set myList = Documents("MyDocument.doc").Lists(4)
myList.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdBulletGallery).ListTemplates(3)
```

This example sets the variable `myLstRange` to the list formatting in the second paragraph of MyDocument.doc. The example then applies the third numbered list template from that point forward in the list.

```
Set myLstRange = Documents("MyDocument.doc").Paragraphs(2).Range.ListFormat
myLstRange.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(3), _
    ApplyTo:=wdListApplyToThisPointForward
```

ApplyOutlineNumberDefault Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyOutlineNumberDefaultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyOutlineNumberDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyOutlineNumberDefaultA "}

Adds the default outline-numbering scheme to the paragraphs in the range for the specified **ListFormat** object. If the paragraphs are already formatted as an outline-numbered list, this method removes the numbers and formatting.

Syntax

expression.**ApplyOutlineNumberDefault**

expression Required. An expression that returns a **ListFormat** object.

Remarks

This method doesn't remove built-in heading styles that have been applied to paragraphs.

ApplyOutlineNumberDefault Method Example

This example adds outline numbering to the paragraphs in the selection. If the selection is already an outline-numbered list, the example removes the numbers and formatting.

```
Selection.Range.ListFormat.ApplyOutlineNumberDefault
```

This example sets the variable `myRange` to include paragraphs three through six of the active document, and then it checks to see whether the range contains list formatting. If there's no list formatting, the default outline-numbered list format is applied.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(3).Range.Start, _
    End:=myDoc.Paragraphs(6).Range.End)
If myRange.ListFormat.ListType = wdListNoNumbering Then
    myRange.ListFormat.ApplyOutlineNumberDefault
End If
```

ApplyNumberDefault Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyNumberDefaultC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyNumberDefaultX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyNumberDefaultA "}

Adds the default numbering scheme to the paragraphs in the range for the specified **ListFormat** object. If the paragraphs are already formatted as a numbered list, this method removes the numbers and formatting.

Syntax

expression.**ApplyNumberDefault**

expression Required. An expression that returns a **ListFormat** object.

ApplyNumberDefault Method Example

This example numbers the paragraphs in the selection. If the selection is already a numbered list, the example removes the numbers and formatting.

```
Selection.Range.ListFormat.ApplyNumberDefault
```

This example sets the variable `myRange` to include paragraphs three through six of the active document, and then it checks to see whether the range contains list formatting. If there's no list formatting, default numbers are applied to the range.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(3).Range.Start, _
    End:=myDoc.Paragraphs(6).Range.End)
If myRange.ListFormat.ListType = wdListNoNumbering Then
    myRange.ListFormat.ApplyNumberDefault
End If
```

Modified Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproModifiedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproModifiedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproModifiedA "}

True if the specified list template is not the built-in list template for that position in the list gallery. Read-only **Boolean**.

Syntax

expression.**Modified(*Index*)**

expression Required. An expression that returns a **ListGallery** object.

Index Required **Long**. A number from 1 to 7 that corresponds to the position of the template in the **Bullets and Numbering** dialog box. Skipping the **None** option, the templates are numbered from left to right, starting with the top row.

Remarks

Use the **Reset** method to set a list template in a list gallery back to the built-in list template.

Modified Property Example

This example checks to see whether the first template on the **Bulleted** tab in the **Bullets and Numbering** dialog box has been changed. If it has, the list template is reset.

```
temp = ListGalleries(wdBulletGallery).Modified(1)
If temp = True Then
    ListGalleries(wdBulletGallery).Reset(1)
Else
   Msgbox "This is the built-in list template."
End If
```

CountNumberedItems Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCountNumberedItemsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCountNumberedItemsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCountNumberedItemsA "}

Returns the number of bulleted or numbered items and LISTNUM fields in the specified **Document**, **List**, or **ListFormat** object.

Syntax

expression.**CountNumberedItems(*NumberType, Level*)**

expression Required. An expression that returns a **Document**, **List**, or **ListFormat** object.

NumberType Optional **Variant**. The type of numbers to be counted. Can be one of the following **WdNumberType** constants: **wdNumberParagraph**, **wdNumberListNum**, or **wdNumberAllNumbers**. The default value is **wdNumberAllNumbers**.

Level Optional **Variant**. A number that corresponds to the numbering level you want to count. If this argument is omitted, all levels are counted.

Remarks

Bulleted items are counted when either **wdNumberParagraph** or **wdNumberAllNumbers** (the default) is specified for ***NumberType***.

There are two types of numbers: preset numbers (**wdNumberParagraph**), which you can add to paragraphs by selecting a template in the **Bullets and Numbering** dialog box; and LISTNUM fields (**wdNumberListNum**), which allow you to add more than one number per paragraph.

CountNumberedItems Method Example

This example formats the current selection as a list, using the second numbered list template. The example then counts the numbered and bulleted items and LISTNUM fields in the active document and displays the result in a message box.

```
Selection.Range.ListFormat.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(2)
Msgbox ActiveDocument.CountNumberedItems
```

This example counts the number of first-level numbered or bulleted items in the active document.

```
Msgbox ActiveDocument.Content.ListFormat.CountNumberedItems(Level:=1)
```

This example displays a message box that reports the number of items in each list in MyLetter.

```
i = 1
Set myDoc = Documents("MyLetter.doc")
For Each li In myDoc.Lists
    Msgbox "List " & i & " has " & li.CountNumberedItems & " items."
    i = i + 1
Next li
```

This example counts the number of LISTNUM fields in the variable `myRange`. The result is displayed in a message box.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(12).Range.Start, _
    End:=myDoc.Paragraphs(50).Range.End)
numfields = myRange.ListFormat.CountNumberedItems(wdNumberListNum)
Msgbox numfields
```

LinkedStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLinkedStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLinkedStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLinkedStyleA "}

Returns or sets the name of the style that's linked to the specified **ListLevel** object. Read/write **String**.

LinkedStyle Property Example

This example sets the variable `myListTemp` to the first list template (excluding **None**) on the **Outline Numbered** tab in the **Bullets and Numbering** dialog box. Each level in the list has a matching heading style linked to it.

```
Set myListTemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(1)
For Each mylevel In myListTemp.ListLevels
    mylevel.LinkedStyle = "Heading " & mylevel.index
Next mylevel
```

List Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListA "}

Returns a **List** object that represents the first formatted list contained in the specified **ListFormat** object. Read-only.

Remarks

If the first paragraph in the range for the **ListFormat** object is not formatted as a list, the **List** property returns nothing.

List Property Example

This example returns the first list in the selection, and then it applies the first list template (excluding **None**) on the **Numbering** tab in the **Bullets and Numbering** dialog box. The selection can only contain one list.

```
Set mylist = Selection.Range.ListFormat.List
mylist.ApplyListTemplate ListTemplate:=ListGalleries(wdNumberGallery) _
    .ListTemplates(1)
```

ListIndent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthListIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthListIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthListIndentA "}

Increases the list level of the paragraphs in the range for the specified **ListFormat** object, in increments of one level.

Syntax

expression.**ListIndent**

expression Required. An expression that returns a **ListFormat** object.

ListIndent Method Example

This example indents each paragraph in the first list in document one by one level.

```
Documents(1).Lists(1).Range.ListFormat.ListIndent
```

This example formats paragraphs four through eight in the active document as an outline-numbered list, and then it indents the paragraphs one level.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(4).Range.Start, _
    End:=myDoc.Paragraphs(8).Range.End)
With myrange.ListFormat
    .ApplyOutlineNumberDefault
    .ListIndent
End With
```

ListGalleries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListGalleriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListGalleriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListGalleriesA "}

Returns a **ListGalleries** collection that represents the three list template galleries (**Bulleted**, **Numbered**, and **Outline Numbered**). Each gallery corresponds to a tab in the **Bullets and Numbering** dialog box. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ListGalleries Property Example

This example sets the variable `mylsttmp` to the second list template on the **Outline Numbered** tab in the **Bullets and Numbering** dialog box. The example then applies that template to the first list in the active document.

```
Set mylsttmp = ListGalleries(wdOutlineNumberGallery).ListTemplates(2)
ActiveDocument.Lists(1).ApplyListTemplate ListTemplate:=mylsttmp
```

This example cycles through the **ListGalleries** collection and changes the templates in each list template gallery back to the built-in template.

```
For Each listgal In ListGalleries
    For i = 1 To 7
        listgal.Reset(i)
    Next i
Next listgal
```

ListLevelNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListLevelNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListLevelNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListLevelNumberA "}

Style object: Returns the list level for the specified style. Read-only **Long**.

ListFormat object: Returns or sets the list level for the first paragraph in the specified **ListFormat** object. Read/write **Long**.

ListLevelNumber Property Example

This example returns the list level for the third paragraph in the active document.

```
lev = ActiveDocument.Paragraphs(3).Range.ListFormat.ListLevelNumber
```

This example displays the list level for the Heading 3 style.

```
Msgbox ActiveDocument.Styles(wdStyleHeading3).ListLevelNumber
```

ListLevels Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListLevelsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListLevelsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListLevelsA "}

Returns a **ListLevels** collection that represents all the levels for the specified **ListTemplate**. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ListLevels Property Example

This example sets the variable `myListTemp` to the first list template (excluding **None**) on the **Outline Numbered** tab in the **Bullets and Numbering** dialog box. Each level in the list has a matching heading style linked to it.

```
Set myListTemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(1)
For Each mylevel In myListTemp.ListLevels
    mylevel.LinkedStyle = "Heading " & mylevel.index
Next mylevel
```

ListOutdent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthListOutdentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthListOutdentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthListOutdentA "}

Decreases the list level of the paragraphs in the range for the specified **ListFormat** object, in increments of one level.

Syntax

expression.**ListOutdent**

expression Required. An expression that returns a **ListFormat** object.

ListOutdent Method Example

This example reduces the indent of each paragraph in first list in the active document by one level.

```
ActiveDocument.Lists(1).Range.ListFormat.ListOutdent
```

This example formats paragraphs four through eight in MyDoc.doc as an outline-numbered list, indents the paragraphs one level, and then removes the indent from the first paragraph in the list.

```
Set myDoc = Documents("MyDoc.doc")
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(4).Range.Start, _
    End:=myDoc.Paragraphs(8).Range.End)
With myrange.ListFormat
    .ApplyOutlineNumberDefault
    .ListIndent
End With
myDoc.Paragraphs(4).Range.ListFormat.ListOutdent
```

Lists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListsA "}

Returns a **Lists** collection that contains all the formatted lists in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Lists Property Example

This example formats the selection as a numbered list. The example then displays a message box that reports the number of lists in the active document.

```
Selection.Range.ListFormat.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(2)
MsgBox "This document has " & ActiveDocument.Lists.Count & " lists."
```

This example formats the third list in the active document with the default bulleted list format. If the list is already formatted with a bulleted list format, the example removes the formatting.

```
If ActiveDocument.Lists.Count >= 3 Then
    ActiveDocument.Lists(3).Range.ListFormat.ApplyBulletDefault
End If
```

This example displays a message box that reports the number of items in each list in MyLetter.doc.

```
Set myDoc = Documents("MyLetter.doc")
i = myDoc.Lists.Count
For each li in myDoc.Lists
    Msgbox "List " & i & " has " & li.CountNumberedItems & " items."
    i = i - 1
Next li
```

ListTemplate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListTemplateA "}

Returns a **ListTemplate** object that represents the list formatting for the specified **Style** or **ListFormat** object. Read-only.

Remarks

A list template includes all the formatting that defines a particular list. Each of the seven formats (excluding **None**) found on each of the tabs in the **Bullets and Numbering** dialog box corresponds to a list template. Documents and templates can also contain collections of list templates.

If the first paragraph in the range for the **ListFormat** object is not formatted as a list, the **ListTemplate** property returns **Nothing**.

ListTemplate Property Example

This example checks to see which list template is used for the second paragraph in the active document, and then it applies that list template to the selection.

```
Set myltemp = ActiveDocument.Paragraphs(2).Range. _
    ListFormat.ListTemplate
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=myltemp
```

ListTemplates Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListTemplatesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListTemplatesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListTemplatesA "}

Returns a **ListTemplates** collection that represents all the list formats for the specified document, template, or list gallery. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ListTemplates Property Example

This example sets the variable `mytemp` to the first list template on the **Outline Numbered** tab in the **Bullets and Numbering** dialog box. The template is modified to use lowercase letters for each level, and it's applied to the second list in the active document.

```
Set mytemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(1)
For each lev in mytemp.ListLevels
    lev.NumberStyle = wdListNumberStyleLowercaseLetter
Next lev
ActiveDocument.Lists(2).ApplyListTemplate ListTemplate:=mytemp
```

This example displays the number of list templates used in the active document.

```
Msgbox ActiveDocument.ListTemplates.Count
```

ListValue Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListValueC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListValueX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListValueA "}

Returns the numeric value of the first paragraph in the range for the specified **ListFormat** object. For example, the **ListValue** property applied to the second paragraph in an alphabetic list would return 2. Read-only **Long**.

Remarks

Use the **ListString** property to return a string that represents the appearance of the list value.

If the **ListFormat** object is in a bulleted list, **ListValue** returns 1.

If the **ListFormat** object is in an outline-numbered list, **ListValue** returns the numeric value of the first paragraph as it occurs in the sequence of paragraphs at the same level. For example, the **ListValue** property applied to a paragraph numbered "A.2" would return 2.

This property will not return the value for a LISTNUM field.

ListValue Property Example

This example displays both the numeric value of the first paragraph in the selection and the string representation of that value.

```
v = Selection.Range.ListFormat.ListValue
lstring = Selection.Range.ListFormat.ListString
MsgBox "List value " & v & " is represented by the string " & lstring
```

ListString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListStringA "}

Returns a string that represents the appearance of the list value of the first paragraph in the range for the specified **ListFormat** object. For example, the second paragraph in an alphabetic list would return B. Read-only **String**.

Remarks

For a bulleted list, you will need to apply the correct font in order to see the string. Most bullets use the Symbol or Wingdings font.

Use the **ListValue** property to return the numeric value of the paragraph.

ListString Property Example

This example displays both the numeric value of the first paragraph in the selection and the string representation of the list value.

```
v = Selection.Range.ListFormat.ListValue
lstring = Selection.Range.ListFormat.ListString
MsgBox "List value " & v & " is represented by the string " & lstring
```

OutlineNumbered Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutlineNumberedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutlineNumberedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutlineNumberedA "}

True if the specified **ListTemplate** object is outline numbered. Read/write **Boolean**.

Remarks

Setting this property to **False** converts the list template to a single-level list, using the formatting of the first level.

You cannot set this property for a **ListTemplate** object returned from a **ListGallery** object.

OutlineNumbered Property Example

This example changes the selected outline-numbered list to a single-level numbered list.

```
Selection.Range.ListFormat.ListTemplate.OutlineNumbered = False
```

This example checks to see whether the third list in MyDoc.doc is an outline-numbered list. If it is, the third outline-numbered list template is applied to it.

```
Set myltemp = Documents("MyDoc.doc").Lists(3).Range.ListFormat.ListTemplate
num = myltemp.OutlineNumbered
If num = True Then ActiveDocument.Lists(3).ApplyListTemplate _
    ListTemplate:=ListGalleries(wdOutlineNumberGallery).ListTemplates(3)
```

ListParagraphs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListParagraphsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListParagraphsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListParagraphsA "}

Returns a **ListParagraphs** collection that represents all the numbered paragraphs in the list, document, or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

ListParagraphs Property Example

This example adds a yellow background to each numbered or bulleted paragraph in the first document.

```
For Each numpar In Documents(1).ListParagraphs
    numpar.Shading.BackgroundPatternColorIndex = wdYellow
Next numpar
```

This example double underlines the paragraphs in the second list in the active document.

```
For Each mypara In ActiveDocument.Lists(2).ListParagraphs
    mypara.Range.Underline = wdUnderlineDouble
Next mypara
```

RemoveNumbers Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRemoveNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRemoveNumbersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRemoveNumbersA "}

Removes numbers or bullets from the specified **Document**, **List**, or **ListFormat** object.

Syntax

expression.**RemoveNumbers(*NumberType*)**

expression Required. An expression that returns a **Document**, **List**, or **ListFormat** object.

NumberType Optional **Variant**. The type of number to be removed. Can be one of the following **WdNumberType** constants: **wdNumberParagraph**, **wdNumberListNum**, or **wdNumberAllNumbers**. The default value is **wdNumberAllNumbers**.

Remarks

When this method is applied to a **List** object, it removes numbers only from paragraphs in the specified list, skipping over any interleaved numbers from other lists. If this method is applied to the **ListFormat** object for a range of text, all numbers from all lists in the range are removed.

RemoveNumbers Method Example

This example removes the bullets or numbers from any numbered paragraphs in the selection.

```
Selection.Range.ListFormat.RemoveNumbers
```

This example removes the numbers from the beginning of any numbered paragraphs in the active document.

```
ActiveDocument.RemoveNumbers wdNumberParagraph
```

This example removes the LISTNUM fields from the selection.

```
Selection.Range.ListFormat.RemoveNumbers wdNumberListNum
```

This example removes the bullets or numbers from the third list in MyDocument.doc.

```
If Documents("MyDocument.doc").Lists.Count >= 3 Then
    Documents("MyDocument.doc").Lists(3).RemoveNumbers
End If
```

ResetOnHigher Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproResetOnHigherC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproResetOnHigherX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproResetOnHigherA "}

True if the specified list level restarts numbering at 1 following a higher list level. **False** if the numbering continues sequentially each time the list level appears. Read/write **Boolean**.

Remarks

This feature allows lists to be interleaved, maintaining numeric sequence.

ResetOnHigher Property Example

This example sets each of the nine list levels in the first outline-numbered list template to continue its sequential numbering whenever that level is used.

```
For Each li In
ListGalleries(wdOutlineNumberGallery).ListTemplates(1).ListLevels
    li.ResetOnHigher = False
Next li
```

StartAt Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStartAtC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStartAtX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStartAtA "}

Returns or sets the starting number for the specified **ListLevel** object. Read/write **Long**.

StartAt Property Example

This example sets the number style and starting number for the third outline-numbered list template. Because the style uses uppercase letters and the starting number is 4, the first letter is D.

```
Set mylev = ListGalleries(wdOutlineNumberGallery) _
    .ListTemplates(3).ListLevels(1)
With mylev
    .NumberStyle = wdListNumberStyleUppercaseLetter
    .StartAt = 4
End With
```

TabPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTabPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTabPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTabPositionA "}

Returns or sets the tab position for the specified **ListLevel** object. Read/write **Single**.

TabPosition Property Example

This example sets each list level number so that it's indented 0.5 inch (36 points) from the previous level, and the tab is set at 0.25 inch (18 points) from the number.

```
r = 0
For Each lev In ListGalleries(wdOutlineNumberGallery) _
    .ListTemplates(1).ListLevels
        lev.Alignment = wdListLevelAlignLeft
        lev.NumberPosition = r
        lev.TrailingCharacter = wdTrailingTab
        lev.TabPosition = r + 18
        r = r + 36
Next lev
```

This example sets the variable `myltemp` to the first numbered list template, and then it sets the tab position at 1 inch. The list template is then applied to the selection.

```
Set myltemp = ListGalleries(wdNumberGallery).ListTemplates(1)
myltemp.ListLevels(1).TabPosition = InchesToPoints(1)
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=myltemp
```

TextPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextPositionA "}

Returns or sets the position for the second line of wrapping text for the specified **ListLevel** object. Read/write **Single**.

TextPosition Property Example

This example sets the indentation for all the levels of the first outline-numbered list template. Each list level number is indented 0.5 inch (36 points) from the previous level, the tab is set at 0.25 inch (18 points) from the number, and wrapping text is indented 0. 25 inch (18 points) from the number.

```
r = 0
For Each lev In ListGalleries(wdOutlineNumberGallery) _
        .ListTemplates(1).ListLevels
    lev.Alignment = wdListLevelAlignLeft
    lev.NumberPosition = r
    lev.TrailingCharacter = wdTrailingTab
    lev.TabPosition = r + 18
    lev.TextPosition = r + 18
    r = r + 36
Next lev
```

TrailingCharacter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTrailingCharacterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTrailingCharacterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTrailingCharacterA "}

Returns or sets the character inserted after the number for the specified list level. Can be one of the following **WdTrailingCharacter** constants: **wdTrailingNone**, **wdTrailingSpace**, or **wdTrailingTab**. Read/write **Long**.

TrailingCharacter Property Example

This example sets the number and text alignment for each level of the sixth outline-numbered list template. The number for each level is followed by a space.

```
r = 0
For Each lev In ListGalleries(wdOutlineNumberGallery) _
        .ListTemplates(6).ListLevels
    lev.Alignment = wdListLevelAlignLeft
    lev.NumberPosition = r
    lev.TextPosition = r
    lev.TrailingCharacter = wdTrailingSpace
    r = r + 18
Next lev
```

ListFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListFormatA "}

Returns a **ListFormat** object that represents all the list formatting characteristics of a range. Read-only.

ListFormat Property Example

This example sets the variable `myDoc` to a range that includes paragraphs three through six of the active document. The example then either applies the default outline-numbered list format to the range or removes it, depending on whether or not the format was already applied to the range.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:= myDoc.Paragraphs(3).Range.Start, _
    End:=myDoc.Paragraphs(6).Range.End)
myRange.ListFormat.ApplyOutlineNumberDefault
```

This example applies the second list template on the **Numbered** tab in the **Bullets and Numbering** dialog box to all the paragraphs in the selection.

```
Selection.Range.ListFormat.ApplyListTemplate _
    ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(2)
```

CanContinuePreviousList Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCanContinuePreviousListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCanContinuePreviousListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCanContinuePreviousListA "}

Returns a **WdContinue** constant (**wdContinueDisabled**, **wdResetList**, or **wdContinueList**) that indicates whether the formatting from the previous list can be continued.

Syntax

expression.**CanContinuePreviousList(*ListTemplate*)**

expression Required. An expression that returns a **List** or **ListFormat** object.

ListTemplate Required **ListTemplate** object. A list template that's been applied to previous paragraphs in the document.

Remarks

This method returns the state of the **Continue previous list** and **Restart numbering** options in the **Bullets and Numbering** dialog box for a specified list format. To change the settings of these options, set the ***ContinuePreviousList*** argument of the **ApplyListTemplate** method.

CanContinuePreviousList Method Example

This example checks to see whether numbering from a previous list is disabled. If it isn't disabled, the current list template is applied with numbering set to continue from the previous list. The selection must be within the second list, or this example creates an error.

```
Set myLF = Selection.Range.ListFormat
temp = myLF.CanContinuePreviousList(ListTemplate:=myLF.ListTemplate)
If temp <> wdContinueDisabled Then
    myLF.ApplyListTemplate ListTemplate:=myLF.ListTemplate,
ContinuePreviousList:=True
End If
```

ListType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproListTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproListTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproListTypeA"}

Returns the type of lists that are contained in the range for the specified **ListFormat** object. Can be one of the following **WdListType** constants: **wdListBullet**, **wdListListNumOnly**, **wdListMixedNumbering**, **wdListNoNumbering**, **wdListOutlineNumbering**, or **wdListSimpleNumbering**. Read-only **Long**.

Remarks

The constant **wdListListNumOnly** refers to LISTNUM fields, which are fields that can be added within the text of a paragraph.

ListType Property Example

This example checks to see if the first list in the active document is a simple numbered list. If it is, the fourth list template on the **Numbered** tab of the **Bullets and Numbering** dialog box is applied.

```
Set myList = ActiveDocument.Lists(1)
If myList.Range.ListFormat.ListType = wdListSimpleNumbering Then
    myList.ApplyListTemplate
ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(4)
End If
```

NumberFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberFormatA "}

Returns or sets the number format for the specified list level. Read/write **String**.

Remarks

The percent sign (%) followed by any number from 1 through 9 represents the number style from the respective list level. For example, if you wanted the format for the first level to be "Article I," "Article II," and so on, the string for the **NumberFormat** property would be "Article %1" and the **NumberStyle** property would be set to **wdListNumberStyleUpperCaseRoman**.

If the **NumberStyle** property is set to **wdListNumberStyleBullet**, the string for the **NumberFormat** property can only contain one character.

NumberFormat Property Example

This example creates a list template that indents each level and formats the level with an arabic numeral and a period. The new list template is then applied to the selection.

```
Set LT = ActiveDocument.ListTemplates.Add(OutlineNumbered:=True)
For x = 1 To 9
    With LT.ListLevels(x)
        .NumberStyle = wdListNumberStyleArabic
        .NumberPosition = InchesToPoints(0.25 * (x - 1))
        .TextPosition = InchesToPoints(0.25 * x)
        .NumberFormat = "%" & x & "."
    End With
Next x
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=LT
```

Apply Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthApplyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthApplyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthApplyA "}

AutoCorrectEntry object: Replaces a range with the value of the specified AutoCorrect entry.

Shape or **ShapeRange** object: Applies to the specified shape formatting that has been copied using the **PickUp** method.

Syntax 1

expression.**Apply(*Range*)**

Syntax 2

expression.**Apply**

expression Required. An expression that returns an **AutoCorrectEntry** object (Syntax 1) or a **Shape** or **ShapeRange** object (Syntax 2).

Range Required **Range** object. The **Range** that's replaced by the AutoCorrect entry.

Remarks

If formatting for the **Shape** or **ShapeRange** object has not previously been copied using the **PickUp** method, using the **Apply** method generates an error.

Apply Method Example

This example adds an AutoCorrect replacement entry, then applies the "sr" AutoCorrect entry to the selected text.

```
AutoCorrect.Entries.Add Name:= "sr", Value:= "Stella Richards"
AutoCorrect.Entries("sr").Apply Selection.Range
```

This example applies the "sr" AutoCorrect entry to the first word in the active document.

```
AutoCorrect.Entries("sr").Apply ActiveDocument.Words(1)
```

This example copies the formatting of shape one on the active document and applies the copied formatting to shape two on the same document.

```
With ActiveDocument
    .Shapes(1).PickUp
    .Shapes(2).Apply
End With
```

PrintPostScriptOverText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintPostScriptOverTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintPostScriptOverTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintPostScriptOverTextA "}

True if PRINT field instructions (such as PostScript commands) in a document are to be printed on top of text and graphics when a PostScript printer is used. Read/write **Boolean**.

Remarks

For Windows, this property controls whether postscript code is printed in a converted Word for the Macintosh document. If the document contains no PRINT fields, this property has no effect.

PrintPostScriptOverText Property Example

This example sets Word to print PRINT field instructions on top of text and graphics, and then it prints the active document.

```
ActiveDocument.PrintPostScriptOverText = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Print PostScript over text** option on the **Print** tab in the **Options** dialog box.

```
currSet = ActiveDocument.PrintPostScriptOverText
```

ClearAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthClearAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthClearAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthClearAllA "}

TabStops object: Clears all the custom tab stops from the specified paragraphs.

KeyBindings object: Clears all the customized key assignments and restores the original Word shortcut key assignments.

Dictionaries object: Unloads all of the custom dictionaries.

Syntax

expression.**ClearAll**

expression Required. An expression that returns a **TabStops**, **KeyBindings**, or **Dictionaries** object.

Remarks

To clear an individual tab stop, use the **Clear** method of the **TabStop** object. The **ClearAll** method doesn't clear the default tab stops. To manipulate the default tab stops, use the **DefaultTabStop** property for the document.

After applying the **ClearAll** method to the **KeyBindings** object, the keys assignments in the specified template or document are reset to the default settings. Use the **CustomizationContext** property to specify a document or template context prior to using the **ClearAll** method.

The **ClearAll** method when used on the **Dictionaries** object does not delete the custom dictionary files. After using this method, the number of custom dictionaries in the collection is 0 (zero).

ClearAll Method Example

This example clears all the custom tab stops in the active document.

```
ActiveDocument.Paragraphs.TabStops.ClearAll
```

This example clears the customized key assignments in the Normal template. The key assignments are reset to the default settings.

```
CustomizationContext = NormalTemplate
KeyBindings.ClearAll
```

This example unloads all of the custom dictionaries.

```
CustomDictionaries.ClearAll
```

ParagraphFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproParagraphFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproParagraphFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproParagraphFormatA "}

Returns or sets a **ParagraphFormat** object that represents the paragraph settings for the specified range, selection, find or replacement operation, or style. Read/write.

ParagraphFormat Property Example

This example sets the paragraph formatting for the current selection to be right-aligned.

```
Selection.ParagraphFormat.Alignment = wdAlignParagraphRight
```

This example sets paragraph formatting for a range that includes the entire contents of MyDoc.doc. Paragraphs in this document are double-spaced and have a custom tab stop at 0.25 inch.

```
Set myRange = Documents("MyDoc.doc").Content
With myRange.ParagraphFormat
    .Space2
    .TabStops.Add Position:=InchesToPoints(.25)
End With
```

This example modifies the Heading 2 style for the active document. Paragraphs formatted with this style are indented to the first tab stop and double-spaced.

```
With ActiveDocument.Styles(wdStyleHeading2).ParagraphFormat
    .TabIndent(1)
    .Space2
End With
```

This example finds all double-spaced paragraphs in the active document and replaces the formatting with 1.5-line spacing.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .ParagraphFormat.Space2
    .Replacement.ClearFormatting
    .Replacement.ParagraphFormat.Space15
    .Execute FindText:="", ReplaceWith:="", _
        Replace:=wdReplaceAll
End With
```

Style Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStyleA "}

Returns or sets the style for the specified object. To set this property, specify either the local name of the style, an integer or a **WdBuiltinStyle** constant (see "Remarks"), or an object that represents the style. Read/write **Variant**.

Remarks

When you return the style for a range that includes more than one style, only the first character or paragraph style is returned.

The **Style** property can be one of the following **WdBuiltinStyle** constants:

wdStyleBlockQuotation
wdStyleBodyText
wdStyleBodyText2
wdStyleBodyText3
wdStyleBodyTextFirstIndent
wdStyleBodyTextFirstIndent2
wdStyleBodyTextIndent
wdStyleBodyTextIndent2
wdStyleBodyTextIndent3
wdStyleCaption
wdStyleClosing
wdStyleCommentReference
wdStyleCommentText
wdStyleDate
wdStyleDefaultParagraphFont
wdStyleEmphasis
wdStyleEndnoteReference
wdStyleEndnoteText
wdStyleEnvelopeAddress
wdStyleEnvelopeReturn
wdStyleFooter
wdStyleFootnoteReference
wdStyleFootnoteText
wdStyleHeader
wdStyleHeading1
wdStyleHeading2
wdStyleHeading3
wdStyleHeading4
wdStyleHeading5
wdStyleHeading6
wdStyleHeading7
wdStyleHeading8
wdStyleHeading9
wdStyleHyperlink
wdStyleList
wdStyleList2
wdStyleList3
wdStyleList4
wdStyleList5
wdStyleListBullet
wdStyleListBullet2
wdStyleListBullet3
wdStyleListBullet4
wdStyleListBullet5
wdStyleListContinue
wdStyleListContinue2
wdStyleListContinue3
wdStyleListContinue4
wdStyleListContinue5
wdStyleListNumber
wdStyleListNumber2
wdStyleListNumber3
wdStyleListNumber4
wdStyleListNumber5
wdStyleMacroText
wdStyleMessageHeader
wdStyleNavPane
wdStyleNormal
wdStyleNormalIndent
wdStyleNoteHeading
wdStylePageNumber
wdStylePlainText
wdStyleSalutation
wdStyleSignature
wdStyleStrong
wdStyleSubtitle
wdStyleTableOfAuthorities
wdStyleTableOfFigures

wdStyleHyperlinkFollowed
wdStyleIndex1
wdStyleIndex2
wdStyleIndex3
wdStyleIndex4
wdStyleIndex5
wdStyleIndex6
wdStyleIndex7
wdStyleIndex8
wdStyleIndex9
wdStyleIndexHeading
wdStyleLineNumber
wdStyleTitle
wdStyleTOAHeading
wdStyleTOC1
wdStyleTOC2
wdStyleTOC3
wdStyleTOC4
wdStyleTOC5
wdStyleTOC6
wdStyleTOC7
wdStyleTOC8
wdStyleTOC9

Style Property Example

This example displays the style for each paragraph in the active document.

```
For Each para in ActiveDocument.Paragraphs
    MsgBox para.Style
Next para
```

This example displays the style for each character in the selection. Each element of the **Characters** collection is a **Range** object.

```
For each c in Selection.Characters
    MsgBox c.Style
Next c
```

This example sets alternating styles of Heading 3 and Normal for all the paragraphs in the active document.

```
For i = 1 To ActiveDocument.Paragraphs.Count
    If i Mod 2 = 0 Then
        ActiveDocument.Paragraphs(i).Style = wdStyleNormal
    Else: ActiveDocument.Paragraphs(i).Style = wdStyleHeading3
    End If
Next i
```

This example finds all instances of the Heading 1 style in the active document and replaces them with the Heading 2 style.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Style = wdStyleHeading1
    .Replacement.ClearFormatting
    .Replacement.Style = wdStyleHeading2
    .Execute FindText:="", ReplaceWith:="", _
        Replace:=wdReplaceAll, Format:=True
End With
```

InUse Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInUseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInUseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInUseA "}

True if the specified style is a built-in style that has been modified or applied in the document or a new style that has been created in the document. Read-only **Boolean**.

Remarks

This property doesn't indicate whether the style is currently applied to any text in the document. For instance, if text that's been formatted with a style is deleted, the **InUse** property of the style remains **True**. For built-in styles that have never been used in the document, this property returns **False**.

InUse Property Example

This example displays a message box that lists the names of all the styles that are currently being used in the active document.

```
Set mydoc = ActiveDocument
msg = "Styles in use:" & vbCr
For Each sty In mydoc.Styles
    If sty.InUse = True Then
        With mydoc.Content.find
            .ClearFormatting
            .Text = ""
            .Style = sty
            .Execute Format:=True
            If .Found = True Then
               msg = msg & sty & vbCr
            End If
        End With
    End If
Next sty
MsgBox msg
```

WordBasic Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWordBasicC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWordBasicX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWordBasicA "}

Returns an Automation object (Word.Basic) that includes methods for all the WordBasic statements and functions available in Word version 6.0 and Word for Windows 95. Read-only.

Remarks

In Word 97, when you open a Word version 6.0 or Word for Windows 95 template that contains WordBasic macros, the macros are automatically converted to Visual Basic modules. Each WordBasic statement and function in the macro is converted to the corresponding Word.Basic method.

For information about WordBasic statements and functions, see the "Microsoft Word Developer's Kit" or WordBasic Help in Word version 6.0 or Word for Windows 95.

For information about converting WordBasic to Visual Basic, see Visual Basic Equivalents for WordBasic Commands.

WordBasic Property Example

This example uses the Word.Basic object to create a new document and insert the available font names. Each font name is formatted in its corresponding font.

```
With WordBasic
    .FileNewDefault
    For aCount = 1 To .CountFonts()
        .Font .[Font$](aCount)
        .Insert .[Font$](aCount)
        .InsertPara
    Next
End With
```

Microsoft Word Developer's Kit

"Microsoft Word Developer's Kit", 3rd ed. (Microsoft Press, 1995), ISBN 1-55615-880-7. A comprehensive guide and reference to programming macros in WordBasic.

LinkToPrevious Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLinkToPreviousC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLinkToPreviousX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLinkToPreviousA "}

True if the specified header or footer is linked to the corresponding header or footer in the previous section. When a header or footer is linked, its contents are the same as in the previous header or footer. Read/write **Boolean**.

Remarks

Because the **LinkToPrevious** property is set to **True** unless you change it, you can add headers, footers, and page numbers to your entire document by working with the headers, footers, and page numbers in the first section. For instance, the following example adds page numbers to all sections of the active document.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).PageNumbers.Add
```

The **LinkToPrevious** property applies to each header or footer individually. For example, the **LinkToPrevious** property could be set to **True** for the even-numbered-page header but **False** for the even-numbered-page footer.

LinkToPrevious Property Example

The first part of this example creates a new document with two sections. The second part creates unique headers for even-numbered and odd-numbered pages in sections one and two in the new document.

```
Documents.Add
With Selection
    For j = 1 to 4
        .TypeParagraph
        .InsertBreak
        .TypeParagraph
    Next j
End With
With ActiveDocument
    .Paragraphs(5).Range.InsertBreak Type:=wdSectionBreakNextPage
    .PageSetup.OddAndEvenPagesHeaderFooter = True
End With
With ActiveDocument.Sections(2)
    With .Headers(wdHeaderFooterPrimary)
        .LinkToPrevious = False
        .Range.InsertBefore "Section 2 Odd Header"
    End With
    With .Headers(wdHeaderFooterEvenPages)
        .LinkToPrevious = False
        .Range.InsertBefore "Section 2 Even Header"
    End With
End With
With ActiveDocument.Sections(1)
    .Headers(wdHeaderFooterPrimary) _
        .Range.InsertBefore "Section 1 Odd Header"
    .Headers(wdHeaderFooterEvenPages) _
        .Range.InsertBefore "Section 1 Even Header"
End With
```

RestartNumberingAtSection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRestartNumberingAtSectionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRestartNumberingAtSectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRestartNumberingAtSectionA "}

True if page numbering starts at 1 again at the beginning of the specified section. Read/write **Boolean**.

Remarks

If set to **False**, the **RestartNumberingAtSection** property will override the **StartingNumber** property so that page numbering can continue from the previous section.

RestartNumberingAtSection Property Example

This example adds page numbers to the headers in the active document, and then it sets page numbering to start at 1 again at the beginning of each section.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary) _
    .PageNumbers.Add Pagenumberalignment:=wdAlignPageNumberCenter
For Each s In ActiveDocument.Sections
    With s.Headers(wdHeaderFooterPrimary).PageNumbers
        .RestartNumberingAtSection = True
        .StartingNumber = 1
    End With
Next s
```

Level Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLevelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLevelA "}

HeadingStyle object: Returns or sets the level for the heading style in a table of contents or table of figures. Read/write **Long**.

Subdocument object: Returns the heading level used to create the subdocument. Read-only **Long**.

Level Property Example

This example adds a table of contents at the insertion point in the active document, and then it changes the levels for the heading styles.

```
ActiveDocument.TablesOfContents.Add _
    Range:=Selection.Range, _
    RightAlignPageNumbers:=True, _
    UseHeadingStyles:=True, _
    UpperHeadingLevel:=1, _
    LowerHeadingLevel:=3, _
    IncludePageNumbers:=True, _
    TableID:=wdTOCFormal
With ActiveDocument.TablesOfContents(1).HeadingStyles
    .Add Style:="Title", Level:=1
    .Add Style:="SubTitle", Level:=2
    .Add Style:="List Bullet", Level:=3
End With
With ActiveDocument.TablesOfContents(1)
    .HeadingStyles(1).Level = 2
    .HeadingStyles(2).Level = 4
    .HeadingStyles(3).Level = 6
End With
```

This example looks through each subdocument in the active document and displays the subdocument's heading level.

```
i = 1
If ActiveDocument.Subdocuments.Count > = 1 Then
    For each s in ActiveDocument.Subdocuments
        MsgBox "The heading level for SubDoc " & i & " is " & s.Level
        i = i + 1
    Next s
Else
    MsgBox "There are no subdocuments defined."
End If
```

TextColumns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextColumnsA "}

Returns or sets a **TextColumns** collection that represents the set of text columns for the specified **PageSetup** object. Read/write.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Remarks

There will always be at least one text column in the collection. When you create new text columns, you're adding to a collection of one column.

TextColumns Property Example

This example creates four evenly-spaced text columns that are applied to section two in the active document.

```
With ActiveDocument.Sections(2).PageSetup.TextColumns
    .SetCount NumColumns:=3
    .Add EvenlySpaced:=True
End With
```

This example creates a document with two text columns. The first column is 1.5 inches wide and the second is 3 inches wide.

```
Set myDoc = Documents.Add
With myDoc.PageSetup.TextColumns
    .SetCount NumColumns:=1
    .Add Width:=InchesToPoints(3)
End With
With myDoc.PageSetup.TextColumns(1)
    .Width = InchesToPoints(1.5)
    .SpaceAfter = InchesToPoints(0.5)
End With
```

IgnoreInternetAndFileAddresses Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIgnoreInternetAndFileAddressesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIgnoreInternetAndFileAddressesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIgnoreInternetAndFileAddressesA "}

True if file name extensions, MS-DOS paths, e-mail addresses, server and share names (also known as UNC paths), and Internet addresses (also known as URLs) are ignored while checking spelling. Read/write **Boolean**.

Note Setting this property to **True** doesn't prevent Macintosh paths from being checked.

IgnoreInternetAndFileAddresses Property Example

This example sets Word to ignore file names and Internet addresses, and then it checks spelling in the active document.

```
Options.IgnoreInternetAndFileAddresses = True
ActiveDocument.CheckSpelling
```

This example returns the current status of the **Ignore Internet and file addresses** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.IgnoreInternetAndFileAddresses
```

NumberPosition Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberPositionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberPositionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberPositionA "}

Returns or sets the position (in points) of the number or bullet for the specified **ListLevel** object. Read/write **Single**.

Remarks

For each list level, you can set the position of the number or bullet, the position of the tab, and the position of the text that wraps.

NumberPosition Property Example

This example sets the indentation for all the levels of the third outline-numbered list template. Each list level is indented 0.25 inch (18 points) more than the preceding level.

```
r = 0
For Each lev In ListGalleries(wdOutlineNumberGallery) _
    .ListTemplates(3).ListLevels
       lev.Alignment = wdListLevelAlignLeft
       lev.NumberPosition = r
      r = r + 18
Next lev
```

This example sets the indent for the the first level of the last numbered list template to 0.5 inch.

```
With ListGalleries(wdNumberGallery).ListTemplates(7).ListLevels(1)
    .Alignment = wdListLevelAlignLeft
    .NumberPosition = InchesToPoints(0.5)
End With
```

SingleListTemplate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSingleListTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSingleListTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSingleListTemplateA "}

True if the entire **List** or **ListFormat** object uses the same list template. Read-only **Boolean**.

SingleListTemplate Property Example

This example checks to see whether the selection is formatted with a single list template. If so, the example applies the second numbered list template to the selection.

```
Set myList = Selection.Range.ListFormat
temp = myList.SingleListTemplate
If temp = True Then
    myList.ApplyListTemplate _
        ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(2)
End If
```

SingleList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSingleListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSingleListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSingleListA "}

True if the specified **ListFormat** object contains only one list. Read-only **Boolean**.

SingleList Property Example

This example checks the selection to see whether it only contains one list. If it does, the example applies the default numbered list template to the selection.

```
temp = Selection.Range.ListFormat.SingleList
If temp = True Then
  Selection.Range.ListFormat.ApplyNumberDefault
End If
```

BackgroundSavingStatus Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackgroundSavingStatusC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackgroundSavingStatusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackgroundSavingStatusA "}

Returns the number of files queued up to be saved in the background. Read-only **Long**.

BackgroundSavingStatus Property Example

This example displays in the status bar the number of documents currently being saved.

```
Options.BackgroundSave =True
Documents.Add
ActiveDocument.SaveAs
    While Application.BackgroundSavingStatus <> 0
        StatusBar = "Documents remaining to save: " & _
        Application.BackgroundSavingStatus
    DoEvents
Wend
```

UserControl Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUserControlC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUserControlX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUserControlA"}

True if the document or application was created or opened by the user. **False** if the document or application was created or opened programmatically with the **Open** method or the **CreateObject** or **GetObject** method. Read/write **Boolean** for the **Document** object; read-only **Boolean** for the **Application** object.

Remarks

If Word is visible to the user, this property will always return **True**.

UserControl Property Example

This example displays the status of the **UserControl** property for the active document.

```
If ActiveDocument.UserControl = True Then
    MsgBox "This document was created or opened by the user."
Else
    MsgBox "This document was created programmatically."
End If
```

ScreenRefresh Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthScreenRefreshC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthScreenRefreshX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthScreenRefreshA "}

Updates the display on the monitor with the current information in the video memory buffer. You can use this method after using the **ScreenUpdating** property to disable screen updates.

Syntax

expression.**ScreenRefresh**

expression Required. An expression that returns an **Application** object.

Remarks

ScreenRefresh turns on screen updating for just one instruction and then immediately turns it off. Subsequent instructions don't update the screen until screen updating is turned on again with the **ScreenUpdating** property.

On the Macintosh, document windows are hidden when screen updating is turned off. The **ScreenRefresh** method displays updated document windows for an instant, but then they're hidden again. Therefore, on the Macintosh, you may want to use the **ScreenUpdating** property instead of the **ScreenRefresh** method to turn screen updating on and off.

ScreenRefresh Method Example

This example turns off screen updating, opens Test.doc, inserts text, refreshes the screen, and then closes the document (with changes saved).

```
ScreenUpdating = False
Documents.Open FileName:="C:\DOCS\TEST.DOC"
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myRange.InsertBefore "new"
Application.ScreenRefresh
ActiveDocument.Close SaveChanges:=wdSaveChanges
ScreenUpdating = True
```

LinkToListTemplate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthLinkToListTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthLinkToListTemplateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthLinkToListTemplateA "}

Links the specified style to a list template so that the style's formatting can be applied to lists.

Syntax

expression.**LinkToListTemplate(*ListTemplate, ListLevelNumber*)**

expression Required. An expression that returns a **Style** object.

ListTemplate Required **ListTemplate** object. The list template that the style is to be linked to.

ListLevelNumber Optional **Variant**. An integer corresponding to the list level that the style is to be linked to. If this argument is omitted, then the level of the style is used.

LinkToListTemplate Method Example

This example creates a new list template and then links heading styles 1 through 9 to levels 1 through 9. The new list template is then applied to the document. Any paragraphs formatted as heading styles will assume the numbering from the list template.

```
Set LT = ActiveDocument.ListTemplates.Add(OutlineNumbered:=True)
For x = 1 To 9
    With LT.ListLevels(x)
        .NumberStyle = wdListNumberStyleArabic
        .NumberPosition = InchesToPoints(0.25 * (x - 1))
        .TextPosition = InchesToPoints(0.25 * x)
        .NumberFormat = "%" & x & "."
    End With
    With ActiveDocument.Styles("Heading " & x)
       .LinkToListTemplate ListTemplate:=LT
    End With
Next x
ActiveDocument.Content.ListFormat.ApplyListTemplate ListTemplate:=LT
```

ConvertNumbersToText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertNumbersToTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertNumbersToTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertNumbersToTextA "}

Changes the list numbers and LISTNUM fields in the specified **Document**, **List**, or **ListFormat** object to text.

Syntax

expression.**ConvertNumbersToText(*NumberType*)**

expression Required. An expression that returns a **Document**, **List**, or **ListFormat** object.

NumberType Optional **Variant**. The type of number to be converted. Can be one of the following **WdNumberType** constants: **wdNumberParagraph**, **wdNumberListNum**, or **wdNumberAllNumbers**. The default value is **wdNumberAllNumbers**.

Remarks

There are two types of numbers: preset numbers (**wdNumberParagraph**), which you can add to paragraphs by selecting a template in the **Bullets and Numbering** dialog box; and LISTNUM fields (**wdNumberListNum**), which allow you to add more than one number per paragraph.

The **ConvertNumbersToText** method is useful if you want to work with a document in another application and that application doesn't recognize list formatting or LISTNUM fields.

Note After you've converted list numbers to text, you can no longer manipulate them in a list.

ConvertNumbersToText Method Example

This example converts the list numbers and LISTNUM fields in the active document to text.

```
ActiveDocument.ConvertNumbersToText
```

This example converts the numbers in the first list to text. Lists are counted in reverse, from the bottom of the document up.

```
ActiveDocument.Lists(1).ConvertNumbersToText
```

This example converts the preset numbers in `myRange` to text without affecting any LISTNUM fields.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(12).Range.Start, _
    End:=myDoc.Paragraphs(20).Range.End)
myRange.ListFormat.ConvertNumbersToText wdNumberParagraph
```

NumberStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberStyleA "}

Returns or sets the number style for the specified object. Read/write **Long**.

For a **CaptionLabel** object, this property can be one of the following **WdCaptionNumberStyle** constants: **wdCaptionNumberStyleArabic**, **wdCaptionNumberStyleLowercaseLetter**, **wdCaptionNumberStyleLowercaseRoman**, **wdCaptionNumberStyleUppercaseLetter**, or **wdCaptionNumberStyleUppercaseRoman**.

For a **Footnotes** or **Endnotes** object, this property can be one of the following **WdNoteNumberStyle** constants: **wdNoteNumberStyleArabic**, **wdNoteNumberStyleLowercaseLetter**, **wdNoteNumberStyleLowercaseRoman**, **wdNoteNumberStyleSymbol**, **wdNoteNumberStyleUppercaseLetter**, or **wdNoteNumberStyleUppercaseRoman**.

For a **PageNumbers** object, this property can be one of the following **WdPageNumberStyle** constants: **wdPageNumberStyleArabic**, **wdPageNumberStyleLowercaseLetter**, **wdPageNumberStyleLowercaseRoman**, **wdPageNumberStyleUppercaseLetter**, or **wdPageNumberStyleUppercaseRoman**.

For a **ListLevel** object, this property can be one of the following **WdListNumberStyle** constants:

wdListNumberStyleArabic	**wdListNumberStyleLowercaseRoman**
wdListNumberStyleArabicLZ	**wdListNumberStyleNone**
wdListNumberStyleBullet	**wdListNumberStyleOrdinal**
wdListNumberStyleCardinalText	**wdListNumberStyleOrdinalText**
wdListNumberStyleLegal	**wdListNumberStyleUppercaseLetter**
wdListNumberStyleLegalLZ	**wdListNumberStyleUppercaseRoman**
wdListNumberStyleLowercaseLetter	

NumberStyle Property Example

This example creates an alternating number style for the third outline-numbered list template.

```
Set myTemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(3)
For i = 1 to 9
    If i Mod 2 = 0 Then
        myTemp.ListLevels(i).NumberStyle = wdListNumberStyleUppercaseRoman
    Else
        myTemp.ListLevels(i).NumberStyle = wdListNumberStyleLowercaseRoman
    End If
Next i
```

This example changes the number style to uppercase letters for every outline-numbered list in the active document.

```
For Each lt In ActiveDocument.ListTemplates
    For Each ll In lt.listlevels
       ll.NumberStyle = wdListNumberStyleUppercaseLetter
    Next ll
Next lt
```

This example inserts a Figure caption at the insertion point. The caption letter is formatted as an uppercase letter.

```
CaptionLabels(wdCaptionFigure).NumberStyle =
wdCaptionNumberStyleUppercaseLetter
Selection.Collapse Direction:=wdCollapseEnd
Selection.InsertCaption Label:="Figure"
```

This example sets the formatting for footnotes and endnotes in the active document.

```
With ActiveDocument
    .Footnotes.NumberStyle = wdNoteNumberStyleLowercaseRoman
    .Endnotes.NumberStyle = wdNoteNumberStyleUppercaseRoman
End With
```

This example formats the page numbers in the active document's footer as lowercase roman numerals.

```
For Each sec In ActiveDocument.Sections
    sec.Footers(wdHeaderFooterPrimary).PageNumbers _
        .NumberStyle = wdPageNumberStyleLowercaseRoman
Next sec
```

LanguageID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguageIDC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguageIDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguageIDA "}

Returns or sets the language for the specified object. Read/write **Long**.

Can be one of the following **WdLanguageID** constants:

wdAfrikaans
wdArabic
wdBasque
wdBelgianDutch
wdBelgianFrench
wdBrazilianPortuguese
wdBulgarian
wdByelorussian
wdCatalan
wdCroatian
wdCzech
wdDanish
wdDutch
wdEnglishAUS
wdEnglishCanadian
wdEnglishNewZealand
wdEnglishSouthAfrica
wdEnglishUK
wdEnglishUS
wdEstonian
wdFarsi
wdFinnish
wdFrench
wdFrenchCanadian
wdGerman
wdGreek
wdHebrew
wdHungarian
wdItalian
wdIcelandic
wdJapanese
wdKorean
wdLanguageNone
wdLatvian
wdMacedonian
wdMalaysian
wdMexicanSpanish
wdNoProofing
wdNorwegianBokmol
wdNorwegianNynorsk
wdPolish
wdPortuguese
wdRomanian
wdRussian
wdSerbianCyrillic
wdSerbianLatin
wdSesotho
wdSimplifiedChinese
wdSlovak
wdSlovenian
wdSpanish
wdSpanishModernSort
wdSwedish
wdSwissFrench
wdSwissGerman
wdSwissItalian
wdTraditionalChinese
wdTsonga
wdTswana
wdTurkish
wdUkrainian
wdVenda
wdXhosa
wdZulu

Remarks

For a custom dictionary, you must first set the **LanguageSpecific** property to **True**, before specifying the **LanguageID**. Custom dictionaries that are language specific only look at text formatted for that language.

LanguageID Property Example

This example formats the second paragraph in the active document as French, then adds a new custom dictionary, that will be used on the French text.

```
ActiveDocument.Paragraphs(2).Range.LanguageId = wdFrench
Set myDictionary = CustomDictionaries.Add(Filename:="French.dic")
With myDictionary
    .LanguageSpecific = True
    .LanguageId = wdFrench
End With
```

This example redefines the Title style to use the Spanish proofing tools. The new style description is then displayed in a message box.

```
ActiveDocument.Styles("Title").LanguageId = wdSpanish
MsgBox ActiveDocument.Styles("Title").Description
```

Dialogs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDialogsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDialogsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDialogsA "}

Returns a **Dialogs** collection that represents all the built-in dialog boxes in Word. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Dialogs Property Example

This example displays the built-in **Find** dialog box, with "Hello" in the **Find What** box.

```
Set dlg = Dialogs(wdDialogEditFind)
With dlg
    .Find = "Hello"
    .Show
End With
```

This example displays the built-in **Open** dialog box showing all file types.

```
With Dialogs(wdDialogFileOpen)
    .Name = "*.*"
    .Show
End With
```

This example prints the active document, using the settings from the **Print** dialog box.

```
Dialogs(wdDialogFilePrint).Execute
```

Help Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthHelpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthHelpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthHelpA "}

Displays on-line Help information.

Syntax

expression.**Help(*HelpType*)**

expression An expression that returns a **Application** object.

HelpType Required **Variant**. The on-line Help topic or window. Can be one of the following **WdHelpType** constants.

Constant	Description
wdHelp	Displays the Help Topics dialog box.
wdHelpAbout	Displays the About Microsoft Word dialog box (Help menu).
wdHelpActiveWindow	Displays Help describing the command associated with the active view or pane.
wdHelpContents	Displays the Help Topics dialog box.
wdHelpIndex	Displays the Help Topics dialog box.
wdHelpPSSHelp	Displays product support information.
wdHelpSearch	Displays the Help Topics dialog box.
wdHelpUsingHelp	Displays a list of Help topics that describe how to use Help.

Help Method Example

This example displays the Help Topics dialog box.

```
Help HelpType:=wdHelp
```

This example displays a list of Help topics that describe how to use Help.

```
Help HelpType:=wdHelpUsingHelp
```

PrintOut Method (Application, Document, and Window Objects)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPrintOutApplicationObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPrintOutApplicationObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPrintOutApplicationObjA "}

Prints all or part of the specified document. The arguments for this method correspond to the options in the **Print** dialog box (**File** menu).

Syntax

expression.**PrintOut(*Background, Append, Range, OutputFileName, From, To, Item, Copies, Pages, PageType, PrintToFile, Collate, FileName, ActivePrinterMacGX, ManualDuplexPrint*)**

expression Required. An expression that returns an **Application**, **Document**, or **Window** object.

Background Optional **Variant**. **True** to have the macro continue while Word prints the document.

Append Optional **Variant**. **True** to append the specified document to the file name specified by the ***OutputFileName*** argument. **False** to overwrite the contents of ***OutputFileName***.

Range Optional **Variant**. The page range. Can be one of the following **WdPrintOutRange** constants: **wdPrintAllDocument**, **wdPrintCurrentPage**, **wdPrintFromTo**, **wdPrintRangeOfPages**, or **wdPrintSelection**.

OutputFileName Optional **Variant**. If ***PrintToFile*** is **True**, this argument specifies the path and file name of the output file. This argument isn't available on the Macintosh unless QuickDraw GX is installed.

From Optional **Variant**. The starting page number when ***Range*** is set to **wdPrintFromTo**.

To Optional **Variant**. The ending page number when ***Range*** is set to **wdPrintFromTo**.

Item Optional **Variant**. The item to be printed. Can be one of the following **WdPrintOutItem** constants: **wdPrintAutoTextEntries**, **wdPrintComments**, **wdPrintDocumentContent**, **wdPrintKeyAssignments**, **wdPrintProperties**, or **wdPrintStyles**.

Copies Optional **Variant**. The number of copies to be printed.

Pages Optional **Variant**. The page numbers and page ranges to be printed, separated by commas. For example, "2, 6-10" prints page 2 and pages 6 through 10.

PageType Optional **Variant**. The type of pages to be printed. Can be one of the following **WdPrintOutPages** constants: **wdPrintAllPages**, **wdPrintEvenPagesOnly**, or **wdPrintOddPagesOnly**.

PrintToFile Optional **Variant**. **True** to send printer instructions to a file. Make sure to specify a file name with ***OutputFileName***. This argument isn't available on the Macintosh unless QuickDraw GX is installed.

Collate Optional **Variant**. When printing multiple copies of a document, **True** to print all pages of the document before printing the next copy.

FileName Optional **Variant**. The path and file name of the document to be printed. If this argument is omitted, Word prints the active document. Available only with the **Application** object.

ActivePrinterMacGX Optional **Variant**. On the Macintosh, if QuickDraw GX is installed, specifies the printer to print to.

ManualDuplexPrint Optional **Variant**. Not used in the U.S. English version of Microsoft Word.

PrintOut Method (Application, Document, and Window Objects) Example

This example prints the current page of the active document.

```
ActiveDocument.PrintOut Range:=wdPrintCurrentPage
```

This example prints all the documents in the current folder. The **Dir** function is used to return all file names that have the file name extension ".doc".

```
adoc = Dir("*.DOC")
While adoc <> ""
    Application.PrintOut FileName:=adoc
    adoc = Dir()
Wend
```

This example prints the first three pages of the document in the active window.

```
ActiveWindow.PrintOut Range:=wdPrintFromTo, From:="1", To:="3"
```

This example prints the comments in the active document.

```
If ActiveDocument.Comments.Count >= 1 Then
    ActiveDocument.PrintOut Item:=wdPrintComments
End If
```

PrintOut Method (Envelope Object)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPrintOutEnvelopeObjC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPrintOutEnvelopeObjX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPrintOutEnvelopeObjA "}

Prints an envelope without adding it to the active document.

Syntax

expression.**PrintOut(*ExtractAddress, Address, AutoText*, *OmitReturnAddress, ReturnAddress, ReturnAutoText, PrintBarCode, PrintFIMA, Size, Height, Width, FeedSource, AddressFromLeft, AddressFromTop, ReturnAddressFromLeft, ReturnAddressFromTop, DefaultFaceUp, DefaultOrientation***)**

expression Required. An expression that returns an **Envelope** object.

ExtractAddress Optional **Variant**. **True** to use the text marked by the "EnvelopeAddress" bookmark (a user-defined bookmark) as the recipient's address.

Address Optional **Variant**. A string that specifies the recipient's address (ignored if ***ExtractAddress*** is **True**).

AutoText Optional **Variant**. The name of the AutoText entry that includes a recipient's address.

OmitReturnAddress Optional **Variant**. **True** to omit the return address.

ReturnAddress Optional **Variant**. A string that specifies the return address.

ReturnAutoText Optional **Variant**. The name of the AutoText entry that includes a return address.

PrintBarCode Optional **Variant**. **True** to add a POSTNET bar code. For U.S. mail only.

PrintFIMA Optional **Variant**. **True** to add a Facing Identification Mark (FIM-A) for use in presorting courtesy reply mail. For U.S. mail only.

Size Optional **Variant**. A string that specifies the envelope size. The string should match one of the sizes listed on the left side of the Envelope size box in the **Envelope Options** dialog box (for example, "Size 10").

Height Optional **Variant**. The height of the envelope (in points) when the ***Size*** argument is set to "Custom size."

Width Optional **Variant**. The width of the envelope (in points) when the ***Size*** argument is set to "Custom size."

FeedSource Optional **Variant**. The paper tray to be used when the envelope is printed. Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed	**wdPrinterManualFeed**
wdPrinterDefaultBin	**wdPrinterMiddleBin**
wdPrinterEnvelopeFeed	**wdPrinterOnlyBin**
wdPrinterFormSource	**wdPrinterPaperCassette**
wdPrinterLargeCapacityBin	**wdPrinterSmallFormatBin**
wdPrinterLargeFormatBin	**wdPrinterTractorFeed**
wdPrinterLowerBin	**wdPrinterUpperBin**
wdPrinterManualEnvelopeFeed	

AddressFromLeft Optional **Variant**. The distance (in points) between the left edge of the envelope and the recipient's address.

AddressFromTop Optional **Variant**. The distance (in points) between the top edge of the envelope and the recipient's address.

ReturnAddressFromLeft Optional **Variant**. The distance (in points) between the left edge of the envelope and the return address.

ReturnAddressFromTop Optional **Variant**. The distance (in points) between the top edge of the

envelope and the return address.

DefaultFaceUp Optional **Variant**. **True** to print the envelope face up, **False** to print it face down.

DefaultOrientation Optional **Variant**. The orientation of the envelope. Can be one of the following **WdEnvelopeOrientation** constants: **wdLeftPortrait**, **wdCenterPortrait**, **wdRightPortrait**, **wdLeftLandscape**, **wdCenterLandscape**, **wdRightLandscape**, **wdLeftClockwise**, **wdCenterClockwise**, or **wdRightClockwise**.

PrintOut Method (Envelope Object) Example

This example prints an envelope using the user address as the return address and a predefined recipient address.

```
recep = "Don Funk" & vbCr & "123 Skye St." & vbCr & _
    "OurTown, WA 98107"
ActiveDocument.Envelope.PrintOut Address:=recep, _
    ReturnAddress:=Application.UserAddress, _
    Size:="Size 10", PrintBarCode:=True
```

Next Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextA "}

Syntax 1: For the **Browser** object, moves the selection to the next item indicated by the browser target. Use the **Target** property to change the browser target.

Syntax 2: Returns the next paragraph as a **Paragraph** object.

Syntax 3: Returns a **Range** object relative to the specified selection or range.

Note If the **Range** or **Selection** is just before the specified ***Unit*** the **Range** or **Selection** is moved to the following unit. For example, if the **Selection** is just before a word, the following instruction moves the **Selection** forward to the following word.

```
Selection.Next(Unit:=wdWord, Count:=1).Select
```

Syntax 1

expression.**Next**

Syntax 2

expression.**Next**(***Count***)

Syntax 3

expression.**Next**(***Unit, Count***)

expression Syntax 1: An expression that returns a **Browser** object.

Syntax 2: An expression that returns a **Paragraph** object.

Syntax 3: An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, **wdLine** can also be used. The default value is **wdCharacter**.

Count Optional **Long**. The number of paragraphs (Syntax 2) or units (Syntax 3) by which you want to move ahead. The default value is 1.

Next Method Example

This example moves the insertion point just before the next comment reference marker in the active document.

```
With Application.Browser
    .Target = wdBrowseComment
    .Next
End With
```

This example inserts a number and a tab before the first nine paragraphs in the active document.

```
For n = 0 To 8
    Set myRange = ActiveDocument.Paragraphs(1).Next(Count:=n).Range
    myRange.Collapse Direction:=wdCollapseStart
    myRange.InsertAfter n + 1 & vbTab
Next n
```

This example selects the paragraph following the current selection.

```
Selection.Next(Unit:=wdParagraph, Count:=1).Select
```

Previous Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPreviousC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPreviousX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPreviousA "}

Syntax 1: For the **Browser** object, moves the selection to the previous item indicated by the browser target. Use the **Target** property to change the browser target.

Syntax 2: Returns the previous paragraph as a **Paragraph** object.

Syntax 3: Returns a **Range** object relative to the specified selection or range.

Note If the **Range** or **Selection** is just after the specified ***Unit*** the **Range** or **Selection** is moved to the previous unit. For example, if the **Selection** is just after a word (before the trailing space), the following instruction moves the **Selection** backwards to the previous word.

```
Selection.Previous(Unit:=wdWord, Count:=1).Select
```

Syntax 1

expression.**Previous**

Syntax 2

expression.**Previous**(***Count***)

Syntax 3

expression.**Previous**(***Unit, Count***)

expression Syntax 1: Required. An expression that returns a **Browser** object.

Syntax 2: Required. An expression that returns a **Paragraph** object.

Syntax 3: Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, **wdLine** can also be used. The default value is **wdCharacter**.

Count Optional **Variant**. The number of paragraphs (Syntax 2) or units (Syntax 3) by which you want to move back. The default value is 1.

Previous Method Example

This example moves the insertion point into the first cell (the cell in the upper-left corner) of the previous table.

```
With Application.Browser
    .Target = wdBrowseTable
    .Previous
End With
```

This example selects the paragraph that precedes the selection in the active document.

```
Selection.Previous(Unit:=wdParagraph, Count:=1).Select
```

This example applies bold formatting to the first word in the active document.

```
ActiveDocument.Words(2).Previous(Unit:=wdWord, Count:=1).Bold = True
```

DefaultSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultSizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultSizeA "}

Returns or sets the default envelope size. Read/write **String**.

Note The string that's returned corresponds to the right-hand side of the string that appears in the **Envelope Size** box in the **Envelope Options** dialog box. If you set either the **DefaultHeight** or **DefaultWidth** property, the envelope size is automatically changed to Custom Size in the **Envelope Options** dialog box (**Tools** menu) and this property returns "Custom size."

DefaultSize Property Example

This example sets the default envelope size to C4 (229 x 324 mm).

```
ActiveDocument.Envelope.DefaultSize = "C4"
```

This example asks the user whether or not they want to change the default envelope size to Size 10. If the answer is yes, the default size is changed accordingly. The **UpdateDocument** method changes the envelope size for the active document. If an envelope has not been added to the active document, a message box is displayed.

```
On Error GoTo errhandler
response = MsgBox("Do you want to set the default envelope to Size 10?", 4)
If response = vbYes Then
    With ActiveDocument.Envelope
        .DefaultSize = "Size 10"
        .UpdateDocument
    End With
End If
errhandler:
If Err = 5852 Then MsgBox "An envelope isn't part of this document"
```

Type Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTypeA "}

Returns or sets the type for the specified object, as described in the following table.

Object	Description
CalloutFormat	Returns or sets the callout type. Can be one of the **MsoCalloutType** constants: **msoCalloutFour**, **msoCalloutMixed**, **msoCalloutOne**, **msoCalloutThree**, or **msoCalloutTwo**. Read/write **Long**.
ColorFormat	Returns or sets the shape color type. Can be one of the following **MsoColorType** constants: **msoColorTypeMixed**, **msoColorTypeRGB**, or **msoColorTypeScheme**. Read-only **Long**.
Dialog	Returns the type of built-in Word dialog box. Can be one of the **WdWordDialog** constants. Read-only **Long**.
Dictionary	Returns the dictionary type. Can be one of the following **WdDictionaryType** constants: **wdGrammar**, **wdHyphenation**, **wdSpelling**, **wdSpellingComplete**, **wdSpellingCustom**, **wdSpellingLegal**, **wdSpellingMedical**, or **wdThesaurus**. Read-only **Long**.
Document	Returns the document type (template or document). Can be either of the following **WdDocumentType** constants: **wdTypeDocument** or **wdTypeTemplate**. Read-only **Long**.
Field, **FormField**, **MailMergeField**	Returns the field type. Can be one of the **WdFieldType** constants. Read-only **Long**.
FillFormat	Returns the shape fill format type. Can be one of the following **MsoFillType** constants: **msoFillBackground**, **msoFillGradient**, **msoFillMixed**, **msoFillPatterned**, **msoFillPicture**, **msoFillSolid** or **msoFillTextured**. Read-only **Long**.
Hyperlink	Returns the hyperlink type. Can be one of the following **MsoHyperlinkType** constants: **msoHyperlinkInlineShape**, **msoHyperLinkRange**, or **msoHyperlinkShape**. Read-only **Long**.
Index	Returns or sets the index type. Can be either of the following **WdIndexType** constants: **wdIndexIndent** or **wdIndexRunin**. Read/write **Long**.
InlineShape	Returns the type of inline shape. Can be one of the following **WdInlineShapeType** constants: **wdInlineShapeEmbeddedOLEObject**, **wdInlineShapeLinkedOLEObject**, **wdInlineShapeLinkedPicture**, **wdInlineShapeOLEControlObject**, or **wdInlineShapePicture**. Read-only **Long**.

LinkFormat	Returns the link type. Can be one of the following **WdLinkType** constants: **wdLinkTypeDDE**, **wdLinkTypeDDEAuto**, **wdLinkTypeImport**, **wdLinkTypeInclude**, **wdLinkTypeOLE**, **wdLinkTypePicture**, **wdLinkTypeReference**, or **wdLinkTypeText**. Read-only **Long**.
MailMergeDataSource	Returns the type of mail merge data source. Can be one of the following **WdMailMergeDataSource** constants: **wdMergeInfoFromAccessDDE**, **wdMergeInfoFromExcelDDE**, **wdMergeInfoFromMSQueryDDE**, **wdMergeInfoFromODBC**, **wdMergeInfoFromWord**, or **wdNoMergeInfo**. Read-only **Long**.
ProofreadingErrors	Returns the type of proofreading error. Can be either of the following **WdProofreadingErrorType** constants: **wdGrammaticalError** or **wdSpellingError**. Read-only **Long**.
Revision	Returns the revision type. Can be one of the following **WdRevisionType** constants: **wdNoRevision**, **wdRevisionDelete**, **wdRevisionDisplayField**, **wdRevisionInsert**, **wdRevisionParagraphNumber**, **wdRevisionProperty**, **wdRevisionReconcile**, **wdRevisionReplace**, or **wdRevisionStyle**. Read-only **Long**.
Selection	Returns the selection type. Can be one of the following **WdSelectionType** constants: **wdNoSelection**, **wdSelectionBlock**, **wdSelectionColumn**, **wdSelectionFrame**, **wdSelectionInlineShape**, **wdSelectionIP**, **wdSelectionNormal**, **wdSelectionRow**, or **wdSelectionShape**. Read-only **Long**.
ShadowFormat	Returns or sets the shape shadow type. Can be one of the **MsoShadowType** constants. Read/write **Long**.
Shape, **ShapeRange**	Returns the shape type. Can be one of the following **MsoShapeType** constants: **msoAutoShape**, **msoCallout**, **msoChart**, **msoComment**, **msoEmbeddedOLEObject**, **msoFormControl**, **msoFreeform**, **msoGroup**, **msoLine**, **msoLinkedOLEObject**, **msoLinkedPicture**, **msoMedia**, **msoOLEControlObject**, **msoPicture**, **msoPlaceholder**, **msoShapeTypeMixed**, **msoTextBox**, or **msoTextEffect**. Read-only **Long**.
Style	Returns the style type. Can be either of the following **WdStyleType** constants: **wdStyleTypeParagraph** or **wdStyleTypeCharacter**. Read-only **Long**.
Template	Returns the template type. Can be one of the following **WdTemplateType** constants: **wdAttachedTemplate**, **wdGlobalTemplate**, or

	wdNormalTemplate. Read-only **Long**.
TextInput	Returns the type of text form field. Can be one of the following **WdTextFormFieldType** constants: **wdCalculationText**, **wdCurrentDateText**, **wdCurrentTimeText**, **wdDateText**, **wdNumberText**, or **wdRegularText**. Read-only **Long**.
View	Returns or sets the view type. Can be one of the following **WdViewType** constants: **wdMasterView**, **wdNormalView**, **wdOnlineView**, **wdOutlineView**, **wdPageView**, or **wdPrintPreview**. Read/write **Long**.
Window	Returns the window type. Can be either of the following **WdWindowType** constants: **wdWindowDocument** or **wdWindowTemplate**. Read-only **Long**.
WrapFormat	Returns the wrap type for the specified shape. Can be one of the following **WdWrapType** constants: **wdWrapNone**, **wdWrapSquare**, **wdWrapThrough**, **wdWrapTight**, or **wdWrapTopBottom**. Read/write **Long**.

Type Property Example

If the active window is a document, this example redefines the Heading 1 style as centered.

```
If ActiveWindow.Type = wdWindowDocument Then
    ActiveDocument.Styles("Heading 1").ParagraphFormat.Alignment =
wdAlignParagraphCenter
End If
```

This example switches the active window to print preview. The **Type** property creates a new print preview window.

```
ActiveWindow.View.Type = wdPrintPreview
```

This example displays a message that indicates the style type of the style named "SubTitle" in the active document.

```
If ActiveDocument.Styles("SubTitle").Type = wdStyleTypeParagraph Then
    MsgBox "Paragraph style"
ElseIf ActiveDocument.Styles("SubTitle").Type = wdStyleTypeCharacter Then
    MsgBox "Character style"
End If
```

This example accepts the next revision in the active document if the revision type is inserted text.

```
Set myRev = Selection.NextRevision
If Not (myRev Is Nothing) Then
    If myRev.Type = wdRevisionInsert Then myRev.Accept
End If
```

This example formats the selection as engraved if the selection isn't an insertion point.

```
If Selection.Type <> wdSelectionIP Then
    Selection.Font.Engrave = True
Else
    MsgBox "You need to select some text."
End If
```

ActivePrinter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActivePrinterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActivePrinterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActivePrinterA "}

Returns or sets the name of the active printer. Read/write **String**.

ActivePrinter Property Example

This example displays the name of the active printer.

```
MsgBox "The name of the active printer is " & ActivePrinter
```

This example makes a network HP LaserJet IIISi printer the active printer.

```
ActivePrinter = "HP LaserJet IIISi on \\printers\laser"
```

This example makes a local HP LaserJet 4 printer on LPT1 the active printer.

```
ActivePrinter = "HP LaserJet 4 local on LPT1:"
```

ComputerType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproComputerTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproComputerTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproComputerTypeA"}

Returns the model of Macintosh computer being used (for example, "Power Macintosh 6100"). Read-only **String**.

Note In Windows, **ComputerType** isn't available and generates an error.

ComputerType Property Example

This example displays the model of computer being used.

```
MsgBox "This computer is a " & System.ComputerType
```

ReadOnly Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReadOnlyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReadOnlyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReadOnlyA "}

Document or **RecentFile** object: **True** if changes to the document cannot be saved to the original document. Read/write **Boolean** for the **RecentFile** object; read-only **Boolean** for the **Document** object.

Dictionary object: **True** if the specified dictionary cannot be changed. Read-only **Boolean**.

Note The active grammar, hyphenation, spelling, and thesaurus dictionaries are read-only. Custom dictionaries are read/write.

ReadOnly Property Example

This example saves the active document if it isn't read-only.

```
If ActiveDocument.ReadOnly = False Then ActiveDocument.Save
```

This example opens the most recently used file as a read-only document.

```
With RecentFiles(1)
    .ReadOnly = True
    .Open
End With
```

Update Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateA "}

Dialog object: Updates the values shown in a built-in Microsoft Word dialog box.

Field or **Fields** object: Updates the result of the specified object. When applied to a **Field** object, returns **True** if the field is updated successfully. When applied to a **Fields** collection, returns 0 (zero) if no errors occur when the fields are updated, or returns the index of the first field that contains an error.

Index, **TableOfAuthorities**, **TableOfContents**, or **TableOfFigures** object: Updates the entries shown in specified index, table of authorities, table of figures or table of contents.

LinkFormat object: Updates the specified link.

Note Use the **UpdatePageNumbers** method to update the page numbers of items in a table of contents or figures.

Syntax

expression.**Update**

expression An expression that returns an object in the Applies To list.

Update Method Example

This example updates all the fields in the active document. A return value of 0 (zero) indicates that the fields were updated without error.

```
If ActiveDocument.Fields.Update = 0 Then
    MsgBox "Update Successful"
Else
    MsgBox "Field " & ActiveDocument.Fields.Update & " has an error"
End If
```

This example updates the first table of figures in the active document.

```
If ActiveDocument.TablesOfFigures.Count >= 1 Then
    ActiveDocument.TableOfFigures(1).Update
End If
```

This example updates the first field in the active document and displays a message in the status bar indicating whether or not the field was updated successfully.

```
If ActiveDocument.Fields(1).Update = True Then
    StatusBar = "Field updated"
Else
    StatusBar = "Error, field not updated"
End If
```

This example returns a **Dialog** object that refers to the Font dialog box. The font applied to the Selection object is changed to Arial, the dialog values are updated, and the Font dialog box is displayed.

```
Set myDialog = Dialogs(wdDialogFormatFont)
Selection.Font.Name = "Arial"
myDialog.Update
myDialog.Show
```

This example updates any fields in the active document that aren't updated automatically.

```
For Each afield In ActiveDocument.Fields
    If afield.LinkFormat.AutoUpdate = False Then afield.LinkFormat.Update
Next afield
```

AddAddress Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddAddressC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddAddressA"}

Adds an entry to the address book. Each entry has values for one or more tag IDs.

Syntax

expression.**AddAddress(*TagID, Value*)**

expression Required. An expression that returns an **Application** object.

TagID Required **String** array. The tag ID values for the new address entry. Each element in the array can contain one of the strings listed in the following table. Only the display name is required; the remaining entries are optional.

Tag ID	Description
PR_DISPLAY_NAME	Name displayed in the **Address Book** dialog box
PR_GIVEN_NAME	First name
PR_SURNAME	Last name
PR_STREET_ADDRESS	Street address
PR_LOCALITY	City or locality
PR_STATE_OR_PROVINCE	State or province
PR_POSTAL_CODE	Postal code
PR_COUNTRY	Country
PR_TITLE	Job title
PR_COMPANY_NAME	Company name
PR_DEPARTMENT_NAME	Department name within the company
PR_OFFICE_LOCATION	Office location
PR_PRIMARY_TELEPHONE_NUMBER	Primary telephone number
PR_PRIMARY_FAX_NUMBER	Primary fax number
PR_OFFICE_TELEPHONE_NUMBER	Office telephone number
PR_OFFICE2_TELEPHONE_NUMBER	Second office telephone number
PR_HOME_TELEPHONE_NUMBER	Home telephone number
PR_CELLULAR_TELEPHONE_NUMBER	Cellular telephone numbe
PR_BEEPER_TELEPHONE_NUMBER	Beeper telephone number
PR_COMMENT	Text included on the **Notes** tab for the address entry
PR_EMAIL_ADDRESS	Electronic mail address
PR_ADDRTYPE	ELectronic mail address type
PR_OTHER_TELEPHONE_NUMBER	Alternate telephone number (other than home or office)
PR_BUSINESS_FAX_NUMBER	Business fax number
PR_HOME_FAX_NUMBER	Home fax number
PR_RADIO_TELEPHONE_NUMBER	Radio telephone number
PR_INITIALS	Initials
PR_LOCATION	Location, in the format buildingnumber/roomnumber

	(for example, 7/3007 represents room 3007 in building 7)
PR_CAR_TELEPHONE_NUMBER	Car telephone number

Value Required **String** array. The values for the new address entry. Each element corresponds to an element in the ***TagID*** array. For more information, see the example.

AddAddress Method Example

This example adds an entry to the address book.

```
Dim tagIDArray(0 To 3) As String
Dim valueArray(0 To 3) As String
tagIDArray(0) = "PR_DISPLAY_NAME"
tagIDArray(1) = "PR_GIVEN_NAME"
tagIDArray(2) = "PR_SURNAME"
tagIDArray(3) = "PR_COMMENT"
valueArray(0) = "Kim Buhler"
valueArray(1) = "Kim"
valueArray(2) = "Buhler"
valueArray(3) = "This is a comment"
Application.AddAddress TagID:=tagIDArray(), Value:=valueArray()
```

BCCRecipients Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBCCRecipientsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBCCRecipientsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBCCRecipientsA"}

Returns or sets the recipients of a blind carbon copy of the mailer. Available only in Word for the Macintosh, with the PowerTalk mail system extension installed. Read/write **Variant**.

Remarks

This property uses an array of strings that specify the recipients' addresses, in one of the following formats:

- A record in the Preferred Personal Catalog. These names are one level deep ("Fred" or "June," for example).
- A full path that specifies either a record in a personal catalog (for example, "HD:Word Folder:My Catalog:Barney") or a plain record (for example, "HD:Folder:Martin").
- A relative path from the current working folder that specifies either a personal catalog record (for example, "My Catalog:Barney") or a plain record (for example, "Martin").
- A path in a PowerShare catalog tree, in the form "CATALOG_NAME:<node>:RECORD_NAME", where <node> is a path to a PowerShare catalog. An example of a complete path is "AppleTalk:North Building Zone:George's Mac."

BCCRecipients Property Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
    .HasMailer = True
    With .Mailer
        .Subject = "Here is the document"
        .ToRecipients = Array("Jean")
        .CCRecipients = Array("Adam", "Bernard")
        .BCCRecipients = Array("Chris")
        .Enclosures = Array("TestFile")
    End With
    .SendMailer
End With
```

CCRecipients Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCCRecipientsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCCRecipientsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCCRecipientsA"}

Returns or sets the recipients of a carbon copy (an indirect copy) of the mailer. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read/write **Variant**.

Remarks

This property uses an array of strings that specify the recipients' addresses, in one of the following formats:

- A record in the Preferred Personal Catalog. These names are one level deep ("Fred" or "June," for example).
- A full path that specifies either a record in a personal catalog (for example, "HD:Word Folder:My Catalog:Barney") or a plain record (for example, "HD:Folder:Martin").
- A relative path from the current working folder that specifies either a personal catalog record (for example, "My Catalog:Barney") or a plain record (for example, "Martin").
- A path in a PowerShare catalog tree, in the form "CATALOG_NAME:<node>:RECORD_NAME", where <node> is a path to a PowerShare catalog. An example of a complete path is "AppleTalk:North Building Zone:George's Mac."

CCRecipients Property Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
    .HasMailer = True
    With .Mailer
        .Subject = "Here is the document"
        .ToRecipients = Array("Jean")
        .CCRecipients = Array("Adam", "Bernard")
        .BCCRecipients = Array("Chris")
        .Enclosures = Array("TestFile")
    End With
    .SendMailer
End With
```

Enclosures Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnclosuresC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnclosuresX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnclosuresA"}

Returns or sets the enclosed files that are attached to the document mailer, as an array of strings, with each string indicating the path of a file to attach as an enclosure. Relative paths are acceptable; they're assumed to be based on the current folder. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read/write **Variant**.

Enclosures Property Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
    .HasMailer = True
    With .Mailer
        .Subject = "Here is the document"
        .ToRecipients = Array("Jean")
        .CCRecipients = Array("Adam", "Bernard")
        .BCCRecipients = Array("Chris")
        .Enclosures = Array("TestFile")
    End With
    .SendMailer
End With
```

ForwardMailer Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthForwardMailerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthForwardMailerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthForwardMailerA"}

Sets up the document mailer for forwarding by creating a new mailer that's preset with the subject and enclosures of the existing mailer. Valid only when the document has a received mailer attached to it (you can only forward a document you've received). Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed.

Syntax

expression.**ForwardMailer**

expression Required. An expression that returns a **Document** object.

Remarks

After you use this method to set up a document mailer for forwarding, you can change the mailer settings (if necessary) by using the **Mailer** property, and then you can use the **SendMailer** method to forward the document.

ForwardMailer Method Example

This example forwards the active document if it has a mailer.

```
With ActiveDocument
   If .HasMailer Then .ForwardMailer
End With
```

HasMailer Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasMailerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasMailerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasMailerA"}

True if the document has a mailer. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read/write **Boolean**.

HasMailer Property Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
   .HasMailer = True
   With .Mailer
      .Subject = "Here is the document"
      .ToRecipients = Array("Jean")
      .CCRecipients = Array("Adam", "Bernard")
      .BCCRecipients = Array("Chris")
      .Enclosures = Array("TestFile")
   End With
   .SendMailer
End With
```

Mailer Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailerA"}

Returns a **Mailer** object that represents the PowerTalk mailer attached to the document. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read-only.

Remarks

The **Mailer** object contains the properties needed to mail a document with PowerTalk. To mail a document, turn on the mailer by using the **HasMailer** property, set the mailer properties, and then send the document and mailer by using the **SendMailer** method.

Mailer Property Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
   .HasMailer = True
   With .Mailer
      .Subject = "Here is the document"
      .ToRecipients = Array("Jean")
      .CCRecipients = Array("Adam", "Bernard")
      .BCCRecipients = Array("Chris")
      .Enclosures = Array("TestFile")
   End With
   .SendMailer
End With
```

MailSystem Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailSystemC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailSystemX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailSystemA"}

Returns the mail system (or systems) installed on the host machine. Can be one of the following **WdMailSystem** constants: **wdMAPI**, **wdMAPIandPowerTalk**, **wdPowerTalk**, or **wdNoMailSystem**. Read-only **Long**.

MailSystem Property Example

This example forwards the active document if PowerTalk is installed on the machine and the document has a mailer.

```
ms = Application.MailSystem
If ms = wdPowerTalk Or ms = wdMAPIandPowerTalk Then
   With ActiveDocument
      If .HasMailer Then .ForwardMailer
   End With
End If
```

Received Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReceivedC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReceivedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReceivedA"}

True if the workbook mailer has been received (if it's been sent by another user to the current user) and the current user hasn't modified the mailer by using the **Reply**, **ReplyAll**, or **ForwardMailer** method. PowerTalk requires that mailers be received before they can be forwarded or replied to. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read-only **Boolean**.

Received Property Example

This example displays the current status of the **Received** property.

```
With ActiveDocument
   If .HasMailer Then
      MsgBox "Received property is " & _
         .Mailer.Received
   Else
      MsgBox "The document has no mailer"
   End If
End With
```

SendDateTime Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSendDateTimeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSendDateTimeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSendDateTimeA"}

Returns the date and time that the mailer was sent. For this property to be valid, the mailer must have already been sent. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read-only **Date**.

SendDateTime Property Example

This example displays the name of the user who sent the active document, plus the date and time it was sent.

```
With ActiveDocument
   If .HasMailer Then
      MsgBox "This document was sent by " & _
         .Mailer.Sender & " at " & _
         Format(.Mailer.SendDateTime, "General Date")
   End If
End With
```

Sender Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSenderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSenderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSenderA"}

Returns the name of the user who sent the document mailer. Available only in Microsoft Word for the Macintosh, with the PowerTalk mail system extension installed. Read-only **String**.

Sender Property Example

This example displays the name of the user who sent the active document, plus the date and time it was sent.

```
With ActiveDocument
   If .HasMailer Then
      MsgBox "This document was sent by " & _
         .Mailer.Sender & " at " & _
         Format(.Mailer.SendDateTime, "General Date")
   End If
End With
```

SendMailer Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSendMailerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSendMailerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSendMailerA"}

Sends the document by using the PowerTalk mailer. This method is available only on the Macintosh, with the PowerTalk system extension installed, and it can only be used on a document that has a mailer attached.

Syntax

expression.**SendMailer**(***FileFormat, Priority***)

expression Required. An expression that returns a **Document** object.

FileFormat Optional **Variant**. The format for the specified file. Can be one of the file format names listed in the **Save as type** box in the **Save As** dialog box (**File** menu).

Priority Optional **Variant**. The message priority. Can be one of the following **WdPriority** constants: **wdPriorityLow**, **wdPriorityNormal**, or **wdPriorityHigh**.

SendMailer Method Example

This example sets up the **Mailer** object for the active document and then sends the document.

```
With ActiveDocument
    .HasMailer = True
    With .Mailer
        .Subject = "Here is the document"
        .ToRecipients = Array("Jean")
        .CCRecipients = Array("Adam", "Bernard")
        .BCCRecipients = Array("Chris")
        .Enclosures = Array("TestFile")
    End With
    .SendMailer
End With
```

AddAsk Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddAskC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddAskX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddAskA "}

Adds an ASK field to a mail merge main document. Returns a **MailMergeField** object. When updated, an ASK field displays a dialog box that prompts you for text to assign to the specified bookmark.

Syntax

expression.**AddAsk**(***Range, Name, Prompt, DefaultAskText, AskOnce***)

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the ASK field.

Name Required **String**. The bookmark name that the response or default text is assigned to. Use a REF field with the bookmark name to display the result in a document.

Prompt Optional **Variant**. The text that's displayed in the dialog box.

DefaultAskText Optional **Variant**. The default response, which appears in the text box when the dialog box is displayed. Corresponds to the \d switch for an ASK field.

AskOnce Optional **Variant**. **True** to display the dialog box only once instead of each time a new data record is merged. Corresponds to the \o switch for an ASK field.

AddAsk Method Example

This example adds an ASK field at the end of the active mail merge main document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.MailMerge.Fields.AddAsk Range:=myRange, _
    Prompt:="Type your company name", Name:="company", AskOnce:=True
```

This example adds an ASK field after the last mail merge field in Main.doc.

```
Set myMMFields = Documents("Main.doc").MailMerge.Fields
myMMFields(myMMFields.Count).Select
Selection.MoveRight Unit:=wdWord, Count:=1, Extend:=wdMove
myMMFields.AddAsk Range:=Selection.Range, Name:="name", _
    Prompt:="What is your name"
```

AddFillIn Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFillInC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFillInX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFillInA "}

Adds a FILLIN field to a mail merge main document. Returns a **MailMergeField** object. When updated, a FILLIN field displays a dialog box that prompts you for text to insert into the document at the location of the FILLIN field.

Note Use the **Add** method with the **Fields** collection object to add a FILLIN field to a document other than a mail merge main document.

Syntax

expression.**AddFillIn**(***Range, Prompt, DefaultFillInText, AskOnce***)

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the FILLIN field.

Prompt Optional **Variant**. The text that's displayed in the dialog box.

DefaultFillinText Optional **Variant**. The default response, which appears in the text box when the dialog box is displayed. Corresponds to the \d switch for an FILLIN field.

AskOnce Optional **Variant**. **True** to display the prompt only once instead of each time a new data record is merged. Corresponds to the \o switch for a FILLIN field. The default value is **False**.

AddFillIn Method Example

This example adds a FILLIN field that prompts you for a name to insert after "Name:".

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .InsertAfter "Name: "
    .Collapse Direction:=wdCollapseEnd
End With
ActiveDocument.MailMerge.Fields.AddFillin Range:=Selection.Range, _
    Prompt:="Your name?", DefaultFillInText:="Joe", AskOnce:=True
```

AddIf Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddIfC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddIfX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddIfA "}

Adds an IF field to a mail merge main document. Returns a **MailMergeField** object. When updated, an IF field compares a field in a data record with a specified value, and then it inserts the appropriate text according to the result of the comparison.

Syntax

expression.**AddIf(*Range, MergeField, Comparison, CompareTo, TrueAutoText, TrueText, FalseAutoText, FalseText*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the IF field.

MergeField Required **String**. The merge field name.

Comparison Required **Long**. The operator used in the comparison. Can be one of the following **WdMailMergeComparison** constants: **wdMergeIfEqual**, **wdMergeIfGreaterThan**, **wdMergeIfGreaterThanOrEqual**, **wdMergeIfIsBlank**, **wdMergeIfIsNotBlank**, **wdMergeIfLessThan**, **wdMergeIfLessThanOrEqual**, or **wdMergeIfNotEqual**.

CompareTo Optional **Variant**. The text to compare with the contents of ***MergeField***.

TrueAutoText Optional **Variant**. The AutoText entry that's inserted if the comparison is true. If this argument is specified, ***TrueText*** is ignored.

TrueText Optional **Variant**. The text that's inserted if the comparison is true.

FalseAutoText Optional **Variant**. The AutoText entry that's inserted if the comparison is false. If this argument is specified, ***FalseText*** is ignored.

FalseText Optional **Variant**. The text that's inserted if the comparison is false.

AddIf Method Example

This example inserts "for your personal use" if the Company merge field is blank and "for your business" if the Company merge field is not blank.

```
ActiveDocument.MailMerge.Fields.AddIf Range:=Selection.Range, _
    MergeField:="Company", Comparison:=wdMergeIfIsBlank, _
    TrueText:="for your personal use", FalseText:="for your business"
```

This example inserts an IF field that compares the contents of the merge field named "Title" with the text "Mr." When the merge is performed, "Hello" is inserted if the comparison is true.

```
ActiveWindow.View.ShowFieldCodes = False
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
With ActiveDocument.MailMerge.Fields
    .AddIf Range:=myRange, MergeField:="Title", _
        Comparison:=wdMergeIfEqual, CompareTo:="Mr.", TrueText:="Hello "
End With
```

AddMergeRec Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddMergeRecC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddMergeRecX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddMergeRecA "}

Adds a MERGEREC field to a mail merge main document. Returns a **MailMergeField** object. A MERGEREC field inserts the number of the current data record (the position of the data record in the current query result) during a mail merge.

Syntax

expression.**AddMergeRec(*Range*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the MERGEREC field.

AddMergeRec Method Example

This example inserts text and a MERGEREC field at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.MailMerge.Fields.AddMergeRec Range:=myRange
myRange.InsertAfter "Record Number: "
```

AddMergeSeq Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddMergeSeqC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddMergeSeqX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddMergeSeqA "}

Adds a MERGESEQ field to a mail merge main document. Returns a **MailMergeField** object. A MERGESEQ field inserts a number based on the sequence in which data records are merged (for example, when record 50 of records 50 to 100 is merged, MERGESEQ inserts the number 1).

Syntax

expression.**AddMergeSeq(*Range*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the MERGESEQ field.

AddMergeSeq Method Example

This example inserts text and a MERGESEQ field at the end of the active document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.MailMerge.Fields.AddMergeSeq Range:=myRange
myRange.InsertAfter "Sequence Number: "
```

AddNext Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddNextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddNextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddNextA "}

Adds a NEXT field to a mail merge main document. Returns a **MailMergeField** object. A NEXT field advances to the next data record so that data from more than one record can be merged into the same merge document (for example, a sheet of mailing labels).

Syntax

expression.**AddNext(*Range*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the NEXT field.

AddNext Method Example

This example adds a NEXT field after the third MERGEFIELD field in Main.doc.

```
Documents("Main.doc").MailMerge.Fields(3).Select
Selection.Collapse Direction:=wdCollapseEnd
Documents("Main.doc").MailMerge.Fields.AddNext Range:=Selection.Range
```

AddNextIf Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddNextIfC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddNextIfX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddNextIfA "}

Adds a NEXTIF field to a mail merge main document. Returns a **MailMergeField** object. A NEXTIF field compares two expressions, and if the comparison is true, the next data record is merged into the current merge document.

Syntax

expression.**AddNextIf(*Range, MergeField, Comparison, CompareTo*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the NEXTIF field.

MergeField Required **String**. The merge field name.

Comparison Required **Long**. The operator used in the comparison. Can be one of the following **WdMailMergeComparison** constants: **wdMergeIfEqual**, **wdMergeIfGreaterThan**, **wdMergeIfGreaterThanOrEqual**, **wdMergeIfIsBlank**, **wdMergeIfIsNotBlank**, **wdMergeIfLessThan**, **wdMergeIfLessThanOrEqual**, or **wdMergeIfNotEqual**.

CompareTo Required **String**. The text to compare with the contents of ***MergeField***.

AddNextIf Method Example

This example adds a NEXTIF field before the first MERGEFIELD field in Main.doc. If the next postal code equals 98004, the next data record is merged into the current merge document.

```
Documents("Main.doc").MailMerge.Fields(1).Select
Selection.Collapse Direction:=wdCollapseStart
Documents("Main.doc").MailMerge.Fields.AddNextIf _
    Range:=Selection.Range, MergeField:="PostalCode", _
    Comparison:=wdMergeIfEqual, CompareTo:="98004"
```

AddSet Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddSetC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddSetX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddSetA "}

Adds a SET field to a mail merge main document. Returns a **MailMergeField** object. A SET field defines the text of the specified bookmark.

Syntax

expression.**AddSet(*Range, Name, ValueText, ValueAutoText*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the SET field.

Name Required **String**. The bookmark name that ***ValueText*** is assigned to.

ValueText Optional **Variant**. The text associated with the bookmark specified by the ***Name*** argument.

ValueAutoText Optional **Variant**. The AutoText entry that includes text associated with the bookmark specified by the ***Name*** argument. If this argument is specified, ***ValueText*** is ignored.

AddSet Method Example

This example adds a SET field at the beginning of the active document and then adds a REF field to display the text after the selection.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.MailMerge.Fields.AddSet Range:=myRange, _
    Name:="Name", ValueText:="Joe Smith"
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Fields.Add Range:=Selection.Range, _
    Type:=wdFieldRef, Text:="Name"
```

AddSkipIf Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddSkipIfC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddSkipIfX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddSkipIfA "}

Adds a SKIPIF field to a mail merge main document. Returns a **MailMergeField** object. A SKIPIF field compares two expressions, and if the comparison is true, SKIPIF moves to the next data record in the data source and starts a new merge document.

Syntax

expression.**AddSkipIf(*Range, MergeField, Comparison, CompareTo*)**

expression Required. An expression that returns a **MailMergeFields** object.

Range Required **Range** object. The location for the SKIPIF field.

MergeField Required **String**. The merge field name.

Comparison Required **Long**. The operator used in the comparison. Can be one of the following **WdMailMergeComparison** constants: **wdMergeIfEqual**, **wdMergeIfGreaterThan**, **wdMergeIfGreaterThanOrEqual**, **wdMergeIfIsBlank**, **wdMergeIfIsNotBlank**, **wdMergeIfLessThan**, **wdMergeIfLessThanOrEqual**, or **wdMergeIfNotEqual**.

CompareTo Optional **Variant**. The text to compare with the contents of ***MergeField***.

AddSkipIf Method Example

This example adds a SKIPIF field before the first MERGEFIELD field in Main.doc. If the next postal code equals 98040, the next data record is skipped and a new merge document is created.

```
Documents("Main.doc").MailMerge.Fields(1).Select
Selection.Collapse Direction:=wdCollapseStart
Documents("Main.doc").MailMerge.Fields.AddSkipIf _
    Range:=Selection.Range, MergeField:="PostalCode", _
    Comparison:=wdMergeIfEqual, CompareTo:="98040"
```

Data Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDataC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDataX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDataA "}

Returns or sets data in an ADDIN field. Read/write **String**.

Note The data is not visible in the field code or result; it is only accessible by returning the value of the **Data** property. If the field isn't an ADDIN field, this property will cause an error.

Data Property Example

This example inserts an ADDIN field at the insertion point in the active document and then sets the data for the field.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldAddin)
myField.Data = "Hidden information"
```

DataFields Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDataFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDataFieldsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDataFieldsA "}

Returns a **MailMergeDataFields** collection that represents the fields in the specified mail merge data source. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

DataFields Property Example

This example displays the name of each field in the data source attached to the active mail merge main document.

```
For Each mField In ActiveDocument.MailMerge.DataSource.DataFields
    MsgBox mField.Name
Next mField
```

This example displays the value of the LastName field from the first record in the data source attached to "Main.doc."

```
With Documents("Main.doc").MailMerge.DataSource
    .ActiveRecord = wdFirstRecord
    MsgBox .DataFields("LastName").Value
End With
```

DataSource Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDataSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDataSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDataSourceA "}

Returns a **MailMergeDataSource** object that refers to the data source attached to a mail merge main document. Read-only.

DataSource Property Example

This example displays the name of the data source attached to the active document.

```
If ActiveDocument.MailMerge.DataSource.Name <> "" Then _
    MsgBox ActiveDocument.MailMerge.DataSource.Name
```

This example displays the next record from the data source attached to Main.doc.

```
ActiveWindow.View.ShowFieldCodes = False
With Documents("Main.doc").MailMerge
    .ViewMailMergeFieldCodes = False
    .DataSource.ActiveRecord = wdNextRecord
End With
```

Destination Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDestinationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDestinationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDestinationA "}

Returns or sets the destination of the mail merge results. Can be one of the following **WdMailMergeDestination** constants: **wdSendToEmail**, **wdSendToFax**, **wdSendToNewDocument**, or **wdSendToPrinter**. Read/write **Long**.

Destination Property Example

This example sends the results of a mail merge operation to a new document.

```
Set mm = ActiveDocument.MailMerge
If mm.State = wdMainAndDataSource Then
    mm.Destination = wdSendToNewDocument
    mm.Execute
End If
```

EditHeaderSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditHeaderSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditHeaderSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditHeaderSourceA "}

Opens the header source attached to a mail merge main document, or activates the header source if it's already open.

Note If the mail merge main document doesn't have a header source, this method causes an error.

Syntax

expression.**EditHeaderSource**

expression Required. An expression that returns a **MailMerge** object.

EditHeaderSource Method Example

This example attaches a header source to the active document and then opens the header source.

```
With ActiveDocument.MailMerge
    .MainDocumentType = wdFormLetters
    .OpenHeaderSource Name:="C:\My Documents\Header.doc"
    .EditHeaderSource
End With
```

This example opens the header source if the active document has an associated header file attached to it.

```
Set MM = ActiveDocument.MailMerge
If MM.State = wdMainAndSourceAndHeader Or _
        MM.State = wdMainAndHeader Then
    MM.EditHeaderSource
End If
```

FieldNames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFieldNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFieldNamesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFieldNamesA "}

Returns a **MailMergeFieldNames** collection that represents the names of all the fields in the specified mail merge data source. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

FieldNames Property Example

This example displays the name of the first field in the data source attached to the active mail merge main document.

```
MsgBox ActiveDocument.MailMerge.DataSource.FieldNames(1).Name
```

This example uses the `mNames()` array to store the names of each merge field contained in the data source attached to the active document.

```
Set MM = ActiveDocument.MailMerge
num = ActiveDocument.MailMerge.DataSource.FieldNames.Count - 1
ReDim mNames(num)
i = 0
For Each aMergeField In MM.DataSource.FieldNames
    mNames(i) = aMergeField.Name
    i = i + 1
Next aMergeField
```

HeaderSourceName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeaderSourceNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeaderSourceNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeaderSourceNameA "}

Returns the path and file name of the header source attached to the specified mail merge main document. Read-only **String**.

HeaderSourceName Property Example

If a header source is attached to the active document, this example displays the file name.

```
hName = ActiveDocument.MailMerge.DataSource.HeaderSourceName
If hName <> "" Then MsgBox hName
```

This example opens the header source attached to the active document if the source is a Word document.

```
Set dSource = ActiveDocument.MailMerge.DataSource
If dSource.HeaderSourceType = wdMergeInfoFromWord Then
    Documents.Open FileName:=dSource.HeaderSourceName
End If
```

HeaderSourceType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeaderSourceTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeaderSourceTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeaderSourceTypeA "}

Returns a value that indicates the way the header source is being supplied for the mail merge operation. Can be one of the following **WdMailMergeDataSource** constants: **wdMergeInfoFromAccessDDE**, **wdMergeInfoFromExcelDDE**, **wdMergeInfoFromMSQueryDDE**, **wdMergeInfoFromODBC**, **wdMergeInfoFromWord**, or **wdNoMergeInfo**. Read-only **Long**.

HeaderSourceType Property Example

This example opens the header source attached to the active document if the source is a Word document.

```
Set dSource = ActiveDocument.MailMerge.DataSource
If dSource.HeaderSourceType = wdMergeInfoFromWord Then
    Documents.Open FileName:=dSource.HeaderSourceName
End If
```

MailMerge Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailMergeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailMergeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailMergeA "}

Returns a **MailMerge** object that represents the mail merge functionality for the specified document. Read-only.

Note The **MailMerge** object is available regardless of whether the specified document is a mail merge main document. Use the **State** property to determine the current state of the mail merge operation.

MailMerge Property Example

This example executes a mail merge if the active document is a main document with an attached data source.

```
Set myMerge = ActiveDocument.MailMerge
If myMerge.State = wdMainAndDataSource Then myMerge.Execute
```

This example merges the main document with data records 1 through 4 and sends the merge documents to the printer.

```
With ActiveDocument.MailMerge
    .DataSource.FirstRecord = 1
    .DataSource.LastRecord = 4
    .Destination = wdSendToPrinter
    .SuppressBlankLines = True
    .Execute
End With
```

MainDocumentType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMainDocumentTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMainDocumentTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMainDocumentTypeA "}

Returns or sets the mail merge main document type. Can be one of the following **WdMailMergeMainDocType** constants: **wdCatalog**, **wdEnvelopes**, **wdFormLetters**, **wdMailingLabels**, or **wdNotAMergeDocument**. Read/write **Long**.

Note If you set this property for a document that's already a main document, the attached data source is removed.

MainDocumentType Property Example

This example creates a new document and makes it a catalog main document for a mail merge operation.

```
Set myDoc = Documents.Add
myDoc.MailMerge.MainDocumentType = wdCatalog
```

This example determines whether the active document is a main document for a mail merge operation, and then it displays a message in the status bar.

```
Set doc = ActiveDocument
If doc.MailMerge.MainDocumentType = wdNotAMergeDocument Then
    StatusBar = "Not a mail merge main document"
Else
    StatusBar = "Document is a mail merge main document.
End If
```

ViewMailMergeFieldCodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproViewMailMergeFieldCodesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproViewMailMergeFieldCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproViewMailMergeFieldCodesA "}

True if merge field names are displayed in a mail merge main document. **False** if information from the current data record is displayed. Read/write **Boolean**.

Note If the active document isn't a mail merge main document, this property causes an error. To view merge field names or their results, set the **ShowFieldCodes** property to **False**.

ViewMailMergeFieldCodes Property Example

This example displays the mail merge fields in Main.doc.

```
ActiveWindow.View.ShowFieldCodes = False
With Documents("Main.doc")
    .Activate
    .MailMerge.ViewMailMergeFieldCodes = True
End With
```

If the active document is set up for a mail merge operation, this example displays the current data record information in the main document.

```
ActiveWindow.View.ShowFieldCodes = False
Set myMerge = ActiveDocument.MailMerge
If myMerge.State = wdMainAndSourceAndHeader Or _
    myMerge.State = wdMainAndDataSource Then
    myMerge.ViewMailMergeFieldCodes = False
End If
```

FirstRecord Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstRecordC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstRecordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstRecordA "}

Returns or sets the number of the first data record to be merged in a mail merge operation. Read/write **Long**.

FirstRecord Property Example

This example merges the main document with data records 1 through 3 and sends the merge documents to the printer.

```
With ActiveDocument.MailMerge
    .DataSource.FirstRecord = 1
    .DataSource.LastRecord = 3
    .Destination = wdSendToPrinter
    .Execute
End With
```

LastRecord Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLastRecordC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLastRecordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLastRecordA "}

Returns or sets the number of the last data record to be merged in a mail merge operation. Read/write **Long**.

LastRecord Property Example

This example merges the main document with data records 2 through 4 and sends the merge documents to a new document.

```
With ActiveDocument.MailMerge
    .DataSource.FirstRecord = 2
    .DataSource.LastRecord = 4
    .Destination = wdSendToNewDocument
    .Execute
End With
```

SuppressBlankLines Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSuppressBlankLinesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSuppressBlankLinesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSuppressBlankLinesA "}

True if blank lines are suppressed when mail merge fields in a mail merge main document are empty. Read/write **Boolean**.

SuppressBlankLines Property Example

This example opens Main.doc and executes the mail merge operation. When merge fields are empty, blank lines are suppressed in the merge document.

```
Set myDoc = Documents.Open(FileName:="C:\My Documents\Main.doc")
With myDoc.MailMerge
    .SuppressBlankLines = True
    .Destination = wdSendToPrinter
    .Execute
End With
```

MailAddressFieldName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailAddressFieldNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailAddressFieldNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailAddressFieldNameA "}

Returns or sets the name of the field that contains electronic mail addresses that are used when the mail merge destination is electronic mail. Read/write **String**.

MailAddressFieldName Property Example

This example merges the document named "Form Letter.doc" with it's attached data document and sends the results to the electronic mail addresses stored in the Email merge field.

```
With Documents("Form Letter.doc").MailMerge
    .MailAddressFieldName = "Email"
    .MailSubject = "Amazing offer"
    .Destination = wdSendToEmail
    .Execute
End With
```

MailMergeDataView Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailMergeDataViewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailMergeDataViewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailMergeDataViewA "}

True if mail merge data is displayed instead of mail merge fields in the specified window. Read/write **Boolean**.

Note If the specified window isn't a main document, an error occurs.

MailMergeDataView Property Example

If the active document includes at least one mail merge field, this example displays mail merge data from the first record in the attached data source.

```
If ActiveDocument.MailMerge.Fields.Count >= 1 Then
    ActiveDocument.MailMerge.DataSource.ActiveRecord = 1
    ActiveWindow.View.ShowFieldCodes = False
    ActiveWindow.View.MailMergeDataView = True
End If
```

This example toggles between viewing mail merge fields and viewing the resulting data.

```
With ActiveWindow.View
    .ShowFieldCodes = False
    .MailMergeDataView = Not .MailMergeDataView
End With
```

MailSubject Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailSubjectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailSubjectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailSubjectA "}

Returns or sets the subject line used when the mail merge destination is electronic mail. Read/write **String**.

MailSubject Property Example

This example merges the document named "Offer.doc" with it's attached data document. The results are sent to the electronic mail addresses stored in the EmailNames merge field, and the subject of the mail message is "Amazing Offer."

```
With Documents("Offer.doc").MailMerge
    .MailAddressFieldName = "EmailNames"
    .MailSubject = "Amazing Offer"
    .Destination = wdSendToEmail
    .Execute
End With
```

UseAddressBook Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUseAddressBookC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUseAddressBookX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUseAddressBookA "}

Selects the address book that's used as the data source for a mail merge operation.

Syntax

expression.**UseAddressBook(*Type*)**

expression Required. An expression that returns a **MailMerge** object.

Type Required **String**. The address book to be used as the mail merge data source. The string corresponds to the application's file extension, as specified in the registry (for example, "olk" for Outlook, "scd" for Schedule+, or "pab" for the Personal Address Book).

UseAddressBook Method Example

This example creates a new main document that uses the Personal Address Book as it's data source.

```
Set myDoc = Documents.Add
With myDoc.MailMerge
    .MainDocumentType = wdFormLetters
    .UseAddressBook Type:="pab"
End With
```

QueryString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproQueryStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproQueryStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproQueryStringA "}

Returns or sets the query string (SQL statement) used to retrieve a subset of the data in a mail merge data source. Read/write **String**.

QueryString Property Example

This example returns the query string for the data source attached to the active document.

```
qString = ActiveDocument.MailMerge.DataSource.QueryString
```

OpenDataSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenDataSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenDataSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenDataSourceA "}

Attaches a data source to the specified document, which becomes a main document if it's not one already.

Syntax

expression.**OpenDataSource(*Name, Format, ConfirmConversions, ReadOnly, LinkToSource, AddToRecentFiles, PasswordDocument, PasswordTemplate, Revert, WritePasswordDocument, WritePasswordTemplate, Connection, SQLStatement, SQLStatement1*)**

expression Required. An expression that returns a **MailMerge** object.

Name Required **String**. The data source file name. In Windows, you can specify a Microsoft Query (.qry) file instead of specifying a data source, a connection string, and a query string.

Format Optional **Variant**. The file converter used to open the document. Can be one of the following **WdOpenFormat** constants: **wdOpenFormatAuto**, **wdOpenFormatDocument**, **wdOpenFormatRTF**, **wdOpenFormatTemplate**, **wdOpenFormatText**, or **wdOpenFormatUnicodeText**. The default value is **wdOpenFormatAuto**.

To specify an external file format, use the **OpenFormat** property with a **FileConverter** object to determine the value to use with this argument.

ConfirmConversions Optional **Variant**. **True** to display the **Convert File** dialog box if the file isn't in Word format.

ReadOnly Optional **Variant**. **True** to open the data source on a read-only basis.

LinkToSource Optional **Variant**. **True** to perform the query specified by ***Connection*** and ***SQLStatement*** each time the main document is opened.

AddToRecentFiles Optional **Variant**. **True** to add the file name to the list of recently used files at the bottom of the **File** menu.

PasswordDocument Optional **Variant**. The password used to open the data source.

PasswordTemplate Optional **Variant**. The password used to open the template.

Revert Optional **Variant**. Controls what happens if ***Name*** is the file name of an open document. **True** to discard any unsaved changes to the open document and reopen the file; **False** to activate the open document.

WritePasswordDocument Optional **Variant**. The password used to save changes to the document.

WritePasswordTemplate Optional **Variant**. The password used to save changes to the template.

Connection Optional **Variant**. A range within which the query specified by ***SQLStatement*** is to be performed. How you specify the range depends on how data is retrieved. For example:

- When retrieving data through ODBC (Windows only), you specify a connection string.
- When retrieving data from Microsoft Excel using dynamic data exchange (DDE), you specify a named range.
- When retrieving data from Microsoft Access (Windows only), you specify the word "Table" or "Query" followed by the name of a table or query.

SQLStatement Optional **Variant**. Defines query options for retrieving data.

SQLStatement1 Optional **Variant**. If the query string is longer than 255 characters, ***SQLStatement*** specifies the first portion of the string, and ***SQLStatement1*** specifies the second portion.

Remarks

To determine the ODBC connection and query strings, set query options manually, and use the **QueryString** property to return the connection string. The following table includes some commonly

used SQL keywords.

Keyword	Description
DSN	The name of the ODBC data source
UID	The user logon ID
PWD	The user-specified password
DBQ	The database file name
FIL	The file type

OpenDataSource Method Example

This example creates a new main document and attaches the Customers table from a Microsoft Access database named "Northwind.mdb."

```
Set myDoc = Documents.Add
With myDoc.MailMerge
    .MainDocumentType = wdFormLetters
    .OpenDataSource Name:="C:\MSOffice\Access\Samples\Northwind.mdb", _
        LinkToSource:=True, AddToRecentFiles:=False, _
        Connection:="TABLE Orders
End With
```

This example creates a new main document and attaches the Microsoft Excel spreadsheet named "Names.xls." The ***Connection*** argument retrieves data from the range named "Sales."

```
Set myDoc = Documents.Add
With myDoc.MailMerge
    .MainDocumentType = wdCatalog
    .OpenDataSource Name:="C:\My Documents\Names.xls", ReadOnly:=True, _
        Connection:="Sales"
End With
```

This example uses ODBC to attach the Microsoft Access database named "NorthWind.mdb" to the active document. The ***SQLStatement*** argument selects the records in the Customers table.

```
With ActiveDocument.MailMerge
    .MainDocumentType = wdFormLetters
    constr = "DSN=MS Access Databases;DBQ=C:\Access\NorthWind.mdb;" _
        & "FIL=RedISAM;"
   .OpenDataSource Name:="C:\MSOffice\Access\NorthWind.mdb", _
        Connection:=constr, SQLStatement:="SELECT * FROM Customers"
End With
```

ConnectString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproConnectStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproConnectStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproConnectStringA "}

Returns the connection string for the specified mail merge data source. Read-only **String**.

ConnectString Property Example

This example creates a new main document and attaches the Customers table from a Microsoft Access database named "Northwind.mdb." The connection string is displayed in a message box.

```
Set myDoc = Documents.Add
With myDoc.MailMerge
    .MainDocumentType = wdFormLetters
    .OpenDataSource Name:="C:\MSOffice\Access\Samples\Northwind.mdb", _
        LinkToSource:=True, AddToRecentFiles:=False, _
        Connection:="TABLE Customers
    MsgBox .DataSource.ConnectString
End With
```

CreateDataSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreateDataSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreateDataSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreateDataSourceA "}

Creates a Word document that uses a table to store data for a mail merge. The new data source is attached to the specified document, which becomes a main document if it's not one already.

Syntax

expression.**CreateDataSource(*Name, PasswordDocument, WritePasswordDocument, HeaderRecord, MSQuery, SQLStatement, SQLStatement1, Connection, LinkToSource*)**

expression Required. An expression that returns a **MailMerge** object.

Name Optional **Variant**. The path and file name for the new data source.

PasswordDocument Optional **Variant**. The password required to open the new data source.

WritePasswordDocument Optional **Variant**. The password required to save changes to the data source.

HeaderRecord Optional **Variant**. Field names for the header record. If this argument is omitted, the standard header record is used: "Title, FirstName, LastName, JobTitle, Company, Address1, Address2, City, State, PostalCode, Country, HomePhone, WorkPhone." To separate field names in Windows, use the list separator specified in **Regional Settings** in **Control Panel**.

MSQuery Optional **Variant**. **True** to launch Microsoft Query, if it's installed (Windows only). The ***FileName***, ***PasswordDoc***, and ***HeaderRecord*** arguments are ignored.

SQLStatement Optional **Variant**. Defines query options for retrieving data.

SQLStatement1 Optional **Variant**. If the query string is longer than 255 characters, ***SQLStatement*** specifies the first portion of the string, and ***SQLStatement1*** specifies the second portion.

Connection Optional **Variant**. A range within which the query specified by ***SQLStatement*** will be performed. How you specify the range depends on how data is retrieved. For example:

- When retrieving data through ODBC (Windows only), you specify a connection string.
- When retrieving data from Microsoft Excel using dynamic data exchange (DDE), you specify a named range.
- When retrieving data from Microsoft Access (Windows only), you specify the word "Table" or "Query" followed by the name of a table or query.

LinkToSource Optional **Variant**. **True** to perform the query specified by ***Connection*** and ***SQLStatement*** each time the main document is opened.

CreateDataSource Method Example

This example creates a new data source document named "Data.doc" and attaches the data source to the active document. The new data source includes a five-column table that has the field names specified by the ***HeaderRecord*** argument.

```
ActiveDocument.MailMerge.CreateDataSource Name:="C:\My Documents\Data.doc",
_
    HeaderRecord:="Name, Address, City, State, Zip"
```

EditDataSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditDataSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditDataSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditDataSourceA "}

Opens or switches to the mail merge data source.

Syntax

expression.**EditDataSource**

expression Required. An expression that returns a **MailMerge** object.

Remarks

If the data source is a Word document, this method opens the data source (or activates the data source if it's already open).

If Word is accessing the data through dynamic data exchange (DDE) − using an application such as Microsoft Excel or Microsoft Access − this method displays the data source in that application.

If Word is accessing the data through open database connectivity (ODBC) (Windows only), this method displays the data in a Word document. Note that if Microsoft Query is installed, a message appears, providing the option to display Microsoft Query instead of converting data.

EditDataSource Method Example

This example opens or activates the data source attached to the document named "Sales.doc."

```
Documents("Sales.doc").MailMerge.EditDataSource
```

This example opens or activates the attached data source if the data source is a Word document.

```
Set myDataSource = ActiveDocument.MailMerge.DataSource
If myDataSource.Type = wdMergeInfoFromWord Then
    ActiveDocument.MailMerge.EditDataSource
End If
```

CreateHeaderSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreateHeaderSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreateHeaderSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreateHeaderSourceA "}

Creates a Word document that stores a header record that's used in place of the data source header record in a mail merge. This method attaches the new header source to the specified document, which becomes a main document if it's not one already.

Note The new header source uses a table to arrange mail merge field names.

Syntax

expression.**CreateHeaderSource(*Name, PasswordDocument, WritePasswordDocument, HeaderRecord*)**

expression Required. An expression that returns a **MailMerge** object.

Name Required **String**. The path and file name for the new header source.

PasswordDocument Optional **Variant**. The password required to open the new header source.

WritePasswordDocument Optional **Variant**. The password required to save changes to the header source.

HeaderRecord Optional **Variant**. A string that specifies the field names for the header record. If this argument is omitted, the standard header record is used: "Title, FirstName, LastName, JobTitle, Company, Address1, Address2, City, State, PostalCode, Country, HomePhone, WorkPhone." To separate field names in Windows, use the list separator specified in **Regional Settings** in **Control Panel**.

CreateHeaderSource Method Example

This example creates a header source with five field names and attaches the new header source named "Header.doc" to the active document.

```
ActiveDocument.MailMerge.CreateHeaderSource Name:="Header.doc", _
    HeaderRecord:="Name, Address, City, State, Zip"
```

This example creates a header source for the document named "Main.doc" (with the standard header record) and opens the data source named "Data.doc."

```
With Documents("Main.doc").MailMerge
    .CreateHeaderSource Name:="Fields.doc"
    .OpenDataSource Name:="C:\My Documents\Data.doc"
End With
```

Check Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCheckC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCheckX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCheckA "}

Simulates the mail merge operation, pausing to report each error as it occurs.

Syntax

expression.**Check**

expression Required. An expression that returns a **MailMerge** object.

Check Method Example

This example checks the active document for mail merge errors.

```
theState = ActiveDocument.MailMerge.State
If theState = wdMainAndDataSource Or _
    theState = wdMainAndSourceAndHeader Then
    ActiveDocument.MailMerge.Check
End If
```

EditMainDocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditMainDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditMainDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditMainDocumentA "}

Activates the mail merge main document associated with the specified header source or data source document.

Note If the main document isn't open, an error occurs. Use the **Open** method if the main document isn't currently open.

Syntax

expression.**EditMainDocument**

expression Required. An expression that returns a **MailMerge** object.

EditMainDocument Method Example

This example attempts to activate the main document associated with the active data source document. If the main document isn't open, the **Open** dialog box is displayed, with a message in the status bar.

```
On Error GoTo errorhandler
Documents("Data.doc").MailMerge.EditMainDocument
errorhandler:
If Err = 4605 Then StatusBar = "Main document is not open"
Dialogs(wdDialogFileOpen).Show
```

OpenHeaderSource Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenHeaderSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenHeaderSourceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenHeaderSourceA "}

Attaches a mail merge header source to the specified document.

Note When a header source is attached, the first record in the header source is used in place of the header record in the data source.

Syntax

expression.**OpenHeaderSource(*Name, Format, ConfirmConversions, ReadOnly, AddToRecentFiles, PasswordDocument, PasswordTemplate, Revert, WritePasswordDocument, WritePasswordTemplate*)**

expression Required. An expression that returns a **MailMerge** object.

Name Required **String**. The file name of the header source.

Format Optional **Variant**. The file converter used to open the document. Can be one of the following **WdOpenFormat** constants: **wdOpenFormatAuto**, **wdOpenFormatDocument**, **wdOpenFormatRTF**, **wdOpenFormatTemplate**, **wdOpenFormatText**, or **wdOpenFormatUnicodeText**. The default value is **wdOpenFormatAuto**.

To specify an external file format, use the **OpenFormat** property with a **FileConverter** object to determine the value to use with this argument.

ConfirmConversions Optional **Variant**. **True** to display the **Convert File** dialog box if the file isn't in Word format.

ReadOnly Optional **Variant**. **True** to open the header source on a read-only basis.

AddToRecentFiles Optional **Variant**. **True** to add the file name to the list of recently used files at the bottom of the **File** menu.

PasswordDocument Optional **Variant**. The password required to open the header source document.

PasswordTemplate Optional **Variant**. The password required to open the header source template.

Revert Optional **Variant**. Controls what happens if ***Name*** is the file name of an open document. **True** to discard any unsaved changes to the open document and reopen the file; **False** to activate the open document.

WritePasswordDocument Optional **Variant**. The password required to save changes to the document data source.

WritePasswordTemplate Optional **Variant**. The password required to save changes to the template data source.

OpenHeaderSource Method Example

This example sets the active document as a main document for form letters, and then it attaches the header source named "Header.doc" and the data document named "Names.doc."

```
With ActiveDocument.MailMerge
    .MainDocumentType = wdFormLetters
    .OpenHeaderSource Name:="C:\My Documents\Header.doc", _
         Revert:=False, AddToRecentFiles:=False
    .OpenDataSource Name:="C:\My Documents\Names.doc"
End With
```

FindRecord Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFindRecordC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFindRecordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFindRecordA "}

Searches the contents of the specified mail merge data source for text in a particular field. Returns **True** if the search text is found.

Note Corresponds to the **Find Record** button on the **Mail Merge** toolbar.

Syntax

expression.**FindRecord(*FindText, Field*)**

expression Required. An expression that returns a **MailMergeDataSource** object.

FindText Required **String**. The text to be looked for.

Field Required **String**. The name of the field to be searched.

FindRecord Method Example

This example displays a merge document for the first data record in which the FirstName field contains "Joe." If the data record is found, the number of the record is stored in the `numRecord` variable.

```
ActiveDocument.MailMerge.ViewMailMergeFieldCodes = False
Set myMMData = ActiveDocument.MailMerge.DataSource
If myMMData.FindRecord(FindText:="Joe", Field:="FirstName") = True Then
    numRecord = myMMData.ActiveRecord
End If
```

MailAsAttachment Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailAsAttachmentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailAsAttachmentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailAsAttachmentA "}

True if the merge documents are sent as attachments when the mail merge destination is an e-mail message or a fax. Read/write **Boolean**.

MailAsAttachment Property Example

This example performs a mail merge operation and sends the merge results as attachments to e-mail messages. The e-mail addresses are stored in the MailAddress merge field.

```
With Documents("Main.doc").MailMerge
    .MailAsAttachment = True
    .Destination = wdSendToEmail
    .MailSubject = "Special offer"
    .MailAddressFieldName = "MailAddress"
    .Execute
End With
```

TwoInitialCapsExceptions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTwoInitialCapsExceptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTwoInitialCapsExceptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTwoInitialCapsExceptionsA "}

Returns a **TwoInitialCapsExceptions** collection that represents the list of terms containing mixed capitalization that Word won't correct automatically. This list corresponds to the list of AutoCorrect exceptions on the **INitial CAps** tab in the **AutoCorrect Exceptions** dialog box (**AutoCorrect** command, **Tools** menu). Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

TwoInitialCapsExceptions Property Example

This example prompts the user to delete or keep each AutoCorrect Initial Caps exception.

```
For Each anEntry In AutoCorrect.TwoInitialCapsExceptions
    response = MsgBox ("Delete entry: " & anEntry.Name, vbYesNoCancel)
    If response = vbYes Then
        anEntry.Delete
    Else
        If response = vbCancel Then End
    End If
Next anEntry
```

DefaultFilePath Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultFilePathC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultFilePathX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultFilePathA "}

Returns or sets default folders for items such as documents, templates, and graphics. Read/write **String**.

Syntax

expression.**DefaultFilePath(*Path*)**

expression Required. An expression that returns an **Options** object.

Path Required **Long**. The default folder to set. Can be one of the following **WdDefaultFilePath** constants:

wdAutoRecoverPath
wdBorderArtPath
wdCurrentFolderPath
wdDocumentsPath
wdGraphicsFiltersPath
wdPicturesPath
wdProgramPath
wdProofingToolsPath
wdStartupPath
wdStyleGalleryPath
wdTempFilePath
wdTextConvertersPath
wdToolsPath
wdTutorialPath
wdUserOptionsPath
wdUserTemplatesPath
wdWorkgroupTemplatesPath

Remarks

The new setting takes effect immediately.

You can use an empty string ("") to remove the setting from the registry (Windows 95 and Windows NT) or Word Settings (8) (Macintosh).

DefaultFilePath Property Example

This example sets the default folder for Word documents.

```
Options.DefaultFilePath(wdDocumentsPath) = "c:\My Documents"
```

This example returns the current default path for user templates (corresponds to the default path setting on the **File Locations** tab in the **Options** dialog box).

```
temp = Options.DefaultFilePath(wdUserTemplatesPath)
```

SendMailAttach Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSendMailAttachC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSendMailAttachX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSendMailAttachA "}

True if the **Send To** command on the **File** menu inserts the active document as an attachment to a mail message. **False** if the **Send To** command inserts the contents of the active document as text in a mail message. Read/write **Boolean**.

SendMailAttach Property Example

This example opens a new mail message that has the active document attached to it.

```
Options.SendMailAttach = True
ActiveDocument.SendMail
```

This example returns the state of the **Mail as attachment** option on the **General** tab of the **Options** dialog box.

```
Msgbox Options.SendMailAttach
```

DisplayScreenTips Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayScreenTipsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayScreenTipsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayScreenTipsA "}

True if comments, footnotes, endnotes, and hyperlinks are displayed as tips. Text marked as having comments is highlighted. Read/write **Boolean**.

DisplayScreenTips Property Example

This example enables Word to display comments, footnotes, and endnotes as tips. Also, text marked as having comments is highlighted.

```
Application.DisplayScreenTips = True
```

This example returns the current status of the **ScreenTips** option on the **View** tab in the **Options** dialog box.

```
temp = Application.DisplayScreenTips
```

DefaultBorderLineStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultBorderLineStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultBorderLineStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultBorderLineStyleA "}

Returns or sets the default border line style. Read/write **Long**.

Can be one of the following **WdLineStyle** constants:

wdLineStyleDashDot
wdLineStyleDashDotDot
wdLineStyleDashDotStroked
wdLineStyleDashLargeGap
wdLineStyleDashSmallGap
wdLineStyleDot
wdLineStyleDouble
wdLineStyleDoubleWavy
wdLineStyleEmboss3D
wdLineStyleEngrave3D
wdLineStyleNone
wdLineStyleSingle
wdLineStyleSingleWavy
wdLineStyleThickThinLargeGap
wdLineStyleThickThinMedGap
wdLineStyleThickThinSmallGap
wdLineStyleThinThickLargeGap
wdLineStyleThinThickMedGap
wdLineStyleThinThickSmallGap
wdLineStyleThinThickThinLargeGap
wdLineStyleThinThickThinMedGap
wdLineStyleThinThickThinSmallGap
wdLineStyleTriple

DefaultBorderLineStyle Property Example

This example sets the default line style to double.

```
Options.DefaultBorderLineStyle = wdLineStyleDouble
```

This example returns the current default line style.

```
temp = Options.DefaultBorderLineStyle
```

EnableSound Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnableSoundC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnableSoundX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnableSoundA "}

True if Word makes the computer respond with a sound whenever an error occurs. Read/write **Boolean**.

EnableSound Property Example

This example sets the **Provide feedback with sound** option on the **General** tab in the **Options** dialog box, based on user input.

```
If MsgBox("Do you want Word to beep on errors?", 36) = vbYes Then
    Options.EnableSound = True
Else
    Options.EnableSound = False
End If
```

NoLineNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNoLineNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNoLineNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNoLineNumberA "}

True if line numbers are repressed for the specified paragraphs. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Remarks

Use the **LineNumbering** property to set line numbers.

NoLineNumber Property Example

This example enables line numbering for the active document. The starting number is set to 1, and the numbering is continuous throughout all sections in the document. Line numbering is then repressed for the second paragraph.

```
With ActiveDocument.PageSetup.LineNumbering
    .Active = True
    .StartingNumber = 1
    .CountBy = 1
    .RestartMode = wdRestartContinuous
End With
ActiveDocument.Paragraphs(2).NoLineNumber = True
```

DifferentFirstPageHeaderFooter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDifferentFirstPageHeaderFooterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDifferentFirstPageHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDifferentFirstPageHeaderFooterA "}

True if a different header or footer is used on the first page. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

DifferentFirstPageHeaderFooter Property Example

This example checks each section in the active document for headers and footers that are different on the first page and displays a message if any are found.

```
For Each sec In ActiveDocument.Sections
    If sec.PageSetup.DifferentFirstPageHeaderFooter = True Then
    Msgbox "Section " & sec.Index & " has different first page headers &
footers."
    End If
Next sec
```

Sections Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSectionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSectionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSectionsA "}

Returns a **Sections** collection that represents the sections in the specified document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Sections Property Example

This example sets the page orientation for all the sections in the active document.

```
For Each sec In ActiveDocument.Sections
    sec.PageSetup.Orientation = wdOrientLandscape
Next sec
```

This example creates a new document then adds some text to the document. It then creates a new section in the document and inserts text into the new section.

```
Set myDoc = Documents.Add
Selection.InsertAfter "This is section 1."
Set mysec = myDoc.Sections.Add
mysec.Range.InsertAfter "This is section 2"
```

Tasks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTasksC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTasksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTasksA "}

Returns a **Tasks** collection that represents all the applications that are running. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Tasks Property Example

This example displays the calculator. If the calculator is not already running, then Word starts the task and then displays the calculator.

```
If Tasks.Exists("Calculator") Then
    With Tasks("Calculator")
        .Activate
        .WindowState = wdWindowStateNormal
    End With
Else
    Shell "calc.exe"
    Tasks("Calculator").WindowState = wdWindowStateNormal
End If
```

This example checks to see whether Microsoft Excel is currently running. If the task is running, the example activates Microsoft Excel; otherwise, a message box is displayed.

```
If Tasks.Exists("Microsoft Excel") = True Then
    With Tasks("Microsoft Excel")
        .Activate
        .WindowState = wdWindowStateMaximize
    End With
Else
    Msgbox "Microsoft Excel is not currently running."
End If
```

PaperSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPaperSizeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPaperSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPaperSizeA "}

Returns or sets the paper size. Read/write **Long**.

Can be one of the following **WdPaperSize** constants:

wdPaper10x14
wdPaper11x17
wdPaperA3
wdPaperA4
wdPaperA4Small
wdPaperA5
wdPaperB4
wdPaperB5
wdPaperCSheet
wdPaperCustom
wdPaperDSheet
wdPaperEnvelope10
wdPaperEnvelope11
wdPaperEnvelope12
wdPaperEnvelope14
wdPaperEnvelope9
wdPaperEnvelopeB4
wdPaperEnvelopeB5
wdPaperEnvelopeB6
wdPaperEnvelopeC3
wdPaperEnvelopeC4
wdPaperEnvelopeC5
wdPaperEnvelopeC6
wdPaperEnvelopeC65
wdPaperEnvelopeDL
wdPaperEnvelopeItaly
wdPaperEnvelopeMonarch
wdPaperEnvelopePersonal
wdPaperESheet
wdPaperExecutive
wdPaperFanfoldLegalGerman
wdPaperFanfoldStdGerman
wdPaperFanfoldUS
wdPaperFolio
wdPaperLedger
wdPaperLegal
wdPaperLetter
wdPaperLetterSmall
wdPaperNote
wdPaperQuarto
wdPaperStatement
wdPaperTabloid

Remarks

Setting the **PageHeight** property or the **PageWidth** property changes the **PaperSize** property to **wdPaperCustom**.

PaperSize Property Example

This example sets the paper size to legal for the first document.

```
Documents(1).PageSetup.PaperSize = wdPaperLegal
```

AutoFormatReplaceSymbols Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplaceSymbolsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplaceSymbolsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplaceSymbolsA "}

True if two consecutive hyphens (--) are replaced by an en dash (–) or an em dash (—) when when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatReplaceSymbols Property Example

This example turns on the replacement of hyphens with symbols, and then it formats the current selection automatically.

```
Options.AutoFormatReplaceSymbols = True
Selection.Range.AutoFormat
```

This example returns the status of the **Symbol characters (--) with symbols (—)** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceFractions
```

AutoFormatAsYouTypeReplaceSymbols Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplaceSymbolsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplaceSymbolsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplaceSymbolsA "}

True if two consecutive hyphens (--) are replaced with an en dash (–) or an em dash (—) as you type. Read/write **Boolean**.

Note If the hyphens are typed with leading and trailing spaces, Word replaces the hyphens with an en dash; if there are no trailing spaces, the hyphens are replaced with an em dash.

AutoFormatAsYouTypeReplaceSymbols Property Example

This example turns on the replacement of hyphens with symbols as you type.

```
Options.AutoFormatAsYouTypeReplaceSymbols = True
```

This example returns the status of the **Symbol characters (--) with symbols (—)** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeReplaceSymbols
```

ShowReadabilityStatistics Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowReadabilityStatisticsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowReadabilityStatisticsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowReadabilityStatisticsA "}

True if the Word displays a list of summary statistics, including measures of readability, when it has finished checking grammar. Read/write **Boolean**.

ShowReadabilityStatistics Property Example

This example sets Word to show readability statistics, and then it checks the spelling and grammar in the active document.

```
Options.ShowReadabilityStatistics = True
ActiveDocument.CheckGrammar
```

This example returns the current status of the **Show readability statistics** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.ShowReadabilityStatistics
```

CheckGrammarWithSpelling Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCheckGrammarWithSpellingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCheckGrammarWithSpellingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCheckGrammarWithSpellingA "}

True if Word checks grammar while checking spelling. Read/write **Boolean**.

Remarks

This property controls whether Word checks grammar when you check spelling by using the **Spelling** command (**Tools** menu).

To check spelling or grammar from a Visual Basic procedure, use the **CheckSpelling** method to check only spelling and use the **CheckGrammar** method to check both grammar and spelling.

CheckGrammarWithSpelling Property Example

This example returns the status of the **Check grammar with spelling** option on the **Spelling & Grammar** tab in the **Options** dialog box. If the option is selected, the procedure checks both spelling and grammar for the active document; otherwise, only spelling is checked.

```
If Options.CheckGrammarWithSpelling = True Then
    ActiveDocument.CheckGrammar
Else
    ActiveDocument.CheckSpelling
End If
```

Assistant Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAssistantC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAssistantX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAssistantA "}

Returns an **Assistant** object that represents the Office Assistant. Read-only.

Assistant Property Example

This example displays the Office Assistant.

```
Assistant.Visible = True
```

This example displays the Office Assistant moves it to the upper-left region of the screen.

```
With Assistant
    .Visible = True
    .Move xLeft:=100, yTop:=100
End With
```

CommandBars Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCommandBarsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCommandBarsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCommandBarsA "}

Returns a **CommandBars** collection that represents the menu bar and all the toolbars in Word. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

CommandBars Property Example

This example enlarges all command bar buttons and enables ToolTips.

```
With CommandBars
    .LargeButtons = True
    .DisplayTooltips = True
End With
```

This example displays the **Drawing** toolbar at the bottom of the application window.

```
With CommandBars("Drawing")
    .Visible = True
    .Position = msoBarBottom
End With
```

BuiltInDocumentProperties Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBuiltInDocumentPropertiesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBuiltInDocumentPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBuiltInDocumentPropertiesA "}

Returns a **DocumentProperties** collection that represents all the built-in document properties for the specified document. Read-only.

Note If Microsoft Word doesn't define a value for one of the built-in document properties, reading the **Value** property for that document property generates an error.

For information about returning a single member of a collection, see Returning an Object from a Collection.

BuiltInDocumentProperties Property Example

This example inserts a list of built-in properties at the end of the active document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
For Each prop In ActiveDocument.BuiltinDocumentProperties
    With myRange
        .InsertParagraphAfter
        .InsertAfter prop.Name & "= "
        On Error Resume Next
        .InsertAfter prop.Value
    End With
Next
```

This example displays the number of words in the active document.

```
wrds = ActiveDocument.BuiltInDocumentProperties(wdPropertyWords)
MsgBox "This document contains " & wrds & " words."
```

CustomDocumentProperties Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCustomDocumentPropertiesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCustomDocumentPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCustomDocumentPropertiesA "}

Returns a **DocumentProperties** collection that represents all the custom document properties for the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

CustomDocumentProperties Property Example

This example inserts a list of custom built-in properties at the end of the active document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
For Each prop In ActiveDocument.CustomDocumentProperties
    With myRange
        .InsertParagraphAfter
        .InsertAfter prop.Name & "= "
        .InsertAfter prop.Value
    End With
Next
```

This example adds a custom built-in property to Sales.doc.

```
thename = InputBox("Please type your name", "Name")
Documents("Sales.doc").CustomDocumentProperties.Add _
    Name:="YourName", LinkToContent:=False, Value:=thename, _
        Type:=msoPropertyTypeString
```

FileSearch Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFileSearchC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFileSearchX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFileSearchA"}

Returns a **FileSearch** object that can be used to search for files using either an absolute or relative path. Read-only.

FileSearch Property Example

This example displays, in a series of message boxes, the file names in the My Documents folder that begin with 97.

```
With Application.FileSearch
    .FileName = "97*.*"
    .LookIn = "C:\My Documents"
    .Execute
    For I = 1 to .FoundFiles.Count
        MsgBox .FoundFiles(I)
    Next I
End With
```

AutoUpdate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoUpdateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoUpdateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoUpdateA "}

True if the specified link is updated automatically when the container file is opened or when the source file is changed. Read/write **Boolean**.

AutoUpdate Property Example

This example updates any shapes in the active document that are linked OLE objects if Word isn't set to update links automatically.

```
For Each s In ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
        If s.LinkFormat.AutoUpdate = False Then s.LinkFormat.Update
    End If
Next s
```

This example updates any fields in the active document that aren't updated automatically.

```
For Each afield In ActiveDocument.Fields
    If afield.LinkFormat.AutoUpdate = False Then afield.LinkFormat.Update
Next afield
```

BreakLink Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthBreakLinkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthBreakLinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthBreakLinkA "}

Breaks the link between the source file and the specified OLE object, picture, or linked field.

Note After you use this method, the link result won't be automatically updated if the source file is changed.

Syntax

expression.**BreakLink**

expression Required. An expression that returns a **LinkFormat** object.

BreakLink Method Example

This example updates and then breaks the links to any shapes that are linked OLE objects in the active document.

```
For Each s In ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
        s.LinkFormat.Update
        s.LinkFormat.BreakLink
    End If
Next s
```

ClassName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproClassNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproClassNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproClassNameA "}

Returns a unique name that identifies the file converter. Read-only **String**.

ClassName Property Example

This example displays the class name and format name of the first converter in the **FileConverters** collection.

```
MsgBox "ClassName= " & FileConverters(1).ClassName & vbCr & _
    "FormatName= " & FileConverters(1).FormatName
```

If an HTML file converter is available, this example sets the HTML format as the default save format.

```
For Each aFile In FileConverters
    If aFile.ClassName = "HTML" Then Application.DefaultSaveFormat = "HTML"
Next aFile
```

DisplayAsIcon Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayAsIconC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayAsIconX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayAsIconA "}

True if the specified object is displayed as an icon. Read/write **Boolean**.

DisplayAsIcon Property Example

This example displays a message box containing the name of each floating shape that's displayed as an icon on the active document.

```
For Each s In ActiveDocument.Shapes
    If s.OLEFormat.DisplayAsIcon Then
        MsgBox s.Name & " is displayed as an icon."
    End If
Next s
```

This example inserts a Microsoft Excel worksheet as a linked OLE object on the active document and then changes the display of the object to an icon.

```
Set myObj = ActiveDocument.Shapes.AddOLEObject _
    (FileName:="c:\program files\microsoft office\office\examples\
samples.xls", LinkToFile:=True)
myObj.OLEFormat.DisplayAsIcon = True
```

Edit Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditA "}

Opens the specified OLE object for editing in the application it was created in.

Syntax

expression.**Edit**

expression Required. An expression that returns an **OLEFormat** object.

Edit Method Example

This example opens (for editing) the first embedded OLE object (defined as a shape) on the active document.

```
Set myS = ActiveDocument.Shapes
If myS.Count >= 1 Then
    If myS(1).Type = msoEmbeddedOLEObject Then
        myS(1).OLEFormat.Edit
    End If
End If
```

This example opens (for editing) the first linked OLE object (defined as an inline shape) in the active document.

```
Set myIS = ActiveDocument.InlineShapes
If myIS.Count >= 1 Then
    If myIS(1).Type = wdInlineShapeLinkedOLEObject Then
        myIS(1).OLEFormat.Edit
    End If
End If
```

IconName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIconNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIconNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIconNameA "}

Returns or sets the program file in which the icon for an OLE object is stored. Read/write **String**.

Note On the Macintosh, you cannot change icons for embedded objects, and Word ignores this property.

IconName Property Example

This example changes the first shape in the selection to be displayed as an icon and sets the text below the icon to the icon's file name.

```
If Selection.ShapeRange.Count >= 1 Then
    Set myOLEF = Selection.ShapeRange(1).OLEFormat
    With myOLEF
        .DisplayAsIcon = True
        .IconLabel = .IconName
    End With
End If
```

IconPath Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIconPathC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIconPathX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIconPathA "}

Returns the path of the file in which the icon for an OLE object is stored. Read-only.

IconPath Property Example

This example displays the path for each floating embedded OLE object that's displayed as an icon on the active document.

```
For each s in ActiveDocument.Shapes
    If s.Type = msoEmbeddedOLEObject Then
        If s.OLEFormat.DisplayAsIcon = True Then Msgbox
s.OLEFormat.IconPath
    End If
Next s
```

IconIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIconIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIconIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIconIndexA "}

Returns or sets the icon that's used when the **DisplayAsIcon** property is **True**: 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. Read/write **Long**.

Note On the Macintosh, you cannot change icons for embedded objects, and Word ignores this property.

IconIndex Property Example

This example returns the icon index number in a message box for the first selected shape that's displayed as an icon.

```
If Selection.ShapeRange.Count >= 1 Then
    Set myOLEF = Selection.ShapeRange(1).OLEFormat
    With myOLEF
        If .DisplayAsIcon = True Then Msgbox .IconIndex
    End With
End If
```

IconLabel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIconLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIconLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIconLabelA "}

Returns or sets the text displayed below the icon for an OLE object. Read/write **String**.

IconLabel Property Example

This example changes the text below the icon for the first shape in the selection.

```
If Selection.ShapeRange.Count >= 1 Then
    Set myOLEF = Selection.ShapeRange(1).OLEFormat
    With myOLEF
        .DisplayAsIcon = True
        .IconLabel = "My Icon"
    End With
End If
```

Label Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLabelA "}

Returns a string that's used to identify the portion of the source file that's being linked. For example, if the source file is a Microsoft Excel workbook, the **Label** property might return "MyWorkbook! R3C1:R4C2" if the OLE object contains only a few cells from the worksheet. Read-only **String**.

Note This property works only for shapes, inline shapes, or fields that are linked OLE objects.

Label Property Example

This example returns the label for the first field in the active document.

```
MsgBox ActiveDocument.Fields(1).OLEFormat.Label
```

OLEFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOLEFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOLEFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOLEFormatA "}

Returns an **OLEFormat** object that represents the OLE characteristics (other than linking) for the specified shape, inline shape, or field. Read-only.

OLEFormat Property Example

This example loops through all the floating shapes on the active document and sets all linked Microsoft Excel worksheets to be updated automatically.

```
For Each s In ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
        If s.OLEFormat.ProgID = "Excel.Sheet.8" Then
            s.LinkFormat.AutoUpdate = True
        End If
    End If
Next
```

DoVerb Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDoVerbC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDoVerbX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDoVerbA "}

Requests that an OLE object peform one of its available verbs – the actions an OLE object takes to activate its contents. Each OLE object supports a set of verbs that pertain to that object.

Syntax

expression.**DoVerb(*VerbIndex*)**

expression Required. An expression that returns an **OLEFormat** object.

VerbIndex Optional **Variant**. The verb that the OLE object should perform. If this argument is omitted, the default verb is sent. Can be one of the **WdOLEVerb** constants.

Constant	Description
wdOLEVerbPrimary	Performs the verb that is invoked when the user double-clicks the object.
wdOLEVerbShow	Shows the object to the user for editing or viewing. Use it to show a newly inserted object for initial editing.
wdOLEVerbOpen	Opens the object in a separate window.
wdOLEVerbHide	Removes the object's user interface from view.
wdOLEVerbUIActivate	Activates the object in place and displays any user-interface tools that the object needs, such as menus or toolbars.
wdOLEVerbInPlaceActivate	Runs the object and installs its window, but doesn't install any user-interface tools.
wdOLEVerbDiscardUndoState	Forces the object to discard any undo state that it might be maintaining; note that the object remains active, however.

DoVerb Method Example

This example sends the default verb to the server for the first floating OLE object on the active document.

```
ActiveDocument.Shapes(1).OLEFormat.DoVerb
```

ActivateAs Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthActivateAsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthActivateAsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthActivateAsA "}

Sets the registery key that determines the default application used to activate the specified OLE object.

Syntax

expression.**ActivateAs**(***ClassType***)

expression Required. An expression that returns an **OLEFormat** object.

ClassType Required **String**. The name of the application in which an OLE object is opened. To see a list of object types that the OLE object can be activated as, click the object and then open the **Convert** dialog box (**Edit** menu, **Object** submenu). You can find the ***ClassType*** string by inserting an object as an inline shape and then viewing the field codes. The class type of the object follows either the word "EMBED" or the word "LINK."

ActivateAs Method Example

This example sets the first floating shape on the active document to open in Microsoft Excel, and then it activates the shape. For the example to work, this shape must be an OLE object that can be opened in Microsoft Excel.

```
With ActiveDocument.Shapes(1).OLEFormat
    .ActivateAs ClassType:="Excel.Sheet.8"
    .Activate
End With
```

ConvertTo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToA "}

Converts the specified OLE object from one class to another, making it possible for you to edit the object in a different server application, or changing how the object is displayed in the document.

Syntax

expression.**ConvertTo(*ClassType, DisplayAsIcon, IconFileName, IconIndex, IconLabel*)**

expression Required. An expression that returns an **OLEFormat** object.

ClassType Optional **Variant**. The name of the application used to activate the OLE object. You can see a list of the available applications in the **Object type** box on the **Create New** tab in the **Object** dialog box (**Insert** menu). You can find the ***ClassType*** string by inserting an object as an inline shape and then viewing the field codes. The class type of the object follows either the word "EMBED" or the word "LINK."

DisplayAsIcon Optional **Variant**. **True** to display the OLE object as an icon. The default value is **False**.

IconFileName Optional **Variant**. The file that contains the icon to be displayed.

IconIndex Optional **Variant**. The index number 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 (**Insert** menu, **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 **Variant**. A label (caption) to be displayed beneath the icon.

ConvertTo Method Example

This example creates a new document, then inserts an embedded Word document with some text. Then, the embedded document is converted to a Word Picture.

```
Documents.Add
Set myEmbedded = ActiveDocument.Shapes.AddOLEObject ClassType:=
"Word.Document.8"
myEmbedded.Activate
Selection.TypeText "Test"
myEmbedded.OLEFormat.OLEFormat.ConvertTo ClassType:="Word.Picture.8"
```

ClassType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproClassTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproClassTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproClassTypeA"}

Returns or sets the class type for the specified OLE object, picture, or field. Read/write **String**.

Remarks

This property is read-only for linked objects other than DDE links.

You can see a list of the available applications in the **Object type** box on the **Create New** tab in the **Object** dialog box (**Insert** menu). You can find the ***ClassType*** string by inserting an object as an inline shape and then viewing the field codes. The class type of the object follows either the word "EMBED" or the word "LINK."

ClassType Property Example

This example loops through all the floating shapes on the active document and sets all linked Microsoft Excel worksheets to be updated automatically.

```
For Each s In ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
        If s.OLEFormat.ClassType = "Excel.Sheet.8" Then
            s.LinkFormat.AutoUpdate = True
        End If
    End If
Next
```

Object Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproObjectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproObjectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproObjectA"}

Returns an object that represents the specified OLE object's top-level interface. This property allows you to access the properties and methods of an ActiveX control or the application in which an OLE object was created. Read-only **Object**.

Object Property Example

This example sets the value of the first shape on the active document. For the example to work, this first shape must be an ActiveX control (a check box or an option button).

```
With ActiveDocument.Shapes(1).OLEFormat
    .Activate
    Set myObj = .Object
End With
myObj.Value = True
```

This example adds a new ActiveX control to the active document. The example then activates the new option button and sets some of its properties.

```
Set myOB =
ActiveDocument.Shapes.AddOLEControl(ClassType:="Forms.OptionButton.1")
With myOB.OLEFormat
    .Activate
    Set myObj = .Object
End With
With myObj
    .Value = False
    .Caption = "My Caption"
    .AutoSize = True
End With
```

ProgID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProgIDC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProgIDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProgIDA"}

Returns the programmatic identifier (ProgID) for the specified OLE object. Read-only **String**.

Remarks

The **ProgID** and **ClassType** properties will (by default) return the same string. However, you can change the **ClassType** property for DDE links.

For information about programmatci identifiers, see OLE Programmatic Identifiers.

ProgID Property Example

This example loops through all the floating shapes in the active document and sets all linked Microsoft Excel worksheets to be updated automatically.

```
For Each s In ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
        If s.OLEFormat.ProgID = "Excel.Sheet.8" Then
            s.LinkFormat.AutoUpdate = True
        End If
    End If
Next
```

programmatic identifier (ProgID)

An identifier in the form *OLEServerName*.*ObjectName* (for example, Excel.Sheet or PowerPoint.Slide) that's used by the system registry to uniquely identify an object. To specify an object created in a specific version of an application, append a period and the version number to the identifier – for example, Excel.Sheet.8 and PowerPoint.Slide.8.

SavePictureWithDocument Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSavePictureWithDocumentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSavePictureWithDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSavePictureWithDocumentA"}

True if the specified picture is saved with the document. Read/write **Boolean**.

Note This property works only with shapes and inline shapes that are linked pictures.

SavePictureWithDocument Property Example

This example saves the linked picture that's defined as the first inline shape in the active document when the document is saved.

```
Set myPic = ActiveDocument.InlineShapes(1)
If myPic.Type = wdInlineShapeLinkedPicture Then
    myPic.LinkFormat.SavePictureWithDocument = True
End If
```

SourceFullName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSourceFullNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSourceFullNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSourceFullNameA"}

Returns or sets the name and path of the source file for the specified linked OLE object, picture, or field. Read/write **String**.

SourceFullName Property Example

This example sets MyExcel.xls as the source file for shape one on the active document and specifies that the OLE object be updated automatically.

```
With ActiveDocument.Shapes(1)
    If .Type = msoLinkedOLEObject Then
        With .LinkFormat
            .SourceFullName = "c:\my documents\myExcel.xls"
            .AutoUpdate = True
        End With
    End If
End With
```

SourceName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSourceNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSourceNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSourceNameA"}

Returns the name of the source file for the specified linked OLE object, picture, or field. Read-only **String**.

Note This property doesn't return the path for the source file.

SourceName Property Example

This example returns the path and name of the source file for any shapes on the active document that are linked OLE objects.

```
For each s in ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
         Msgbox s.LinkFormat.SourcePath & "\" & s.LinkFormat.SourceName
    End If
Next s
```

SourcePath Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSourcePathC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSourcePathX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSourcePathA"}

Returns the path of the source file for the specified linked OLE object, picture, or field. Read-only **String**.

Note This property doesn't return the name of the source file.

SourcePath Property Example

This example returns the path and name of the source file for any shapes on the active document that are linked OLE objects.

```
For each s in ActiveDocument.Shapes
    If s.Type = msoLinkedOLEObject Then
         Msgbox s.LinkFormat.SourcePath & "\" & s.LinkFormat.SourceName
    End If
Next s
```

AddOLEObject Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddOLEObjectC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddOLEObjectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddOLEObjectA"}

Shapes object: Creates an OLE object. Returns the **Shape** object that represents the new OLE object.

InlineShapes object: Creates an OLE object. Returns the **InlineShape** object that represents the new OLE object.

Syntax 1

expression.**AddOLEObject(*ClassType, FileName, LinkToFile, DisplayAsIcon, IconFileName, IconIndex, IconLabel, Left, Top, Width, Height, Anchor*)**

Syntax 2

expression.**AddOLEObject(*ClassType, FileName, LinkToFile, DisplayAsIcon, IconFileName, IconIndex, IconLabel, Range*)**

expression Syntax 1: Required. An expression that returns a **Shapes** object.

Syntax 2: Required. An expression that returns an **InlineShapes** object.

ClassType Optional **String**. The name of the application used to activate the specified OLE object. You can see a list of the available applications in the **Object type** box on the **Create New** tab in the **Object** dialog box (**Insert** menu). You can find the ***ClassType*** string by inserting an object as an inline shape and then viewing the field codes. The class type of the object follows either the word "EMBED" or the word "LINK." You must specify either the ***ClassType*** or ***FileName*** argument for the object, but not both.

For information about available class types, see OLE Programmatic Identifiers.

FileName Optional **Variant**. The file from which the object is to be created. If this argument is omitted, the current folder is used. You must specify either the ***ClassType*** or ***FileName*** argument for the object, but not both.

LinkToFile Optional **Variant**. **True** to link the OLE object to the file from which it was created. **False** to make the OLE object an independent copy of the file. If you specified a value for ***ClassType***, the ***LinkToFile*** argument must be **False**. The default value is **False**.

DisplayAsIcon Optional **Variant**. **True** to display the OLE object as an icon. The default value is **False**.

IconFileName Optional **Variant**. The file that contains the icon to be displayed.

IconIndex Optional **Variant**. The index number 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 (**Insert** menu, **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 **Variant**. A label (caption) to be displayed beneath the icon.

Left, ***Top*** Optional **Variant**. The position (in points) of the upper-left corner of the new object relative to the anchor.

Width, ***Height*** Optional **Variant**. The width and height of the OLE object, in points.

Anchor Optional **Variant**. The range to which the OLE object is bound. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph of the anchoring range. If ***Anchor*** is not specified, the anchor is placed automatically and the OLE Object is positioned relative to the top and left edges of the page.

Range Optional **Variant**. The range where the OLE object will be placed in the text. The OLE object replaces the range, unless the range is collapsed. If this argument is omitted, the object is placed automatically.

AddOLEObject Method Example

This example adds a new floating bitmap image to the active document. The bitmap is linked to another file.

```
ActiveDocument.Shapes.AddOLEObject _
    FileName:="c:\my documents\MyDrawing.bmp", _
    LinkToFile:=True
```

This example adds a new Microsoft Excel worksheet to the active document. The worksheet will replace of the second paragraph.

```
ActiveDocument.InlineShapes.AddOLEObject _
    ClassType:="Excel.Sheet.8", DisplayAsIcon:=False, _
    Range:=ActiveDocument.Paragraphs(2).Range
```

AddPicture Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddPictureC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddPictureX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddPictureA"}

Shapes object: Creates a picture from an existing file. Returns the **Shape** object that represents the new picture.

InlineShapes object: Creates a picture from an existing file. Returns the **InlineShape** object that represents the new picture.

Syntax 1

expression.**AddPicture(*FileName, LinkToFile, SaveWithDocument, Left, Top, Width, Height, Anchor*)**

Syntax 2

expression.**AddPicture(*FileName, LinkToFile, SaveWithDocument, Range*)**

expression Syntax 1: Required. An expression that returns a **Shapes** object.

Syntax 2: Required. An expression that returns an **InlineShapes** object.

FileName Required **String**. The file from which the object is to be created. If this argument is omitted, the current folder is used.

LinkToFile Optional **Variant**. **True** to link the picture to the file from which it was created. **False** to make the picture an independent copy of the file. The default value is **False**.

SaveWithDocument Optional **Variant**. **True** to save the linked picture with the document. The default value is **False**.

Left, ***Top*** Optional **Variant**. The position (in points) of the upper-left corner of the new picture relative to the anchor.

Width, ***Height*** Optional **Variant**. The width and height of the picture, in points.

Anchor Optional **Variant**. The range to which the picture is bound. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, however, the anchor is placed automatically and the picture is positioned relative to the top and left edges of the page.

Range Optional **Variant**. The range where the picture will be placed in the text. The picture replaces the range, if the range isn't collapsed. If this argument is omitted, the picture is placed automatically.

AddPicture Method Example

This example adds a new picture to `myDoc`. The inserted picture is linked to the file from which it was created and is saved with the document.

```
Set myDoc = ActiveDocument
ActiveDocument.Shapes.AddPicture "c:\program files\microsoft office\
clipart\music.bmp", True, True
```

AddOLEControl Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddOLEControlC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddOLEControlX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddOLEControlA"}

Shapes object: Creates an ActiveX control. Returns the **Shape** object that represents the new ActiveX control.

InlineShapes object: Creates an ActiveX control. Returns the **InlineShape** object that represents the new ActiveX control.

Syntax 1

expression.**AddOLEControl(*ClassType, Left, Top, Width, Height, Anchor*)**

Syntax 2

expression.**AddOLEControl(*ClassType, Range*)**

expression Syntax 1: Required. An expression that returns a **Shapes** object.

Syntax 2: Required. An expression that returns an **InlineShapes** object.

ClassType Optional **String**. The OLE long class name or the programmatic identifier for the ActiveX control to be created.

For information about available class types, see OLE Programmatic Identifiers

Left, ***Top*** Optional **Variant**. The position (in points) of the upper-left corner of the new object relative to the anchor.

Width, ***Height*** Optional **Variant**. The width and height of the ActiveX control, in points.

Anchor Optional **Variant**. The range to which the ActiveX control is bound. If ***Anchor*** is specified, the anchor is positioned at the beginning of the first paragraph in the anchoring range. If this argument is omitted, however, the anchor is placed automatically and the ActiveX control is positioned relative to the top and left edges of the page.

Range Optional **Variant**. The range where the ActiveX control will be placed in the text. The ActiveX control replaces the range, if the range isn't collapsed. If this argument is omitted, the Active X control is placed automatically.

Remarks

ActiveX controls are represented as either **Shape** objects or **InlineShape** objects in Word. To modify the properties for an ActiveX control, you use the **Object** property of the **OLEFormat** object for the specified shape or inline shape.

AddOLEControl Method Example

This example adds a check box to the active document.

```
ActiveDocument.Shapes.AddOLEControl ClassType:="Forms.CheckBox.1"
```

This example adds a combo box to the active document.

```
ActiveDocument.Shapes.AddOLEControl ClassType:="Forms.ComboBox.1"
```

This example adds a check box to the active document, clears the check box, and then adds a caption for it.

```
Set myCB =
ActiveDocument.Shapes.AddOLEControl(ClassType:="Forms.CheckBox.1")
With myCB.OLEFormat.Object
    .Value = False
    .Caption = "Check if over 21"
End With
```

Duplicate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddOLEControlC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddOLEControlX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddOLEControlA"}

Creates a duplicate of the specified **Shape** or **ShapeRange** object, adds the new range of shapes to the **Shapes** collection at a standard offset from the original shapes, and then returns the new **Shape** object.

Syntax

expression.**Duplicate**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Duplicate Method Example

This example creates a duplicate of shape one on the active document and then changes the fill for the new shape.

```
Set newShape = ActiveDocument.Shapes(1).Duplicate
With newShape
    .Fill.PresetGradient msoGradientVertical, 1, msoGradientGold
End With
```

Shape, **InlineShape**, **Field**

LinkFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLinkFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLinkFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLinkFormatM"}

LinkFormat

Represents the linking characteristics for an OLE object or picture.

Using the LinkFormat Object

Use the **LinkFormat** property for a shape, inline shape, or field to return the **LinkFormat** object. The following example breaks the link for the first shape on the active document.

```
ActiveDocument.Shapes(1).LinkFormat.BreakLink
```

Remarks

Not all types of shapes, inline shapes, and fields can be linked to a source. Use the **Type** property for the **Shape** and **InlineShape** objects to determine whether a particular shape can be linked. The **Type** property for a **Field** object returns the type of field.

You can use both the **Update** method and the **AutoUpdate** property to update links. To return or set the full path for a particular link's source file, use the **SourceFullName** property.

OLEFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjOLEFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjOLEFormatP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjOLEFormatM"}

OLEFormat

Represents the OLE characteristics (other than linking) for an OLE object, ActiveX control, or field.

Using the OLEFormat Object

Use the **OLEFormat** property for a shape, inline shape, or field to return the **OLEFormat** object. The following example displays the class type for the first shape on the active document.

```
MsgBox ActiveDocument.Shapes(1).OLEFormat.ClassType
```

Remarks

Not all types of shapes, inline shapes, and fields have OLE capabilities. Use the **Type** property for the **Shape** and **InlineShape** objects to determine what category the specified shape or inline shape falls into. The **Type** property for a **Field** object returns the type of field.

You can use the **Activate**, **Edit**, **Open**, and **DoVerb** methods to automate an OLE object.

Use the **Object** property to return an object that represents an ActiveX control or OLE object. With this object, you can use the properties and methods of the container application or the ActiveX control.

Shape, **InlineShape**, **Field**

AllowAccentedUppercase Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAllowAccentedUppercaseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAllowAccentedUppercaseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAllowAccentedUppercaseA "}

True if accents are retained when a French language character is changed to uppercase. Read/write **Boolean**.

Remarks

This property affects only text that's been marked as standard French. For all other languages, accents are always retained even if the **AllowAccentedUppercase** property is set to **False**.

If you change a character back to lowercase after an accent mark has been stripped from it, the accent won't reappear.

AllowAccentedUppercase Property Example

This example sets Word to remove accent marks when characters in French text are changed to uppercase.

```
Options.AllowAccentedUppercase = False
```

This example returns the status of the **Allow accented uppercase in French** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.AllowAccentedUppercase
```

AutoWordSelection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoWordSelectionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoWordSelectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoWordSelectionA "}

True if dragging selects one word at a time instead of one character at a time. Read/write **Boolean**.

AutoWordSelection Property Example

This example sets Word to select individual characters instead of entire words when you select by dragging.

```
Options.AutoWordSelection = False
```

This example returns the status of the **When selecting, automatically select entire word** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.AutoWordSelection
```

AllowDragAndDrop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAllowDragAndDropC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAllowDragAndDropX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAllowDragAndDropA "}

True if dragging and dropping can be used to move or copy a selection. Read/write **Boolean**.

AllowDragAndDrop Property Example

This example turns on the drag-and-drop editing feature.

```
Options.AllowDragAndDrop = True
```

This example returns the status the **Drag-and-Drop text editing** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.AllowDragAndDrop
```

INSKeyForPaste Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproINSKeyForPasteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproINSKeyForPasteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproINSKeyForPasteA "}

True if the INS key can be used for pasting the Clipboard contents. Read/write **Boolean**.

INSKeyForPaste Property Example

This example enables the INS key to be used for pasting the contents of the Clipboard.

```
Options.INSKeyForPaste = True
```

This example returns the status of the **Use the INS key for paste** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.INSKeyForPaste
```

PictureEditor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPictureEditorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPictureEditorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPictureEditorA "}

Returns or sets the name of the application to use to edit pictures. Read/write **String**.

Remarks

You must use the exact wording displayed in the **Picture editor** box on the **Edit** tab in the **Options** dialog box. Otherwise, the default setting "Microsoft Word" is used.

If the name of your graphics application doesn't appear in the list, contact the manufacturer of the graphics application for instructions.

PictureEditor Property Example

This example sets the application to use to edit pictures.

```
Options.PictureEditor = "Microsoft Imager 3.0 Picture"
```

This example returns the name of the application to use to edit pictures.

```
MsgBox Options.PictureEditor
```

ReplaceSelection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReplaceSelectionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReplaceSelectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReplaceSelectionA "}

True if the result of typing or pasting replaces the selection. **False** if the result of typing or pasting is added before the selection, leaving the selection intact. Read/write **Boolean**.

ReplaceSelection Property Example

This example sets Word to leave the add the result of typing or pasting before the selection, leaving the selection intact.

```
Options.ReplaceSelection = False
```

This example returns the status of the **Typing replaces selection** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.ReplaceSelection
```

SmartCutPaste Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSmartCutPasteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSmartCutPasteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSmartCutPasteA "}

True if Word automatically adjusts the spacing between words and punctuation when cutting and pasting occurs. Read/write **Boolean**.

SmartCutPaste Property Example

This example sets Word to automatically adjust the spacing between words and punctuation when cutting and pasting occurs, and then it cuts and pastes some text in a newly created document. If the **SmartCutPaste** property were set to **False**, the second and third words would run together.

```
Options.SmartCutPaste = True
Set myDoc = Documents.Add
With myDoc
    .Content.InsertAfter("The brown quick fox")
    .Words(2).Cut
    .Characters(10).Paste
End With
```

This example returns the status of the **Use smart cut and paste** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.SmartCutPaste
```

TabIndentKey Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTabIndentKeyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTabIndentKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTabIndentKeyA "}

True if the TAB and BACKSPACE keys can be used to increase and decrease the left indent of paragraphs, respectively, and if the BACKSPACE key can be used to change right-aligned paragraphs to centered paragraphs and centered paragraphs to left-aligned paragraphs. Read/write **Boolean**.

TabIndentKey Property Example

This example sets Word so that the TAB and BACKSPACE keys set the left indent, instead of inserting and deleting tabs.

```
Options.TabIndentKey = True
```

This example returns the status of the **Tabs and Backspace set left indent** option on the **Edit** tab in the **Options** dialog box.

```
temp = Options.TabIndentKey
```

DisplayScrollBars Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayScrollBarsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayScrollBarsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayScrollBarsA "}

True if Word displays a scroll bar in at least one document window. **False** if there are no scroll bars displayed in any window. Read/write **Boolean**.

Remarks

Setting the **DisplayScrollBars** property to **True** displays horizontal and vertical scroll bars in all windows. Setting this property to **False** turns off all scroll bars in all windows.

Use the **DisplayHorizontalScrollBar** and **DisplayVerticalScrollBar** properties to display individual scroll bars in the specified window.

DisplayScrollBars Property Example

This example displays horizontal and vertical scroll bars in all windows.

```
Application.DisplayScrollBars = True
```

This example returns **True** if there's a scroll bar currently displayed in any window.

```
temp = Application.DisplayScrollBars
```

AnimateScreenMovements Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAnimateScreenMovementsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAnimateScreenMovementsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAnimateScreenMovementsA"}

True if Word animates mouse movements, uses animated cursors, and animates actions such as background saving and find and replace operations. Read/write **Boolean**.

AnimateScreenMovements Property Example

This example sets Word to animate movements on the screen.

```
Options.AnimateScreenMovements = True
```

This example returns the current status of the **Provide feedback with animation** option on the **General** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.AnimateScreenMovements
```

AutoFormatAsYouTypeApplyTables Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeApplyTablesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeApplyTablesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeApplyTablesA"}

True if Word automatically creates a table when you type a plus sign, a series of hyphens, another plus sign, and so on, and then press ENTER. The plus signs become the column borders, and the hyphens become the column widths. Read/write **Boolean**.

AutoFormatAsYouTypeApplyTables Example

This example sets Word to automatically create tables as you type.

```
Options.AutoFormatAsYouTypeApplyTables = True
```

This example returns the status of the **Tables** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeApplyTables
```

AutoFormatAsYouTypeFormatListItemBeginning Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeFormatListItemBeginningC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeFormatListItemBeginningX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeFormatListItemBeginningA"}

True if Word repeats character formatting applied to the beginning of a list item to the next list item. Read/write **Boolean**.

AutoFormatAsYouTypeFormatListItemBeginning Example

This example sets Word to automatically repeat character formatting at the beginning of list items.

```
Options.AutoFormatAsYouTypeFormatListItemBeginning = True
```

This example returns the status of the **Format beginning of list item like the one before it** option in the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Options** menu).

```
temp = Options.AutoFormatAsYouTypeFormatListItemBeginning
```

DefaultBorderColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultBorderColorIndexC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultBorderColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultBorderColorIndexA"}

Returns or sets the default line color for borders. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto	**wdGray50**
wdBlack	**wdGreen**
wdBlue	**wdPink**
wdBrightGreen	**wdRed**
wdDarkBlue	**wdTeal**
wdDarkRed	**wdTurquoise**
wdDarkYellow	**wdViolet**
wdGray25	**wdWhite**
	wdYellow

Note If the **Enable** property of the **Borders** object is set to **True**, the default line width, line style, and line color for borders are used.

DefaultBorderColorIndex Property Example

This example changes the default line color and style for borders and then applies a border around the first paragraph in the active document.

```
With Options
    .DefaultBorderColorIndex = wdRed
    .DefaultBorderLineStyle = wdLineStyleDouble
End With
ActiveDocument.Paragraphs(1).Borders.Enable = True
```

GridDistanceHorizontal Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGridDistanceHorizontalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGridDistanceHorizontalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGridDistanceHorizontalA"}

Returns or sets the amount of horizontal space between the invisible gridlines that Word uses when you draw, move, and resize AutoShapes. Read/write **Single**.

GridDistanceHorizontal Property Example

This example sets the horizontal and vertical distance between gridlines and then enables the **Snap to grid** feature.

```
With Options
    .GridDistanceHorizontal = InchesToPoints(0.2)
    .GridDistanceVertical = InchesToPoints(0.2)
    .SnapToGrid = True
End With
```

GridDistanceVertical Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGridDistanceVerticalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGridDistanceVerticalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGridDistanceVerticalA"}

Returns or sets the amount of vertical space between the invisible gridlines that Word uses when you draw, move, and resize AutoShapes. Read/write **Single**.

GridDistanceVertical Property Example

This example sets the horizontal and vertical distance between gridlines and then enables the **Snap to grid** feature.

```
With Options
    .GridDistanceHorizontal = InchesToPoints(0.2)
    .GridDistanceVertical = InchesToPoints(0.2)
    .SnapToGrid = True
End With
```

GridOriginHorizontal Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGridOriginHorizontalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGridOriginHorizontalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGridOriginHorizontalA"}

Returns or sets the point, relative to the left edge of the page, where you want the invisible grid for drawing, moving, and resizing AutoShapes to begin. Read/write **Single**.

GridOriginHorizontal Property Example

This example sets the horizontal and vertical point of origin for the grid, sets the horizontal and vertical distance between gridlines, and then enables the **Snap to grid** feature.

```
With Options
    .GridOriginHorizontal = InchesToPoints(1)
    .GridOriginVertical = InchesToPoints(2)
    .GridDistanceHorizontal = InchesToPoints(0.1)
    .GridDistanceVertical = InchesToPoints(0.1)
    .SnapToGrid = True
End With
```

GridOriginVertical Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGridOriginVerticalC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGridOriginVerticalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGridOriginVerticalA"}

Returns or sets the point, relative to the top of the page, where you want the invisible grid for drawing, moving, and resizing AutoShapes to begin. Read/write **Single**.

GridOriginVertical Property Example

This example sets the horizontal and vertical point of origin for the grid, sets the horizontal and vertical distance between gridlines, and then enables the **Snap to grid** feature.

```
With Options
    .GridOriginHorizontal = InchesToPoints(1)
    .GridOriginVertical = InchesToPoints(2)
    .GridDistanceHorizontal = InchesToPoints(0.2)
    .GridDistanceVertical = InchesToPoints(0.2)
    .SnapToGrid = True
End With
```

MapPaperSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMapPaperSizeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMapPaperSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMapPaperSizeA"}

True if documents formatted for another country's standard paper size (for example, A4) are automatically adjusted so that they're printed correctly on your country's standard paper size (for example, Letter). Read/write **Boolean**.

Remarks

This property affects only the printout of your document; its formatting is left unchanged.

MapPaperSize Property Example

This example allows Word to adjust paper size according to the country setting.

```
Options.MapPaperSize = True
```

This example returns the status of the **Allow A4/Letter paper resizing** option on the **Print** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.MapPaperSize
```

SnapToGrid Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSnapToGridC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSnapToGridX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSnapToGridA"}

True if AutoShapes are automatically aligned with an invisible grid when they are drawn, moved, or resized. Read/write **Boolean**.

Remarks

You can temporarily override this setting by pressing ALT while drawing, moving, or resizing an AutoShape.

SnapToGrid Property Example

This example sets Word so that AutoShapes are automatically aligned with the invisible grid.

```
Options.SnapToGrid = True
```

This example returns the status of the **Snap to grid** option in the **Snap to Grid** dialog box (**Drawing** toolbar, **Draw** menu, **Grid** command).

```
Temp = Options.SnapToGrid
```

SnapToShapes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSnapToShapesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSnapToShapesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSnapToShapesA"}

True if Word automatically aligns AutoShapes with invisible gridlines that go through the vertical and horizontal edges of other AutoShapes. Read/write **Boolean**.

Remarks

This property creates additional invisible gridlines for each AutoShape. **SnapToShapes** works independently of the **SnapToGrid** property.

SnapToShapes Property Example

This example sets Word to automatically align AutoShapes with invisible gridlines that go through the vertical and horizontal edges of other AutoShapes.

```
Options.SnapToShapes = TrueSct
```

This example returns the status of the **Snap to shapes** option in the **Snap to Grid** dialog box (**Drawing** toolbar, **Draw** menu, **Grid** command).

```
Temp = Options.SnapToShapes
```

VirusProtection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVirusProtectionC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVirusProtectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVirusProtectionA"}

True if Word displays a built-in warning message whenever you open a document or template that might contain macro viruses. Read/write **Boolean**.

Remarks

Any document that contains macros or customized toolbars, menus, or shortcuts will automatically display this warning. You can then decide whether to open the document with or without its macros.

VirusProtection Property Example

This example sets Word to automatically display a warning message whenever you open a document that might contain macro viruses.

```
Options.VirusProtection = True
```

This example returns the current status of the **Enable macro virus protection** option on the **General** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.VirusProtection
```

BlueScreen Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBlueScreenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBlueScreenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBlueScreenA "}

True if Word displays text as white characters on a blue background. Read/write **Boolean**.

BlueScreen Property Example

This example asks users whether they want white text on a blue background and presents Yes and No buttons for their response.

```
If MsgBox("Do you want white on blue?", 36, "BlueScreen?") = vbYes Then
    Options.BlueScreen = True
Else
    Options.BlueScreen = False
End If
```

ButtonFieldClicks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproButtonFieldClicksC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproButtonFieldClicksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproButtonFieldClicksA "}

Returns or sets the number of clicks (either one or two) required to run a GOTOBUTTON or MACROBUTTON field. Read/write **Long**.

ButtonFieldClicks Property Example

This example sets the number of clicks required to run a MACROBUTTON or GOTOBUTTON field to one.

```
Options.ButtonFieldClicks = 1
```

ConfirmConversions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproConfirmConversionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproConfirmConversionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproConfirmConversionsA "}

True if Word displays the **Convert File** dialog box before it opens or inserts a file that isn't a Word document or template. In the **Convert File** dialog box, the user chooses the format to convert the file from. Read/write **Boolean**.

ConfirmConversions Property Example

This example sets Word to display the **Convert File** dialog box whenever a file that isn't a Word document or template is opened.

```
Options.ConfirmConversions = True
```

This example returns the current status of the **Confirm conversion at Open** option on the **General** tab in the **Options** dialog box.

```
temp = Options.ConfirmConversions
```

Pagination Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPaginationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPaginationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPaginationA "}

True if Word repaginates documents in the background. Read/write **Boolean**.

Pagination Property Example

This example sets Word to perform background repagination.

```
Options.Pagination = True
```

This example returns the current status of the **Background repagination** option on the **General** tab in the **Options** dialog box.

```
temp = Options.Pagination
```

RTFInClipboard Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRTFInClipboardC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRTFInClipboardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRTFInClipboardA "}

Macintosh only. **True** if all text copied from Word to the Clipboard retains its character and paragraph formatting. Read/write **Boolean**.

RTFInClipboard Property Example

This example preserves the character and paragraph formatting of text copied from Word to the Clipboard.

```
Options.RTFInClipboard = True
```

ShortMenuNames Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShortMenuNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShortMenuNamesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShortMenuNamesA "}

Macintosh only. **True** if Word displays command names in abbreviated form on menus. For example, Word displays "Ins" instead of "Insert," "Fmt" instead of "Format," and "Wnd" instead of "Window." Read/write **Boolean**.

ShortMenuNames Property Example

This example spells out Macintosh command names on menus in full, rather than abbreviating them.

```
Options.ShortMenuNames = False
```

MeasurementUnit Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMeasurementUnitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMeasurementUnitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMeasurementUnitA "}

Returns or sets the standard measurement unit for Word. Can be one of the following **WdMeasurementUnits** constants: **wdCentimeters**, **wdInches**, **wdPicas**, or **wdPoints**. Read/write **Long**.

MeasurementUnit Property Example

This example sets thestandard measurement unit for Word to points.

```
Options.MeasurementUnit = wdPoints
```

This example returns the current measurement unit selected on the **General** tab in the **Options** dialog box.

```
CurrUnit = Options.MeasurementUnit
```

UpdateLinksAtOpen Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUpdateLinksAtOpenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUpdateLinksAtOpenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUpdateLinksAtOpenA "}

True if Word automatically updates all embedded OLE links in a document when it's opened. Read/write **Boolean**.

UpdateLinksAtOpen Property Example

This example sets Word to update embedded OLE links when it opens files.

```
Options.UpdateLinksAtOpen = True
```

This example returns the current status of the **Update automatic links at Open** option on the **General** tab in the **Options** dialog box.

```
temp = Options.UpdateLinksAtOpen
```

WPDocNavKeys Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWPDocNavKeysC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWPDocNavKeysX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWPDocNavKeysA "}

True if navigation keys for WordPerfect users are enabled. Read/write **Boolean**.

WPDocNavKeys Property Example

This example sets Word to use WordPerfect navigation keys.

```
Options.WPDocNavKeys = True
```

This example returns the status of the **Navigation keys for WordPerfect users** option on the **General** tab in the **Options** dialog box.

```
temp = Options.WPDocNavKeys
```

WPHelp Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWPHelpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWPHelpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWPHelpA "}

True if pressing Word key combinations that produce actions in WordPerfect displays dialog boxes that describe how to perform the equivalent actions in Word. Read/write **Boolean**.

WPHelp Property Example

This example toggles WordPerfect help.

```
Options.WPHelp = Not Options.WPHelp
```

This example displays the status of the **Help for WordPerfect users** option on the **General** tab in the **Options** dialog box.

```
Msgbox Options.WPHelp
```

PrintComments Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintCommentsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintCommentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintCommentsA "}

True if Word prints comments, starting on a new page at the end of the document. Read/write **Boolean**.

Remarks

Setting the **PrintComments** property to **True** automatically sets the **PrintHiddenText** property to **True**. However, setting the **PrintComments** property to **False** has no effect on the setting of the **PrintHiddenText** property.

PrintComments Property Example

This example sets Word to print comments, and then it prints the active document.

```
Options.PrintComments = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Comments** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintComments
```

PrintBackground Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintBackgroundC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintBackgroundX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintBackgroundA "}

True if Word prints in the background. Read/write **Boolean**.

PrintBackground Property Example

This example sets Word to print documents in the background, and then it prints the active document.

```
Options.PrintBackground = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Background printing** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintBackground
```

PrintDraft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintDraftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintDraftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintDraftA "}

True if Word prints using minimal formatting. Read/write **Boolean**.

Note Not all printers support draft printing.

PrintDraft Property Example

This example sets Word to use draft printing, and then it prints the active document.

```
Options.PrintDraft = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Draft output** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintDraft
```

PrintDrawingObjects Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintDrawingObjectsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintDrawingObjectsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintDrawingObjectsA "}

True if Word prints drawing objects. Read/write **Boolean**.

PrintDrawingObjects Property Example

This example sets Word to print drawing objects, and then it prints the active document.

```
Options.PrintDrawingObjects = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Drawing objects** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintDrawingObjects
```

EnvelopeFeederInstalled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEnvelopeFeederInstalledC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEnvelopeFeederInstalledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEnvelopeFeederInstalledA "}

True if the current printer has a special feeder for envelopes. Read-only **Boolean**.

EnvelopeFeederInstalled Property Example

This example prints the active document as an envelope, provided that there's an envelope feeder installed.

```
If Options.EnvelopeFeederInstalled = True Then
    ActiveDocument.Envelope.PrintOut _
        AddressFromLeft:=InchesToPoints(3), _
        AddressFromTop:=InchesToPoints(1.5)
Else
    Msgbox "No envelope feeder available."
End If
```

PrintFormsData Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintFormsDataC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintFormsDataX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintFormsDataA "}

True if Word prints onto a preprinted form only the data entered in the corresponding online form. Read/write **Boolean**.

PrintFormsData Property Example

This example sets Word to print only the data from an online form, and then it prints the active document.

```
ActiveDocument.PrintFormsData = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Print data only for forms** on the **Print** tab in the **Options** dialog box.

```
temp = ActiveDocument.PrintFormsData
```

PrintReverse Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintReverseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintReverseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintReverseA "}

True if Word prints pages in reverse order. Read/write **Boolean**.

PrintReverse Property Example

This example sets Word to print pages in reverse order, and then it prints the active document.

```
Options.PrintReverse = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Reverse print order** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintReverse
```

PrintFieldCodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintFieldCodesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintFieldCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintFieldCodesA "}

True if Word prints field codes instead of field results. Read/write **Boolean**.

PrintFieldCodes Property Example

This example sets Word to print field codes, and then it prints the active document.

```
Options.PrintFieldCodes = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Field codes** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintFieldCodes
```

PrintHiddenText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintHiddenTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintHiddenTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintHiddenTextA "}

True if hidden text is printed. Read/write **Boolean**.

Remarks

Setting the **PrintHiddenText** property to **False** automatically sets the **PrintComments** property to **False**. However, setting the **PrintHiddenText** property to **True** has no effect on the setting of the **PrintComments** property.

PrintHiddenText Property Example

This example sets Word to print hidden text, and then it prints the active document.

```
Options.PrintHiddenText = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Hidden text** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintHiddenText
```

UpdateFieldsAtPrint Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUpdateFieldsAtPrintC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUpdateFieldsAtPrintX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUpdateFieldsAtPrintA "}

True if Word updates fields automatically before printing a document. Read/write **Boolean**.

UpdateFieldsAtPrint Property Example

This example sets Word to update fields automatically before printing, and then it prints the active document.

```
Options.UpdateFieldsAtPrint = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Update fields** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.UpdateFieldsAtPrint
```

UpdateLinksAtPrint Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUpdateLinksAtPrintC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUpdateLinksAtPrintX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUpdateLinksAtPrintA "}

True if Word updates embedded links to other files before printing a document. Read/write **Boolean**.

UpdateLinksAtPrint Property Example

This example sets Word to update embedded links automatically before printing, and then it prints the active document.

```
Options.UpdateLinksAtPrint = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Update links** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.UpdateLinksAtPrint
```

DeletedTextColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDeletedTextColorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDeletedTextColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDeletedTextColorA "}

Returns or sets the color of text that is deleted while change tracking is enabled. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto	**wdGreen**
wdBlack	**wdNoHighlight**
wdBlue	**wdPink**
wdBrightGreen	**wdRed**
wdByAuthor	**wdTeal**
wdDarkBlue	**wdTurquoise**
wdDarkRed	**wdViolet**
wdDarkYellow	**wdWhite**
wdGray25	**wdYellow**
wdGray50	

Remarks

If the **DeletedTextColor** property is set to **wdByAuthor**, Word automatically assigns a unique color to each of the first eight authors who revise a document.

DeletedTextColor Property Example

This example sets the color of deleted text to bright green.

```
Options.DeletedTextColor = wdBrightGreen
```

This example returns the current status of the **Color** option under **Deleted Text** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.DeletedTextColor
```

DeletedTextMark Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDeletedTextMarkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDeletedTextMarkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDeletedTextMarkA "}

Returns or sets the format of text that is deleted while change tracking is enabled. Can be one of the following **WdDeletedTextMark** constants: **wdDeletedTextMarkCaret**, **wdDeletedTextMarkHidden**, **wdDeletedTextMarkPound**, or **wdDeletedTextMarkStrikeThrough**. Read/write **Long**.

DeletedTextMark Property Example

This example applies strikethrough formatting to deleted text.

```
Options.DeletedTextMark = wdDeletedTextMarkStrikeThrough
```

This example returns the current status of the **Mark** option under **Deleted Text** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.DeletedTextMark
```

InsertedTextColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsertedTextColorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsertedTextColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsertedTextColorA "}

Returns or sets the color of text that is inserted while change tracking is enabled. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto
wdBlack
wdBlue
wdBrightGreen
wdByAuthor
wdDarkBlue
wdDarkRed
wdDarkYellow
wdGray25
wdGray50
wdGreen
wdNoHighlight
wdPink
wdRed
wdTeal
wdTurquoise
wdViolet
wdWhite
wdYellow

Remarks

If the **InsertedTextColor** property is set to **wdByAuthor**, Word automatically assigns a unique color to each of the first eight authors who revise a document.

InsertedTextColor Property Example

This example sets the color of inserted text to dark red.

```
Options.InsertedTextColor = wdDarkRed
```

This example returns the current status of the **Color** option under **Inserted Text** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.InsertedTextColor
```

InsertedTextMark Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInsertedTextMarkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInsertedTextMarkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInsertedTextMarkA "}

Returns or sets the formatting of text that is inserted while change tracking is enabled. Can be one of the following **WdInsertedTextMark** constants: **wdInsertedTextMarkNone**, **wdInsertedTextMarkBold**, **wdInsertedTextMarkDoubleUnderline**, **wdInsertedTextMarkItalic**, or **wdInsertedTextMarkUnderline**. Read/write **Long**.

InsertedTextMark Property Example

This example applies double-underline formatting to inserted text.

```
Options.InsertedTextMark = wdInsertedTextMarkDoubleUnderline
```

This example returns the current status of the **Mark** option under **Inserted Text** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.InsertedTextMark
```

RevisedLinesColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRevisedLinesColorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRevisedLinesColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRevisedLinesColorA "}

Returns or sets the color of changed lines in a document with tracked changes. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto
wdBlack
wdBlue
wdBrightGreen
wdByAuthor
wdDarkBlue
wdDarkRed
wdDarkYellow
wdGray25
wdGray50
wdGreen
wdNoHighlight
wdPink
wdRed
wdTeal
wdTurquoise
wdViolet
wdWhite
wdYellow

RevisedLinesColor Property Example

This example sets the color of changed lines to pink.

```
Options.RevisedLinesColor = wdPink
```

This example returns the current status of the **Color** option under **Chnaged Lines** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.RevisedLinesColor
```

RevisedLinesMark Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRevisedLinesMarkC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRevisedLinesMarkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRevisedLinesMarkA "}

Returns or sets the placement of changed lines in a document with tracked changes. Can be one of the following **WdRevisedLinesMark** constants: **wdRevisedLinesMarkNone**, **wdRevisedLinesMarkLeftBorder**, **wdRevisedLinesMarkOutsideBorder**, or **wdRevisedLinesMarkRightBorder**. Read/write **Long**.

RevisedLinesMark Property Example

This example sets changed lines to appear in the left margin of every page.

```
Options.RevisedLinesMark = wdRevisedLinesMarkLeftBorder
```

This example returns the current status of the **Mark** option under **Changed Lines** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.RevisedLinesMark
```

BackgroundSave Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBackgroundSaveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBackgroundSaveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBackgroundSaveA "}

True if Word saves documents in the background. When Word is saving in the background, users can continue to type and to choose commands. Read/write **Boolean**.

BackgroundSave Property Example

This example allows users to continue working in a document while Word is saving it.

```
Options.BackgroundSave = True
```

This example returns the current status of the **Allow background saves** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.BackgroundSave
```

CreateBackup Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCreateBackupC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCreateBackupX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCreateBackupA "}

True if Word creates a backup copy each time a document is saved. Read/write **Boolean**.

Remarks

The **CreateBackup** and **AllowFastSave** properties cannot be set to **True** concurrently.

CreateBackup Property Example

This example sets Word to automatically create a backup copy, and then it saves the active document.

```
Options.CreateBackup = True
ActiveDocument.Save
```

This example returns the current status of the **Always create backup copy** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.CreateBackup
```

EmbedTrueTypeFonts Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEmbedTrueTypeFontsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEmbedTrueTypeFontsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEmbedTrueTypeFontsA "}

True if Word embeds TrueType fonts in a document when it's saved. This allow others to view the document with the same fonts that were used to create it. Read/write **Boolean**.

EmbedTrueTypeFonts Property Example

This example sets Word to automatically embed TrueType fonts when saving a document, and then it saves the active document.

```
ActiveDocument.EmbedTrueTypeFonts = True
ActiveDocument.Save
```

This example returns the current status of the **Embed TrueType fonts** option on the **Save** tab in the **Options** dialog box.

```
temp = ActiveDocument.EmbedTrueTypeFonts
```

AllowFastSave Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAllowFastSaveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAllowFastSaveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAllowFastSaveA "}

True if Word saves only changes to a document. When reopening the document, Word uses the saved changes to reconstruct the document. Read/write **Boolean**.

Remarks

The **AllowFastSave** and **CreateBackup** properties cannot be set to **True** concurrently.

AllowFastSave Property Example

This example sets Word to save the complete document, and then it saves the active document.

```
Options.AllowFastSave = False
ActiveDocument.Save
```

This example returns the current status of the **Allow fast saves** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.AllowFastSave
```

SaveFormsData Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSaveFormsDataC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSaveFormsDataX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSaveFormsDataA "}

True if Word saves the data entered in a form as a tab-delimited record for use in a database. Read/write **Boolean**.

SaveFormsData Property Example

This example sets Word to save only the data entered in a form

```
ActiveDocument.SaveFormsData = True
```

This example returns the current status of the **Save data only for forms** option on the **Save** tab in the **Options** dialog box.

```
temp = ActiveDocument.SaveFormsData
```

SaveNormalPrompt Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSaveNormalPromptC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSaveNormalPromptX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSaveNormalPromptA "}

True if Word prompts the user for confirmation to save changes to the Normal template before it quits. **False** if Word automatically saves changes to the Normal template before it quits. Read/write **Boolean**.

SaveNormalPrompt Property Example

This example sets Word to save the Normal template automatically before quitting, and then it quits.

```
Options.SaveNormalPrompt = False
Application.Quit
```

This example returns the current status of the **Prompt to save Normal template** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.SaveNormalPrompt
```

ReadOnlyRecommended Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReadOnlyRecommendedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReadOnlyRecommendedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReadOnlyRecommendedA "}

True if Word displays a message box whenever a user opens the document, suggesting that it be opened as read-only. Read/write **Boolean**.

ReadOnlyRecommended Property Example

This example sets Word to suggest, when it's opening the document, that the document be opened as read-only.

```
ActiveDocument.ReadOnlyRecommended = True
```

This example returns the current status of the **Read-only recommended** option on the **Save** tab in the **Options** dialog box.

```
temp = ActiveDocument.ReadOnlyRecommended
```

SaveInterval Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSaveIntervalC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSaveIntervalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSaveIntervalA "}

Returns or sets the time interval for saving AutoRecover information, in minutes. Read/write **Long**.

Remarks

Set the **SaveInterval** property to 0 (zero) to turn off saving AutoRecover information.

SaveInterval Property Example

This example sets Word to save AutoRecover information for all open documents every five minutes.

```
Options.SaveInterval = 5
```

This example prevents Word from saving AutoRecover information.

```
Options.SaveInterval = 0
```

This example returns the current status of the **Save AutoRecover info every** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.SaveInterval
```

SaveSubsetFonts Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSaveSubsetFontsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSaveSubsetFontsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSaveSubsetFontsA "}

True if Word saves a subset of the embedded TrueType fonts with the document. Read/write **Boolean**.

Remarks

If fewer than 32 characters of a TrueType font are used in a document, Word embeds the subset (only the characters used) in the document. If more than 32 characters are used, Word embeds the entire font.

SaveSubsetFonts Property Example

This example sets a document named "MyDoc" to save only a subset of its embedded TrueType fonts (when just a few characters are used), and then it saves "MyDoc."

```
With Documents("MyDoc")
    .EmbedTrueTypeFonts = True
    .SaveSubsetFonts = True
    .Save
End With
```

SavePropertiesPrompt Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSavePropertiesPromptC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSavePropertiesPromptX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSavePropertiesPromptA "}

True if Word prompts for document property information when saving a new document. Read/write **Boolean**.

SavePropertiesPrompt Property Example

This example causes Word to prompt for document property information when saving a new document.

```
Options.SavePropertiesPrompt = True
```

This example returns the current status of the **Prompt for document properties** option on the **Save** tab in the **Options** dialog box.

```
temp = Options.SavePropertiesPrompt
```

UserAddress Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUserAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUserAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUserAddressA "}

Returns or sets the user's mailing address. Read/write **String**.

Remarks

The mailing address is used as a return address on envelopes.

UserAddress Property Example

This example sets the user's return address. The **Chr** function is used to return a line feed character.

```
Application.UserAddress = "4200 Third Street NE" & Chr(10) _
    & "Anytown, WA  98999"
```

This example returns the address found in the **Mailing address** option on the **User Information** tab in the **Options** dialog box.

```
Msgbox Application.UserAddress
```

UserInitials Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUserInitialsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUserInitialsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUserInitialsA "}

Returns or sets the user's initials, which Word uses to construct comment marks. Read/write **String**.

UserInitials Property Example

This example sets the user's initials.

```
Application.UserInitials = "baa"
```

This example returns the letters found in the **Initials** option on the **User Info** tab in the **Options** dialog box.

```
Msgbox Application.UserInitials
```

UserName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUserNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUserNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUserNameA "}

Returns or sets the user's name, which is used on envelopes and for the Author document property. Read/write **String**.

UserName Property Example

This example sets the user's name.

```
Application.UserName = "Andrew Fuller"
```

This example returns the name found in the **Name** option on the **User Info** tab in the **Options** dialog box.

```
Msgbox Application.UserName
```

AutoFormatAsYouTypeApplyBorders Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeApplyBordersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeApplyBordersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeApplyBordersA "}

True if a series of three or more hyphens (-), equal signs (=), or underscore characters (_) are automatically replaced by a specific border line when the ENTER key is pressed. Read/write **Boolean**.

Remarks

Hyphens (-) are replaced by a 0.75-point line, equal signs (=) are replaced by a 0.75-point double line, and underscore characters (_) are replaced by a 1.5-point line.

AutoFormatAsYouTypeApplyBorders Property Example

This example causes sequences of three or more hyphens (-), equal signs (=), or underscore characters (_) to be transformed into borders.

```
Options.AutoFormatAsYouTypeApplyBorders = True
```

This example returns the current setting for the **Borders** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
MsgBox Options.AutoFormatAsYouTypeApplyBorders
```

AutoFormatAsYouTypeApplyBulletedLists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeApplyBulletedListsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeApplyBulletedListsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeApplyBulletedListsA "}

True if bullet characters (such as asterisks, hyphens, and greater-than signs) are replaced with bullets from the **Bullets And Numbering** dialog box (**Format** menu) as you type. Read/write **Boolean**.

AutoFormatAsYouTypeApplyBulletedLists Property Example

This example causes characters to be replaced with bullets when typed in a list.

```
Options.AutoFormatAsYouTypeApplyBulletedLists = True
```

This example returns the status of the **Automatic bulleted lists** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeApplyBulletedLists
```

AutoFormatApplyBulletedLists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatApplyBulletedListsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatApplyBulletedListsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatApplyBulletedListsA "}

True if characters (such as asterisks, hyphens, and greater-than signs) at the beginning of list paragraphs are replaced with bullets from the **Bullets and Numbering** dialog box (**Format** menu) when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatApplyBulletedLists Property Example

This example replaces any characters used at the beginning of list paragraphs in the current selection with bullets.

```
Options.AutoFormatApplyBulletedLists = True
Selection.Range.AutoFormat
```

This example returns the status of the **Automatic bulleted lists** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (Tools menu).

```
temp = Options.AutoFormatApplyBulletedLists
```

AutoFormatAsYouTypeApplyNumberedLists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeApplyNumberedListsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeApplyNumberedListsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeApplyNumberedListsA "}

True if paragraphs are automatically formatted as numbered lists with a numbering scheme from the **Bullets and Numbering** dialog box (**Format** menu), according to what's typed. For example, if a paragraph starts with "1.1" and a tab character, Word automatically inserts "1.2" and a tab character after the ENTER key is pressed. Read/write **Boolean**.

AutoFormatAsYouTypeApplyNumberedLists Property Example

This example causes lists to be automatically numbered as you type.

```
Options.AutoFormatAsYouTypeApplyNumberedLists = True
```

This example returns the status of the **Automatic numbered lists** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeApplyNumberedLists
```

AutoFormatApplyHeadings Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatApplyHeadingsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatApplyHeadingsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatApplyHeadingsA "}

True if styles are automatically applied to headings when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatApplyHeadings Property Example

This example applies the Heading 1 through Heading 9 styles to headings in the current selection.

```
Options.AutoFormatApplyHeadings = True
Selection.Range.AutoFormat
```

This example returns the status of the **Headings** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatApplyHeadings
```

AutoFormatApplyLists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatApplyListsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatApplyListsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatApplyListsA "}

True if styles are automatically applied to lists when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatApplyLists Property Example

This example applies styles to any lists in the current selection.

```
Options.AutoFormatApplyLists = True
Selection.Range.AutoFormat
```

This example returns the status of the **Lists** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatApplyLists
```

AutoFormatApplyOtherParas Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatApplyOtherParasC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatApplyOtherParasX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatApplyOtherParasA "}

True if styles are automatically applied to paragraphs that aren't headings or list items when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatApplyOtherParas Property Example

This example automatically applies styles to paragraphs in the current selection.

```
Options.AutoFormatApplyOtherParas = True
Selection.Range.AutoFormat
```

This example returns the status of the **Other paragraphs** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatApplyOtherParas
```

AutoFormatPreserveStyles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatPreserveStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatPreserveStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatPreserveStylesA "}

True if previously applied styles are preserved when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatPreserveStyles Property Example

This example sets Word to preserve existing styles and to format headings, lists, and other paragraphs with styles when formatting automatically. Word then formats the current selection automatically.

```
With Options
    .AutoFormatPreserveStyles = True
    .AutoFormatApplyHeadings = True
    .AutoFormatApplyLists = True
    .AutoFormatApplyOtherParas = True
End With
Selection.Range.AutoFormat
```

This example returns the status of the **Styles** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatPreserveStyles
```

AutoFormatReplaceFractions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplaceFractionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplaceFractionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplaceFractionsA "}

True if typed fractions are replaced with fractions from the current character set when Word formats a document or range automatically. For example, "1/2" is replaced with "½." Read/write **Boolean**.

AutoFormatReplaceFractions Property Example

This example turns on the replacement of typed fractions, and thenit formats the current selection automatically.

```
Options.AutoFormatReplaceFractions = True
Selection.Range.AutoFormat
```

This example returns the status of the **Fractions (1/2) with fraction character (½)** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceFractions
```

AutoFormatReplaceOrdinals Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplaceOrdinalsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplaceOrdinalsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplaceOrdinalsA "}

True if the ordinal number suffixes "st", "nd", "rd", and "th" are replaced with the same letters in superscript when Word formats a document or range automatically. For example, "1st" is replaced with "1st." Read/write **Boolean**.

AutoFormatReplaceOrdinals Property Example

This example turns on the automatic replacement of ordinals with superscript, and then it formats the current selection automatically.

```
Options.AutoFormatReplaceOrdinals = True
Selection.Range.AutoFormat
```

This example returns the status of the **Ordinals (1st) with superscript** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceOrdinals
```

AutoFormatReplaceQuotes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplaceQuotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplaceQuotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplaceQuotesA "}

True if straight quotation marks are automatically changed to smart (curly) quotation marks when Word formats a document or range automatically. Read/write **Boolean**.

AutoFormatReplaceQuotes Property Example

This example turns on the automatic replacement of straight quotation marks with smart (curly) quotation marks, and then it formats the current selection automatically.

```
Options.AutoFormatReplaceQuotes = True
Selection.Range.AutoFormat
```

This example returns the status of the **Straight quotes with smart quotes** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceQuotes
```

AutoFormatAsYouTypeApplyHeadings Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeApplyHeadingsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeApplyHeadingsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeApplyHeadingsA "}

True if styles are automatically applied to headings as you type. Read/write **Boolean**.

AutoFormatAsYouTypeApplyHeadings Property Example

This example sets Word to automatically apply the Heading1 through Heading 9 styles to headings as you type.

```
Options.AutoFormatAsYouTypeApplyHeadings = True
```

This example returns the status of the **Headings** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeApplyHeadings
```

AutoFormatAsYouTypeReplaceFractions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplaceFractionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplaceFractionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplaceFractionsA "}

True if typed fractions are replaced with fractions from the current character set as you type. For example, "1/2" is replaced with "½." Read/write **Boolean**.

AutoFormatAsYouTypeReplaceFractions Property Example

This example turns off the automatic replacement of typed fractions.

```
Options.AutoFormatAsYouTypeReplaceFractions = False
```

This example returns the status of the **Fractions (1/2) with fraction character (½)** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeReplaceFractions
```

AutoFormatAsYouTypeReplaceOrdinals Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplaceOrdinalsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplaceOrdinalsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplaceOrdinalsA "}

True if the ordinal number suffixes "st", "nd", "rd", and "th" are replaced with the same letters in superscript as you type. For example, "1st" is replaced with "1st." Read/write **Boolean**.

AutoFormatAsYouTypeReplaceOrdinals Property Example

This example turns on the automatic replacement of ordinals with superscript letters.

```
Options.AutoFormatAsYouTypeReplaceOrdinals = True
```

This example returns the status of the **Ordinals (1st) with superscript** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeReplaceOrdinals
```

AutoFormatAsYouTypeReplaceQuotes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplaceQuotesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplaceQuotesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplaceQuotesA "}

True if straight quotation marks are automatically changed to smart (curly) quotation marks as you type. Read/write **Boolean**.

AutoFormatAsYouTypeReplaceQuotes Property Example

This example turns on the automatic replacement of straight quotation marks with smart (curly) quotation marks as you type.

```
Options.AutoFormatAsYouTypeReplaceQuotes = True
```

This example returns the status of the **Straight quotes with smart quotes** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceQuotes
```

AutoFormatAsYouTypeReplacePlainTextEmphasis Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplacePlainTextEmphasisC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplacePlainTextEmphasisX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplacePlainTextEmphasisA "}

True if manual emphasis characters are automatically replaced with character formatting as you type. For example, "*bold*" is changed to "**bold**" and "_underline_" is changed to "underline." Read/write **Boolean**.

AutoFormatAsYouTypeReplacePlainTextEmphasis Property Example

This example turns on the replacement of manual emphasis characters with character formatting.

```
Options.AutoFormatAsYouTypeReplacePlainTextEmphasis = True
```

This example returns the status of the ***Bold* and _underline_ with real formatting** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeReplacePlainTextEmphasis
```

AutoFormatReplacePlainTextEmphasis Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplacePlainTextEmphasisC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplacePlainTextEmphasisX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplacePlainTextEmphasisA "}

True if manual emphasis characters are replaced with character formatting when Word formats a document or range automatically. For example, "*bold*" is changed to "**bold**" and "_underline_" is changed to "underline." Read/write **Boolean**.

AutoFormatReplacePlainTextEmphasis Property Example

This example turns on the replacement of manual emphasis characters with character formatting

```
Options.AutoFormatReplacePlainTextEmphasis = True
Selection.Range.AutoFormat
```

This example returns the status of the ***Bold* and _underline_ with real formatting** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplacePlainTextEmphasis
```

AutoFormatAsYouTypeDefineStyles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeDefineStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeDefineStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeDefineStylesA "}

True if Word automatically creates new styles based on manual formatting. Read/write **Boolean**.

AutoFormatAsYouTypeDefineStyles Property Example

This example sets Word to automatically create styles as you type.

```
Options.AutoFormatAsYouTypeDefineStyles = True
```

This example returns the status of the **Define styles based on your formatting** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeDefineStyles
```

DisplayAlerts Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayAlertsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayAlertsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayAlertsA "}

Returns or sets the way certain alerts and messages are handled while a macro is running. Read/write **Long**.

Can be one of the following **WdAlertLevel** constants.

Constant	Description
wdAlertsNone	No alerts or message boxes are displayed. If a macro encounters a message box, the default value is chosen and the macro continues.
wdAlertsMessageBox	Only message boxes are displayed; errors are trapped and returned to the macro.
wdAlertsAll	All message boxes and alerts are displayed; errors are returned to the macro.

Note If you set this property to **wdAlertsNone** or **wdAlertsMessageBox**, Word doesn't set it back to **wdAlertsAll** when your macro stops running. You should write your macro in such a way that it always sets the **DisplayAlerts** property back to **wdAlertsAll** when it stops running.

DisplayAlerts Property Example

This example sets Word to display all alerts and message boxes when it's running macros.

```
Application.DisplayAlerts = wdAlertsAll
```

This example returns the current setting of the **DisplayAlerts** property.

```
temp = Application.DisplayAlerts
```

DisplayAutoCompleteTips Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayAutoCompleteTipsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayAutoCompleteTipsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayAutoCompleteTipsA "}

True if Word displays tips that suggest text for completing words, dates, or phrases as you type. Read/write **Boolean**.

DisplayAutoCompleteTips Property Example

This example sets Word to display tips that suggest text for completing words, dates, or phrases as you type.

```
Application.DisplayAutoCompleteTips = True
```

This example returns the status of the **Suggest the rest of the word or date with a tip as you type** option on the **AutoText** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Application.DisplayAutoCompleteTips
```

SetWPHelpOptions Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetWPHelpOptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetWPHelpOptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetWPHelpOptionsA "}

Sets the options for the WordPerfect Help feature.

Note On the Macintosh, this feature isn't available and the method generates an error.

Syntax

expression.**SetWPHelpOptions(*CommandKeyHelp, DocNavigationKeys, MouseSimulation, DemoGuidance, DemoSpeed, HelpType*)**

expression Required. An expression that returns an **Options** object.

CommandKeyHelp Optional **Variant**. **True** to display instructions or demonstrate a Word equivalent when a WordPerfect® for DOS key combination is pressed. WordPerfect Help is displayed in the status bar.

DocNavigationKeys Optional **Variant**. **True** to make the arrow keys and the PAGE UP, PAGE DOWN, HOME, END, and ESC keys function as they would in WordPerfect.

MouseSimulation Optional **Variant**. **True** to have the pointer move to each option that WordPerfect Help selects during demonstrations.

DemoGuidance Optional **Variant**. **True** to display help text when user input is required during WordPerfect Help demonstrations.

DemoSpeed Optional **Variant**. The speed of WordPerfect Help demonstrations. Can be one of the following values.

Value	Speed
0 (zero)	Fast
1	Medium
2	Slow

HelpType Optional **Variant**. Specifies whether WordPerfect Help displays help text or demonstrates the WordPerfect command. Can be either 0 (zero), for Help text, or 1, for a demonstration.

SetWPHelpOptions Method Example

This example sets the WordPerfect Help options.

```
Options.SetWPHelpOptions _
    CommandKeyHelp:=True, _
    DocNavigationKeys:=True, _
    MouseSimulation:=True, _
    DemoGuidance:=True, _
    DemoSpeed:=0, _
    HelpType:=0
```

SuggestSpellingCorrections Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSuggestSpellingCorrectionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSuggestSpellingCorrectionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSuggestSpellingCorrectionsA "}

True if Word always suggests alternative spellings for each misspelled word when checking spelling. Read/write **Boolean**.

SuggestSpellingCorrections Property Example

This example sets Word to always suggest alternative spellings for misspelled words, and then it checks spelling in the active document.

```
Options.SuggestSpellingCorrections = True
ActiveDocument.CheckSpelling
```

This example returns the current status of the **Always suggest corrections** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.SuggestSpellingCorrections
```

ShowSpellingErrors Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowSpellingErrorsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowSpellingErrorsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowSpellingErrorsA "}

True if Word underlines spelling errors in the document. Read/write **Boolean**.

Remarks

To view spelling errors in a document, you must set the **CheckSpellingAsYouType** property to **True**.

ShowSpellingErrors Property Example

This example sets Word to hide the wavy red line that denotes possible spelling errors in the active document.

```
ActiveDocument.ShowSpellingErrors = False
```

This example sets Word to show spelling errors in the active document.

```
Options.CheckSpellingAsYouType = True
ActiveDocument.ShowSpellingErrors = True
```

This example returns the current status of the **Hide spelling errors in this document** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = ActiveDocument.ShowSpellingErrors
```

IgnoreUppercase Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIgnoreUppercaseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIgnoreUppercaseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIgnoreUppercaseA "}

True if words in all uppercase letters are ignored while checking spelling. Read/write **Boolean**.

IgnoreUppercase Property Example

This example sets Word to ignore words in all uppercase letters, and then it checks spelling in the active document.

```
Options.IgnoreUppercase = True
ActiveDocument.CheckSpelling
```

This example returns the current status of the **Ignore words in UPPERCASE** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.IgnoreUppercase
```

IgnoreMixedDigits Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIgnoreMixedDigitsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIgnoreMixedDigitsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIgnoreMixedDigitsA "}

True if words that contain numbers are ignored while checking spelling. Read/write **Boolean**.

IgnoreMixedDigits Property Example

This example sets Word to ignore words that contain numbers, and then it checks spelling in the active document.

```
Options.IgnoreMixedDigits = True
ActiveDocument.CheckSpelling
```

This example returns the current status of the **Ignore words with numbers** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.IgnoreMixedDigits
```

SuggestFromMainDictionaryOnly Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMainDictionaryOnlyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSuggestFromMainDictionaryOnlyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSuggestFromMainDictionaryOnlyA "}

True if Word draws spelling suggestions from the main dictionary only. **False** if it draws spelling suggestions from the main dictionary and any custom dictionaries that have been added. Read/write **Boolean**.

SuggestFromMainDictionaryOnly Property Example

This example sets Word to suggest words from the main dictionary only, and then it checks spelling in the active document.

```
Options.SuggestFromMainDictionaryOnly = True
ActiveDocument.CheckSpelling
```

This example returns the current status of the **Suggest from main dictionary only** option on the **Spelling & Grammar** tab in the **Options** dialog box.

```
temp = Options.SuggestFromMainDictionaryOnly
```

PrintProperties Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintPropertiesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintPropertiesA "}

True if Word prints document summary information on a separate page at the end of the document. **False** if document summary information is not printed. Summary information is found in the **Properties** dialog box (**File** menu). Read/write **Boolean**.

PrintProperties Property Example

This example sets Word to print document summary information on a separate page at the end of the document, and then it prints the active document.

```
Options.PrintProperties = True
ActiveDocument.PrintOut
```

This example returns the current status of the **Document properties** option on the **Print** tab in the **Options** dialog box.

```
temp = Options.PrintProperties
```

DefaultHighlightColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultHighlightColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultHighlightColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultHighlightColorIndexA "}

Returns or sets the color used to highlight text formatted with the **Highlight** button (**Formatting** toolbar). Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdBlack
wdBlue
wdBrightGreen
wdDarkBlue
wdDarkRed
wdDarkYellow
wdGray25
wdGray50
wdGreen
wdNoHighlight
wdPink
wdRed
wdTeal
wdTurquoise
wdViolet
wdWhite
wdYellow

DefaultHighlightColorIndex Property Example

This example sets the default highlight color to bright green. The new color doesn't apply to any previously highlighted text.

```
Options.DefaultHighlightColorIndex = wdBrightGreen
```

This example returns the current default highlight color index.

```
temp = Options.DefaultHighlightColorIndex
```

DefaultTray Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultTrayC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultTrayX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultTrayA "}

Returns or sets the default tray your printer uses to print documents. Read/write **String**.

Remarks

When setting this property, you must specify a string found in the **Default tray** box on the **Print** tab in the **Options** dialog box. You can use the **DefaultTrayID** property and specify a **WdPaperTray** constant to set this same option.

DefaultTray Property Example

This example sets Word up to use the lower print tray.

```
Options.DefaultTray = "Lower tray"
```

This example returns the string found in the **Default tray** box on the **Print** tab in the **Options** dialog box.

```
Msgbox Options.DefaultTray
```

DefaultTrayID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultTrayIdC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultTrayIdX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultTrayIdA "}

Returns or sets the default tray your printer uses to print documents. Read/write **Long**.

Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed	**wdPrinterManualFeed**
wdPrinterDefaultBin	**wdPrinterMiddleBin**
wdPrinterEnvelopeFeed	**wdPrinterOnlyBin**
wdPrinterFormSource	**wdPrinterPaperCassette**
wdPrinterLargeCapacityBin	**wdPrinterSmallFormalBin**
wdPrinterLargeFormatBin	**wdPrinterTractorFeed**
wdPrinterLowerBin	**wdPrinterUpperBin**
wdPrinterManualEnvelopeFeed	

Remarks

You can use the **DefaultTray** property with a string from the **Default tray** box on the **Print** tab in the **Options** dialog box to set this same option.

DefaultTrayID Property Example

This example sets Word to use the upper print tray, and then it prints the active document.

```
Options.DefaultTrayID = wdPrinterUpperBin
ActiveDocument.PrintOut
```

This example returns the current setting of the **Default tray** option on the **Print** tab in the **Options** dialog box.

```
currTray = Options.DefaultTrayID
```

DefaultSaveFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultSaveFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultSaveFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultSaveFormatA "}

Returns or sets the default format that will appear in the **Save as type** box in the **Save As** dialog box (**File** menu). Corresponds to the **Save Word files as** box on the **Save** tab in the **Options** dialog box (**Tools** menu). Read/write **String**.

Remarks

The string used with this property is the file converter class name. The class names for internal Word formats are listed in the following table.

Word format	File converter class name
Word Document	""
Document Template	"Dot"
Text Only	"Text"
Text Only with Line Breaks	"CRText"
MS-DOS Text	"8Text"
MS-DOS Text with Line Breaks	"8CRText"
Rich Text Format	"Rtf"
Unicode Text	"Unicode"

Use the **ClassName** property with a **FileConverter** object to determine the class name of an external file converter.

DefaultSaveFormat Property Example

This example sets the Word document format as the default save format.

```
Application.DefaultSaveFormat = ""
```

This example returns the current setting the **Save Word files as** box on the **Save** tab in the **Options** dialog box (**Tools** menu).

```
Msgbox Application.DefaultSaveFormat
```

AutoFormatAsYouTypeReplaceHyperlinks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatAsYouTypeReplaceHyperlinksC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatAsYouTypeReplaceHyperlinksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatAsYouTypeReplaceHyperlinksA"}

True if e-mail addresses, server and share names (also known as UNC paths), and Internet addresses (also known as URLs) are automatically changed to hyperlinks as you type. Read/write **Boolean**.

Remarks

Word changes any text that looks like an e-mail address, UNC, or URL to a hyperlink. Word doesn't check the validity of the hyperlink.

AutoFormatAsYouTypeReplaceHyperlinks Property Example

This example enables Word to automatically replace any Internet or network paths with hyperlinks when the paths are typed.

```
Options.AutoFormatAsYouTypeReplaceHyperlinks = True
```

This example returns the status of the **Internet and network paths with hyperlinks** option on the **AutoFormat As You Type** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatAsYouTypeReplaceHyperlinks
```

AutoFormatPlainTextWordMail Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatPlainTextWordMailC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatPlainTextWordMailX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatPlainTextWordMailA"}

True if Word automatically formats plain-text WordMail messages when you open them in Word. Read/write **Boolean**.

AutoFormatPlainTextWordMail Property Example

This example sets Word to automatically format any plain-text WordMail messages that are opened.

```
Options.AutoFormatPlainTextWordMail = True
```

This example returns the status of the **Plain text WordMail documents** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatPlainTextWordMail
```

AutoFormatReplaceHyperlinks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatReplaceHyperlinksC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatReplaceHyperlinksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatReplaceHyperlinksA"}

True if e-mail addresses, server and share names (also known as UNC paths), and Internet addresses (also known as URLs) are automatically formatted whenever Word AutoFormats a document or range. Read/write **Boolean**.

Remarks

Word changes any text that looks like an e-mail address, UNC, or URL to a hyperlink. Word doesn't check the validity of the hyperlink.

AutoFormatReplaceHyperlinks Property Example

This example enables replacement of any Internet or network paths with hyperlinks, and then it formats the selection automatically.

```
Options.AutoFormatReplaceHyperlinks = True
Selection.Range.AutoFormat
```

This example returns the status of the **Internet and network paths with hyperlinks** option on the **AutoFormat** tab in the **AutoCorrect** dialog box (**Tools** menu).

```
temp = Options.AutoFormatReplaceHyperlinks
```

RevisedPropertiesColor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRevisedPropertiesColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRevisedPropertiesColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRevisedPropertiesColorA"}

Returns or sets the color used to mark formatting changes while change tracking is enabled. Read/write **Long**.

Can be one of the following **WdColorIndex** constants:

wdAuto
wdBlack
wdBlue
wdBrightGreen
wdByAuthor
wdDarkBlue
wdDarkRed
wdDarkYellow
wdGray25
wdGray50
wdGreen
wdNoHighlight
wdPink
wdRed
wdTeal
wdTurquoise
wdViolet
wdWhite
wdYellow

Remarks

If deleted or inserted text has formatting changes, the **RevisedPropertiesColor** property is overridden by the **DeletedTextColor** or **InsertedTextColor** property.

RevisedPropertiesColor Property Example

This example tracks changes in the active document, sets the color of text with changed formatting to teal, and applies bold formatting to the selection.

```
ActiveDocument.TrackRevisions = True
Options.RevisedPropertiesColor = wdTeal
Selection.Font.Bold = True
```

This example returns the option selected in the **Color** box under **Changed formatting** on the **Track Changes** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.RevisedPropertiesColor
```

RevisedPropertiesMark Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRevisedPropertiesMarkC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRevisedPropertiesMarkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRevisedPropertiesMarkA"}

Returns or sets the mark used to show formatting changes while change tracking is enabled. Can be one of the following **WdRevisedPropertiesMark** constants: **wdRevisedPropertiesMarkBold**, **wdRevisedPropertiesMarkDoubleUnderline**, **wdRevisedPropertiesMarkItalic**, **wdRevisedPropertiesMarkNone**, or **wdRevisedPropertiesMarkUnderline**. Read/write **Long**.

RevisedPropertiesMark Property Example

This example causes text with changed formatting to be double-underlined when change tracking is enabled.

```
Options.RevisedPropertiesMark = wdRevisedPropertiesMarkDoubleUnderline
```

This example returns the option selected in the **Mark** box under **Changed formatting** on the **Track Changes** tab in the **Options** dialog box.

```
temp = Options.RevisedPropertiesColor
```

CloseUp Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCloseUpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCloseUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCloseUpA "}

Removes any spacing before the specified paragraphs.

Syntax

expression.**CloseUp**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Remarks

The following two statements are equivalent:

```
ActiveDocument.Paragraphs(1).CloseUp
ActiveDocument.Paragraphs(1).SpaceBefore = 0
```

CloseUp Method Example

This example removes any space before the first paragraph in the selection.

```
Selection.Paragraphs(1).CloseUp
```

This example changes the Heading 1 style in the active document so that there's no space before Heading 1 paragraphs.

```
ActiveDocument.Styles("Heading 1").ParagraphFormat.CloseUp
```

DefaultTabStop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDefaultTabStopC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDefaultTabStopX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDefaultTabStopA "}

Returns or sets the interval (in points) between the default tab stops in the specified document. Read/write **Single**.

DefaultTabStop Property Example

This example sets the default tab stops in the active document to 1 inch. The **InchesToPoints** method is used to convert inches to points.

```
ActiveDocument.DefaultTabStop = InchesToPoints(1)
```

Hyphenation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHyphenationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHyphenationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHyphenationA "}

True if the specified paragraphs are included in automatic hyphenation. **False** if the specified paragraphs are to be excluded from automatic hyphenation. Can be **True**, **False** or **wdUndefined**. Read/write **Long**.

Hyphenation Property Example

This example turns off automatic hyphenation for all paragraphs in the active document that have the Normal style.

```
ActiveDocument.Styles("Normal").ParagraphFormat.Hyphenation = False
```

FirstLineIndent Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstLineIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstLineIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstLineIndentA "}

Returns or sets the value (in points) for a first line or hanging indent. Use a positive value to set a first-line indent, and use a negative value to set a hanging indent. Read/write **Single**.

FirstLineIndent Property Example

This example sets a first-line indent of 1 inch for the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).FirstLineIndent = _
    InchesToPoints(1)
```

This example sets a hanging indent of 0.5 inch for the second paragraph in the active document. The **InchesToPoints** method is used to convert inches to points.

```
ActiveDocument.Paragraphs(2).FirstLineIndent = _
    InchesToPoints(-0.5)
```

KeepTogether Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeepTogetherC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeepTogetherX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeepTogetherA "}

True if all lines in the specified paragraphs remain on the same page when Word repaginates the document. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

KeepTogether Property Example

This example formats the paragraphs in the active document so that all the lines in each paragraph are on the same page when Word repaginates the document.

```
ActiveDocument.Paragraphs.KeepTogether = True
```

KeepWithNext Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproKeepWithNextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproKeepWithNextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproKeepWithNextA "}

True if the specified paragraph remains on the same page as the paragraph that follows it when Word repaginates the document. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

KeepWithNext Property Example

This example keeps the third paragraph through sixth paragraph in the active document on the same page.

```
For i = 3 To 5
    ActiveDocument.Paragraphs(i).KeepWithNext = True
Next i
```

Leader Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLeaderC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLeaderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLeaderA "}

Returns or sets the leader for the specified **TabStop** object. Can be one of the following **WdTabLeader** constants: **wdTabLeaderDashes**, **wdTabLeaderDots**, **wdTabLeaderHeavy**, **wdTabLeaderLines**, or **wdTabLeaderSpaces**. Read/write **Long**.

Leader Property Example

This example changes the leader for all tab stops with a leader to dashes for all the paragraphs in the active document.

```
For each tb in ActiveDocument.Paragraphs.TabStops
    If tb.Leader <> wdTabLeaderSpaces Then
        tb.Leader = wdTabLeaderDashes
    End If
Next tb
```

LeftIndent Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLeftIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLeftIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLeftIndentA "}

Paragraph, **Paragraphs**, or **ParagraphFormat** object: Returns or sets the left indent value (in points) for the specified paragraphs. Read/write **Single**.

Row or **Rows** object: Returns or sets the distance (in points) between the left page margin and the left edge of the text in the specified rows. Read/write **Single**.

LeftIndent Property Example

This example sets the left indent of the first paragraph in the active document to 1 inch. The **InchesToPoints** method is used to convert inches to points.

```
ActiveDocument.Paragraphs(1).LeftIndent = InchesToPoints(1)
```

This example sets the left indent for all rows in the first table in the active document.

```
ActiveDocument.Tables(1).Rows.LeftIndent = InchesToPoints(1)
```

OpenOrCloseUp Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenOrCloseUpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenOrCloseUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenOrCloseUpA "}

If spacing before the specified paragraphs is 0 (zero), this method sets spacing to 12 points. If spacing before the paragraphs is greater than 0 (zero), this method sets spacing to 0 (zero).

Syntax

expression.**OpenOrCloseUp**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

OpenOrCloseUp Method Example

This example toggles the formatting of the first paragraph in the active document to either add 12 points of space before the paragraph or leave no space before it.

```
ActiveDocument.Paragraphs(1).OpenOrCloseUp
```

OpenUp Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOpenUpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOpenUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOpenUpA "}

Sets spacing before the specified paragraphs to 12 points.

Syntax

expression.**OpenUp**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Remarks

The following two statements are equivalent:

```
ActiveDocument.Paragraphs(1).OpenUp
ActiveDocument.Paragraphs(1).SpaceBefore = 12
```

OpenUp Method Example

This example changes the formatting of the second paragraph in the active document to leave 12 points of space before the paragraph.

```
ActiveDocument.Paragraphs(2).OpenUp
```

PageBreakBefore Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageBreakBeforeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageBreakBeforeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageBreakBeforeA "}

True if a page break is forced before the specified paragraphs. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

PageBreakBefore Property Example

This example forces a page break before the first paragraph in the selection.

```
Selection.Paragraphs(1).PageBreakBefore = True
```

Paragraphs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproParagraphsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproParagraphsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproParagraphsA "}

Returns a **Paragraphs** collection that represents all the paragraphs in the specified document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Paragraphs Property Example

This example sets the line spacing to single for the collection of all paragraphs in section one in the active document.

```
ActiveDocument.Sections(1).Range.Paragraphs.LineSpacingRule = _
wdLineSpaceSingle
```

This example sets the line spacing to double for the first paragraph in the selection.

```
Selection.Paragraphs(1).LineSpacingRule = wdLineSpaceDouble
```

RightIndent Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRightIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRightIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRightIndentA "}

Returns or sets the right indent (in points) for the specified paragraphs. Read/write **Single**.

RightIndent Property Example

This example sets the right indent for all paragraphs in the active document to 1 inch from the right margin. The **InchesToPoints** method is used to convert inches to points.

```
ActiveDocument.Paragraphs.RightIndent = InchesToPoints(1)
```

Space1 Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSpace1C "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSpace1X":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSpace1A "}

Single-spaces the specified paragraphs. The exact spacing is determined by the font size of the largest characters in each paragraph.

Syntax

expression.**Space1**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Remarks

The following two statements are equivalent:

```
ActiveDocument.Paragraphs(1).Space1
ActiveDocument.Paragraphs(1).LineSpacingRule = wdLineSpaceSingle
```

Space1 Method Example

This example changes the first paragraph in the active document to single spacing.

```
ActiveDocument.Paragraphs(1).Space1
```

Space15 Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSpace15C "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSpace15X":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSpace15A "}

Formats the specified paragraphs with 1.5-line spacing. The exact spacing is determined by adding 6 points to the font size of the largest character in each paragraph.

Syntax

expression.**Space15**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Remarks

The following two statements are equivalent:

```
ActiveDocument.Paragraphs(1).Space15
ActiveDocument.Paragraphs(1).LineSpacingRule = wdLineSpace1pt5
```

Space15 Method Example

This example changes the first paragraph in the active document to 1.5-line spacing.

```
ActiveDocument.Paragraphs(1).Space15
```

Space2 Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSpace2C "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSpace2X":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSpace2A "}

Double-spaces the specified paragraphs. The exact spacing is determined by adding 12 points to the font size of the largest character in each paragraph.

Syntax

expression.**Space2**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Remarks

The following two statements are equivalent:

```
ActiveDocument.Paragraphs(1).Space2
ActiveDocument.Paragraphs(1).LineSpacingRule = wdLineSpaceDouble
```

Space2 Method Example

This example changes the first paragraph in the selection to double spacing.

```
Selection.Paragraphs(1).Space2
```

SpaceBefore Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpaceBeforeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpaceBeforeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpaceBeforeA "}

Returns or sets the spacing (in points) before the specified paragraphs. Read/write **Single**.

SpaceBefore Property Example

This example sets the spacing before the second paragraph in the active document to 12 points.

```
ActiveDocument.Paragraphs(2).SpaceBefore = 12
```

TabStops Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTabStopsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTabStopsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTabStopsA "}

Returns or sets a **TabStops** collection that represents all the custom tab stops for the specified paragraphs. Read/write.

For information about returning a single member of a collection, see Returning an Object from a Collection.

TabStops Property Example

This example adds a centered tab stop at 2 inches to all the paragraphs in the active document. The **InchesToPoints** method is used to convert inches to points.

```
With ActiveDocument.Paragraphs.TabStops
    .Add Position:= InchesToPoints(2), Alignment:= wdAlignTabCenter
End With
```

This example sets the tab stops for every paragrah in the document to match the tab stops in the first paragraph.

```
Set para1Tabs = ActiveDocument.Paragraphs(1).TabStops
ActiveDocument.Paragraphs.TabStops = para1Tabs
```

TabIndent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTabIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTabIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTabIndentA "}

Sets the left indent for the specified paragraphs to a specified number of tab stops. Can also be used to remove the indent if the value of ***Count*** is a negative number.

Syntax

expression.**TabIndent(*Count*)**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Count Required **Integer**. The number of tab stops to indent (if positive) or the number of tab stops to remove from the indent (if negative).

TabIndent Method Example

This example indents the first paragraph in the active document to the second tab stop.

```
ActiveDocument.Paragraphs(1).TabIndent(2)
```

This example moves the indent of the first paragraph in the active document back one tab stop.

```
ActiveDocument.Paragraphs(1).TabIndent(-1)
```

TabHangingIndent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTabHangingIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTabHangingIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTabHangingIndentA "}

Sets a hanging indent to a specified number of tab stops. Can be used to remove tab stops from a hanging indent if the value of ***Count*** is a negative number.

Syntax

expression.**TabHangingIndent(*Count*)**

expression Required. An expression that returns a **Paragraph**, **Paragraphs**, or **ParagraphFormat** object.

Count Required **Integer**. The number of tab stops to indent (if positive) or the number of tab stops to remove from the indent (if negative).

TabHangingIndent Method Example

This example sets a hanging indent to the second tab stop for the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).TabHangingIndent(2)
```

This example moves the hanging indent back one tab stop for the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).TabHangingIndent(-1)
```

CustomTab Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCustomTabC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCustomTabX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCustomTabA "}

True if the specified tab stop is a custom tab stop. Read-only **Boolean**.

CustomTab Property Example

This example cycles through the collection of tab stops in the first paragraph in the active document, and left-aligns any custom tab stops that it finds.

```
For each ts in ActiveDocument.Paragraphs(1).TabStops
    If ts.CustomTab = True Then ts.Alignment = wdAlignTabLeft
Next ts
```

WidowControl Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWidowControlC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWidowControlX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWidowControlA "}

True if the first and last lines in the specified paragraph remain on the same page as the rest of the paragraph when Word repaginates the document. Can be **True**, **False** or **wdUndefined**. Read/write **Long**.

WidowControl Property Example

This example formats the paragraphs in the active document so that the first or last line in a paragraph can appear by itself at the top or bottom of a page.

```
ActiveDocument.Paragraphs.WidowControl = False
```

After Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAfterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAfterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAfterA "}

Returns the next **TabStop** object to the right of ***Position***.

Syntax

expression.**After(*Position*)**

expression Required. An expression that returns a **TabStops** collection.

Position Required **Single**. A location on the ruler, in points.

After Method Example

This example changes the alignment of the first custom tab stop in the first paragraph in the active document that's more than 1 inch from the left margin.

```
Set ts = ActiveDocument.Paragraphs(1).TabStops.After(InchesToPoints(1))
ts.Alignment = wdAlignTabCenter
```

Before Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthBeforeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthBeforeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthBeforeA "}

Returns the next **TabStop** object to the left of ***Position***.

Syntax

expression.**Before(*Position*)**

expression Required. An expression that returns a **TabStops** collection.

Position Required **Single**. A location on the ruler, in points.

Before Method Example

This example changes the alignment of the first custom tab stop in the first paragraph in the active document that's less than 2 inches from the left margin.

```
Set ts = ActiveDocument.Paragraphs(1) _
    .TabStops.Before(InchesToPoints(2))
ts.Alignment = wdAlignTabCenter
```

DropCap Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDropCapC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDropCapX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDropCapA "}

Returns a **DropCap** object that represents a dropped capital letter for the specified paragraph. Read-only.

DropCap Property Example

This example sets a dropped capital letter for the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).DropCap
    .FontName = "Arial"
    .Position = wdDropNormal
    .LinesToDrop = 3
    .DistanceFromText = InchesToPoints(0.1)
End With
```

FontName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFontNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFontNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFontNameA "}

Returns or sets the name of the font for the dropped capital letter. Read/write **String**.

FontName Property Example

This example sets Arial as the font for the dropped capital letter for the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).DropCap
    .FontName = "Arial"
    .Position = wdDropNormal
    .LinesToDrop = 3
    .DistanceFromText = InchesToPoints(0.1)
End With
```

Alignment Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAlignmentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAlignmentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAlignmentA "}

ListLevel object: Returns or sets the alignment for the list level for the list template. Can be one of the following **WdListLevelAlignment** constants: **wdListLevelAlignCenter**, **wdListLevelAlignLeft**, or **wdListLevelAlignRight**. Read/write **Long**.

PageNumber object: Returns or sets the alignment for the page number. Can be one of the following **WdPageNumberAlignment** constants: **wdAlignPageNumberCenter**, **wdAlignPageNumberInside**, **wdAlignPageNumberLeft**, **wdAlignPageNumberOutside**, or **wdAlignPageNumberRight**. Read/write **Long**.

Paragraph, **Paragraphs**, or **ParagraphFormat** object: Returns or sets the alignment for the specified paragraphs. Can be one of the following **WdParagraphAlignment** constants: **wdAlignParagraphLeft**, **wdAlignParagraphCenter**, **wdAlignParagraphRight**, or **wdAlignParagraphJustify**. Read/write **Long**.

Row or **Rows** object: Returns or sets the alignment for the specified rows. Can be one of the following **WdRowAlignment** constants: **wdAlignRowLeft**, **wdAlignRowCenter**, or **wdAlignRowRight**. Read/write **Long**.

TabStop object: Returns or sets the alignment for the specified tab stop. Can be one of the following **WdTabAlignment** constants: **wdAlignTabBar**, **wdAlignTabCenter**, **wdAlignTabDecimal**, **wdAlignTabLeft**, **wdAlignTabList**, or **wdAlignTabRight**. Read/write **Long**.

Alignment Property Example

This example right-aligns the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Alignment = wdAlignParagraphRight
```

This example centers all the rows in the first table in the active document.

```
ActiveDocument.Tables(1).Rows.Alignment = wdAlignRowCenter
```

This example centers the first tab stop in the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).TabStops(1).Alignment = wdAlignTabCenter
```

LineSpacing Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineSpacingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineSpacingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineSpacingA "}

Returns or sets the line spacing (in points) for the specified paragraphs. Read/write **Single**.

Remarks

The **LineSpacing** property can be set after the **LineSpacingRule** property has been set to **wdLineSpaceAtLeast**, **wdLineSpaceExactly**, or **wdLineSpaceMultiple**. If **wdLineSpaceAtLeast** is used, the line spacing can be greater than or equal to, but never less than, the specified **LineSpacing** value. If **wdLineSpaceExactly** is used, the line spacing never changes from the specified **LineSpacing** value, even if a larger font is used within the paragraph. If **wdLineSpaceMultiple** is used, a **LineSpacing** property value must be specified, in points.

Use the **LinesToPoints** method to convert a number of lines to the corresponding value in points. For example, `LinesToPoints(2)` returns the value 24.

LineSpacing Property Example

This example sets the line spacing for the first paragraph in the active document to always be at least 12 points.

```
With ActiveDocument.Paragraphs(1)
    .LineSpacingRule = wdLineSpaceAtLeast
    .LineSpacing = 12
End With
```

This example triple-spaces the lines in the selected paragraphs.

```
With Selection.Paragraphs
    .LineSpacingRule = wdLineSpaceMultiple
    .LineSpacing = LinesToPoints(3)
End With
```

LineSpacingRule Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineSpacingRuleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineSpacingRuleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineSpacingRuleA "}

Returns or sets the line spacing for the specified paragraphs. Can be one of the following **WdLineSpacing** constants: **wdLineSpace1pt5**, **wdLineSpaceAtLeast**, **wdLineSpaceDouble**, **wdLineSpaceExactly**, **wdLineSpaceMultiple**, or **wdLineSpaceSingle**. Read/write **Long**.

Remarks

Use **wdLineSpaceSingle**, **wdLineSpace1pt5**, or **wdLineSpaceDouble** to set the line spacing to one of these values. To set the line spacing to an exact number of points or to a multiple number of lines, you must also set the **LineSpacing** property.

LineSpacingRule Property Example

This example double-spaces the lines in the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).LineSpacingRule = wdLineSpaceDouble
```

This example returns the line spacing rule used for the first paragraph in the selection.

```
lrule = Selection.Paragraphs(1).LineSpacingRule
```

SpaceAfter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpaceAfterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpaceAfterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpaceAfterA "}

Returns or sets the amount of spacing (in points) after the specified paragraph or text column. Read/write **Single**.

SpaceAfter Property Example

This example sets the spacing after the first paragraph in the active document to 12 points.

```
ActiveDocument.Paragraphs(1).SpaceAfter = 12
```

This example sets the active document to three columns, with a 0.5-inch space after the first column. The **InchesToPoints** method is used to convert inches to points.

```
With ActiveDocument.PageSetup.TextColumns
    .SetCount NumColumns:=3
    .LineBetween = False
    .EvenlySpaced = True
    .Item(1).SpaceAfter = InchesToPoints(0.5)
End With
```

OutlineLevel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutlineLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutlineLevelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutlineLevelA "}

Returns or sets the outline level for the specified paragraphs. Can be one of the following **WdOutlineLevel** constants: **wdOutLineLevel1**, **wdOutLineLevel2**, **wdOutLineLevel3**, **wdOutLineLevel4**, **wdOutLineLevel5**, **wdOutLineLevel6**, **wdOutLineLevel7**, **wdOutLineLevel8**, **wdOutLineLevel9**, or **wdOutLineLevelBodyText**. Read/write **Long**.

Remarks

If a paragraph has a heading style applied to it (Heading 1 through Heading 9), the outline level is the same as the heading style and cannot be changed.

Outline levels are visible only in outline view or the document map pane.

OutlineLevel Property Example

This example returns the outline level of the first paragraph in the active document.

```
temp = ActiveDocument.Paragraphs(1).OutlineLevel
```

This example sets the outline level for each paragraph in the active document. First the Normal style is applied to all paragraphs. The **Mod** operator is used to determine which outline level (1, 2, or 3) to apply to successive paragraphs in the document, and then the view is changed to outline view.

```
Set myParas = ActiveDocument.Paragraphs
ActiveDocument.Paragraphs.Style = wdStyleNormal
For x = 1 To myParas.Count
    If x Mod 3 = 1 Then
        myParas(x).OutlineLevel = wdOutlineLevel1
    ElseIf x Mod 3 = 2 Then
        myParas(x).OutlineLevel = wdOutlineLevel2
    Else
        myParas(x).OutlineLevel = wdOutlineLevel3
    End If
Next x
ActiveWindow.View.Type = wdOutlineView
```

CreatePublisher Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCreatePublisherC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCreatePublisherX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCreatePublisherA "}

On the Macintosh, this method publishes the selected text in an edition, which you can then subscribe to in other documents.

In Windows, the **CreatePublisher** method isn't available and generates an error.

EditionOptions Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEditionOptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEditionOptionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEditionOptionsA "}

On the Macintosh, this method sets options for the selected publisher.

In Windows, the **EditionOptions** method isn't available and generates an error.

SubscribeTo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSubscribeToC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSubscribeToX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSubscribeToA "}

On the Macintosh, this method subscribes to a published edition.

In Windows, the **SubscribeTo** method isn't available and generates an error.

MountVolume Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMountVolumeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMountVolumeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMountVolumeA"}

On the Macintosh, this method connects to a disk or folder shared across a network.

In Windows, the **MountVolume** method isn't available and generates an error. Use the **Connect** method instead.

ShowClipboard Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShowClipboardC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShowClipboardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShowClipboardA"}

On the Macintosh, this method displays the Clipboard and its contents.

In Windows, the **ShowClipboard** method is not available and generates an error. Instead, you can use the **Shell** function to display the Clipboard.

MacintoshName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMacintoshNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMacintoshNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMacintoshNameA"}

Returns the Macintosh name as set in the Sharing Setup control panel. Read-only **String**.

In Windows, the **MacintoshName** property isn't available and generates an error.

InsertParagraph Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertParagraphC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertParagraphX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertParagraphA "}

Replaces the specified range or selection with a new paragraph.

Note After this method has been used, the range or selection is the new paragraph.

Syntax

expression.**InsertParagraph**

expression Required. An expression that returns a **Range** or **Selection** object.

Remarks

If you don't want to replace the range or selection, use the **Collapse** method before using this method. The **InsertParagraphAfter** method inserts a new paragraph following a **Range** or **Selection** object.

InsertParagraph Method Example

This example inserts a new paragraph at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(0, 0)
With myRange
    .InsertParagraph
    .InsertBefore "Dear Sirs,"
End With
```

This example collapses the selection and then inserts a paragraph mark at the insertion point.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .InsertParagraph
    .Collapse Direction:=wdCollapseEnd
End With
```

InsertAfter Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertAfterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertAfterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertAfterA "}

Inserts the specified text at the end of a range or selection. After this method is applied, the range or selection expands to include the new text.

Note You can insert characters such as quotation marks, tab characters, and nonbreaking hyphens by using the **Chr** function with the **InsertAfter** method. You can also use the following Visual Basic constants: **vbCr**, **vbLf**, **vbCrLf** and **vbTab**.

Syntax

expression.**InsertAfter(*Text*)**

expression Required. An expression that returns a **Selection** or **Range** object.

Text Required **String**. The text to be inserted.

Remarks

If you use this method with a range or selection that refers to an entire paragraph, the text is inserted after the ending paragraph mark (the text will appear at the beginning of the next paragraph). To insert text at the end of a paragraph, determine the ending point and subtract 1 from this location (the paragraph mark is one character), as shown in the following example.

```
Set Doc = ActiveDocument
Set myRange = Doc.Range(Start:=Doc.Paragraphs(1).Range.End - 1, _
    End:=Doc.Paragraphs(1).Range.End - 1)
myRange.InsertAfter " the end."
```

InsertAfter Method Example

This example inserts text at the end of the active document. The **Content** property returns a **Range** object.

```
ActiveDocument.Content.InsertAfter "end of document"
```

This example inserts text at the end of the selection and then collapses the selection to an insertion point.

```
With Selection
    .InsertAfter "appended text"
    .Collapse Direction:=wdCollapseEnd
End With
```

This example inserts text from an input box as the second paragraph in the active document.

```
response = InputBox("Type some text")
With ActiveDocument.Paragraphs(1).Range
    .InsertAfter "1." & Chr(9) & response
    .InsertParagraphAfter
End With
```

InsertParagraphAfter Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertParagraphAfterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertParagraphAfterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertParagraphAfterA "}

Inserts a paragraph mark after a range or selection.

Note After this method is applied, the range or selection expands to include the new paragraph.

Syntax

expression.**InsertParagraphAfter**

expression Required. An expression that returns a **Range** or **Selection** object.

InsertParagraphAfter Method Example

This example inserts a new paragraph after the current paragraph.

```
With Selection
    .Move Unit:=wdParagraph
    .InsertParagraphAfter
    .Collapse Direction:=wdCollapseStart
End With
```

This example inserts text as a new paragraph at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(0, 0)
With myRange
    .InsertBefore "Title"
    .ParagraphFormat.Alignment = wdAlignParagraphCenter
    .InsertParagraphAfter
End With
```

This example inserts a paragraph at the end of the active document. The **Content** property returns a **Range** object.

```
ActiveDocument.Content.InsertParagraphAfter
```

Calculate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCalculateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCalculateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCalculateA "}

Calculates a mathematical expression within a range or selection. Returns the result as a **Single**.

Syntax

expression.**Calculate**

expression Required. An expression that returns a **Range** or **Selection** object.

Calculate Method Example

This example inserts a mathematical expression at the beginning of the active document, calculates the expression, and then appends the results to the range. The result is "1 + 1 = 2".

```
Set myRange = ActiveDocument.Range(0, 0)
myRange.InsertBefore "1 + 1 "
myRange.InsertAfter "= " & myRange.Calculate
```

This example calculates the selected mathematical expression and displays the result.

```
MsgBox "And the answer is... " & Selection.Calculate
```

Case Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCaseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCaseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCaseA "}

Returns or sets the case of the text in the specified range. Can be one of the following **WdCharacterCase** constants: **wdLowerCase**, **wdNextCase**, **wdTitleSentence**, **wdTitleWord**, **wdToggleCase**, or **wdUpperCase**. Read/write **Long**.

Case Property Example

This example changes the first word in the selection to uppercase.

```
Selection.Words(1).Case = wdUpperCase
```

This example capitalizes the first letter of each sentence in the first paragraph of the document..

```
Set myRange = ActiveDocument.Paragraphs(1).Range
For Each Sent In myRange.Sentences
    Sent.Case = wdTitleSentence
Next Sent
```

Characters Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCharactersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCharactersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCharactersA "}

Returns a **Characters** collection that represents the characters in a document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Characters Property Example

This example displays the first character in the selection. If nothing is selected, the character immediately after the insertion point is displayed.

```
char = Selection.Characters(1).Text
MsgBox "The first character is... " & char
```

This example returns the number of characters in the first sentence in the active document (spaces are included in the count).

```
numchars = ActiveDocument.Sentences(1).Characters.Count
```

CheckSynonyms Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCheckSynonymsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCheckSynonymsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCheckSynonymsA "}

Displays the **Thesaurus** dialog box (**Language** submenu, **Tools** menu), which lists alternative word choices, or synonyms, for the text in the specified range.

Syntax

expression.**CheckSynonyms**

expression Required. An expression that returns a **Range** object.

CheckSynonyms Method Example

This example displays the **Thesaurus** dialog box with synonyms for the selected text.

```
Selection.Range.CheckSynonyms
```

This example displays the **Thesaurus** dialog box with synonyms for the first word in the active document.

```
ActiveDocument.Words(1).CheckSynonyms
```

Collapse Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCollapseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCollapseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCollapseA "}

Collapses a range or selection to the starting or ending position. After a range or selection is collapsed, the starting and ending points are equal.

Syntax

expression.**Collapse(*Direction*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Direction Optional **Variant**. The direction in which to collapse the range or selection. Can be either of the following **WdCollapseDirection** constants: **wdCollapseEnd** or **wdCollapseStart**. The default value is **wdCollapseStart**.

Remarks

If you use **wdCollapseEnd** to collapse a range that refers to an entire paragraph, the range is located after the ending paragraph mark (the beginning of the next paragraph). However, you can move the range back one character by using the **MoveEnd** method after the range is collapsed, as shown in the following example.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.Collapse Direction:=wdCollapseEnd
myRange.MoveEnd Unit:=wdCharacter, Count:=-1
```

Collapse Method Example

This example collapses the selection to an insertion point at the beginning of the previous selection.

```
Selection.Collapse Direction:=wdCollapseStart
```

This example sets `myRange` equal to the contents of the active document, collapses `myRange`, and then inserts a 2x2 table at the end of the document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.Tables.Add Range:=myRange, NumRows:=2, NumColumns:=2
```

ColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColorIndexA "}

Returns or sets the color for the specified border or font object. Read/write **Long**.

Can be one of the following **WdColorIndex** constants.

wdAuto	**wdGreen**
wdBlack	**wdNoHighlight**
wdBlue	**wdPink**
wdBrightGreen	**wdRed**
wdDarkBlue	**wdTeal**
wdDarkRed	**wdTurquoise**
wdDarkYellow	**wdViolet**
wdGray25	**wdWhite**
wdGray50	**wdYellow**

Remarks

The **wdByAuthor** constant is not valid for border and font objects.

ColorIndex Property Example

This example changes the color of the text in the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Range.Font.ColorIndex = wdGreen
```

This example formats the selected text to appear in red.

```
Selection.Font.ColorIndex = wdRed
```

This example adds a dotted red border around each cell in the first table.

```
If ActiveDocument.Tables.Count >= 1 Then
    For Each aBorder In ActiveDocument.Tables(1).Borders
        aBorder.ColorIndex = wdRed
        aBorder.LineStyle = wdLineStyleDashDot
        aBorder.LineWidth = wdLineWidth075pt
    Next aBorder
End If
```

Content Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContentA "}

Returns a **Range** object that represents the main document story. Read-only.

Remarks

The following two statements are equivalent:

```
Set mainStory = ActiveDocument.Content
Set mainStory = ActiveDocument.StoryRanges(wdMainTextStory)
```

Content Property Example

This example changes the font and font size of the text in the active document to Arial 10 point.

```
Set myRange = ActiveDocument.Content
With myRange.Font
    .Name = "Arial"
    .Size = 10
End With
```

This example inserts text at the end of the document named "Changes.doc." The **For Each...Next** statement is used to determine whether the document is open.

```
For Each aDocument In Documents
    If InStr(LCase$(aDocument.Name), "changes.doc") Then
        Set myRange = Documents("Changes.doc").Content
        myRange.InsertAfter "the end."
    End If
Next aDocument
```

ConvertToTable Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToTableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToTableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToTableA "}

Converts text within a range or selection to a table. Returns the table as a **Table** object.

Syntax

expression.**ConvertToTable(*Separator, NumRows, NumColumns, InitialColumnWidth, Format, ApplyBorders, ApplyShading, ApplyFont, ApplyColor, ApplyHeadingRows, ApplyLastRow, ApplyFirstColumn, ApplyLastColumn, AutoFit*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Separator Optional **Variant**. Specifies the character used to separate text into cells. Can be a character or one of the following **WdTableFieldSeparator** constants: **wdSeparateByCommas**, **wdSeparateByDefaultListSeparator**, **wdSeparateByParagraphs**, or **wdSeparateByTabs**. If this argument is omitted, the value of the **DefaultTableSeparator** property is used.

NumRows Optional **Variant**. The number of rows in the table. If this argument is omitted, Word sets the number of rows, based on the contents of the range or selection.

NumColumns Optional **Variant**. The number of columns in the table. If this argument is omitted, Word sets the number of columns, based on the contents of the range or selection.

InitialColumnWidth Optional **Variant**. The initial width of each column, in points. If this argument is omitted, Word calculates and adjusts the column width so that the table stretches from margin to margin.

Format Optional **Variant**. Specifies one of the predefined formats listed in the **Table AutoFormat** dialog box (**Table** menu). Can be one of the following **WdTableFormat** constants:

wdTableFormat3DEffects1	**wdTableFormatGrid4**
wdTableFormat3DEffects2	**wdTableFormatGrid5**
wdTableFormat3DEffects3	**wdTableFormatGrid6**
wdTableFormatClassic1	**wdTableFormatGrid7**
wdTableFormatClassic2	**wdTableFormatGrid8**
wdTableFormatClassic3	**wdTableFormatList1**
wdTableFormatClassic4	**wdTableFormatList2**
wdTableFormatColorful1	**wdTableFormatList3**
wdTableFormatColorful2	**wdTableFormatList4**
wdTableFormatColorful3	**wdTableFormatList5**
wdTableFormatColumns1	**wdTableFormatList6**
wdTableFormatColumns2	**wdTableFormatList7**
wdTableFormatColumns3	**wdTableFormatList8**
wdTableFormatColumns4	**wdTableFormatNone**
wdTableFormatColumns5	**wdTableFormatProfessional**
wdTableFormatContemporary	**wdTableFormatSimple1**
wdTableFormatElegant	**wdTableFormatSimple2**
wdTableFormatGrid1	**wdTableFormatSimple3**
wdTableFormatGrid2	**wdTableFormatSubtle1**
wdTableFormatGrid3	**wdTableFormatSubtle2**

ApplyBorders Optional **Variant**. **True** to apply the border properties of the specified format.

ApplyShading Optional **Variant**. **True** to apply the shading properties of the specified format.

ApplyFont Optional **Variant**. **True** to apply the font properties of the specified format.

ApplyColor Optional **Variant**. **True** to apply the color properties of the specified format.

ApplyHeadingRows Optional **Variant**. **True** to apply the heading-row properties of the specified format.

ApplyLastRow Optional **Variant**. **True** to apply the last-row properties of the specified format.

ApplyFirstColumn Optional **Variant**. **True** to apply the first-column properties of the specified format.

ApplyLastColumn Optional **Variant**. **True** to apply the last-column properties of the specified format.

AutoFit Optional **Variant**. **True** to decrease the width of the table columns as much as possible without changing the way text wraps in the cells.

ConvertToTable Method Example

This example converts the first three paragraphs in the active document to a table.

```
Set aDoc = ActiveDocument
Set myRange = aDoc.Range(Start:=aDoc.Paragraphs(1).Range.Start, _
    End:=aDoc.Paragraphs(3).Range.End)
myRange.ConvertToTable Separator:=wdSeparateByParagraphs
```

This example inserts text at the insertion point and then converts the comma-delimited text to a table with formatting.

```
With Selection
    .Collapse
    .InsertBefore "one, two, three"
    .InsertParagraphAfter
    .InsertAfter "one, two, three"
    .InsertParagraphAfter
End With
Set myTable = Selection.ConvertToTable(Separator:=wdSeparateByCommas, _
    Format:=wdTableFormatList8)
```

End Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEndC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEndX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEndA "}

Returns or sets the ending character position of a selection, range, or bookmark. Read/write **Long**.

Note If this property is set to a value smaller than the **Start** property, the **Start** property is set to the same value (that is, the **Start** and **End** property are equal).

Remarks

The **Selection**, **Range**, and **Bookmark** objects all have a starting position and an ending position. The ending position is the point farthest away from the beginning of the story.

This property returns the ending character position relative to the beginning of the story. The main document story (**wdMainTextStory**) begins with character position 0 (zero). You can change the size of a selection, range, or bookmark by setting this property.

End Property Example

This example compares the ending position of the "temp" bookmark with the starting position of the "begin" bookmark.

```
Set Book1 = ActiveDocument.Bookmarks("begin")
Set Book2 = ActiveDocument.Bookmarks("temp")
If Book2.End > Book1.Start Then Book1.Select
```

This example retrieves the ending position of the selection. This value is used to create a range so that a field can be inserted after the selection.

```
pos = Selection.End
Set myRange = ActiveDocument.Range(Start:=pos, End:=pos)
ActiveDocument.Fields.Add Range:=myRange, Type:=wdFieldAuthor
```

This example changes the ending position of `myRange` by one character.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.End = myRange.End - 1
```

EndOf Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEndOfC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEndOfX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEndOfA "}

Moves or extends the ending character position of a range or selection to the end of the nearest specified text unit. This method returns a value that indicates the number of character positions the range or selection was moved or extended (movement is forward in the document).

Syntax

expression.**EndOf(*Unit, Extend*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. The unit by which to move the ending character position. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, **wdLine** can also be used. The default value is **wdWord**.

Extend Optional **Variant**. Can be either of the following **WdMovementType** constants: **wdMove** or **wdExtend**. If **wdMove**, both ends of the range or selection object are moved to the end of the specified unit. If **wdExtend** is used, the end of the range or selection is extended to the end of the specified unit. The default value is **wdMove**.

Remarks

If the both the starting and ending positions for the range or selection are already at the end of the specified unit, this method doesn't move or extend the range or selection. For example, if the selection is at the end of a word and the trailing space, the following instruction doesn't change the selection (`char` equals 0 (zero)).

```
char = Selection.EndOf(Unit:=wdWord, Extend:=wdMove)
```

EndOf Method Example

This example extends the selection to the end of the paragraph.

```
charmoved = Selection.EndOf(Unit:=wdParagraph, Extend:=wdExtend)
If charmoved = 0 Then MsgBox "Selection unchanged"
```

This example moves `myRange` to the end of the first word in the selection (after the trailing space).

```
Set myRange = Selection.Characters(1)
myRange.EndOf Unit:=wdWord, Extend:=wdMove
```

This example adds a table, selects the first cell in row two, and then extends the selection to the end of the column.

```
Set myRange = ActiveDocument.Range(0, 0)
Set myTable = ActiveDocument.Tables.Add(Range:=myRange, _
    NumRows:=5, NumColumns:=3)
myTable.Cell(2, 1).Select
Selection.EndOf Unit:=wdColumn, Extend:=wdExtend
```

Expand Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExpandC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExpandX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExpandA "}

Expands the specified range or selection. Returns the number of characters added to the range or selection.

Syntax

expression.**Expand(*Unit*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. The unit by which to expand the range. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* represents a **Selection** object, **wdLine** can also be used. The default value is **wdWord**.

Expand Method Example

This example creates a range that refers to the first word in the active document, and then it expands the range to reference the first paragraph in the document.

```
Set myRange = ActiveDocument.Words(1)
myRange.Expand Unit:=wdParagraph
```

This example capitalizes the first character in the selection and then expands the selection to include the entire sentence.

```
With Selection
    .Characters(1).Case = wdTitleSentence
    .Expand Unit:=wdSentence
End With
```

InRange Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInRangeA "}

Returns **True** if the range or selection to which the method is applied is contained in the range specified by the ***Range*** argument.

Syntax

expression.**InRange(*Range*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Range Required **Range** object. The range to which you want to compare *expression*.

Remarks

This method determines whether the range or selection returned by *expression* is contained in the specified ***Range*** by comparing the starting and ending character positions, as well as the story type.

InRange Method Example

This example determines whether the selection is contained in the first paragraph in the active document.

```
status = Selection.InRange(ActiveDocument.Paragraphs(1).Range)
```

This example sets `myRange` equal to the first word in the active document. If `myRange` isn't contained in the selection, `myRange` is selected.

```
Set myRange = ActiveDocument.Words(1)
If myRange.InRange(Selection.Range) = False Then myRange.Select
```

This example displays a message if the selection is in the footnote story.

```
If Selection.InRange(ActiveDocument.StoryRanges(wdFootnotesStory)) Then
    MsgBox "Selection in footnotes"
End If
```

InsertBreak Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertBreakC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertBreakX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertBreakA "}

Inserts a page, column, or section break.

Syntax

expression.**InsertBreak(*Type*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Type Optional **Variant**. The type of break to be inserted. Can be one of the following **WdBreakType** constants: **wdPageBreak**, **wdColumnBreak**, **wdSectionBreakNextPage**, **wdSectionBreakContinuous**, **wdSectionBreakEvenPage**, **wdSectionBreakOddPage**, or **wdLineBreak**. The default value is **wdPageBreak**.

Remarks

When you insert a page or column break, the range or selection is replaced by the break. If you don't want to replace the range or selection, use the **Collapse** method before using the **InsertBreak** method. When you insert a section break, the break is inserted immediately preceding the **Range** or **Selection** object.

InsertBreak Method Example

This example inserts a continuous section break immediately preceding the selection.

```
Selection.InsertBreak Type:=wdSectionBreakContinuous
```

This example inserts a page break immediately following the second paragraph in the active document.

```
Set myRange = ActiveDocument.Paragraphs(2).Range
With myRange
    .Collapse Direction:=wdCollapseEnd
    .InsertBreak Type:=wdPageBreak
End With
```

InsertCaption Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertCaptionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertCaptionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertCaptionA "}

Inserts a caption immediately preceding or following the specified range or selection.

Syntax

expression.**InsertCaption(*Label, Title, TitleAutoText, Position*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Label Required **Variant**. The caption label to be inserted. Can be a string or one of the following **WdCaptionLabelID** constants: **wdCaptionEquation**, **wdCaptionFigure**, or **wdCaptionTable**. If the label hasn't yet been defined, an error occurs. Use the **Add** method with the **CaptionLabels** object to define new caption labels.

Title Optional **Variant**. The string to be inserted immediately following the label in the caption (ignored if ***TitleAutoText*** is specified).

TitleAutoText Optional **Variant**. The AutoText entry whose contents you want to insert immediately following the label in the caption (overrides any text specified by ***Title***).

Position Optional **Variant**. Specifies whether the caption will be inserted above or below the **Selection** or **Range** object. Can be either of the following **WdCaptionPosition** constants: **wdCaptionPositionAbove** or **wdCaptionPositionBelow**.

InsertCaption Method Example

This example inserts a caption below the first table in the active document.

```
ActiveDocument.Tables(1).Range.InsertCaption Label:=wdCaptionTable, _
    Position:=wdCaptionPositionBelow
```

This example inserts a Figure caption at the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.InsertCaption Label:="Figure", _
    Title:=": Sales Results", Position:=wdCaptionPositionBelow
```

InsertFile Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertFileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertFileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertFileA "}

Inserts all or part of the specified file.

Syntax

expression.**InsertFile(*FileName, Range, ConfirmConversions, Link, Attachment*)**

expression Required. An expression that returns a **Range** or **Selection** object.

FileName Required **String**. The path and file name of the file to be inserted. If you don't specify a path, Word assumes the file is in the current folder.

Range Optional **Variant**. If the specified file is a Word document, this parameter refers to a bookmark. If the file is another type (for example, a Microsoft Excel worksheet), this parameter refers to a named range or a cell range (for example, R1C1:R3C4).

ConfirmConversions Optional **Variant**. **True** to have Word prompt you to confirm conversion when inserting files in formats other than the Word Document format.

Link Optional **Variant**. **True** to insert the file by using an INCLUDETEXT field.

Attachment Optional **Variant**. **True** to insert the file as an attachment to a WordMail message.

InsertFile Method Example

This example uses an INCLUDETEXT field to insert the TEST.DOC file at the insertion point.

```
Selection.Collapse Direction:=wdCollapseEnd
Selection.InsertFile FileName:="C:\TEST.DOC", Link:=True
```

This example creates a new document and then inserts the contents of each text file in the C:\TMP folder into the new document.

```
Documents.Add
ChDir "C:\TMP"
myName = Dir("*.TXT")
While myName <> ""
    With Selection
        .InsertFile FileName:=myName, ConfirmConversions:=False
        .InsertParagraphAfter
        .InsertBreak Type:=wdSectionBreakNextPage
        .Collapse Direction:=wdCollapseEnd
    End With
    myName = Dir()
Wend
```

IsEqual Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthIsEqualC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthIsEqualX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthIsEqualA "}

True if the selection or range to which this method is applied is equal to the range specified by the ***Range*** argument. This method compares the starting and ending character positions, as well as the story type. If all three of these items are the same for both objects, the objects are equal.

Syntax

expression.**IsEqual(*Range*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Range Required **Range** object. The **Range** object that's compared with *expression*.

IsEqual Method Example

This example compares the selection with the second paragraph in the active document. If the selection isn't equal to the second paragraph, the second paragraph is selected.

```
If Selection.IsEqual(ActiveDocument.Paragraphs(2).Range) = False Then
    ActiveDocument.Paragraphs(2).Range.Select
End If
```

This example compares `Range1` with `Range2` to determine whether they're equal. If the two ranges are equal, the content of `Range1` is deleted.

```
Set Range1 = Selection.Words(1)
Set Range2 = ActiveDocument.Words(3)
If Range1.IsEqual(Range:=Range2) = True Then
    Range1.Delete
End If
```

MoveEnd Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveEndC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveEndX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveEndA "}

Moves the ending character position of a range or selection. This method returns an integer that indicates the number of units the range or selection actually moved, or it returns 0 (zero) if the move was unsuccessful.

Syntax

expression.**MoveEnd(*Unit, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. The unit by which to move the ending character position. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, **wdLine** can also be used. The default value is **wdCharacter**.

Count Optional **Variant**. The number of units to move. If this number is positive, the ending character position is moved forward in the document. If this number is negative, the end is moved backward. If the ending position overtakes the starting position, the range collapses and both character positions move together. The default value is 1.

MoveEnd Method Example

This example moves the end of the selection one character backward (the selection size is reduced by one character). A space is considered a character.

```
Selection.MoveEnd Unit:=wdCharacter, Count:=-1
```

This example moves the end of the selection to the end of the line (the selection is extended to the end of the line).

```
Selection.MoveEnd Unit:=wdLine, Count:=1
```

This example sets `myRange` to be equal to the second word in the active document. The **MoveEnd** method is used to move the ending position of `myRange` (a range object) forward one word. After this macro is run, the second and third words in the document are selected.

```
If ActiveDocument.Words.Count >= 3 Then
    Set myRange = ActiveDocument.Words(2)
    With myRange
        .MoveEnd Unit:=wdWord, Count:=1
        .Select
    End With
End If
```

Outline Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOutlineC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOutlineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOutlineA "}

True if the font is formatted as outline. Returns **True**, **False**, or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Outline Property Example

This example applies outline font formatting to the first three words in the active document.

```
Set myRange = ActiveDocument.Range(Start:= _
    ActiveDocument.Words(1).Start, End:=ActiveDocument.Words(3).End)
myRange.Font.Outline = True
```

This example toggles outline formatting for the selected text.

```
Selection.Font.Outline = wdToggle
```

This example removes outline font formatting from the selection if outline formatting is partially applied to the selection.

```
Set myFont = Selection.Font
If myFont.Outline = wdUndefined Then
    myFont.Outline = False
End If
```

Paste Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPasteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPasteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPasteA "}

Inserts the contents of the Clipboard at the specified range or selection. If you don't want to replace the contents of the range or selection, use the **Collapse** method before using this method.

Syntax

expression.**Paste**

expression Required. An expression that returns a **Range** or **Selection** object.

Remarks

When this method is used with a range object, the range expands to include the contents of the Clipboard. When this method is used with a selection object, the selection doesn't expand to include the Clipboard contents; instead, the selection is positioned after the pasted Clipboard contents.

Paste Method Example

This example copies and pastes the first table in the active document into a new document.

```
If ActiveDocument.Tables.Count >= 1 Then
    ActiveDocument.Tables(1).Range.Copy
    Documents.Add.Content.Paste
End If
```

This example copies the first paragraph in the document and pastes it at the insertion point.

```
ActiveDocument.Paragraphs(1).Range.Copy
Selection.Collapse Direction:=wdCollapseStart
Selection.Paste
```

This example copies the selection and pastes it at the end of the document.

```
If Selection.Type <> wdSelectionIP Then
    Selection.Copy
    Set Range2 = ActiveDocument.Content
    Range2.Collapse Direction:=wdCollapseEnd
    Range2.Paste
End If
```

Range Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRangeA "}

Returns a **Range** object that represents the portion of a document that's contained in the specified object. Read-only.

For information about returning a range from a document or returning a shape range from a collection of shapes, see the **Range** method.

Range Property Example

This example applies the Heading 1 style to the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Range.Style = wdStyleHeading1
```

This example copies the first row in table one.

```
If ActiveDocument.Tables.Count >= 1 Then _
    ActiveDocument.Tables(1).Rows(1).Range.Copy
```

This example changes the text of the first comment in the document.

```
With ActiveDocument.Comments(1).Range
    .Delete
    .InsertBefore "new comment text"
End With
```

This example inserts text at the end of section one.

```
Set myRange = ActiveDocument.Sections(1).Range
With myRange
    .MoveEnd Unit:=wdCharacter, Count:=-1
    .Collapse Direction:=wdCollapseEnd
    .InsertParagraphAfter
    .InsertAfter "End of section"
End With
```

Relocate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRelocateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRelocateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRelocateA "}

In outline view, moves the paragraphs within the specified range after the next visible paragraph or before the previous visible paragraph. Body text moves with a heading only if the body text is collapsed in outline view or if it's part of the range.

Syntax

expression.**Relocate(*Direction*)**

expression Required. An expression that returns a **Range** object.

Direction Required **Long**. The direction of the move. Can be either of the following **WdRelocate** constants: **wdRelocateUp** or **wdRelocateDown**.

Relocate Method Example

This example moves the third, fourth, and fifth paragraphs in the active document below the next (sixth) paragraph.

```
theStart = ActiveDocument.Paragraphs(3).Range.Start
theEnd = ActiveDocument.Paragraphs(5).Range.End
Set myRange = ActiveDocument.Range(Start:=theStart, End:=theEnd)
ActiveWindow.View.Type = wdOutlineView
myRange.Relocate Direction:=wdRelocateDown
```

This example moves the first paragraph in the selection above the previous paragraph.

```
ActiveWindow.View.Type = wdOutlineView
Selection.Paragraphs(1).Range.Relocate Direction:=wdRelocateUp
```

SetRange Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetRangeA "}

Sets the starting and ending character positions for the range or selection.

Note Character position values start at the beginning of the story, with the first value being 0 (zero). All characters are counted, including nonprinting characters. Hidden characters are counted even if they're not displayed.

Syntax

expression.**SetRange**(***Start, End***)

expression Required. An expression that returns a **Range** or **Selection** object.

Start Required **Long**. The starting character position of the range or selection.

End Required **Long**. The ending character position of the range or selection.

Remarks

The **SetRange** method redefines the starting and ending positions of an existing **Selection** or **Range** object. This method differs from the **Range** method, which is used to create a range, given a starting and ending position.

SetRange Method Example

This example selects the first 10 characters in the document.

```
Selection.SetRange Start:=0, End:=10
```

This example uses **SetRange** to redefine `myRange` to refer to the first three paragraphs in the active document.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
myRange.SetRange Start:=myRange.Start, _
    End:=ActiveDocument.Paragraphs(3).Range.End
```

This example uses **SetRange** to redefine `myRange` to refer to the area starting at the beginning of the document and ending at the end of the current selection.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myRange.InsertAfter "Hello "
myRange.SetRange Start:=myRange.Start, End:=Selection.End
```

This example extends the selection to the end of the document.

```
Selection.SetRange Start:=Selection.Start, _
    End:=ActiveDocument.Content.End
```

Start Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStartC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStartX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStartA "}

Returns or sets the starting character position of a selection, range, or bookmark. Read/write **Long**.

Note If this property is set to a value larger than that of the **End** property, the **End** property is set to the same value as that of **Start** property.

Remarks

Selection, **Range**,and **Bookmark** objects have starting and ending character positions. The starting position refers to the character position closest to the beginning of the story.

This property returns the starting character position relative to the beginning of the story. The main text story (**wdMainTextStory**) begins with character position 0 (zero). You can change the size of a selection, range, or bookmark by setting this property.

Start Property Example

This example returns the starting position of the second paragraph and the ending position of the fourth paragraph in the active document. The character positions are used to create the range `myRange`.

```
pos = ActiveDocument.Paragraphs(2).Range.Start
pos2 = ActiveDocument.Paragraphs(4).Range.End
Set myRange = ActiveDocument.Range(Start:=pos, End:=pos2)
```

This example determines the length of the selection by comparing the starting and ending character positions.

```
SelLength = Selection.End - Selection.Start
```

This example moves the starting position of `myRange` one character to the right (this reduces the size of the range by one character).

```
Set myRange = Selection.Range
myRange.SetRange Start:=myRange.Start + 1, End:=myRange.End
```

StoryType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStoryTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStoryTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStoryTypeA "}

Returns the story type for the specified range, selection, or bookmark. Can be one of the following **WdStoryType** constants: **wdCommentsStory**, **wdEndnotesStory**, **wdEvenPagesFooterStory**, **wdEvenPagesHeaderStory**, **wdFirstPageFooterStory**, **wdFirstPageHeaderStory**, **wdFootnotesStory**, **wdMainTextStory**, **wdPrimaryFooterStory**, **wdPrimaryHeaderStory**, or **wdTextFrameStory**. Read-only **Long**.

StoryType Property Example

This example returns the story type of the selection.

```
story = Selection.StoryType
```

This example closes the footnote pane if the selection is contained in the footnote story.

```
ActiveWindow.View.Type = wdNormalView
If Selection.StoryType = wdFootnotesStory Then _
    ActiveWindow.ActivePane.Close
```

This example selects the bookmark named "temp" if the bookmark is contained in the main story in the active document.

```
If ActiveDocument.Bookmarks.Exists("temp") = True Then
    Set myBookmark = ActiveDocument.Bookmarks("temp")
    If myBookmark.StoryType = wdMainTextStory Then myBookmark.Select
End If
```

StoryLength Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStoryLengthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStoryLengthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStoryLengthA "}

Returns the number of characters in the story that contains the specified range or selection. Read-only **Long**.

StoryLength Property Example

This example determines whether the header in the active document is empty. If the header story isn't empty, a message box displays the contents of the header. If the document header is empty, **StoryLength** returns 1 for the final paragraph mark.

```
Set myRange = ActiveDocument.Sections(1) _
    .Headers(wdHeaderFooterPrimary).Range
If myRange.StoryLength > 1 Then MsgBox myRange.Text
```

This example closes the document without saving changes if it's empty.

```
If ActiveDocument.Content.StoryLength = 1 Then _
    ActiveDocument.Close SaveChanges:=wdDoNotSaveChanges
```

Text Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextA "}

Range or **Selection** object: Returns or sets the text in the specified range or selection. Read/write **String**.

Find or **Replacement** object: Returns or sets the text to find or replace in the specified range or selection. Read/write **String**.

Remarks

The **Text** property returns the plain, unformatted text of the selection or range. When you set this property, the text of the range or selection is replaced.

Text Property Example

This example displays the text in the selection. If nothing is selected, the character following the insertion point is displayed.

```
MsgBox Selection.Text
```

This example replaces the first word in the active document with "Dear."

```
Set myRange = ActiveDocument.Words(1)
myRange.Text = "Dear "
```

This example inserts 10 lines of text into a new document.

```
Documents.Add
For i = 1 To 10
    Selection.Text = "Line" & Str(i) & Chr(13)
    Selection.MoveDown Unit:=wdParagraph, Count:=1
Next i
```

This example replaces "Hello" with "Goodbye" in the active document.

```
Set myRange = ActiveDocument.Content
With myRange.Find
    .ClearFormatting
    .Replacement.ClearFormatting
    .Text = "Hello"
    .Replacement.Text = "Goodbye"
    .Execute Replace:=wdReplaceAll
End With
```

WholeStory Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthWholeStoryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthWholeStoryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthWholeStoryA "}

Expands a range or selection to include the entire story.

Syntax

expression.**WholeStory**

expression Required. An expression that returns a **Range** or **Selection** object.

Remarks

The following instructions, where `myRange` is a valid **Range** object, are functionally equivalent:

```
myRange.WholeStory
myRange.Expand Unit:=wdStory
```

WholeStory Method Example

This example expands `myRange` to include the entire story and then applies the Arial font to the range.

```
Set myRange = Selection.Range
myRange.WholeStory
myRange.Font.Name = "Arial"
```

This example expands `myRange` to include the entire comments story (**wdCommentsStory**) and then copies the comments into a new document.

```
If ActiveDocument.Comments.Count >= 1 Then
    Set myRange = Activedocument.Comments(1).Range
    myRange.WholeStory
    myRange.Copy
    Documents.Add.Content.Paste
End If
```

Words Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWordsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWordsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWordsA "}

Returns a **Words** collection that represents all the words in a range, selection, or document. Read-only.

Note Punctuation and paragraph marks in a document are included in the **Words** collection.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Words Property Example

This example displays the number of words in the selection. Paragraphs marks, partial words, and punctuation are included in the count.

```
MsgBox "There are " & Selection.Words.Count & " words."
```

This example steps through the words in `myRange` (which spans from the beginning of the active document to the end of the selection) and deletes the word "Franklin" (including the trailing space) wherever it occurs in the range.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=Selection.End)
For Each aWord In myRange.Words
    If aWord.Text = "Franklin " Then aWord.Delete
Next aWord
```

StartOf Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthStartOfC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthStartOfX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthStartOfA "}

Moves or extends the start position of the specified range or selection to the beginning of the nearest specified text unit. This method returns a value that indicates the number of characters by which the range or selection was moved or extended. The method returns a negative number if the movement is backward through the document.

Syntax

expression.**StartOf(*Unit, Extend*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. The unit by which the start position of the specified range or selection is to be moved. Can be one of the following **WdUnits** constants: **wdCell**, **wdCharacter**, **wdColumn**, **wdParagraph**, **wdRow**, **wdSection**, **wdSentence**, **wdStory**, **wdTable**, or **wdWord**. If *expression* returns a **Selection** object, you can also use **wdLine**. The default value is **wdWord**.

Extend Optional **Variant**. Can be either of the following **WdMovementType** constants: **wdMove** or **wdExtend**. If you use **wdMove**, both ends of the range or selection are moved to the beginning of the specified unit. If you use **wdExtend**, the beginning of the range or selection is extended to the beginning of the specified unit. The default value is **wdMove**.

Remarks

If the beginning of the specified range or selection is already at the beginning of the specified unit, this method doesn't move or extend the range or selection. For example, if the selection is at the beginning of a line, the following example returns 0 (zero) and doesn't change the selection.

```
char = Selection.StartOf(Unit:=wdLine, Extend:=wdMove)
```

StartOf Method Example

This example selects the text from the insertion point to the beginning of the line. The number of characters selected is stored in `charmoved`.

```
Selection.Collapse Direction:=wdCollapseStart
charmoved = Selection.StartOf(Unit:=wdLine, Extend:=wdExtend)
```

This example moves `myRange` to the beginning of the second sentence in the document (`myRange` is collapsed and positioned at the beginning of the second sentence). The example uses the **Select** method to show the location of `myRange`.

```
Set myRange = ActiveDocument.Sentences(2)
myRange.StartOf Unit:=wdSentence, Extend:=wdMove
myRange.Select
```

This example moves the selection to the beginning of the paragraph.

```
Selection.StartOf Unit:=wdParagraph, Extend:=wdMove
```

InsertDatabase Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertDatabaseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertDatabaseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertDatabaseA "}

Retrieves data from a data source (for example, a separate Word document, a Microsoft Excel worksheet, or a Microsoft Access database) and inserts the data as a table in place of the specified range.

Syntax

expression.**InsertDatabase(*Format, Style, LinkToSource, Connection, SQLStatement, SQLStatement1, PasswordDocument, PasswordTemplate, WritePasswordDocument, WritePasswordTemplate, DataSource, From, To, IncludeFields*)**

expression Required. An expression that returns a **Range** object.

Format Optional **Variant**. A format listed in the **Formats** box in the **Table AutoFormat** dialog box (**Table** menu). Can be any of the **WdTableFormat** constants. A border is applied to the cells in the table by default.

Style Optional **Variant**. The attributes of the AutoFormat specified by ***Format*** that are applied to the table. Use the sum of any combination of the following values:

Value	Meaning
0 (zero)	None
1	Borders
2	Shading
4	Font
8	Color
16	Auto Fit
32	Heading Rows
64	Last Row
128	First Column
256	Last Column

LinkToSource Optional **Variant**. **True** to establish a link between the new table and the data source.

Connection Optional **Variant**. A range within which to perform the query specified by ***SQLStatement***. How you specify the range depends on how data is retrieved. For example:

- When retrieving data through ODBC (Windows only), you specify a connection string.
- When retrieving data from Microsoft Excel by using dynamic data exchange (DDE), you specify a named range or "Entire Spreadsheet."
- When retrieving data from Microsoft Access (Windows only), you specify the word "Table" or "Query" followed by the name of a table or query.

SQLStatement Optional **String**. An optional query string that retrieves a subset of the data in a primary data source to be inserted into the document (Windows only).

SQLStatement1 Optional **String**. If the query string is longer than 255 characters, ***SQLStatement*** denotes the first portion of the string and ***SQLStatement1*** denotes the second portion (Windows only).

PasswordDocument Optional **Variant**. The password (if any) required to open the data source.

PasswordTemplate Optional **Variant**. If the data source is a Word document, this argument is the password (if any) required to open the attached template.

WritePasswordDocument Optional **Variant**. The password required to save changes to the document.

WritePasswordTemplate Optional **Variant**. The password required to save changes to the template.

DataSource Optional **Variant**. The path and file name of the data source.

From Optional **Variant**. The number of the first data record in the range of records to be inserted.

To Optional **Variant**. The number of the last data record in the range of records to be inserted.

IncludeFields Optional **Variant**. **True** to include field names from the data source in the first row of the new table.

InsertDatabase Method Example

This example inserts a Microsoft Excel spreadsheet named "Data.xls" after the selection . The ***Style*** value (191) is a combination of the numbers 1, 2, 4, 8, 16, 32, and 128.

```
With Selection
    .Collapse Direction:=wdCollapseEnd
    .Range.InsertDatabase Format:=wdTableFormatSimple2, Style:=191, _
        LinkToSource:=False, Connection:="Entire Spreadsheet", _
        DataSource:="C:\MSOffice\Excel\Data.xls"
End With
```

MoveEndWhile Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveEndWhileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveEndWhileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveEndWhileA "}

Moves the ending character position of a range or selection while any of the specified characters are found in the document.

Remarks

While any character in ***Cset*** is found, the end position of the specified range or selection is moved. This method returns the number of characters that the end position of the range or selection moved as a **Long** value. If no ***Cset*** characters are found, the range or selection isn't changed and the method returns 0 (zero). If the end position is moved backward to a point that precedes the original start position, the start position is set to the new end position.

Syntax

expression.**MoveEndWhile(*Cset, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Cset Required **Variant**. One or more characters. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the range or selection is moved forward in the document. If it's a negative number, the range or selection is moved backward. The default value is **wdForward**.

MoveEndWhile Method Example

This example moves the end position of the selection forward while the space character is found.

```
Selection.MoveEndWhile Cset:=" ", Count:=wdForward
```

This example moves the end position of the selection forward while ***Count*** is less than or equal to 10 and any letter from "a" through "h" is found.

```
Selection.MoveEndWhile Cset:="abcdefgh", Count:=10
```

MoveStartWhile Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveStartWhileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveStartWhileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveStartWhileA "}

Moves the start position of the specified range or selection while any of the specified characters are found in the document.

Remarks

While any character in ***Cset*** is found, the start position of the range or selection is moved. This method returns the number of characters that the start position of the range or selection moved as a **Long** value. If not ***Cset*** characters are found, the range or selection isn't changed and the method returns 0 (zero). If the start position is moved forward to a position beyond the original end position, the end position is set to the new start position.

Syntax

expression.**MoveStartWhile(*Cset, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Cset Required **Variant**. One or more characters. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the specified range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the range or selection is moved forward in the document. If it's a negative number, the range or selection is moved backward. The default value is **wdForward**.

MoveStartWhile Method Example

This example moves the start position of the selection backward through the document while the space character is found.

```
Selection.MoveStartWhile Cset:=" ", Count:=wdBackward
```

This example moves the start position of the selection backward through the document while ***Count*** is less than or equal to 10 and any letter from "a" through "h" is found.

```
Selection.MoveStartWhile Cset:="abcdefgh", Count:=-10
```

MoveEndUntil Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveEndUntilC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveEndUntilX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveEndUntilA "}

Moves the end position of the specified range or selection until any of the specified characters are found in the document. If the movement is forward in the document, the range or selection is expanded.

Remarks

This method returns the number of characters by which the end position of the specified range or selection was moved, as a **Long** value. If ***Count*** is greater than 0 (zero), this method returns the number of characters moved plus 1. If ***Count*** is less than 0 (zero), this method returns the number of characters moved minus 1. If no ***Cset*** characters are found, the range or selection isn't changed and the method returns 0 (zero). If the end position is moved backward to a point that precedes the original start position, the start position is set to the new ending position.

Syntax

expression.**MoveEndUntil(*Cset, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Cset Required **Variant**. One or more characters. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the specified range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the range or selection is moved forward in the document. If it's a negative number, the range or selection is moved backward. The default value is **wdForward**.

MoveEndUntil Method Example

This example extends the selection forward in the document until the letter "a" is found. The example then expands the selection by one character to include the letter "a".

```
With Selection
    .MoveEndUntil Cset:="a", Count:=wdForward
    .MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
End With
```

This example extends the selection forward in the document until a tab is found. If a tab character isn't found in the next 100 characters, the selection isn't moved.

```
char = Selection.MoveEndUntil(Cset:=vbTab, Count:=100)
If char = 0 Then StatusBar = "Selection not moved"
```

MoveWhile Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveWhileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveWhileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveWhileA "}

Moves the specified range or selection while any of the specified characters are found in the document.

Remarks

While any character in ***Cset*** is found, the specified range or selection is moved. The resulting **Range** or **Selection** object is positioned as an insertion point after whatever ***Cset*** characters were found. This method returns the number of characters by which the specified range or selection was moved, as a **Long** value. If no ***Cset*** characters are found, the range or selection isn't changed and the method returns 0 (zero).

Syntax

expression.**MoveWhile(*Cset, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Cset Required **Variant**. One or more characters. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the specified range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the specified range or selection is moved forward in the document, beginning at the end position. If it's a negative number, the range or selection is moved backward, beginning at the start position. The default value is **wdForward**.

MoveWhile Method Example

This example moves the selection after consecutive tabs.

```
Selection.MoveWhile Cset:=vbTab, Count:=wdForward
```

This example moves `aRange` while any of the following (uppercase or lowercase) letters are found: "a", "t", or "i".

```
Set aRange = ActiveDocument.Characters(1)
aRange.MoveWhile Cset:="atiATI", Count:=wdForward
```

MoveStart Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveStartC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveStartX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveStartA "}

Moves the start position of the specified range or selection. This method returns an integer that indicates the number of units by which the start position or the range or selection actually moved, or it returns 0 (zero) if the move was unsuccessful.

Syntax

expression.**MoveStart(*Unit, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Unit Optional **Variant**. The unit by which start position of the specified range or selection is to be moved. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, you can also use **wdLine**. The default value is **wdCharacter**.

Count Optional **Variant**. The maximum number of units by which the specified range or selection is to be moved. If ***Count*** is a positive number, the start position of the range or selection is moved forward in the document. If it's a negative number, the start position is moved backward. If the start position is moved forward to a position beyond the end position, the range or selection is collapsed and both the start and end positions are moved together. The default value is 1.

MoveStart Method Example

This example moves the start position of the selection one character forward (the selection size is reduced by one character). Note that a space is considered a character.

```
Selection.MoveStart Unit:=wdCharacter, Count:=1
```

This example moves the start position of the selection to the beginning of the line (the selection is extended to the start of the line).

```
Selection.MoveStart Unit:=wdLine, Count:=-1
```

This example sets `myRange` to be equal to the second word in the active document. The example uses the **MoveStart** method to move the start position of `myRange` (a **Range** object) backward one word. After this macro is run, the first and second words in the document are selected.

```
If ActiveDocument.Words.Count >= 2 Then
    Set myRange = ActiveDocument.Words(2)
    With myRange
        .MoveStart Unit:=wdWord, Count:=-1
        .Select
    End With
End If
```

MoveStartUntil Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveStartUntilC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveStartUntilX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveStartUntilA "}

Moves the start position of the specified range or selection until one of the specified characters is found in the document. If the movement is backward through the document, the range or selection is expanded.

Remarks

This method returns the number of characters by which the start position of the specified range or selection moved, as a **Long** value. If ***Count*** is greater than 0 (zero), this method returns the number of characters moved plus 1. If ***Count*** is less than 0 (zero), this method returns the number of characters moved minus 1. If no ***Cset*** characters are found, the specified range or selection isn't changed and the method returns 0 (zero). If the start position is moved forward to a point beyond the end position, the range or selection is collapsed and both the start and end positions are moved together.

Syntax

expression.**MoveStartUntil(*Cset, Count*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Cset Required **Variant**. One or more characters. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the specified range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the range or selection is moved forward in the document. If it's a negative number, the range or selection is moved backward. The default value is **wdForward**.

MoveStartUntil Method Example

This example extends the selection backward until a capital "I" is found.

```
Selection.MoveStartUntil Cset:="I", Count:=wdBackward
```

If there's a dollar sign character ($) in the first paragraph in the selection, this example moves `myRange` just before the dollar sign.

```
Set myRange = Selection.Paragraphs(1).Range
leng = myRange.End - myRange.Start
myRange.Collapse Direction:=wdCollapseStart
myRange.MoveStartUntil Cset:="$", Count:=leng
```

MoveUntil Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveUntilC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveUntilX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveUntilA "}

Moves the specified range or selection until one of the specified characters is found in the document.

Remarks

This method returns the number of characters by which the specified range or selection was moved, as a **Long** value. If ***Count*** is greater than 0 (zero), this method returns the number of characters moved plus 1. If ***Count*** is less than 0 (zero), this method returns the number of characters moved minus 1. If no ***Cset*** characters are found, the range or selection isn't not changed and the method returns 0 (zero).

Syntax

expression.**MoveUntil(*Cset, Count*)**

expression Required. An expression that returns a **Range** of **Selection** object.

Cset Required **Variant**. One or more characters. If any character in ***Cset*** is found before the ***Count*** value expires, the specified range or selection is positioned as an insertion point immediately before that character. This argument is case sensitive.

Count Optional **Variant**. The maximum number of characters by which the specified range or selection is to be moved. Can be a number or either the **wdForward** or **wdBackward** constant. If ***Count*** is a positive number, the range or selection is moved forward in the document, beginning at the end position. If it's a negative number, the range or selection is moved backward, beginning at the start position. The default value is **wdForward**.

MoveUntil Method Example

This example moves `myRange` forward through the next 100 characters in the document until the character "t" is found.

```
Set myRange = ActiveDocument.Words(1)
myRange.MoveUntil Cset:="t", Count:=100
```

This example moves the selection forward to the end of the active paragraph and then displays the number of characters by which the selection was moved.

```
x = Selection.MoveUntil(Cset:=Chr$(13), Count:=wdForward)
MsgBox x-1 & " character positions were moved"
```

Flags Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFlagsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFlagsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFlagsA"}

Returns or sets properties of the selection. Can be set to one of the following **WdSelectionFlags** constants: **wdSelActive**, **wdSelAtEOL**, **wdSelOvertype**, **wdSelReplace**, or **wdSelStartActive**. The return value of the **Flags** property is the sum of the **WdSelectionFlags** constants that apply to the selection. Read/write **Long**.

Note The **wdSelAtEOL** constant is functionally equivalent to the **IPAtEndOfLine** property.

Flags Property Example

This example selects the first word in the active document. The first message box displays "False" because the end of the selection is active. The **Flags** property makes the beginning of the selection active, and the second message box displays "True."

```
ActiveDocument.Words(1).Select
MsgBox Selection.StartIsActive
Selection.Flags = wdSelStartActive
MsgBox Selection.StartIsActive
```

This example turns on overtype mode for the selection.

```
Selection.Flags = wdSelOvertype
```

StartIsActive Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStartIsActiveC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStartIsActiveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStartIsActiveA"}

True if the beginning of the selection is active. The active end of the selection moves when you press SHIFT+an arrow key. Read/write **Boolean**.

StartIsActive Property Example

This example selects the first word in the active document. The message box displays "False" because the end of the selection is active.

```
ActiveDocument.Words(1).Select
MsgBox Selection.StartIsActive
```

This example collapses the selection and selects the previous word. The message box displays "True" because the beginning of the selection is active.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.MoveLeft Unit:=wdWord, Extend:=wdExtend
MsgBox Selection.StartIsActive
```

Move Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveA "}

Range or **Selection** object: Collapses the specified range or selection to its start position or end position and then moves the collapsed object by the specified number of units. This method returns a value that indicates the number of units by which the object was actually moved, or it returns 0 (zero) if the move was unsuccessful.

Application or **Task** object: Positions a task window or the Word application window.

Syntax 1

expression.**Move(*Unit, Count*)**

Syntax 2

expression.**Move(*Left, Top*)**

expression Required. An expression that returns a **Range** or **Selection** object (Syntax 1) or an expression that returns an **Application** or **Task** object (Syntax 2).

Unit Optional **Variant**. The unit by which the collapsed range or selection is to be moved. Can be one of the following **WdUnits** constants: **wdCharacter**, **wdWord**, **wdSentence**, **wdParagraph**, **wdSection**, **wdStory**, **wdCell**, **wdColumn**, **wdRow**, or **wdTable**. If *expression* returns a **Selection** object, you can also use **wdLine**. The default value is **wdCharacter**.

Count Optional **Variant**. The number of units by which the specified range or selection is to be moved. If ***Count*** is a positive number, the object is collapsed to its end position and moved forward in the document by the specified number of units. If ***Count*** is a negative number, the object is collapsed to its start position and moved backward by the specified number of units. The default value is 1. You can also control the collapse direction by using the **Collapse** method before using the **Move** method.

If the range or selection is in the middle of a unit or isn't collapsed, moving it to the beginning or end of the unit counts as moving it one full unit.

Left Required **Long**. The horizontal screen position of the specified window.

Top Required **Long**. The vertical screen position of the specified window.

Remarks

The start position and end position of a collapsed range or selection are equal.

Applying the **Move** method to a range doesn't rearrange text in the document. Instead, it redefines the range to refer to a new location in the document.

If you apply the **Move** method to any range other than a **Range** object variable (for example,`Selection.Paragraphs(3).Range.Move`), the method has no effect.

Moving a **Selection** object collapses the selection and moves the insertion point either forward or backward in the document.

Move Method Example

This example moves the selection two words to the right and positions the insertion point after the second word's trailing space. If the move is unsuccessful, a message box indicates that the selection is at the end of the document.

```
If Selection.StoryType = wdMainTextStory Then
    wUnits = Selection.Move(Unit:=wdWord, Count:=2)
    If wUnits < 2 Then MsgBox "Selection is at the end of the document"
End If
```

This example sets `Range1` to the first paragraph in the active document and then moves the range forward three paragraphs. After this macro is run, the insertion point is positioned at the beginning of the fourth paragraph.

```
Set Range1 = ActiveDocument.Paragraphs(1).Range
With Range1
    .Collapse Direction:=wdCollapseStart
    .Move Unit:=wdParagraph, Count:=3
    .Select
End With
```

This example moves the selection forward three cells in the table.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Move Unit:=wdCell, Count:=3
End If
```

This example starts the Calculator application (Calc.exe) and uses the **Move** method to reposition the application window.

```
Shell "Calc.exe"
With Tasks("Calculator")
    .WindowState = wdWindowStateNormal
    .Move Top:=50, Left:=50
End With
```

InsertDateTime Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertDateTimeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertDateTimeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertDateTimeA "}

Inserts the current date or time, or both, either as text or as a TIME field.

Syntax

expression.**InsertDateTime(*DateTimeFormat, InsertAsField, InsertAsFullWidth*)**

expression Required. An expression that returns a **Range** or **Selection** object.

DateTimeFormat Optional **Variant**. The format to be used for displaying the date or time, or both. If this argument is omitted, Word uses the short-date style from the Windows Control Panel (**Regional Settings** icon).

InsertAsField Optional **Variant**. **True** to insert the specified information as a TIME field. The default value is **True**.

InsertAsFullWidth Optional **Variant**. Not used in the U.S. English version of Microsoft Word.

InsertDateTime Method Example

This example inserts a TIME field for the current date. A possible result might be "November 18, 1996."

```
Selection.InsertDateTime DateTimeFormat:="MMMM dd, yyyy", _
    InsertAsField:=True
```

This example inserts the current date at the end of the active document. A possible result might be "01/12/97."

```
With ActiveDocument.Content
    .Collapse Direction:=wdCollapseEnd
    .InsertDateTime DateTimeFormat:="MM/dd/yy", InsertAsField:=False
End With
```

This example inserts a TIME field for the current date in the footer for the active document.

```
ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary).Range _
    .InsertDateTime DateTimeFormat:="MMMM dd, yyyy", _
    InsertAsField:=True
```

FormattedText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFormattedTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFormattedTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFormattedTextA "}

Returns or sets a **Range** object that includes the formatted text in the specified range or selection. Read/write.

Remarks

This property returns a **Range** object with the character formatting and text from the specified range or selection. Paragraph formatting is included in the **Range** object if there's a paragraph mark in the range or selection.

When you set this property, the text in the range is replaced with formatted text. If you don't want to replace the existing text, use the **Collapse** method before using this property (see the first example).

FormattedText Property Example

This example copies the first paragraph in the document, including its formatting, and inserts the formatted text at the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.FormattedText = ActiveDocument.Paragraphs(1).Range
```

This example copies the text and formatting from the selection into a new document.

```
Set myRange = Selection.FormattedText
Documents.Add.Content.FormattedText = myRange
```

GoToNext Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGoToNextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGoToNextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGoToNextA "}

Range or **Selection** object: Returns a **Range** object that refers to the start position of the next item or location specified by the ***What*** argument. If you apply this method to the **Selection** object, the method moves the selection to the specified item (except for the **wdGoToGrammaticalError**, **wdGoToProofreadingError**, and **wdGoToSpellingError** constants).

Note When you use this method with the **wdGoToGrammaticalError**, **wdGoToProofreadingError**, or **wdGoToSpellingError** constant, the **Range** object that's returned includes any grammar error text or spelling error text.

MailMessage object: Displays the next mail message if WordMail is available.

Syntax 1

expression.**GoToNext(*What*)**

Syntax 2

expression.**GoToNext**

expression Required. An expression that returns a **Range** or **Selection** object (Syntax 1) or a **MailMessage** object (Syntax 2).

What Required **Long**. The item that the specified range or selection it to be moved to. Can be one of the following **WdGoToItem** constants: **wdGoToBookmark**, **wdGoToComment**, **wdGoToEndnote**, **wdGoToEquation**, **wdGoToField**, **wdGoToFootnote**, **wdGoToGrammaticalError**, **wdGoToGraphic**, **wdGoToHeading**, **wdGoToLine**, **wdGoToObject**, **wdGoToPage**, **wdGoToPercent**, **wdGoToProofreadingError**, **wdGoToSection**, **wdGoToSpellingError**, or **wdGoToTable**.

GoToNext Method Example

This example adds a bookmark at the top of page 2 in the active document.

```
Set myRange = ActiveDocument.Words(1).GoToNext(What:=wdGoToPage)
ActiveDocument.Bookmarks.Add Name:="Page2", Range:=myRange
```

This example moves to the next field and selects it.

```
With Selection
    Set myRange = .GoToNext(What:=wdGoToField)
    .MoveRight Unit:=wdWord, Extend:=wdExtend
    .Fields(1).Select
End With
```

GoToPrevious Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGoToPreviousC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGoToPreviousX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGoToPreviousA "}

Range or **Selection** object: Returns a **Range** object that refers to the start position of the previous item or location specified by the ***What*** argument. If applied to a **Selection** object, **GoToPrevious** moves the selection to the specified item.

MailMessage object: Displays the previous mail message if WordMail is available.

Syntax 1

expression.**GoToPrevious(*What*)**

Syntax 2

expression.**GoToPrevious**

expression Required. An expression that returns a **Range** or **Selection** object (Syntax 1) or a **MailMessage** object (Syntax 2).

What Required **Long**. The item that the specified range or selection is to be moved to. Can be one of the following **WdGoToItem** constants: **wdGoToBookmark**, **wdGoToComment**, **wdGoToEndnote**, **wdGoToEquation**, **wdGoToField**, **wdGoToFootnote**, **wdGoToGrammaticalError**, **wdGoToGraphic**, **wdGoToHeading**, **wdGoToLine**, **wdGoToObject**, **wdGoToPage**, **wdGoToPercent**, **wdGoToProofreadingError**, **wdGoToSection**, **wdGoToSpellingError**, or **wdGoToTable**.

GoToPrevious Method Example

This example moves to the previous field in the active document.

```
Selection.GoToPrevious What:=wdGoToField
```

This example creates a range that references the last footnote reference marker in the active document.

```
Set myRange = ActiveDocument.Words.Last.GoToPrevious(What:=wdGoToFootnote)
myRange.Expand Unit:=wdCharacter
```

CopyAsPicture Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCopyAsPictureC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCopyAsPictureX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCopyAsPictureA "}

On the Macintosh, this method copies the specified range or selection to the Clipboard as a picture. In Windows, **CopyAsPicture** works the same way as the **Copy** method.

Syntax

expression.**CopyAsPicture**

expression Required. An expression that returns a **Range** or **Selection** object.

CopyAsPicture Method Example

On the Macintosh, this example copies the first paragraph in the active document to the Clipboard as a picture, and then it pastes the picture into the next document.

```
ActiveDocument.Paragraphs(1).Range.CopyAsPicture
If Documents.Count >= 2 Then
    Documents(2).Activate
    ActiveDocument.Content.Paste
End If
```

InsertSymbol Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertSymbolC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertSymbolX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertSymbolA "}

Inserts a symbol in place of the specified range or selection.

Note If you don't want to replace the range or selection, use the **Collapse** method before you use this method.

Syntax

expression.**InsertSymbol(*CharacterNumber, Font, Unicode*)**

expression Required. An expression that returns a **Range** or **Selection** object.

CharacterNumber Required **Long**. The character number for the specified symbol. This value will always be the sum of 31 and the number that corresponds to the position of the symbol in the table of symbols (counting from left to right). For example, to specify a delta character at position 37 in the table of symbols in the Symbol font, set ***CharacterNumber*** to 68.

Font Optional **Variant**. The name of the font that contains the symbol.

Unicode Optional **Variant**. **True** to insert the unicode character specifed by ***CharacterNumber***; **False** to insert the ANSI character specified by ***CharacterNumber***. The default value is **False**.

InsertSymbol Method Example

This example inserts a double-headed arrow at the insertion point.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .InsertSymbol CharacterNumber:=171, Font:="Symbol", Unicode:=False
End With
```

This example inserts a bullet and a tab stop at the beginning of the first paragraph in the selection.

```
Set myRange = Selection.Paragraphs(1).Range
With myRange
    .Collapse Direction:=wdCollapseStart
    .InsertSymbol CharacterNumber:=183, Font:="Symbol", Unicode:=False
    .MoveStart Unit:=wdCharacter, Count:=1
    .InsertAfter vbTab
End With
```

PasteSpecial Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPasteSpecialC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPasteSpecialX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPasteSpecialA "}

Inserts the contents of the Clipboard. Unlike with the **Paste** method, with **PasteSpecial** you can control the format of the pasted information and (optionally) establish a link to the source file (for example, a Microsoft Excel worksheet).

Note If you don't want to replace the contents of the specified range or selection, use the **Collapse** method before you use this method. When you use this method, the range or selection doesn't expand to include the contents of the Clipboard.

Syntax

expression.**PasteSpecial(*IconIndex, Link, Placement, DisplayAsIcon, DataType, IconFileName, IconLabel*)**

expression Required. An expression that returns a **Range** or **Selection** object.

IconIndex Optional **Variant**. If ***DisplayAsIcon*** is **True**, 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 (**Insert** menu, **Object** command, **Create New** tab): 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. On the Macintosh, this argument is ignored, and therefore you cannot change icons for embedded objects.

Link Optional **Variant**. **True** to create a link to the source file of the Clipboard contents. The default value is **False**.

Placement Optional **Variant**. Can be either of the following **WdOLEPlacement** constants: **wdFloatOverText** or **wdInLine**. The default value is **wdInLine**.

DisplayAsIcon Optional **Variant**. **True** to display the link as an icon. The default value is **False**.

DataType Optional **Variant**. A format for the Clipboard contents when they're inserted into the document. Can be one of the following **WdPasteDataType** constants: **wdPasteBitmap**, **wdPasteDeviceIndependentBitmap**, **wdPasteEnhancedMetafile**, **wdPasteHyperlink**, **wdPasteMetafilePicture**, **wdPasteOLEObject**, **wdPasteRTF**, **wdPasteShape** or **wdPasteText**. The default format varies, depending on the contents of the Clipboard.

IconFileName Optional **Variant**. If ***DisplayAsIcon*** is **True**, this argument is the path and file name for the file in which the icon to be displayed is stored.

IconLabel Optional **Variant**. If ***DisplayAsIcon*** is **True**, this argument is the text that appears below the icon.

PasteSpecial Method Example

This example inserts the Clipboard contents at the insertion point as unformatted text.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.Range.PasteSpecial DataType:=wdPasteText
```

This example copies the selected text and pastes it into a new document as a hyperlink. The source document must first be saved for this example to work.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Copy
    Documents.Add.Content.PasteSpecial Link:=True,
DataType:=wdPasteHyperlink
End If
```

GoTo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGoToC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGoToX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGoToA "}

Document or **Range** object: Returns a **Range** object that represents the start position of the specified item, such as a page, bookmark, or field.

Selection object: Moves the insertion point to the character position immediately preceding the specified item, and returns a **Range** object (except for the **wdGoToGrammaticalError**, **wdGoToProofreadingError**, or **wdGoToSpellingError** constant).

Note When you use this method with the **wdGoToGrammaticalError**, **wdGoToProofreadingError**, or **wdGoToSpellingError** constant, the **Range** that's returned includes any grammar error text or spelling error text.

Syntax

expression.**GoTo(*What, Which, Count, Name*)**

expression Required. An expression that returns a **Document**, **Range**, or **Selection** object.

What Optional **Variant**. The kind of item that the range or selection is to be moved to. Can be one of the following **WdGoToItem** constants:

wdGoToBookmark	**wdGoToLine**
wdGoToComment	**wdGoToObject**
wdGoToEndnote	**wdGoToPage**
wdGoToEquation	**wdGoToPercent**
wdGoToField	**wdGoToProofreadingError**
wdGoToFootnote	**wdGoToSection**
wdGoToGrammaticalError	**wdGoToSpellingError**
wdGoToGraphic	**wdGoToTable**
wdGoToHeading	

Which Optional **Variant**. The item that the range or selection is to be moved to. Can be one of the following **WdGoToDirection** constants: **wdGoToAbsolute**, **wdGoToFirst**, **wdGoToLast**, **wdGoToNext**, **wdGoToPrevious**, or **wdGoToRelative**.

The following examples are functionally equivalent, they both move the selection to the first heading in the document.

```
Selection.GoTo What:=wdGoToHeading, Which:=wdGoToFirst
Selection.GoTo What:=wdGoToHeading, Which:=wdGoToAbsolute, Count:=1
```

Count Optional **Variant**. The number of the item in the document. The default value is 1. The following example moves the selection to the fourth line in the document.

```
Selection.GoTo What:=wdGoToLine, Which:=wdGoToAbsolute, Count:=4
```

Only positive values are valid. To specify an item that precedes the range or selection, use **wdGoToPrevious** as the ***Which*** argument and specify a ***Count*** value. The following example moves the selection up two lines.

```
Selection.GoTo What:=wdGoToLine, Which:=wdGoToPrevious, Count:=2
```

Name Optional **Variant**. If the ***What*** argument is **wdGoToBookmark**, **wdGoToComment**, **wdGoToField**, or **wdGoToObject**, this argument specifies a name. The following example moves to the next DATE field.

```
Selection.GoTo What:=wdGoToField, Name:="Date"
```

GoTo Method Example

This example moves the selection to the first cell in the next table.

```
Selection.GoTo What:=wdGoToTable, Which:=wdGoToNext
```

This example moves the insertion point just before the fifth endnote reference mark in the active document.

```
If ActiveDocument.Endnotes.Count >= 5 Then
    Selection.GoTo What:=wdGoToEndnote, Which:=wdGoToAbsolute, Count:=5
End If
```

This example sets `R1` equal to the first footnote reference mark in the active document.

```
If ActiveDocument.Footnotes.Count >= 1 Then
    Set R1 = ActiveDocument.GoTo(What:=wdGoToFootnote, Which:=wdGoToFirst)
    R1.Expand Unit:=wdCharacter
End If
```

This example moves the selection down four lines.

```
Selection.GoTo What:=wdGoToLine, Which:=wdGoToRelative, Count:=4
```

This example moves the selection back two pages.

```
Selection.GoTo What:=wdGoToPage, Which:=wdGoToPrevious, Count:=2
```

EscapeKey Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEscapeKeyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEscapeKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEscapeKeyA"}

Cancels a mode such as extend or column select (equivalent to pressing the ESC key).

Syntax

expression.**EscapeKey**

expression Required. An expression that returns a **Selection** object.

EscapeKey Method Example

This example turns on and then cancels extend mode.

```
With Selection
    .ExtendMode = True
    .EscapeKey
End With
```

LanguageIDFarEast Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguageIDFarEastC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguageIDFarEastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguageIDFarEastA"}

Returns or sets a Far East language for the specified object. Can be any of the **WdLanguageID** constants. Read/write **Long**.

Note This is the recommended way to apply a Far East language to text in a document created in a Far East version of Word.

LanguageIDFarEast Property Example

This example sets the language of the selection to Korean.

```
Selection.LanguageIDFarEast = wdKorean
```

LanguageIDOther Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguageIDOtherC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguageIDOtherX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguageIDOtherA"}

Returns or sets the language for the specified object. Can be any of the **WdLanguageID** constants. Read/write **Long**.

Note This is the recommended way to determine the language of a document created in a Far East version of Word.

LanguageIDOther Property Example

This example sets the language of the selection to French.

```
Selection.LanguageIDOther = wdFrench
```

NameFarEast Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNameFarEastC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNameFarEastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNameFarEastA"}

Returns or sets a Far East font name. Read/write **String**.

Note In the U.S. English version of Microsoft Word, this property defaults to Times New Roman. This is the recommended way to apply a Far East font to text in a document created in a Far East version of Word.

NameFarEast Property Example

This example displays the Far East font name that's applied to the selection.

```
MsgBox Selection.Font.NameFarEast
```

NameOther Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNameOtherC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNameOtherX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNameOtherA"}

Returns or sets the font used for characters whose character set numbers are greater than 127. Read/write **String**.

Note In the U.S. English version of Microsoft Word, this property defaults to Times New Roman. Use the **Name** property to change the font that's applied to the text and that appears on the **Formatting** toolbar.

NameOther Property Example

This example sets the font used for characters whose character set numbers are greater than 127.

```
Selection.Font.NameOther = "Century"
```

NameAscii Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNameAsciiC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNameAsciiX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNameAsciiA"}

Returns or set the font used for characters whose character set numbers are from 0 (zero) through 127. Read/write **String**.

Note In the U.S. English version of Microsoft Word, this property defaults to Times New Roman. Use the **Name** property to change the font that's applied to the text and that appears on the **Formatting** toolbar.

NameAscii Property Example

This example sets the font used for characters whose character set numbers are from 0 (zero) through 127.

```
Selection.Font.NameAscii = "Century"
```

NextStoryRange Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNextStoryRangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNextStoryRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNextStoryRangeA"}

Returns a **Range** object that refers to the next story, as shown in the following table. Read-only.

Story type	Item returned by the NextStoryRange method
wdMainTextStory, **wdFootnotesStory**, **wdEndnotesStory**, and **wdCommentsStory**	Always returns **Nothing**
wdTextFrameStory	The story of the next set of linked text boxes
wdEvenPagesHeaderStory, **wdPrimaryHeaderStory**, **wdEvenPagesFooterStory**, **wdPrimaryFooterStory**, **wdFirstPageHeaderStory**, **wdFirstPageFooterStory**	The next section's story of the same type

NextStoryRange Property Example

This example adds text to the even headers in the first two sections in the active document.

```
If ActiveDocument.Sections.Count >= 2 Then
    With ActiveDocument
        .PageSetup.OddAndEvenPagesHeaderFooter = True
        .Sections(1).Headers(wdHeaderFooterEvenPages).Range.Text = "Even
Header"
        .Sections(2).Headers(wdHeaderFooterEvenPages).LinkToPrevious =
False
        .StoryRanges(wdEvenPagesHeaderStory).NextStoryRange.Text = "Even
Header 2"
    End With
End If
```

This example searches each story in the active document for the text "Microsoft Word." The example also applies italic formatting to any instances of this text that it finds.

```
For Each myStoryRange In ActiveDocument.StoryRanges
    myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
    While myStoryRange.Find.Found
        myStoryRange.Italic = True
        myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
    Wend
    While Not (myStoryRange.NextStoryRange Is Nothing)
        Set myStoryRange = myStoryRange.NextStoryRange
        myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
        While myStoryRange.Find.Found
            myStoryRange.Italic = True
            myStoryRange.Find.Execute FindText:="Microsoft Word",
Forward:=True
        Wend
    Wend
Next myStoryRange
```

CommandParameter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCommandParameterC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCommandParameterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCommandParameterA"}

Returns the command parameter assigned to the specified shortcut key. Read-only **String**.

Note For information about commands that take parameters, see Add Method (KeyBindings Object). Use the **Command** property to return the command name assigned to the specified shortcut key.

CommandParameter Property Example

This example assigns a shortcut key to the **FontSize** command, with a command parameter of 8. Use the **CommandParameter** property to display the command parameter along with the command name and key string.

```
Set aKey = KeyBindings.Add(KeyCategory:=wdKeyCategoryCommand,
Command:="FontSize", _
    KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyS),
CommandParameter:="8")
MsgBox aKey.Command & Chr$(32) & aKey.CommandParameter & vbCr &
aKey.KeyString
```

ColumnSelectMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColumnSelectModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColumnSelectModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColumnSelectModeA "}

True if column selection mode is active. When this mode is active, the letters "COL" appear on the status bar. Read/write **Boolean**.

ColumnSelectMode Property Example

This example selects a column of text that's two words across and three lines deep. The example copies the selection to the Clipboard and cancels column selection mode.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .ColumnSelectMode = True
    .MoveRight Unit:=wdWord, Count:=2, Extend:=wdExtend
    .MoveDown Unit:=wdLine, Count:=2, Extend:=wdExtend
    .Copy
    .ColumnSelectMode = False
End With
```

CopyFormat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCopyFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCopyFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCopyFormatA "}

Copies the character formatting of the first character in the selected text. If a paragraph mark is selected, Word copies paragraph formatting in addition to character formatting.

Note You can apply the copied formatting to another selection by using the **PasteFormat** method.

Syntax

expression.**CopyFormat**

expression Required. An expression that returns a **Selection** object.

CopyFormat Method Example

This example copies the formatting of the first paragraph to the second paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Range.Select
Selection.CopyFormat
ActiveDocument.Paragraphs(2).Range.Select
Selection.PasteFormat
```

This example collapses the selection and copies its character formatting to the next word.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .CopyFormat
    .Next(Unit:=wdWord, Count:=1).Select
    .PasteFormat
End With
```

ExtendMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExtendModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExtendModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExtendModeA "}

True if extend selection mode is active. When this mode is active, the letters "EXT" appear on the status bar. Read/write **Boolean**.

ExtendMode Property Example

This example moves the selection to the beginning of the active line and then extends the selection down two lines. The example applies italic formatting to the selection and cancels extend selection mode.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .HomeKey Unit:=wdLine, Extend:=wdMove
    .ExtendMode = True
    .MoveDown Unit:=wdLine, Count:=2
    .Font.Italic = True
    .ExtendMode = False
End With
```

This example selects all text in the selection from the insertion point through the bookmark named "Temp."

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .ExtendMode = True
    Selection.GoTo What:=wdGoToBookmark, Name:="temp"
    .ExtendMode = False
End With
```

HighlightColorIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHighlightColorIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHighlightColorIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHighlightColorIndexA "}

Returns or sets the highlight color for the specified range. Can be one of the following **WdColorIndex** constants: **wdNoHighlight**, **wdBlack**, **wdBlue**, **wdBrightGreen**, **wdDarkBlue**, **wdDarkRed**, **wdDarkYellow**, **wdGray25**, **wdGray50**, **wdGreen**, **wdPink**, **wdRed**, **wdTeal**, **wdTurquoise**, **wdViolet**, **wdWhite**, or **wdYellow**. Read/write **Long**.

Note Setting this property to **wdNoHighlight** removes the highlight color (if any) from the specified range.

HighlightColorIndex Property Example

This example removes highlight formatting from the selection.

```
Selection.Range.HighlightColorIndex = wdNoHighlight
```

This example applies yellow highlighting to each bookmark in the active document.

```
For Each abookmark In ActiveDocument.Bookmarks
    abookmark.Range.HighlightColorIndex = wdYellow
Next abookmark
```

InsertAutoText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertAutoTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertAutoTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertAutoTextA "}

Attempts to match the text in the specified range or the text surrounding the range with an existing AutoText entry name. If any such match is found, **InsertAutoText** inserts the AutoText entry to replace that text. If a match cannot be found, an error occurs.

Note You can use the **Insert** method with an **AutoTextEntry** object to insert a specific AutoText entry.

Syntax

expression.**InsertAutoText**

expression Required. An expression that returns a **Range** object.

InsertAutoText Method Example

This example inserts an AutoText entry that matches the text around a selection.

```
Documents.Add
Selection.TypeText "Best w"
Selection.Range.InsertAutoText
```

This example inserts an AutoText entry with a name that matches the first word in the active document.

```
Documents.Add
Selection.TypeText "In "
Set myRange = ActiveDocument.Words(1)
myRange.InsertAutoText
```

PasteFormat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPasteFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPasteFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPasteFormatA "}

Applies formatting copied with the **CopyFormat** method to the selection. If a paragraph mark was selected when the **CopyFormat** method was used, Word applies paragraph formatting in addition to character formatting.

Syntax

expression.**PasteFormat**

expression Required. An expression that returns a **Selection** object.

PasteFormat Method Example

This example copies the paragraph and character formatting from the first paragraph in the selection to the next paragraph in the selection.

```
With Selection
    .Paragraphs(1).Range.Select
    .CopyFormat
    .Paragraphs(1).Next.Range.Select
    .PasteFormat
End With
```

This example collapses the selection and copies the character formatting to the next word.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .CopyFormat
    .Next(Unit:=wdWord, Count:=1).Select
    .PasteFormat
End With
```

TextRetrievalMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTextRetrievalModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTextRetrievalModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTextRetrievalModeA "}

Returns a **TextRetrievalMode** object that controls how text is retrieved from the specified **Range**. Read/write.

TextRetrievalMode Property Example

This example retrieves the selected text (excluding any hidden text) and inserts it at the beginning of the third paragraph in the active document.

```
If Selection.Type = wdSelectionNormal Then
    Set Range1 = Selection.Range
    Range1.TextRetrievalMode.IncludeHiddenText = False
    Set Range2 = ActiveDocument.Paragraphs(2).Range
    Range2.InsertAfter Range1.Text
End If
```

This example retrieves and displays the first three paragraphs as they appear in outline view.

```
Set myRange =
ActiveDocument.Range(Start:=ActiveDocument.Paragraphs(1).Range.Start, _
    End:=ActiveDocument.Paragraphs(3).Range.End)
myRange.TextRetrievalMode.ViewType = wdOutlineView
MsgBox myRange.Text
```

This example excludes field codes and hidden text from the range that refers to the selected text. The example then displays the text in a message box.

```
If Selection.Type = wdSelectionNormal Then
    Set aRange = Selection.Range
    With aRange.TextRetrievalMode
        .IncludeHiddenText = False
        .IncludeFieldCodes = False
    End With
    MsgBox aRange.Text
End If
```

InsertBefore Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertBeforeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertBeforeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertBeforeA "}

Inserts the specified text before the specified selection or range. After the text is inserted, the selection or range is expanded to include the new text.

Note You can insert characters such as quotation marks, tab characters, and nonbreaking hyphens by using the **Chr** function with the **InsertBefore** method. You can also use the following Visual Basic constants: **vbCr**, **vbLf**, **vbCrLf** and **vbTab**.

Syntax

expression.**InsertBefore(*Text*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Text Required **String**. The text to be inserted.

InsertBefore Method Example

This example inserts the text "Hamlet" (enclosed in quotation marks) before the selection and then collapses the selection.

```
With Selection
    .InsertBefore Chr(34) & "Hamlet" & Chr(34) & Chr(32)
    .Collapse Direction:=wdCollapseEnd
End With
```

This example inserts the text "Introduction" as a separate paragraph at the beginning of the active document.

```
With ActiveDocument.Content
    .InsertParagraphBefore
    .InsertBefore "Introduction"
End With
```

This example inserts all the font names in the **FontNames** collection into a new document.

```
Documents.Add
For Each aFont In FontNames
    With Selection
        .InsertBefore aFont
        .Collapse Direction:=wdCollapseEnd
        .TypeParagraph
    End With
Next aFont
```

BookmarkID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBookmarkIDC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBookmarkIDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBookmarkIDA"}

Returns the number of the bookmark that encloses the beginning of the specified selection or range; returns 0 (zero) if there's no corresponding bookmark. The number corresponds to the position of the bookmark in the document − 1 for the first bookmark, 2 for the second one, and so on. Read-only **Long**.

BookmarkID Property Example

This example displays the number of the bookmark that encloses the beginning of the selection.

```
MsgBox "Bookmark " & Selection.BookmarkID
```

This example adds a bookmark named "temp" at the beginning of the document if there's not already a bookmark set for that location.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseStart
If myRange.BookmarkID = 0 Then
    ActiveDocument.Bookmarks.Add Name:="temp", Range:=myRange
End If
```

InsertParagraphBefore Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertParagraphBeforeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertParagraphBeforeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertParagraphBeforeA"}

Inserts a new paragraph before the specified selection or range.

Note After this method is applied, the range or selection expands to include the new paragraph.

Syntax

expression.**InsertParagraphBefore**

expression Required. An expression that returns a **Selection** or **Range** object.

InsertParagraphBefore Method Example

This example inserts a new paragraph at the beginning of the active document.

```
ActiveDocument.Range(Start:=0, End:=0).InsertParagraphBefore
```

This example inserts the text "Hello" as a new paragraph before the selection.

```
With Selection
    .InsertParagraphBefore
    .InsertBefore "Hello"
End With
```

PreviousBookmarkID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPreviousBookmarkIDC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPreviousBookmarkIDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPreviousBookmarkIDA"}

Returns the number of the last bookmark that starts before or at the same place as the specified selection or range; returns 0 (zero) if there's no corresponding bookmark. Read-only **Long**.

PreviousBookmarkID Property Example

This example selects the previous bookmark in the active document.

```
num = Selection.PreviousBookmarkID
If num <> 0 Then ActiveDocument.Content.Bookmarks(num).Select
```

This example displays the name of the bookmark that precedes the second paragraph.

```
num = ActiveDocument.Paragraphs(2).Range.PreviousBookmarkID
If num <> 0 Then MsgBox ActiveDocument.Content.Bookmarks(num).Name
```

HeaderFooter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeaderFooterC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeaderFooterA"}

Returns a **HeaderFooter** object for the specified selection or range. Read-only.

Note An error occurs if the selection isn't located within a header or footer.

HeaderFooter Property Example

This example adds a centered page number to the current page footer.

```
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageFooter
End With
Selection.HeaderFooter.PageNumbers.Add
PageNumberAlignment:=wdAlignPageNumberCenter
```

IPAtEndOfLine Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIPAtEndOfLineC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIPAtEndOfLineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIPAtEndOfLineA"}

True if the insertion point is at the end of a line that wraps to the next line. **False** if the selection isn't collapsed, if the insertion point isn't at the end of a line, or if the insertion point is positioned before a paragraph mark. Read-only **Boolean**.

IPAtEndOfLine Property Example

If the insertion point isn't already at the end of the line, this example moves it there.

```
Selection.Collapse Direction:=wdCollapseEnd
If Selection.IPAtEndOfLine = False Then
    Selection.EndKey Unit:=wdLine, Extend:=wdMove
End If
```

IsEndOfRowMark Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsEndOfRowMarkC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsEndOfRowMarkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsEndOfRowMarkA"}

True if the specified selection or range is collapsed and is located at the end-of-row mark in a table. Read-only **Boolean**.

Note This property is the equivalent of the following expression:

```
Selection.Information(wdAtEndOfRowMarker)
```

IsEndOfRowMark Property Example

This example collapses the selection and selects the current row if the insertion point is at the end of the row (just before the end-of-row mark).

```
Selection.Collapse Direction:=wdCollapseEnd
If Selection.IsEndOfRowMark = True Then
    Selection.Rows(1).Select
End If
```

EndKey Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthEndKeyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthEndKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthEndKeyA"}

Moves or extends the seletion to the end of the specified unit. This method returns an integer that indicates the number of characters the selection or active end was actually moved, or it returns 0 (zero) if the move was unsuccessful.

Note This method corresponds to functionality of the END key.

Syntax

expression.**EndKey(*Unit, Extend*)**

expression Required. An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which the selection is to be moved or extended. Can be one of the following **WdUnits** constants: **wdStory**, **wdColumn**, **wdLine**, or **wdRow**. The default value is **wdLine**.

Extend Optional **Variant**. Specifies the way the selection is moved. Can be one of the following **WdMovementType** constants: **wdMove** or **wdExtend**. If the value of this argument is **wdMove**, the selection is collapsed to an insertion point and moved to the end of the specified unit. If it's **wdExtend**, the end of the selection is extended to the end of the specified unit. The default value is **wdMove**.

EndKey Method Example

This example moves the selection to the end of the current line and assigns the number of characters moved to the `pos` variable.

```
pos = Selection.EndKey(Unit:=wdLine, Extend:=wdMove)
```

This example moves the selection to the beginning of the current table column and then extends the selection to the end of the column.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.HomeKey Unit:=wdColumn, Extend:=wdMove
    Selection.EndKey Unit:=wdColumn, Extend:=wdExtend
End If
```

This example moves the selection to the end of the current story. If the selection is in the main text story, the example moves the selection to the end of the document.

```
Selection.EndKey Unit:=wdStory, Extend:=wdMove
```

HomeKey Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthHomeKeyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthHomeKeyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthHomeKeyA"}

Moves or extends the selection to the beginning of the specified unit. This method returns an integer that indicates the number of characters the selection was actually moved, or it returns 0 (zero) if the move was unsuccessful.

Note This method corresponds to functionality of the HOME key.

Syntax

expression.**HomeKey**(***Unit, Extend***)

expression An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which the selection is to be moved or extended. Can be one of the following **WdUnits** constants: **wdStory**, **wdColumn**, **wdLine**, or **wdRow**. The default value is **wdLine**.

Extend Optional **Variant**. Specifies the way the selection is moved. Can be one of the following **WdMovementType** constants: **wdMove** or **wdExtend**. If the value of this argument is **wdMove**, the selection is collapsed to an insertion point and moved to the beginning of the specified unit. If it's **wdExtend**, the beginning of the selection is extended to the beginning of the specified unit. The default value is **wdMove**.

HomeKey Method Example

This example moves the selection to the beginning of the current story. If the selection is in the main text story, the selection is moved to the beginning of the document.

```
Selection.HomeKey Unit:=wdStory, Extend:=wdMove
```

This example moves the selection to the beginning of the current line and assigns the number of characters moved to the `pos` variable.

```
pos = Selection.HomeKey(Unit:=wdLine, Extend:=wdMove)
If pos = 0 Then StatusBar = "Selection was not moved"
```

LookupNameProperties Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthLookupNamePropertiesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthLookupNamePropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthLookupNamePropertiesA"}

Looks up a name in the global address book list and displays the **Properties** dialog box, which includes information about the specified name. If this method finds more than one match, it displays the **Check Names** dialog box.

Syntax 1

expression.**LookupNameProperties(*Name*)**

Syntax 2

expression.**LookupNameProperties**

expression Required. An expression that returns an **Application** object (Syntax 1) or a **Range** object (Syntax 2).

Name Required **String**. A name in the global address book.

LookupNameProperties Method Example

This example looks up the name Don Funk in the address book and displays the **Properties** dialog box for Don Funk.

```
Application.LookupNameProperties Name:="Don Funk"
```

This example looks up the selected name in the address book and displays the **Properties** dialog box for that person.

```
Selection.Range.LookupNameProperties
```

SelectCurrentAlignment Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentAlignmentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentAlignmentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentAlignmentA"}

Extends the selection forward until text with a different paragraph alignment is encountered.

Syntax

expression.**SelectCurrentAlignment**

expression Required. An expression that returns a **Selection** object.

Remarks

There are four types of paragraph alignment: left, centered, right, and justified.

SelectCurrentAlignment Method Example

This example positions the insertion point at the beginning of the first paragraph after the current paragraph that doesn't have the same alignment as the current paragraph. If the alignment is the same from the selection to the end of the document, the example moves the selection to the end of the document and displays a message.

```
Selection.SelectCurrentAlignment
Selection.Collapse Direction:=wdCollapseEnd
If Selection.End = ActiveDocument.Content.End - 1 Then
    MsgBox "No change in alignment found."
End If
```

SelectCurrentColor Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentColorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentColorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentColorA"}

Extends the selection forward until text with a different color is encountered.

Syntax

expression.**SelectCurrentColor**

expression Required. An expression that returns a **Selection** object.

SelectCurrentColor Method Example

This example extends the selection from the beginning of the document to the first character formatted with a different color and then displays the number of characters in the resulting selection.

```
Selection.HomeKey Unit:=wdStory, Extend:=wdMove
Selection.SelectCurrentColor
n = Len(Selection.Text)
MsgBox "Contiguous characters with the same color: " & n
```

SelectCurrentFont Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentFontC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentFontX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentFontA"}

Extends the selection forward until text in a different font or font size is encountered.

Syntax

expression.**SelectCurrentFont**

expression Required. An expression that returns a **Selection** object.

SelectCurrentFont Method Example

This example extends the selection until text in a different font or font size is encountered. The example uses the **Grow** method to increase the size of the selected text to the next available font size.

```
With Selection
    .SelectCurrentFont
    .Font.Grow
End With
```

SelectCurrentIndent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentIndentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentIndentA"}

Extends the selection forward until text with different left or right paragraph indents is encountered.

Syntax

expression.**SelectCurrentIndent**

expression Required. An expression that returns a **Selection** object.

SelectCurrentIndent Method Example

This example jumps to the beginning of the first paragraph in the document that has different indents than the first paragraph in the active document.

```
With Selection
    .HomeKey Unit:=wdStory, Extend:=wdMove
    .SelectCurrentIndent
    .Collapse Direction:=wdCollapseEnd
End With
```

This example determines whether all the paragraphs in the active document are formatted with the same left and right indents and then displays a message box indicating the result.

```
With Selection
    .HomeKey Unit:=wdStory, Extend:=wdMove
    .SelectCurrentIndent
    .Collapse Direction:=wdCollapseEnd
End With
If Selection.End = ActiveDocument.Content.End - 1 Then
    MsgBox "All paragraphs share the same left and right indents."
Else
    MsgBox "Not all paragraphs share the same left and right indents."
End If
```

SelectCurrentSpacing Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentSpacingC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentSpacingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentSpacingA"}

Extends the selection forward until a paragraph with different line spacing is encountered.

Syntax

expression.**SelectCurrentSpacing**

expression Required. An expression that returns a **Selection** object.

SelectCurrentSpacing Method Example

This example selects all consecutive paragraphs that have the same line spacing and changes the line spacing to single spacing.

```
With Selection
    .SelectCurrentSpacing
    .ParagraphFormat.Space1
End With
```

SelectCurrentTabs Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectCurrentTabsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectCurrentTabsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectCurrentTabsA"}

Extends the selection forward until a paragraph with different tab stops is encountered.

Syntax

expression.**SelectCurrentTabs**

expression Required. An expression that returns a **Selection** object.

SelectCurrentTabs Method Example

This example selects the second paragraph in the active document and then extends the selection to include all other paragraphs that have the same tab stops.

```
Set myRange = ActiveDocument.Paragraphs(2).Range
myRange.Select
Selection.SelectCurrentTabs
```

This example selects paragraphs that have the same tab stops and retrieves the position of the first tab stop. The example moves the selection to the next range of paragraphs that have the same tab stops. The example then adds the tab stop setting from the first group of paragraphs to the current selection.

```
With Selection
    .SelectCurrentTabs
    pos = .Paragraphs.TabStops(1).Position
    .Collapse Direction:=wdCollapseEnd
    .SelectCurrentTabs
    .Paragraphs.TabStops.Add Position:=pos
End With
```

TypeBackspace Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTypeBackspaceC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTypeBackspaceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTypeBackspaceA"}

Deletes the character preceding a collapsed selection (an insertion point). If the selection isn't collapsed to an insertion point, the selection is deleted.

Note This method corresponds to functionality of the BACKSPACE key.

Syntax

expression.**TypeBackspace**

expression Required. An expression that returns a **Selection** object.

TypeBackspace Method Example

This example deletes the character preceding the insertion point (the collapsed selection).

```
With Selection
    .Collapse Direction:=wdCollapseEnd
    .TypeBackspace
End With
```

This example extends the selection to the end of the current paragraph (including the paragraph mark) and then deletes the selection.

```
With Selection
    .EndOf Unit:=wdParagraph, Extend:=wdExtend
    .TypeBackspace
End With
```

TypeParagraph Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTypeParagraphC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTypeParagraphX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTypeParagraphA"}

Inserts a new, blank paragraph. If the selection isn't collapsed to an insertion point, it's replaced by the new paragraph. Use the **InsertParagraphAfter** or **InsertParagraphBefore** method to insert a new paragraph without deleting the contents of the selection.

Note This method corresponds to the functionality of the ENTER key.

Syntax

expression.**TypeParagraph**

expression Required. An expression that returns a **Selection** object.

TypeParagraph Method Example

This example collapses the selection to its end and then inserts a new paragraph following it.

```
With Selection
    .Collapse Direction:=wdCollapseEnd
    .TypeParagraph
End With
```

StoryRanges Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStoryRangesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStoryRangesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStoryRangesA "}

Returns a **StoryRanges** collection that represents all the stories in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

StoryRanges Property Example

This example steps through the **StoryRanges** collection to determine whether **wdPrimaryFooterStory** is part of the **StoryRanges** collection.

```
For Each aStory In ActiveDocument.StoryRanges
    If aStory.StoryType = wdEvenPagesFooterStory Then
        MsgBox "Document includes a even page footer"
    End If
Next aStory
```

This example adds text to the primary header story and then displays the text.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).Range _
    .Text = "Header text"
MsgBox ActiveDocument.StoryRanges(wdPrimaryHeaderStory).Text
```

InsertCrossReference Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertCrossReferenceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertCrossReferenceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertCrossReferenceA "}

Inserts a cross-reference to a heading, bookmark, footnote, or endnote, or to an item for which a caption label is defined (for example, an equation, figure, or table).

Syntax

expression.**InsertCrossReference(*ReferenceType, ReferenceKind, ReferenceItem, InsertAsHyperLink, IncludePosition*)**

expression Required. An expression that returns a **Range** or **Selection** object.

ReferenceType Required **Variant**. The type of item for which a cross-reference is to be inserted. Can be one of the following **WdReferenceType** constants: **wdRefTypeBookmark**, **wdRefTypeEndnote**, **wdRefTypeFootnote**, **wdRefTypeHeading**, or **wdRefTypeNumberedItem**. Can also be a **WdCaptionLabelID** constant (**wdCaptionEquation**, **wdCaptionFigure**, or **wdCaptionTable**) or a user defined caption label.

ReferenceKind Required **Long**. The information to be included in the cross-reference. Can be one of the following **WdReferenceKind** constants: **wdContentText**, **wdEndnoteNumber**, **wdEndnoteNumberFormatted**, **wdEntireCaption**, **wdFootnoteNumber**, **wdFootnoteNumberFormatted**, **wdNumberFullContext**, **wdNumberNoContext**, **wdNumberRelativeContext**, **wdOnlyCaptionText**, **wdOnlyLabelAndNumber**, **wdPageNumber**, or **wdPosition**.

ReferenceItem Required **Variant**. If ***ReferenceType*** is **wdRefTypeBookmark**, this argument specifies a bookmark name. For all other ***ReferenceType*** values, this argument specifies the item number or name in the **Reference type** box in the **Cross-reference** dialog box. Use the **GetCrossReferenceItems** method to return a list of item names that can be used with this argument.

InsertAsHyperLink Optional **Variant**. **True** to insert the cross-reference as a hyperlink to the referenced item.

IncludePosition Optional **Variant**. **True** to insert "above" or "below," depending on the location of the reference item in relation to the cross-reference.

InsertCrossReference Method Example

This example inserts at the beginning of the active document a cross-reference to the page that includes the first bookmark in the document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myBookmarks = ActiveDocument.GetCrossReferenceItems(wdRefTypeBookmark)
With myRange
    .InsertBefore "Page "
    .Collapse Direction:=wdCollapseEnd
    .InsertCrossReference ReferenceType:=wdRefTypeBookmark, _
        ReferenceKind:=wdPageNumber, ReferenceItem:=myBookmarks(1)
End With
```

This example inserts a sentence that contains two cross-references: one cross-reference to heading text, and another one to the page where the heading text appears.

```
With Selection
    .Collapse Direction:=wdCollapseStart
    .InsertBefore "For more information, see "
    .Collapse Direction:=wdCollapseEnd
    .InsertCrossReference ReferenceType:=wdRefTypeHeading, _
        ReferenceKind:=wdContentText, ReferenceItem:=1
    .InsertAfter " on page "
    .Collapse Direction:=wdCollapseEnd
    .InsertCrossReference ReferenceType:=wdRefTypeHeading, _
        ReferenceKind:=wdPageNumber, ReferenceItem:=1
    .InsertAfter "."
End With
```

InStory Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInStoryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInStoryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInStoryA "}

True if the selection or range to which this method is applied is in the same story as the range specified by the ***Range*** argument.

Note A range can belong to only one story.

Syntax

expression.**InStory(*Range*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Range Required **Range** object. The **Range** object whose story is compared with the story that contains *expression*.

story

A document area that contains a range of text distinct from other areas of text in a document. For example, if a document includes body text, footnotes, and headers, it contains a main text story, footnotes story, and headers story.

There are 11 different types of stories that can be part of a document, corresponding to the following **WdStoryType** constants: **wdCommentsStory**, **wdEndnotesStory**, **wdEvenPagesFooterStory**, **wdEvenPagesHeaderStory**, **wdFirstPageFooterStory**, **wdFirstPageHeaderStory**, **wdFootnotesStory**, **wdMainTextStory**, **wdPrimaryFooterStory**, **wdPrimaryHeaderStory**, and **wdTextFrameStory**. The **StoryRanges** collection contains the first story for each story type available in a document. Use the **NextStoryRange** method to return subsequent stories.

InStory Method Example

This example determines whether the selection is in the same story as the first paragraph in the active document. The message box displays "False" because the selection is in the primary header story and the first paragraph is in the main text story.

```
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageHeader
End With
same = Selection.InStory(ActiveDocument.Paragraphs(1).Range)
MsgBox same
```

This example determines whether `Range1` and `Range2` are in the same story. If they are, bold formatting is applied to `Range1`.

```
Set Range1 = Selection.Words(1)
Set Range2 = ActiveDocument.Range(Start:=20, End:=100)
If Range1.InStory(Range:=Range2) = True Then
    Range1.Font.Bold = True
End If
```

DoubleStrikeThrough Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDoubleStrikeThroughC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDoubleStrikeThroughX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDoubleStrikeThroughA"}

True if the specified font is formatted as double strikethrough text. Returns **True**, **False**, or **wdUndefined** (a mixture of **True** and **False**). Can be set to **True**, **False**, or **wdToggle**. Read/write **Long**.

Note To set or return single-line strikethrough formatting, use the **StrikeThrough** property. Setting **DoubleStrikeThrough** to **True** sets **StrikeThrough** to **False**, and vice versa.

DoubleStrikeThrough Property Example

This example applies double strikethrough formatting to the selected text.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Font.DoubleStrikeThrough = True
Else
    MsgBox "You need to select some text."
End If
```

This example removes double strikethrough formatting from the first word in the active document and capitalizes the first letter in the word.

```
With ActiveDocument.Words(1)
    .Font.DoubleStrikeThrough = False
    .Case = wdTitleSentence
End With
```

Extend Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExtendC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExtendX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExtendA"}

Turns extend mode on and extends the selection to the next unit of text. The progression is as follows: word, sentence, paragraph, section, entire document. If ***Character*** is specified, extends the selection through the next instance of the specified character.

Syntax

expression.**Extend(*Character*)**

expression Required. An expression that returns a **Selection** object.

Character Optional **Variant**. The character that the selection is to be extended through. This argument is case sensitive.

Extend Method Example

This example extends the selection through the next instance of a capital "R" and then decreases the selection by one character.

```
With Selection
    .Extend Character:="R"
    .MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
End With
```

First Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstA"}

Sentences, **Words**, and **Characters** objects: Returns a **Range** object that represents the first sentence, word, or character in a document, selection or range. Read-only.

Columns, **Paragraphs**, **Rows**, and **Sections** objects: Returns the first item in the specified collection. Read-only. For example, both of the following instructions return a **Paragraph** object that represents the first paragraph in the selection.

```
Selection.Paragraphs.First
Selection.Paragraphs(1)
```

First Property Example

This example right aligns the first paragraph in the selection.

```
Selection.Paragraphs.First.Alignment = wdAlignParagraphRight
```

This example applies shading and a bottom border to the first row in table one in the active document.

```
ActiveDocument.Tables(1).Borders.Enable = False
With ActiveDocument.Tables(1).Rows.First
    .Shading.Texture = wdTexture10Percent
    .Borders(wdBorderBottom).LineStyle = wdLineStyleSingle
End With
```

Last Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLastC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLastA"}

Sentences, **Words**, and **Characters** objects: Returns a **Range** object that represents the last sentence, word, or character in a document, selection, or range. Read-only.

Columns, **Paragraphs**, **Rows**, and **Sections** objects: Returns the last item in the specified collection. Read-only. For example, both of the following instructions return a **Paragraph** object that represents the last paragraph in the selection.

```
Selection.Paragraphs.Last
Selection.Paragraphs(Selection.Paragraphs.Count)
```

Last Property Example

This example formats the last paragraph in the active document to be right aligned.

```
ActiveDocument.Paragraphs.Last.Alignment = wdAlignParagraphRight
```

This example deletes the last row in table one.

```
ActiveDocument.Tables(1).Rows.Last.Cells.Delete
```

This example applies bold formatting to the last word in the selection.

```
If Selection.Words.Count >= 2 Then
    Selection.Words.Last.Bold = True
End If
```

LinesToDrop Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLinesToDropC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLinesToDropX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLinesToDropA"}

Returns or sets the height of the specified dropped capital letter, in lines. Read/write **Long**.

LinesToDrop Property Example

This example formats the first character in the active document as a dropped capital letter with a height of three lines.

```
With ActiveDocument.Paragraphs(1).DropCap
    .Enable
    .Position = wdDropNormal
    .LinesToDrop = 3
End With
```

NextField Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextFieldC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextFieldX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextFieldA"}

Selects the next field. If a field is found, this method returns a **Field** object; if not, it returns **Nothing**.

Syntax

expression.**NextField**

expression Required. An expression that returns a **Selection** object.

NextField Method Example

This example updates the next field in the selection.

```
If Not (Selection.NextField Is Nothing) Then
    Selection.Fields.Update
End If
```

This example selects the next field in the selection, and if a field is found, displays a message in the status bar.

```
Set myField = Selection.NextField
If Not (myField Is Nothing) Then StatusBar = "Field found"
```

NextSubdocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextSubdocumentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextSubdocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextSubdocumentA"}

Moves the range or selection to the next subdocument. If there isn't another subdocument, an error occurs.

Syntax

expression.**NextSubdocument**

expression Required. An expression that returns a **Range** or **Selection** object.

NextSubdocument Method Example

This example switches the active document to master document view and selects the first subdocument.

```
If ActiveDocument.Subdocuments.Count >= 1 Then
    ActiveWindow.View.Type = wdMasterView
    Selection.HomeKey unit:=wdStory, Extend:=wdMove
    Selection.NextSubdocument
End If
```

OutlineDemote Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOutlineDemoteC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOutlineDemoteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOutlineDemoteA"}

Applies the next heading level style (Heading 1 through Heading 8) to the specified paragraph or paragraphs. For example, if a paragraph is formatted with the Heading 2 style, this method demotes the paragraph by changing the style to Heading 3.

Syntax

expression.**OutlineDemote**

expression Required. An expression that returns a **Paragraph** or **Paragraphs** object.

OutlineDemote Method Example

This example demotes the selected paragraphs.

```
Selection.Paragraphs.OutlineDemote
```

This example demotes the third paragraph in the active document.

```
ActiveDocument.Paragraphs(3).OutlineDemote
```

OutlineDemoteToBody Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOutlineDemoteToBodyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOutlineDemoteToBodyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOutlineDemoteToBodyA"}

Demotes the specified paragraph or paragraphs to body text by applying the Normal style.

Syntax

expression.**OutlineDemoteToBody**

expression Required. An expression that returns a **Paragraph** or **Paragraphs** object.

OutlineDemoteToBody Method Example

This example demotes the selected paragraphs to body text by applying the Normal style.

```
Selection.Paragraphs.OutlineDemoteToBody
```

This example switches the active window to outline view and demotes the first paragraph in the selection to body text.

```
ActiveWindow.View.Type = wdOutlineView
Selection.Paragraphs(1).OutlineDemoteToBody
```

OutlinePromote Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthOutlinePromoteC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthOutlinePromoteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthOutlinePromoteA"}

Applies the previous heading level style (Heading 1 through Heading 8) to the specified paragraph or paragraphs. For example, if a paragraph is formatted with the Heading 2 style, this method promotes the paragraph by changing the style to Heading 1.

Syntax

expression.**OutlinePromote**

expression Required. An expression that returns a **Paragraph** or **Paragraphs** object.

OutlinePromote Method Example

This example promotes the selected paragraphs.

```
Selection.Paragraphs.OutlinePromote
```

This example switches the active window to outline view and promotes the first paragraph in the active document.

```
ActiveWindow.View.Type = wdOutlineView
ActiveDocument.Paragraphs(1).OutlinePromote
```

PreviousField Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPreviousFieldC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPreviousFieldX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPreviousFieldA"}

Selects the previous field. If a field is found, this method returns a **Field** object; if not, it returns **Nothing**.

Syntax

expression.**PreviousField**

expression Required. An expression that returns a **Selection** object.

PreviousField Method Example

This example updates the previous field (the field immediately preceding the selection).

```
If Not (Selection.PreviousField Is Nothing) Then
    Selection.Fields.Update
End If
```

This example selects the previous field, and if a field is found, displays a message in the status bar.

```
Set myField = Selection.PreviousField
If Not (myField Is Nothing) Then StatusBar = "Field found"
```

PreviousSubdocument Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPreviousSubdocumentC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPreviousSubdocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPreviousSubdocumentA"}

Moves the range or selection to the previous subdocument. If there isn't another subdocument, an error occurs.

Syntax

expression.**PreviousSubdocument**

expression Required. An expression that returns a **Range** or **Selection** object.

PreviousSubdocument Method Example

This example switches the active document to master document view and selects the previous subdocument.

```
If ActiveDocument.Subdocuments.Count >= 1 Then
    ActiveWindow.View.Type = wdMasterView
    Selection.EndKey Unit:=wdStory, Extend:=wdMove
    Selection.PreviousSubdocument
End If
```

Scaling Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproScalingC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproScalingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproScalingA"}

Returns or sets the scaling percentage applied to the font. This property stretches or compresses text horizontally as a percentage of the current size (the scaling range is from 1 through 600). Read/write **Long**.

Scaling Property Example

This example horizontally streches the text in the active document to 110 percent of its original size.

```
ActiveDocument.Content.Font.Scaling = 110
```

This example compresses the text in the first paragraph in Sales.doc to 90 percent of its original size.

```
With Documents("Sales.doc").Paragraphs(1).Range.Font
    .Scaling = 90
    .Bold = False
End With
```

SetCount Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetCountC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetCountX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetCountA"}

Arranges text into the specified number of text columns.

Note You can also use the **Add** method of the **TextColumns** object to add a single column to the **TextColumns** collection.

Syntax

expression.**SetCount(*NumColumns*)**

expression Required. An expression that returns a **TextColumns** object.

NumColumns Required **Long**. The number of columns the text is to be arranged into.

SetCount Method Example

This example arranges the text in the active document into two columns of equal width.

```
ActiveDocument.PageSetup.TextColumns.SetCount NumColumns:=2
```

This example arranges the text in the first section of Brochure.doc into three columns of equal width.

```
Documents("Brochure.doc").Sections(1).PageSetup.TextColumns.SetCount
NumColumns:=3
```

SpellingErrorType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpellingErrorTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpellingErrorTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpellingErrorTypeA"}

Returns the spelling error type. Can be one of the following **WdSpellingErrorType** constants: **wdSpellingCapitalization**, **wdSpellingCorrect**, or **wdSpellingNotInDictionary**. Read-only **Long**.

Note Use the **GetSpellingSuggestions** method to return a collection of words suggested as spelling replacements. If a word is misspelled, the **CheckSpelling** method returns **True**.

SpellingErrorType Property Example

If the first word in the active document isn't in the dictionary, this example displays "Unknown word" in the status bar.

```
Set suggs = ActiveDocument.Content.GetSpellingSuggestions
If suggs.SpellingErrorType = wdSpellingNotInDictionary Then
    StatusBar = "Unknown word"
End If
```

SplitTable Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSplitTableC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSplitTableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSplitTableA"}

Inserts an empty paragraph above the first row in the selection. If the selection isn't in the first row of the table, the table is split into two tables.

Note If the selection isn't in a table, an error occurs.

Syntax

expression.**SplitTable**

expression Required. An expression that returns a **Selection** object.

SplitTable Method Example

If the selection is in a table, this example splits the table.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.SplitTable
End If
```

This example splits the first table in the active document between the first and second rows.

```
ActiveDocument.Tables(1).Rows(2).Select
Selection.SplitTable
```

Sentences Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSentencesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSentencesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSentencesA "}

Returns a **Sentences** collection that represents all the sentences in the range, selection, or document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Sentences Property Example

This example copies the first sentences in the active document.

```
ActiveDocument.Sentences(1).Copy
```

This example deletes the last sentence in the active document.

```
ActiveDocument.Sentences.Last.Delete
```

This example displays the number of sentences in the first paragraph in the active document.

```
MsgBox ActiveDocument.Paragraphs(1).Range.Sentences.Count & " sentences"
```

FullName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFullNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFullNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFullNameA"}

Returns the name of the specified document or template, including its path on disk. Read-only **String**.

Remarks

Using this property is equivalent to using the **Path**, **PathSeparator**, and **Name** properties, in sequence.

FullName Property Example

This example displays the path and file name of the active document (assuming that the document has been saved).

```
MsgBox ActiveDocument.FullName
```

This example displays the path and file name of the template attached to the active document.

```
MsgBox ActiveDocument.AttachedTemplate.FullName
```

Range Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRangeA "}

Document object: Returns a **Range** object by using the specified starting and ending character positions. For information about **Range** objects, see Working with **Range** objects.

Shapes object: Returns a **ShapeRange** object that represents a subset of shapes in the **Shapes** collection.

Syntax 1

expression.**Range(*Start, End*)**

Syntax 2

expression.**Range(*Index*)**

expression Required. An expression that returns a **Document** object (Syntax 1) or a **Shapes** object (Syntax 2).

Start Optional **Long**. The starting character position.

End Optional **Long**. The ending character position.

Index Required **Variant**. Specifies which shapes are to be included in the specified range. Can be an an integer that specifies the index number of a shape within the **Shapes** collection, a string that specifies the name of a shape, or a **Variant** array that contains integers or strings.

Remarks

Character position values begin with 0 (zero) at the beginning of the document. All characters are counted, including nonprinting characters. Hidden characters are counted even if they're not displayed. If you don't specify starting and ending character positions for the **Range** method, the entire document is returned as a **Range** object.

ShapeRange objects don't include **InlineShape** objects. An **InlineShape** object is equivalent to a character and is positioned as a character within a range of text. **Shape** objects are anchored to a range of text (the selection, by default), but they can be positioned anywhere on the page. A **Shape** object will always appear on the same page as the range it's anchored to.

Most operations that you can do with a **Shape** object you can also do with a **ShapeRange** object that contains a single shape. Some operations, when performed on a **ShapeRange** object that contains multiple shapes, produce an error.

Range Method Example

This example applies bold formatting to the first 10 characters in the active document.

```
ActiveDocument.Range(Start:=0, End:=10).Bold = True
```

This example creates a range that starts at the beginning of the active document and ends at the end of the selection.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=Selection.End)
```

This example creates sets the variable `myRange` to paragraphs three through six in the active document, and then it right aligns the paragraphs in the range.

```
Set aDoc = ActiveDocument
If aDoc.Paragraphs.Count >= 6 Then
    Set myRange = aDoc.Range(aDoc.Paragraphs(3).Range.Start, _
        aDoc.Paragraphs(6).Range.End)
    myRange.Paragraphs.Alignment = wdAlignParagraphRight
End If
```

This example sets the fill foreground color to purple for the first shape in the active document.

```
ActiveDocument.Shapes.Range(1).Fill.ForeColor.RGB = RGB(255, 0, 255)
```

This example applies a shadow to the shape named "myshape" in the active document.

```
ActiveDocument.Shapes.Range("myshape").Shadow.Type = msoShadow6
```

This example selects shapes one and three in the active document.

```
ActiveDocument.Shapes.Range(Array(1, 3)).Select
```

Copy Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCopyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCopyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCopyA "}

Syntax 1: Copies the specified object to the Clipboard.

Syntax 2: Sets the bookmark specified by the ***Name*** argument to the location marked by another bookmark, and returns a **Bookmark** object.

Syntax 1

expression.**Copy**

Syntax 2

expression.**Copy(*Name*)**

expression Syntax 1: Required. An expression that returns a **Field**, **FormField**, **Frame**, **MailMergeField**, **PageNumber**, **Range**, or **Selection** object.

Syntax 2: Required. An expression that returns a **Bookmark** object.

Name Required **String**. The name of the new bookmark.

Copy Method Example

This example copies the contents of the selection into a new document.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Copy
    Documents.Add.Content.Paste
End If
```

This example sets the Book2 bookmark to the location marked by the Book1 bookmark.

```
ActiveDocument.Bookmarks("Book1").Copy Name:="Book2"
```

This example sets the Selection bookmark to the \Sel predefined bookmark in the active document.

```
ActiveDocument.Bookmarks("\Sel").Copy Name:="Selection"
```

This example copies the first paragraph in the active document and pastes it at the end of the document.

```
ActiveDocument.Paragraphs(1).Range.Copy
Set myRange = ActiveDocument.Range(Start:=ActiveDocument.Content.End - 1, _
    End:=ActiveDocument.Content.End - 1)
myRange.Paste
```

This example copies the comments in the active document to the Clipboard.

```
If ActiveDocument.Comments.Count >= 1 Then
    ActiveDocument.StoryRanges(wdCommentsStory).Copy
End If
```

Cut Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCutC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCutX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCutA "}

Removes the specified object from the document and places it on the Clipboard.

Syntax

expression.**Cut**

expression Required. An expression that returns a **Field**, **FormField**, **Frame**, **MailMergeField**, **PageNumber**, **Range**, or **Selection** object.

Remarks

If *expression* returns a **Range** or **Selection** object, the contents of the object are cut to the Clipboard but the collapsed object remains in the document.

Cut Method Example

This example cuts the first field in the active document and pastes the field at the insertion point.

```
If ActiveDocument.Fields.Count >= 1 Then
    ActiveDocument.Fields(1).Cut
    Selection.Collapse Direction:=wdCollapseEnd
    Selection.Paste
End If
```

This example cuts the first word in the first paragraph and pastes the word at the end of the paragraph.

```
With ActiveDocument.Paragraphs(1).Range
    .Words(1).Cut
    .Collapse Direction:=wdCollapseEnd
    .Move Unit:=wdCharacter, Count:=-1
    .Paste
End With
```

This example cuts the contents of the selection and pastes them into a new document.

```
If Selection.Type = wdSelectionNormal Then
    Selection.Cut
    Documents.Add.Content.Paste
End If
```

Insert Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertA "}

AutoTextEntry object: Inserts the AutoText entry in place of the specified range. If you don't want to replace the range, use the **Collapse** method before using this method. The inserted AutoText entry is returned as a **Range** object.

Envelope object: Inserts an envelope as a separate section at the beginning of the specified document. The arguments for this method correspond to the options on the **Envelopes** tab in the **Envelopes and Labels** dialog box (**Tools** menu).

Syntax 1

expression.**Insert(*Where, RichText*)**

Syntax 2

expression.**Insert(*ExtractAddress, Address, AutoText, OmitReturnAddress, ReturnAddress, ReturnAutoText, PrintBarCode, PrintFIMA, Size, Height, Width, FeedSource, AddressFromLeft, AddressFromTop, ReturnAddressFromLeft, ReturnAddressFromTop, DefaultFaceUp, DefaultOrientation*)**

expression Required. An expression that returns an **AutoTextEntry** object (Syntax 1) or an **Envelope** object (Syntax 2).

Where Required **Range** object. The location for the AutoText entry.

RichText Optional **Variant**. **True** to insert the AutoText entry with its original formatting.

ExtractAddress Optional **Variant**. **True** to use the text marked by the EnvelopeAddress bookmark (a user-defined bookmark) as the recipient's address.

Address Optional **Variant**. A string that specifies the recipient's address (ignored if ***ExtractAddress*** is **True**).

AutoText Optional **Variant**. A string that specifies an AutoText entry to use for the address. If specified, ***Address*** is ignored.

OmitReturnAddress Optional **Variant**. **True** to omit the return address.

ReturnAddress Optional **Variant**. A string that specifies the return address.

ReturnAutoText Optional **Variant**. A string that specifies an AutoText entry to use for the return address. If specified, ***ReturnAddress*** is ignored.

PrintBarCode Optional **Variant**. **True** to add a POSTNET bar code. For U.S. mail only.

PrintFIMA Optional **Variant**. **True** to add a Facing Identification Mark (FIMA) for use in presorting courtesy reply mail. For U.S. mail only.

Size Optional **Variant**. A string that specifies the envelope size. The string should match one of the sizes listed on the left in the **Envelope size** box in the **Envelope Options** dialog box (for example, "Size 10" or "C4").

Height Optional **Variant**. The height of the envelope (in points) when the ***Size*** argument is set to "Custom size."

Width Optional **Variant**. The width of the envelope (in points) when the ***Size*** argument is set to "Custom size."

FeedSource Optional **Variant**. The paper tray to be used when the envelope is printed. Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed	**wdPrinterManualFeed**
wdPrinterDefaultBin	**wdPrinterMiddleBin**
wdPrinterEnvelopeFeed	**wdPrinterOnlyBin**
wdPrinterFormSource	**wdPrinterPaperCassette**

wdPrinterLargeCapacityBin **wdPrinterSmallFormatBin**
wdPrinterLargeFormatBin **wdPrinterTractorFeed**
wdPrinterLowerBin **wdPrinterUpperBin**
wdPrinterManualEnvelopeFeed

AddressFromLeft Optional **Variant**. The distance (in points) between the left edge of the envelope and the recipient's address.

AddressFromTop Optional **Variant**. The distance (in points) between the top edge of the envelope and the recipient's address.

ReturnAddressFromLeft Optional **Variant**. The distance (in points) between the left edge of the envelope and the return address.

ReturnAddressFromTop Optional **Variant**. The distance (in points) between the top edge of the envelope and the return address.

DefaultFaceUp Optional **Variant**. **True** to print the envelope face up, **False** to print it face down.

DefaultOrientation Optional **Variant**. The orientation for the envelope. Can be one of the following **WdEnvelopeOrientation** constants: **wdLeftPortrait**, **wdCenterPortrait**, **wdRightPortrait**, **wdLeftLandscape**, **wdCenterLandscape**, **wdRightLandscape**, **wdLeftClockwise**, **wdCenterClockwise**, or **wdRightClockwise**.

Insert Method Example

This example inserts the formatted AutoText entry named "one" after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.AttachedTemplate.AutoTextEntries("one").Insert _
    Where:=Selection.Range, RichText:=True
```

This example adds a Size 10 envelope to the active document by using the addresses stored in the `tAddr` and `fAddr` variables.

```
rtn = VbCr
tAddr = "Max Benson" & rtn & "123 Skye St." & rtn & "OurTown, WA 98107"
fAddr = "Paul Borm" & rtn & "456 Erde Lane" & rtn & "OurTown, WA 98107"
ActiveDocument.Envelope.Insert Address:=fAddr, ReturnAddress:=tAddr,
Size:="Size 10"
```

Accept Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAcceptC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAcceptX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAcceptA "}

Accepts the specified tracked change. The revision marks are removed, and the change is incorporated into the document.

Syntax

expression.**Accept**

expression Required. An expression that returns a **Revision** object.

Accept Method Example

This example accepts the next tracked change found if the change type is inserted text.

```
Set myRev = Selection.NextRevision(Wrap:=True)
If Not (myRev Is Nothing) Then
    If myRev.Type = wdRevisionInsert Then myRev.Accept
End If
```

This example accepts all the tracked changes in the selection.

```
Set myRange = Selection.Range
For Each myRev In myRange.Revisions
    myRev.Accept
Next myRev
```

AcceptAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAcceptAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAcceptAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAcceptAllA "}

Accepts all the tracked changes in a document or range. The revision marks are removed, and the changes are incorporated into the document.

Syntax

expression.**AcceptAll**

expression Required. An expression that returns a **Revisions** object.

Remarks

Use the **AcceptAllRevisions** method to accept all revisions in a document.

AcceptAll Method Example

This example accepts all the tracked changes in the active document.

```
If ActiveDocument.Revisions.Count >= 1 Then _
    ActiveDocument.Revisions.AcceptAll
```

This example accepts all the tracked changes in the selection.

```
Selection.Range.Revisions.AcceptAll
```

Author Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAuthorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAuthorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAuthorA "}

Comment object: Returns or sets the author name for a comment. Read/write **String**.

Revision object: Returns the name of the user who made the specified tracked change. Read-only **String**.

Author Property Example

This example sets the author name and initials for the first comment in the active document.

```
If ActiveDocument.Comments.Count >= 1 Then
    With ActiveDocument.Comments(1)
        .Author = "Joe Smith"
        .Initial = "JAS"
    End With
End If
```

This example returns the author name for the first comment in the selection.

```
If Selection.Comments.Count >= 1 Then _
    aAuthor = Selection.Comments(1).Author
```

This example displays the author name for the first tracked change in the first selected section.

```
Set myRange = Selection.Sections(1).Range
MsgBox "Revisions made by " & myRange.Revisions(1).Author
```

Date Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDateA "}

Revision object: The date and time that the tracked change was made. Read-only **Date**.

Version object: The date and time that the document version was saved. Read-only **Date**.

Date Property Example

This example displays the date and time that the last version of the active document was saved.

```
Set myDoc = ActiveDocument
If myDoc.Path <> "" Then MsgBox _
    myDoc.Versions(myDoc.Versions.Count).Date
```

This example displays the date and time of the next tracked change found in the active document.

```
If ActiveDocument.Revisions.Count >= 1 Then
    Set myRev = Selection.NextRevision
    If Not (myRev Is Nothing) Then MsgBox myRev.Date
End If
```

PrintRevisions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrintRevisionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrintRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrintRevisionsA "}

True if revision marks are printed with the document. **False** if revision marks aren't printed (that is, tracked changes are printed as if they'd been accepted). Read/write **Boolean**.

PrintRevisions Property Example

This example prints the active document without revision marks.

```
With ActiveDocument
    .PrintRevisions = False
    .PrintOut
End With
```

Reject Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRejectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRejectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRejectA "}

Rejects the specified tracked change. The revision marks are removed, leaving the original text intact.

Note Formatting changes cannot be rejected.

Syntax

expression.**Reject**

expression Required. An expression that returns a **Revision** object.

Reject Method Example

This example rejects the next tracked change found in the active document.

```
If ActiveDocument.Revisions.Count >= 1 Then
    Set myRev = Selection.NextRevision
    If Not (myrev Is Nothing) Then myrev.Reject
End If
```

This example rejects the tracked changes in the first paragraph.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
For Each myRev In myRange.Revisions
    myRev.Reject
Next myRev
```

This example rejects the first tracked change in the selection.

```
Set myRange = Selection.Range
If myRange.Revisions.Count >= 1 Then myRange.Revisions(1).Reject
```

RejectAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRejectAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRejectAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRejectAllA "}

Rejects all the tracked changes in a range. The revision marks are removed, leaving the original text intact.

Syntax

expression.**RejectAll**

expression Required. An expression that returns a **Revisions** object.

Remarks

Use the **RejectAllRevisions** method to reject all revisions in a document. Formatting changes cannot be rejected.

RejectAll Method Example

This example rejects all the tracked changes in the active document.

```
ActiveDocument.Revisions.RejectAll
```

This example rejects all the tracked changes in the selection.

```
Set myRange = Selection.Range
myRange.Revisions.RejectAll
```

Revisions Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRevisionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRevisionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRevisionsA "}

Returns a **Revisions** collection that represents the tracked changes in the document or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Revisions Property Example

This example displays the number of tracked changes in the first section in the active document.

```
MsgBox ActiveDocument.Sections(1).Range.Revisions.Count
```

This example accepts all tracked changes in the first paragraph in the selection.

```
Set myRange = Selection.Paragraphs(1).Range
myRange.Revisions.AcceptAll
```

This example inserts text if change tracking isn't enabled.

```
If ActiveDocument.TrackRevisions = False Then
    Selection.InsertBefore "new text"
End If
```

NextRevision Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextRevisionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextRevisionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextRevisionA "}

Locates and returns the next tracked change as a **Revision** object.

Syntax

expression.**NextRevision(*Wrap*)**

expression Required. An expression that returns a **Selection** object.

Wrap Optional **Variant**. **True** to continue searching for a revision at the beginning of the document when the end of the document is reached. The default value is **False**.

NextRevision Method Example

This example rejects the next tracked change found after the fifth paragraph in the active document. The `myRev` variable is set to **Nothing** if a change is not found.

```
If ActiveDocument.Paragraphs.Count >= 5 Then
    Set myRange = ActiveDocument.Paragraphs(5).Range
    myRange.Select
    Set myRev = Selection.NextRevision
    If Not (myRev Is Nothing) Then myRev.Reject
End If
```

This example accepts the next tracked change found if the change type is inserted text.

```
Set myRev = Selection.NextRevision(Wrap:=True)
If Not (myRev Is Nothing) Then
    If myRev.Type = wdRevisionInsert Then myRev.Accept
End If
```

PreviousRevision Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPreviousRevisionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPreviousRevisionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPreviousRevisionA "}

Locates and returns the previous tracked change as a **Revision** object.

Syntax

expression.**PreviousRevision(*Wrap*)**

expression Required. An expression that returns a **Selection** object.

Wrap Optional **Variant**. **True** to continue searching for a revision at the end of the document when the beginning of the document is reached. The default value is **False**.

PreviousRevision Method Example

This example selects the last tracked change in the first section in the active document and displays the date and time of the change.

```
Selection.EndOf Unit:=wdStory, Extend:=wdMove
Set myRev = Selection.PreviousRevision
If Not (myRev Is Nothing) Then MsgBox myRev.Date
```

This example rejects the previous tracked change found if the change type is deleted or inserted text. If the tracked change is a style change, the change is accepted.

```
Set myRev = Selection.PreviousRevision(Wrap:=True)
If Not (myRev Is Nothing) Then
    Select Case myRev.Type
        Case wdRevisionDelete
            myRev.Reject
        Case wdRevisionInsert
            myRev.Reject
        Case wdRevisionStyle
            myRev.Accept
    End Select
End If
```

Compare Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCompareC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCompareX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCompareA "}

Displays revision marks that indicate where the specified document differs from another document.

Syntax

expression.**Compare(*Name*)**

expression Required. An expression that returns a **Document** object.

Name Required **String**. The name of the document that the document specified by *expression* is compared with.

Compare Method Example

This example compares the active document with the document named "First Rev.doc" in the Draft folder. On the Macintosh, use a colon as the path separator (for example, HD:Draft:First Rev).

```
ActiveDocument.Compare Name:="C:\Draft\First Rev.doc"
```

Merge Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMergeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMergeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMergeA "}

Cell object: Merges the specified table cell with another cell. The result is a single table cell.

Cells object: Merges the specified table cells with one another. The result is a single table cell.

Document object: Merges the changes marked with revision marks from one document to another.

Subdocument object: Merges the specified subdocuments of a master document into a single subdocument.

Syntax 1

expression.**Merge(*MergeTo*)**

Syntax 2

expression.**Merge**

Syntax 3

expression.**Merge(*FileName*)**

Syntax 4

expression.**Merge(*FirstSubdocument, LastSubdocument*)**

expression Syntax 1: An expression that returns a **Cell** object.
Syntax 2: An expression that returns a **Cells** object.
Syntax 3: An expression that returns a **Document** object.
Syntax 4: An expression that returns a **Subdocuments** object.

MergeTo Required **Cell**. The cell to be merged with.

FileName Required **String**. The path and file name of the original document you want to merge revisions with.

FirstSubdocument Optional **Variant**. The first subdocument in a range of subdocuments to be merged.

LastSubdocument Optional **Variant**. The last subdocument in a range of subdocuments to be merged.

Merge Method Example

This example merges the first two cells in table one in the active document with one another and then removes the table borders.

```
If ActiveDocument.Tables.Count >= 1 Then
    With ActiveDocument.Tables(1)
        .Cell(Row:=1, Column:=1).Merge MergeTo:=.Cell(Row:=1, Column:=2)
        .Borders.Enable = False
    End With
End If
```

This example merges changes from Sales2.doc (the active document) into Sales1.doc.

```
If InStr(1, ActiveDocument.Name, "sales2.doc", 1) Then _
    ActiveDocument.Merge FileName:="C:\Docs\Sales1.doc"
```

This example merges the cells in row one of the selection into a single cell and then applies shading to the row.

```
If Selection.Information(wdWithInTable) = True Then
    Set myrow = Selection.Rows(1)
    myrow.Cells.Merge
    myrow.Shading.Texture = wdTexture10Percent
End If
```

This example merges the first and second subdocuments in the active document into one subdocument.

```
If ActiveDocument.Subdocuments.Count >= 2 Then
    Set aDoc = ActiveDocument
    aDoc.Subdocuments.Merge FirstSubdocument:=aDoc.Subdocuments(1), _
        LastSubdocument:=aDoc.Subdocuments(2)
End If
```

Wrap Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWrapC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWrapX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWrapA "}

Returns or sets what happens if the search begins at a point other than the beginning of the document and the end of the document is reached (or vice versa if **Forward** is set to **False**) or if the search text isn't found in the specified selection or range. Read/write **Long**.

Can be one of the following **WdFindWrap** constants:

Constant	Description
wdFindAsk	After searching the selection or range, Word displays a message asking whether to search the remainder of the document.
wdFindContinue	The find operation continues when the beginning or end of the search range is reached.
WdFindStop	The find operation ends when the beginning or end of the search range is reached.

Wrap Property Example

The following example searches forward through the document for the word "aspirin." When the end of the document is reached, the search continues at the beginning of the document. If the word "aspirin" is found, it's selected.

```
With Selection.Find
    .Forward = True
    .ClearFormatting
    .MatchWholeWord = True
    .MatchCase = False
    .Wrap = wdFindContinue
    .Execute FindText:="aspirin"
End With
```

ShowBy Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowByC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowByX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowByA"}

Returns or sets the name of the reviewer whose comments are shown in the comments pane. You can choose to show comments either by a single reviewer or by all reviewers. Read/write **String**.

Note To view the comments by all reviewers, set this property to "All Reviewers."

Syntax

expression.**ShowBy(*Author*)**

expression Required. An expression that returns a **Comments** object.

Author Required **String**. The name of the reviewer.

ShowBy Property Example

The following example displays comments made by Don Funk.

```
If ActiveDocument.Comments.Count >= 1 Then
    ActiveWindow.View.SplitSpecial = wdPaneComments
    ActiveDocument.Comments.ShowBy = "Don Funk"
End If
```

Delivery Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDeliveryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDeliveryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDeliveryA "}

Returns or sets the delivery method used for routing the document. Can be either of the following **WdRoutingSlipDelivery** constants: **wdAllAtOnce** or **wdOneAfterAnother**. Read/write **Long** before routing starts; read-only **Long** while routing is in progress.

Delivery Property Example

This example routes the document named "Status.doc" to two recipients, one after the other.

```
Documents("Status.doc").HasRoutingSlip = True
With Documents("Status.doc").RoutingSlip
    .Subject = "Status Doc"
    .AddRecipient Recipient:="Don Funk"
    .AddRecipient Recipient:="Eric Maffei"
    .Delivery = wdOneAfterAnother
End With
Documents("Status.doc").Route
```

HasRoutingSlip Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasRoutingSlipC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasRoutingSlipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasRoutingSlipA "}

True if the specified document has a routing slip attached to it. Setting this property to **True** creates a routing slip; setting it to **False** deletes the routing slip. Read/write **Boolean**.

HasRoutingSlip Property Example

This example removes the routing slip from Sales 1995.doc.

```
Documents("Sales 1995.doc").HasRoutingSlip = False
```

If the active document has a routing slip attached to it, this example routes the document.

```
If ActiveDocument.HasRoutingSlip = True Then
    ActiveDocument.Route
End If
```

Message Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMessageC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMessageX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMessageA "}

Returns or sets the message text for the specified routing slip. The text is used as the body text of the mail message used to route the document. Read/write **String**.

Message Property Example

This example adds a routing slip to the active document, sets the subject and message text, adds a recipient, and then routes the document.

```
ActiveDocument.HasRoutingSlip = True
With ActiveDocument.RoutingSlip
    .Subject = "Status Doc"
    .Message = "Please fill in your status."
    .AddRecipient Recipient:="Kate Dresen"
End With
ActiveDocument.Route
```

If Monthly Report.doc has a routing slip attached to it, this example displays the message text.

```
Set myDoc = Documents("Monthly Report.doc")
If myDoc.HasRoutingSlip = True Then MsgBox myDoc.RoutingSlip.Message
```

Protect Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthProtectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthProtectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthProtectA "}

Protects the specified document from changes. When a document is protected, the user can make only limited changes, such as adding annotations, making revisions, or completing a form.

Note If the document is already protected when you use this method, an error occurs.

Syntax

expression.**Protect(*Type, NoReset, Password*)**

expression Required. An expression that returns a **Document** object.

Type Required **Long**. The protection type for the specified document. Can be one of the following **WdProtectionType** constants: **wdAllowOnlyComments**, **wdAllowOnlyFormFields**, **wdAllowOnlyRevisions**, or **wdNoProtection**.

NoReset Optional **Variant**. **False** to reset form fields to their default values. **True** to retain the current form field values if the specified document is protected. If ***Type*** isn't **wdAllowOnlyFormFields**, the ***NoReset*** argument is ignored.

Password Optional **Variant**. The password required to "unprotect" the specified document.

Protect Method Example

This example protects the active document for forms without resetting the contents of the form fields.

```
If ActiveDocument.ProtectionType = wdNoProtection Then
    ActiveDocument.Protect Type:=wdAllowOnlyFormFields, NoReset:=True
End If
```

This example protects Monthly Report.doc so that only comments can be added to it. The password "free" is required to unprotect the document.

```
Set myDoc = Documents("Monthly Report.doc")
myDoc.Protect Type:=wdAllowOnlyComments, Password:="free"
```

Protect Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProtectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProtectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProtectA "}

Returns or sets the protection type for the document associated with the specified routing slip. Can be one of the following **WdProtectionType** constants: **wdAllowOnlyComments**, **wdAllowOnlyFormFields**, **wdAllowOnlyRevisions**, or **wdNoProtection**. Read/write **Long**.

Protect Property Example

If the active document has a routing slip, this example protects the document (only allows comments) and then routes it.

```
With ActiveDocument
    If .HasRoutingSlip = True Then
        .RoutingSlip.Protect = wdAllowOnlyComments
        .Route
    End If
End With
```

AddRecipient Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddRecipientC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddRecipientX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddRecipientA "}

Adds a recipient name to the specified routing slip.

Note If the recipient name isn't in the global address book, an error occurs.

Syntax

expression.**AddRecipient(*Recipient*)**

expression Required. An expression that returns a **RoutingSlip** object.

Recipient Required **String**. The recipient name.

AddRecipient Method Example

This example routes the active document to two recipients, one after the other.

```
ActiveDocument.HasRoutingSlip = True
With ActiveDocument.RoutingSlip
    .Subject = "Status Document"
    .AddRecipient Recipient:="Tim O' Brien"
    .AddRecipient Recipient:="Karin Gallagher"
    .Delivery = wdOneAfterAnother
End With
ActiveDocument.Route
```

ReturnWhenDone Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproReturnWhenDoneC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproReturnWhenDoneX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproReturnWhenDoneA "}

True if the document associated with the specified routing slip is sent back to the original sender when the routing is finished. Read/write **Boolean** before routing begins; read-only **Boolean** while routing is in progress.

ReturnWhenDone Property Example

This example sets the routing slip for Sales 1995.doc to return the document back to the original sender after the last recipient reviews it.

```
If Documents("Sales 1995.doc").HasRoutingSlip = True Then
    With Documents("Sales 1995.doc").RoutingSlip
        .Delivery = wdOneAfterAnother
        .ReturnWhenDone = True
    End With
End If
```

Route Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthRouteC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthRouteX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthRouteA "}

Routes the specified document, using the document's current routing slip.

Remarks

If the document doesn't have a routing slip, an error occurs. Use the **HasRoutingSlip** property to determine whether there's a routing slip attached to the document. Routing a document sets the **Routed** property to **True**.

Syntax

expression.**Route**

expression Required. An expression that returns a **Document** object.

Route Method Example

If the active document has a routing slip attached to it, this example routes the document.

```
If ActiveDocument.HasRoutingSlip = True Then ActiveDocument.Route
```

This example routes Feedback.doc to two recipients, one after the other.

```
Documents("Feedback.doc").HasRoutingSlip = True
With Documents("Feedback.doc").RoutingSlip
    .Subject = "Your feedback please..."
    .AddRecipient Recipient:="Tad Orman"
    .AddRecipient Recipient:="David Simpson"
    .Delivery = wdOneAfterAnother
End With
Documents("Status.doc").Route
```

Routed Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRoutedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRoutedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRoutedA "}

True if the specified document has been routed to the next recipient. **False** if the document has yet to be routed (for example, if the document has no routing slip, or if a routing slip was just created). Read-only **Boolean**.

Routed Property Example

This example routes the active document if it hasn't yet been routed.

```
If ActiveDocument.Routed = False Then ActiveDocument.Route
```

RoutingSlip Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRoutingSlipC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRoutingSlipX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRoutingSlipA "}

Returns a **RoutingSlip** object that represents the routing slip information for the specified document. A routing slip is used to send a document through an electronic mail system. Read-only.

RoutingSlip Property Example

This example adds a routing slip to Status.doc and then routes the document to the specified recipients.

```
Documents("Status.doc").HasRoutingSlip = True
With Documents("Status.doc").RoutingSlip
    .Subject = "Status Doc "
    .AddRecipient Recipient:="Don Funk"
    .AddRecipient Recipient:="Frida Ebbeson"
    .Delivery = wdAllAtOnce
End With
Documents("Status.doc").Route
```

Status Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStatusC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStatusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStatusA "}

Returns the routing status of the specified routing slip. Can be one of the following **WdRoutingSlipStatus** constants: **wdNotYetRouted**, **wdRouteComplete**, or **wdRouteInProgress**. Read-only **Long**.

Status Property Example

If the active document has a routing slip attached to it, this example displays a message indicating the routing status.

```
If ActiveDocument.HasRoutingSlip = True Then
    Select Case ActiveDocument.RoutingSlip.Status
        Case wdNotYetRouted
            MsgBox "The document hasn't been routed yet."
        Case wdRouteInProgress
            MsgBox "Routing is in progress."
        Case wdRouteComplete
            MsgBox "Routing is complete."
    End Select
End If
```

This example resets the routing slip for Sales.doc if the routing is complete.

```
With Documents("Sales.doc").RoutingSlip
    If .Status = wdRouteComplete Then
        .Reset
    Else
        MsgBox "Cannot reset routing; not yet complete."
    End If
End With
```

Subject Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSubjectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSubjectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSubjectA "}

RoutingSlip object: Returns or sets the subject text of mail messages used to route a document. Read/write **String**.

LetterContent object: Returns or sets the subject text of a letter created by the Letter Wizard. Read/write **String**.

Mailer object: Returns or sets the mail message subject line. Available only on the Macintosh and only if the PowerTalk mail system extension is installed. Read/write **String**.

Subject Property Example

This example sets the subject and message text for the routing slip associated with Month End.doc.

```
If Documents("Month End.doc").HasRoutingSlip = True Then
    With Documents("Month End.doc").RoutingSlip
        .Subject = "End of month report"
        .Message = "I need your response on this."
    End With
End If
```

This example displays the subject of a letter created by the Letter Wizard, unless the subject is an empty string.

```
If ActiveDocument.GetLetterContent.Subject <> "" Then
    MsgBox ActiveDocument.GetLetterContent.Subject
End If
```

This example sets the subject for the mailer associated with the active document.

```
If ActiveDocument.HasMailer = True Then
    ActiveDocument.Mailer.Subject = "Here is the document"
End If
```

TrackStatus Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTrackStatusC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTrackStatusX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTrackStatusA "}

True if a mail message is sent back to the original sender each time the routed document is forwarded. Read/write **Boolean** before routing begins; read-only **Boolean** while routing is in progress.

TrackStatus Property Example

This example adds a routing slip to the active document, adds two recipients, enables status tracking, and routes the document.

```
ActiveDocument.HasRoutingSlip = True
With ActiveDocument.RoutingSlip
    .AddRecipient Recipient:="James Allard"
    .AddRecipient Recipient:="Rich Andrews"
    .TrackStatus = True
    .Parent.Route
End With
```

Recipients Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRecipientsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRecipientsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRecipientsA "}

Returns a recipient name from the specified routing slip. Read-only **String**.

Syntax

expression.**Recipients(*Index*)**

expression Required. An expression that returns a **RoutingSlip** object.

Index Optional **Variant**. A number that specifies the recipient (in the list of recipients).

Recipients Property Example

This example adds a recipient to the routing slip attached to Sales.doc and then displays the name of the first recipient.

```
If Documents("Sales.doc").HasRoutingSlip = True Then
    Documents("Sales.doc").RoutingSlip.AddRecipient Recipient:="Aaron Con"
    MsgBox Documents("Sales.doc").RoutingSlip.Recipients(1)
End If
```

CheckName Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCheckNameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCheckNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCheckNameA"}

Validates the e-mail addresses that appear in the **To:**, **Cc:**, and **Bcc:** lines in the active WordMail message. This method is available only if WordMail is installed.

Note If the names cannot be validated, the **Check Names** dialog box is displayed.

Syntax

expression.**CheckName**

expression Required. An expression that returns a **MailMessage** object.

CheckName Method Example

This example validates the e-mail addresses that appear in the active WordMail message.

```
Application.MailMessage.CheckName
```

DisplayMoveDialog Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDisplayMoveDialogC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDisplayMoveDialogX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDisplayMoveDialogA"}

Displays the **Move** dialog box, in which the user can specify a new location for the active WordMail message in an available message store. This method is available only if WordMail is installed.

Syntax

expression.**DisplayMoveDialog**

expression Required. An expression that returns a **MailMessage** object.

DisplayMoveDialog Method Example

This example displays the **Move** dialog box for the active WordMail message.

```
Application.MailMessage.DisplayMoveDialog
```

DisplayProperties Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDisplayPropertiesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDisplayPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDisplayPropertiesA"}

Displays the **Properties** dialog box for the active WordMail message. This method is available only if WordMail is installed.

Syntax

expression.**DisplayProperties**

expression Required. An expression that returns a **MailMessage** object.

DisplayProperties Method Example

This example displays the **Properties** dialog box for the active WordMail message.

```
Application.MailMessage.DisplayProperties
```

DisplaySelectNamesDialog Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDisplaySelectNamesDialogC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDisplaySelectNamesDialogX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDisplaySelectNamesDialogA"}

Displays the **Select Names** dialog box, in which the user can add addresses to the **To:**, **Cc:**, and **Bcc:** lines in the active, unsent WordMail message. This method is available only if WordMail is installed.

Syntax

expression.**DisplaySelectNamesDialog**

expression Required. An expression that returns a **MailMessage** object.

DisplaySelectNamesDialog Method Example

This example displays the **Select Names** dialog box for the active WordMail message.

```
Application.MailMessage.DisplaySelectNamesDialog
```

Forward Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthForwardC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthForwardX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthForwardA"}

Opens a new WordMail message with an empty **To:** line for forwarding the active message. This method is available only if WordMail is installed.

Syntax

expression.**Forward**

expression Required. An expression that returns a **MailMessage** object.

Forward Method Example

This example opens a new WordMail message for forwarding the active message.

```
Application.MailMessage.Forward
```

MailingLabel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailingLabelC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailingLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailingLabelA"}

Returns a **MailingLabel** object that represents a mailing label. Read-only.

MailingLabel Property Example

This example creates a new Avery 2160 minilabel document, using a predefined address.

```
addr = "Dave Edson" & vbCr & "123 Skye St." & vbCr & "Our Town, WA  98004"
Application.MailingLabel.CreateNewDocument _
    Name:="2160 mini", Address:=addr, ExtractAddress:=False
```

MailMessage Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMailMessageC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMailMessageX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMailMessageA"}

Returns a **MailMessage** object that represents the active WordMail message. Read-only.

MailMessage Property Example

This example displays the **Select Names** dialog box for the active WordMail message.

```
Application.MailMessage.DisplaySelectNamesDialog
```

MAPIAvailable Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMAPIAvailableC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMAPIAvailableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMAPIAvailableA"}

True if MAPI is installed. Read-only **Boolean**.

MAPIAvailable Property Example

This example displays a message if MAPI is installed.

```
If Application.MAPIAvailable = True Then
    MsgBox "MAPI is available"
End If
```

Post Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPostC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPostX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPostA"}

Posts the specified document to a public folder in Microsoft Exchange. This method displays the **Send to Exchange Folder** dialog box so that a folder can be selected.

Syntax

expression.**Post**

expression Required. An expression that returns a **Document** object.

Post Method Example

This example displays the **Send to Exchange Folder** dialog box so that the active document can be posted to a public folder.

```
ActiveDocument.Post
```

Reply Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthReplyC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthReplyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthReplyA"}

MailMessage object: Opens a new WordMail message – with the sender's address on the **To:** line – for replying to the active message.

Document object: Replies to the document by creating a copy of it and preinitializing the new document's mailer to send to the originator. Valid only when the document has a received mailer attached to it (as you can only reply to a document if you've received it). Available only on the Macintosh, with the PowerTalk mail system extension installed.

Syntax

expression.**Reply**

expression Required. An expression that returns a **Document** or **MailMessage** object.

Remarks

To reply to a document, use this method to set up the mailer, use the **Mailer** property to adjust the mailer settings (if necessary), and then use the **SendMailer** method to send the reply.

Reply Method Example

This example replies to the sender of the active document.

```
Set original = ActiveDocument
If original.HasMailer Then
    original.Reply
    original.Mailer.Subject = "Here's my reply"
    ActiveDocument.SendMailer
End If
```

This example opens a new WordMail message for replying to the active message.

```
Application.MailMessage.Reply
```

ReplyAll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthReplyAllC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthReplyAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthReplyAllA"}

MailMessage object: Opens a new WordMail message – with the sender's and all other recipients' addresses on the **To:** and **Cc:** lines, as appropriate – for replying to the active message.

Document object: Replies to the document by creating a copy of it and preinitializing the new document's mailer to send to all recipients of the original. Valid only when the document has a received mailer attached (as you can only reply to a document if you've received it). Available only on the Macintosh, with the PowerTalk mail system extension installed.

Syntax

expression.**ReplyAll**

expression Required. An expression that returns a **Document** or **MailMessage** object.

Remarks

To reply to all recipients of a document, use this method to set up the mailer, use the **Mailer** property to adjust the mailer settings (if necessary), and then use the **SendMailer** method to send the reply.

ReplyAll Method Example

This example replies to all recipients of the active document.

```
Set original = ActiveDocument
If original.HasMailer Then
    original.ReplyAll
    original.Mailer.Subject = "Here's my reply"
    ActiveDocument.SendMailer
End If
```

This example opens a new WordMail message for replying to the active message.

```
Application.MailMessage.ReplyAll
```

ToggleHeader Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthToggleHeaderC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthToggleHeaderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthToggleHeaderA"}

Toggles the display of the header in the active WordMail message.

Syntax

expression.**ToggleHeader**

expression Required. An expression that returns a **MailMessage** object.

ToggleHeader Method Example

This example toggles the display of the header in the active WordMail message.

```
Application.MailMessage.ToggleHeader
```

GetAddress Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGetAddressC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGetAddressX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGetAddressA "}

Returns an address from the default address book.

Syntax

expression.**GetAddress(*Name, AddressProperties, UseAutoText, DisplaySelectDialog, SelectDialog, CheckNamesDialog, RecentAddressesChoice, UpdateRecentAddresses*)**

expression Required. An expression that returns an **Application** object.

Name Optional **Variant**. The name of the addressee, as specified in the **Search Name** dialog box in the address book.

AddressProperties Optional **Variant**. If ***UseAutoText*** is **True**, this argument denotes the name of an AutoText entry that defines a sequence of address book properties. If ***UseAutoText*** is **False** or omitted, this argument defines a custom layout. Valid address book property names or sets of property names are surrounded by angle brackets ("<" and ">") and separated by a space or a paragraph mark (for example, "<PR_GIVEN_NAME> <PR_SURNAME>" & vbCr & "<PR_OFFICE_TELEPHONE_NUMBER>").

If this argument is omitted, default AutoText entry named "AddressLayout" is used. If "AddressLayout" hasn't been defined, the following address layout definition is used: "<PR_GIVEN_NAME> <PR_SURNAME>" & vbCr & "<PR_STREET_ADDRESS>" & vbCr & "<PR_LOCALITY>" & ", " & "<PR_STATE_OR_PROVINCE>" & " " & "<PR_POSTAL_CODE>" & vbCr & "<PR_COUNTRY>".

For a list of the valid address book property names, see the **AddAddress** method.

UseAutoText Optional **Variant**. **True** if ***AddressProperties*** specifies the name of an AutoText entry that defines a sequence of address book properties; **False** if it specifies a custom layout.

DisplaySelectDialog Optional **Variant**. Specifies whether the **Select Name** dialog box is displayed, as shown in the following table.

Value	Result
0 (zero)	The **Select Name** dialog box isn't displayed.
1 or omitted	The **Select Name** dialog box is displayed.
2	The **Select Name** dialog box isn't displayed, and no search for a specific name is performed. The address returned by this method will be the previously selected address.

SelectDialog Optional **Variant**. Specifies how the **Select Name** dialog box should be displayed (that is, in what mode), as shown in the following table.

Value	Display mode
0 (zero) or omitted	Browse mode
1	Compose mode, with only the **To:** box
2	Compose mode, with both the **To:** and **CC:** boxes

CheckNamesDialog Optional **Variant**. **True** to display the **Check Names** dialog box when the value of the ***Name*** argument isn't specific enough.

RecentAddressesChoice Optional **Variant**. **True** to use the list of recently used return addresses.

UpdateRecentAddresses Optional **Variant**. **True** to add an address to the list of recently used addresses; **False** to not add the address. If ***SelectDialog*** is set to 1 or 2, this argument is ignored.

GetAddress Method Example

This example sets the variable `letAddress` to John Smith's address, moves the insertion point to the beginning of the document, and inserts the address. The inserted address will include the default address book properties.

```
letAddress = Application.GetAddress(Name:="John Smith",
CheckNamesDialog:=True)
ActiveDocument.Range(Start:=0, End:=0).InsertAfter letAddress
```

The following example returns John Smith's address, using the "My Address Layout" AutoText entry as the layout definition. "My Address Layout" is defined in the active template and contains a set of address properties assigned to the `text$` variable. The example also adds John Smith's address to the list of recently used addresses.

```
Dim TagIDArray(0 To 3) As String
Dim ValueArray(0 To 3) As String
TagIDArray(0) = "PR_DISPLAY_NAME"
TagIDArray(1) = "PR_GIVEN_NAME"
TagIDArray(2) = "PR_SURNAME"
TagIDArray(3) = "PR_COMMENT"
ValueArray(0) = "Display_Name"
ValueArray(1) = "Kim"
ValueArray(2) = "Buhler"
ValueArray(3) = "This is a comment"
Application.AddAddress TagID:=TagIDArray(), Value:=ValueArray()
addr = Application.GetAddress(Name:="Kim Buhler",
UpdateRecentAddresses:=True)
```

MailMessage Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMessageC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMessageP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMessageM"}

Application

Multiple Objects
MailMessage

Represents the active WordMail message.

Using the MailMessage Object

Use the **MailMessage** property to return the **MailMessage** object. The following example validates the e-mail addresses that appear in the active WordMail message.

```
Application.MailMessage.CheckName
```

Remarks

The methods of the **MailMessage** object require that WordMail be installed and that a WordMail message be active. If either of these conditions isn't true, an error occurs.

NextLetter Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextLetterC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextLetterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextLetterA"}

Opens the oldest unread Microsoft Word letter from the **In Tray**. Available only on the Macintosh, with the PowerTalk mail system extension installed.

Note In Windows, **NextLetter** isn't available and generates an error.

Syntax

expression.**NextLetter**

expression Required. An expression that returns an **Application** object.

NextLetter Method Example

This example opens the oldest unread Microsoft Word letter from the **In Tray**.

```
If Application.MailSystem = wdPowerTalk Then _
    Application.NextLetter
```

FocusInMailHeader Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMAPIAvailableC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMAPIAvailableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMAPIAvailableA"}

True if the insertion point is in a WordMail header field (the **To:** field, for example). Read-only **Boolean**.

FocusInMailHeader Property Example

This example displays a message in the status bar if the insertion point is in a WordMail header field.

```
If Application.FocusInMailHeader = True Then
    StatusBar = "Selection is in message header"
End If
```

BottomMargin Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBottomMarginC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBottomMarginX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBottomMarginA "}

Returns or sets the distance (in points) between the bottom edge of the page and the bottom boundary of the body text. Read/write **Single**.

BottomMargin Property Example

This example sets the bottom margin to 72 points (1 inch) and the top margin to 2 inches for the active document. The **InchesToPoints** method is used to convert inches to points.

```
With ActiveDocument.PageSetup
    .BottomMargin = 72
    .TopMargin = InchesToPoints(2)
End With
```

This example sets the bottom margin to 2.5 inches for all the sections in the current selection.

```
Selection.PageSetup.BottomMargin = InchesToPoints(2.5)
```

This example returns the bottom margin for section 1 in the selection. The **PointsToInches** method is used to convert the result to inches.

```
myvar = Selection.Sections(1).PageSetup.BottomMargin
MsgBox PointsToInches(myvar) & " inches"
```

CountBy Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCountByC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCountByX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCountByA "}

Returns or sets the numeric increment for line numbers. For example, if the **CountBy** property is set to 5, every fifth line will display the line number. Line numbers are only displayed in page layout view and print preview. Read/write **Long**.

Remarks

This property has no effect unless the **Active** property of the **LineNumbering** object is set to **True**.

CountBy Property Example

This example turns on line numbering for the active document. The line number is displayed on every fifth line and line numbering starts over for each new section.

```
With ActiveDocument.PageSetup.LineNumbering
    .Active = True
    .CountBy = 5
    .RestartMode = wdRestartSection
End With
```

FirstPageTray Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFirstPageTrayC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFirstPageTrayX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFirstPageTrayA "}

Returns or sets the paper tray to use for the first page of a document or section. Read/write **Long**.

Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed
wdPrinterDefaultBin
wdPrinterEnvelopeFeed
wdPrinterFormSource
wdPrinterLargeCapacityBin
wdPrinterLargeFormatBin
wdPrinterLowerBin
wdPrinterManualEnvelopeFeed
wdPrinterManualFeed
wdPrinterMiddleBin
wdPrinterOnlyBin
wdPrinterPaperCassette
wdPrinterSmallFormatBin
wdPrinterTractorFeed
wdPrinterUpperBin

FirstPageTray Property Example

This example sets the tray to use for printing the first page of each section in the active document.

```
ActiveDocument.PageSetup.FirstPageTray = wdPrinterLowerBin
```

This example sets the tray to use for printing the first page of each section in the selection.

```
Selection.PageSetup.FirstPageTray = wdPrinterUpperBin
```

FooterDistance Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFooterDistanceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFooterDistanceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFooterDistanceA "}

Returns or sets the distance (in points) between the footer and the bottom of the page. Read/write **Single**.

FooterDistance Property Example

This example sets the distance between the footer and the bottom of the page to 0.5 inch. The **InchesToPoints** method is used to convert inches to points.

```
ActiveDocument.PageSetup.FooterDistance = InchesToPoints(0.5)
```

This example sets the distance between the footer and the bottom of the page for all the sections in the selection to 1 inch.

```
Selection.Range.PageSetup.FooterDistance = 72
```

Footers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFootersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFootersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFootersA "}

Returns a **HeadersFooters** collection that represents the footers in the specified section. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Footers Property Example

This example adds a right-aligned page number to the primary footer in the first section in the active document.

```
With ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary)
    .PageNumbers.Add PageNumberAlignment:=wdAlignPageNumberRight
End With
```

Gutter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproGutterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproGutterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproGutterA "}

Returns or sets the amount (in points) of extra margin space added to each page in a document or section for binding. Read/write **Single**.

Remarks

If the **MirrorMargins** property is set to **True**, the **Gutter** property adds the extra space to the inside margins. Otherwise, the extra space is added to the left margin.

Gutter Property Example

This example adds 1 inch (72 points) to the inside margins of the active document.

```
With ActiveDocument.PageSetup
    .MirrorMargins = True
    .Gutter = 72
End With
```

HeaderDistance Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeaderDistanceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeaderDistanceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeaderDistanceA "}

Returns or sets the distance (in points) between the header and the top of the page. Read/write **Single**.

HeaderDistance Property Example

This example displays the distance between the header and the top of the page. The **PointsToInches** method is used to convert points to inches.

```
dist = ActiveDocument.PageSetup.HeaderDistance
Msgbox PointsToInches(dist) & " inches"
```

LeftMargin Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLeftMarginC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLeftMarginX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLeftMarginA":0)}

Returns or sets the distance (in points) between the left edge of the page and the left boundary of the body text. Read/write **Single**.

Remarks

If the **MirrorMargins** property is set to **True**, the **LeftMargin** property controls the setting for inside margins and the **RightMargin** property controls the setting for outside margins.

LeftMargin Property Example

This example sets the left margin to 1 inch (72 points) for the second section in the active document.

```
ActiveDocument.Sections(2).PageSetup.LeftMargin = 72
```

LineBetween Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineBetweenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineBetweenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineBetweenA "}

True if vertical lines appear between all the columns in the **TextColumns** collection. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

LineBetween Property Example

This example cycles through each section in the active document and displays a message box if the text columns in the section are separated by vertical lines.

```
i = 1
For each s in ActiveDocument.Sections
    If s.PageSetup.TextColumns.LineBetween = True Then
        MsgBox "The columns in section " & i & " contain lines."
    End If
    i = i + 1
Next s
```

OtherPagesTray Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOtherPagesTrayC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOtherPagesTrayX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOtherPagesTrayA "}

Returns or sets the paper tray to be used for all but the first page of a document or section. Read/write **Long**.

Can be one of the following **WdPaperTray** constants:

wdPrinterAutomaticSheetFeed
wdPrinterDefaultBin
wdPrinterEnvelopeFeed
wdPrinterFormSource
wdPrinterLargeCapacityBin
wdPrinterLargeFormatBin
wdPrinterLowerBin
wdPrinterManualEnvelopeFeed
wdPrinterManualFeed
wdPrinterMiddleBin
wdPrinterOnlyBin
wdPrinterPaperCassette
wdPrinterSmallFormatBin
wdPrinterTractorFeed
wdPrinterUpperBin

OtherPagesTray Property Example

This example sets the tray to be used for printing all but the first page of each section in the active document.

```
ActiveDocument.PageSetup.OtherPagesTray = wdPrinterUpperBin
```

This example sets the tray to be used for printing all but the first page of each section in the selection.

```
Selection.PageSetup.OtherPagesTray = wdPrinterLowerBin
```

PageNumbers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageNumbersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageNumbersA "}

Returns a **PageNumbers** collection that represents all the page number fields included in the specified header or footer. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

PageNumbers Property Example

This example creates a new document and adds page numbers to the footer.

```
Set myDoc = Documents.Add
With myDoc.Sections(1).Footers(wdHeaderFooterPrimary)
    .PageNumbers.Add PageNumberAlignment := wdAlignPageNumberCenter
End With
```

RightMargin Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRightMarginC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRightMarginX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRightMarginA "}

Returns or sets the distance (in points) between the right edge of the page and the right boundary of the body text. Read/write **Single**.

Remarks

If the **MirrorMargins** property is set to **True**, the **RightMargin** property controls the setting for outside margins and the **LeftMargin** property controls the setting for inside margins.

RightMargin Property Example

This example displays the right margin setting for the active document. The **PointsToInches** method is used to convert the result to inches.

```
With ActiveDocument.PageSetup
    Msgbox "The right margin is set to " & _
        PointsToInches(.RightMargin) & " inches."
End With
```

This example sets the right margin for section two in the selection. The **InchesToPoints** method is used to convert inches to points.

```
Selection.Sections(2).PageSetup.RightMargin = InchesToPoints(1)
```

TogglePortrait Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTogglePortraitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTogglePortraitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTogglePortraitA "}

Switches between portrait and landscape page orientations for a document or section.

Syntax

expression.**TogglePortrait**

expression Required. An expression that returns a **PageSetup** object.

Remarks

If the specified sections have different page orientations, an error occurs.

TogglePortrait Method Example

This example changes the page orientation for the active document.

```
ActiveDocument.PageSetup.TogglePortrait
```

This example changes the page orientation for all the sections in the selection. If the initial orientation of each section is not the same as the orientation of the other sections, an error occurs.

```
Selection.PageSetup.TogglePortrait
```

VerticalAlignment Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalAlignmentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalAlignmentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalAlignmentA "}

Returns or sets the vertical alignment of text on each page in a document or section, or in one or more cells of a table. Can be one of the following **WdVerticalAlignment** constants: **wdAlignVerticalBottom**, **wdAlignVerticalCenter**, **wdAlignVerticalJustify**, or **wdAlignVerticalTop**. Read/write **Long**.

VerticalAlignment Property Example

This example changes the vertical alignment of the first document so that the text is centered between the top and bottom margins.

```
Documents(1).PageSetup.VerticalAlignment = wdAlignVerticalCenter
```

This example creates a new document and then inserts the same paragraph 10 times. The vertical alignment of the new document is then set so that the 10 paragraphs are equally spaced (justified) between the top and bottom margins.

```
Set myDoc = Documents.Add
With myDoc.Content
    For i = 1 to 9
        .InsertAfter "This is a sentence."
        .InsertParagraphAfter
    Next i
    .InsertAfter "This is a sentence."
End With
myDoc.PageSetup.VerticalAlignment = wdAlignVerticalJustify
```

This example creates a 3x3 table in a new document and assigns a sequential cell number to each cell in the table. The example then sets the height of the first row to 20 points and vertically aligns the text at the top of the cells.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
i = 1
For Each c In myTable.Range.Cells
    c.Range.InsertAfter "Cell " & i
    i = i + 1
Next
With myTable.Rows(1)
    .Height = 20
    .Cells.VerticalAlignment = wdAlignVerticalTop
End With
```

MirrorMargins Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMirrorMarginsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMirrorMarginsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMirrorMarginsA "}

True if the inside margins of facing pages are the same width and the outside margins are the same width. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Remarks

If the **MirrorMargins** property is set to **True**, the **LeftMargin** property controls the setting for inside margins and the **RightMargin** property controls the setting for outside margins.

MirrorMargins Property Example

This example sets the inside margins in the active document to 1 inch (72 points) and the outside margins to 0.5 inch. The **InchesToPoints** method is used to convert inches to points.

```
With ActiveDocument.PageSetup
    .MirrorMargins = True
    .LeftMargin = 72
    .RightMargin = InchesToPoints(0.5)
End With
```

TopMargin Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTopMarginC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTopMarginX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTopMarginA "}

Returns or sets the distance (in points) between the top edge of the page and the top boundary of the body text. Read/write **Single**.

TopMargin Property Example

This example sets the top margin to 72 points (1 inch) for the first section in the active document.

```
ActiveDocument.Sections(1).PageSetup.TopMargin = 72
```

This example creates a new custom label and sets several properties, including the top margin, and then it creates a new document using the custom labels.

```
Set newlbl = Application.MailingLabel. _
    CustomLabels.Add(Name:="My Label")
With newlbl
    .Height = InchesToPoints(1.25)
    .NumberAcross = 2
    .NumberDown = 7
    .PageSize = wdCustomLabelLetter
    .SideMargin = InchesToPoints(0)
    .TopMargin = InchesToPoints(1)
    .Width = InchesToPoints(4.25)
End With
Application.MailingLabel.CreateNewDocument Name:="My Label"
```

EvenlySpaced Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproEvenlySpacedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproEvenlySpacedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproEvenlySpacedA "}

True if text columns are evenly spaced. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Remarks

If you set the **Spacing** or **Width** property of the **TextColumns** object, the **EvenlySpaced** property is automatically set to **True**. Also, setting the **EvenlySpaced** property may change the settings for the **Spacing** and **Width** properties of the **TextColumns** object.

EvenlySpaced Property Example

This example topic sets columns in the active document to be evenly spaced.

```
Set myCols = ActiveDocument.PageSetup.TextColumns
    If myCols.Count > 1 Then myCols.EvenlySpaced = True
```

This example returns the status of the **Equal column width** option in the **Columns** dialog box (**Format** menu).

```
temp = ActiveDocument.PageSetup.TextColumns.EvenlySpaced
```

Headers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeadersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeadersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeadersA "}

Returns a **HeadersFooters** collection that represents the headers for the specified section. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Remarks

To return a **HeadersFooters** collection that represents the footers for the specified section, use the **Footers** property.

Headers Property Example

This example adds centered page numbers to every page in the active document except the first. (A separate header is created for the first page.)

```
With ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary)
    .PageNumbers.Add PageNumberAlignment:=wdAlignPageNumberCenter, _
        FirstPage:=False
End With
```

This example adds text to the first-page header in the active document.

```
ActiveDocument.PageSetup.DifferentFirstPageHeaderFooter = True
With ActiveDocument.Sections(1).Headers(wdHeaderFooterFirstPage)
    .Range.InsertAfter("First Page Text")
    .Range.Paragraphs.Alignment = wdAlignParagraphRight
End With
```

IsHeader Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsHeaderC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsHeaderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsHeaderA "}

True if the specified **HeaderFooter** object is a header. Read-only **Boolean**.

IsHeader Property Example

This example selects the footer and adds a page number.

```
With ActiveWindow.ActivePane.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageHeader
End With
If Selection.HeaderFooter.IsHeader = True Then
    ActiveWindow.ActivePane.View _
        .SeekView = wdSeekCurrentPageFooter
End If
Selection.HeaderFooter.PageNumbers.Add
```

LineNumbering Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLineNumberingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLineNumberingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLineNumberingA "}

Returns or sets the **LineNumbering** object that represents the line numbers for the specified **PageSetup** object. Read/write.

Remarks

You must be in page layout view to see line numbering.

LineNumbering Property Example

This example enables line numbering for the active document.

```
ActiveDocument.PageSetup.LineNumbering.Active = True
```

This example enables line numbering for a document named "My Document." The starting number is set to 1, every fifth line number is shown, and the numbering is continuous throughout all sections in the document.

```
set myDoc = Documents("MyDocument")
With myDoc.PageSetup.LineNumbering
    .Active = True
    .StartingNumber = 1
    .CountBy = 5
    .RestartMode = wdRestartContinuous
End With
```

This example sets the line numbering in the active document equal to the line numbering in MyDocument.

```
ActiveDocument.PageSetup.LineNumbering = Documents("MyDocument") _
    .PageSetup.LineNumbering
```

OddAndEvenPagesHeaderFooter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOddAndEvenPagesHeaderFooterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOddAndEvenPagesHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOddAndEvenPagesHeaderFooterA "}

True if the specified **PageSetup** object has different headers and footers for odd-numbered pages and even-numbered pages. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

OddAndEvenPagesHeaderFooter Property Example

This example creates different headers and footers for odd-numbered pages and even-numbered pages in Document1.

```
Set myDoc = Documents("Document1")
myDoc.PageSetup.OddAndEvenPagesHeaderFooter = True
With myDoc.Sections(1)
    .Headers(wdHeaderFooterPrimary).Range.InsertAfter "Odd Header"
    .Headers(wdHeaderFooterEvenPages).Range.InsertAfter "Even Header"
End With
```

PageHeight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageHeightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageHeightA "}

Returns or sets the height of the page, in points. Read/write **Single**.

Remarks

Setting the **PageHeight** property changes the **PaperSize** property to **wdPaperCustom**.

Use the **PaperSize** property to set the page height and page width to those of a predefined paper size, such as Letter or A4.

PageHeight Property Example

This example sets the page height for the active document to 9 inches.

```
With ActiveDocument.PageSetup
    .PageHeight = InchesToPoints(9)
    .PageWidth = InchesToPoints(7)
End With
```

PageWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageWidthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageWidthA "}

Returns or sets the width of the page, in points. Read/write **Single**.

Remarks

Setting the **PageWidth** property changes the **PaperSize** property to **wdPaperCustom**.

Use the **PaperSize** property to set the page height and page width to those of a predefined paper size, such as Letter or A4.

PageWidth Property Example

This example returns the page width for Document1. The **PointsToInches** method is used to convert points to inches.

```
Set doc1set = Documents("Document1").PageSetup
Msgbox "The page width is " & _
    PointsToInches(doc1set.PageWidth) & " inches."
```

RestartMode Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRestartModeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRestartModeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRestartModeA "}

Returns or sets the way line numbering runs – that is, whether it starts over at the beginning of a new page or section or runs continuously. Can be one of the following **WdNumberingRule** constants: **wdRestartContinuous**, **wdRestartPage**, or **wdRestartSection**. Read/write **Long**.

Remarks

You must be in page layout view to see line numbering.

RestartMode Property Example

This example enables line numbering for the active document. The starting number is set to 1, every tenth line number is shown, and the numbering starts over at the beginning of each section.

```
set myDoc = ActiveDocument
With myDoc.PageSetup.LineNumbering
    .Active = True
    .StartingNumber = 1
    .CountBy = 10
    .RestartMode = wdRestartSection
End With
```

SectionStart Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSectionStartC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSectionStartX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSectionStartA "}

Returns or sets the type of section break for the specified **PageSetup** object. Can be one of the following **WdSectionStart** constants: **wdSectionContinuous**, **wdSectionEvenPage**, **wdSectionNewColumn**, **wdSectionNewPage**, or **wdSectionOddPage**. Read/write **Long**.

SectionStart Property Example

This example changes the type of section break to continuous for all sections in the active document.

```
ActiveDocument.PageSetup.SectionStart = wdSectionContinuous
```

This example returns the type of section break used at the beginning of the second section in MyDoc and applies it to all the sections in the active document.

```
mytype = Documents("MyDoc").Sections(2).PageSetup.SectionStart
ActiveDocument.PageSetup.SectionStart = mytype
```

StartingNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStartingNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStartingNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStartingNumberA "}

Returns or sets the starting note number, line number, or page number. Read/write **Long**.

Remarks

You must be in page layout view to see line numbering.

When applied to page numbers, this property returns or sets the beginning page number for the specified **HeaderFooter** object. This number may or may not be visible on the first page, depending on the setting of the **ShowFirstPageNumber** property. The **RestartNumberingAtSection** property, if set to **False**, will override the **StartingNumber** property so that page numbering can continue from the previous section.

StartingNumber Property Example

This example creates a new document, sets the starting number for footnotes to 10, and then adds a footnote at the insertion point.

```
Set myDoc =Documents.Add
With myDoc.Footnotes
    .StartingNumber = 10
    .Add Range:=Selection.Range, Text:="Text for a footnote"
End With
```

This example enables line numbering for the active document. The starting number is set to 5, every fifth line number is shown, and the numbering starts over at the beginning of each section in the document.

```
With ActiveDocument.PageSetup.LineNumbering
    .Active = True
    .StartingNumber = 5
    .CountBy = 5
    .RestartMode = wdRestartSection
End With
```

This example sets properties for page numbers, and then it adds page numbers to the header of the active document.

```
With ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).PageNumbers
    .NumberStyle = wdPageNumberStyleArabic
    .IncludeChapterNumber = False
    .RestartNumberingAtSection = True
    .StartingNumber = 5
    .Add PageNumberAlignment:=wdAlignPageNumberCenter, _
        FirstPage:=True
End With
```

ChapterPageSeparator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproChapterPageSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproChapterPageSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproChapterPageSeparatorA "}

Returns or sets the separator character used between the chapter number and the page number. Can be one of the following **WdSeparatorType** constants: **wdSeparatorColon**, **wdSeparatorEmDash**, **wdSeparatorEnDash**, **wdSeparatorHyphen**, or **wdSeparatorPeriod**. Read/write **Long**.

Remarks

Before you can create page numbers that include chapter numbers, the document headings must have a numbered outline format applied that uses styles from the **Bullets and Numbering** dialog box. To do this in Visual Basic, use the **ApplyListTemplate** method.

ChapterPageSeparator Property Example

The first part of this example creates a new document, adds chapter titles and page breaks, and then formats the document by using the first numbered outline format listed in the **Bullets and Numbering** dialog box. The second part of the example adds centered page numbers – including the chapter number – to the header; an en dash separates the chapter number and the page number.

```
Documents.Add
For i = 1 To 5
    With Selection
        .TypeText "Chapter Number " & i
        .TypeParagraph
        .InsertBreak
    End With
Next i
ActiveDocument.Content.ListFormat.ApplyListTemplate _

ListTemplate:=ListGalleries(wdOutlineNumberGallery).ListTemplates(1)

Set myHead = ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary)
With myHead.PageNumbers
    .Add PageNumberAlignment:= wdAlignPageNumberCenter
    .NumberStyle = wdPageNumberStyleArabic
    .IncludeChapterNumber = True
    .HeadingLevelForChapter = 0
    .ChapterPageSeparator = wdSeparatorEnDash
End With
```

HeadingLevelForChapter Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeadingLevelForChapterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeadingLevelForChapterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeadingLevelForChapterA "}

Returns or sets the heading level style that's applied to the chapter titles in the document. Can be a number from 0 (zero) through 8, corresponding to heading levels 1 through 9. Read/write **Long**.

Remarks

Before you can create page numbers that include chapter numbers, the document headings must have a numbered outline format applied that uses styles from the **Bullets and Numbering** dialog box. To do this in Visual Basic, use the **ApplyListTemplate** method.

HeadingLevelForChapter Property Example

The first part of this example creates a new document, adds chapter titles and page breaks, and then formats the document by using the fifth numbered outline format listed in the **Bullets and Numbering** dialog box. The second part of the example adds centered page numbers - including the chapter number - to the footer; an en dash separates the chapter number and the page number. The first heading level is used for the chapter number, and lowercase roman numerals are used for the page number.

```
Documents.Add
For i = 1 To 5
    With Selection
        .TypeText "Chapter Number " & i
        .TypeParagraph
        .InsertBreak
    End With
Next i
ActiveDocument.Content.ListFormat.ApplyListTemplate _

ListTemplate:=ListGalleries(wdOutlineNumberGallery).ListTemplates(5)

Set myFoot = ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary)
With myFoot.PageNumbers
    .Add PageNumberAlignment:= wdAlignPageNumberCenter
    .NumberStyle = wdPageNumberStyleLowercaseRoman
    .IncludeChapterNumber = True
    .HeadingLevelForChapter = 0
    .ChapterPageSeparator = wdSeparatorEnDash
End With
```

ShowFirstPageNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowFirstPageNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowFirstPageNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowFirstPageNumberA "}

True if the page number appears on the first page in the section. Read/write **Boolean**.

Remarks

Setting this property to **True** automatically adds page numbers to a section.

ShowFirstPageNumber Property Example

This example checks to see whether the page number appears on the first page in the active document.

```
Set myDoc = ActiveDocument
first = myDoc.Sections(1).Headers(wdHeaderFooterPrimary). _
    PageNumbers.ShowFirstPageNumber
Msgbox "This document shows numbers on the first page - " & first
```

This example adds page numbers to the active document.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).PageNumbers _
    .ShowFirstPageNumber = True
```

ActiveSpellingDictionary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveSpellingDictionaryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveSpellingDictionaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveSpellingDictionaryA "}

Returns a **Dictionary** object that represents the active spelling dictionary for the specified language. Read-only.

Remarks

If there's no spelling dictionary installed for the specified language, this property returns **Nothing**.

ActiveSpellingDictionary Property Example

This example returns the full path and file name of the active spelling dictionary.

```
myLang = Selection.LanguageID
Set mySpell = Languages(myLang).ActiveSpellingDictionary
MsgBox mySpell.Path & Application.PathSeparator & mySpell.Name
```

CheckSpelling Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCheckSpellingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCheckSpellingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCheckSpellingA "}

Syntax 1: Begins a spelling check for the specified document or range. If the document or range contains errors, this method displays the **Spelling and Grammar** dialog box (**Tools** menu), with the **Check grammar** check box cleared. For a document, this method checks all available stories (such as headers, footers, and text boxes).

Syntax 2: Checks a string for spelling errors. Returns **True** if the string has no spelling errors.

Syntax 1

expression.**CheckSpelling(*CustomDictionary, IgnoreUppercase, AlwaysSuggest, CustomDictionary2 – CustomDictionary10*)**

Syntax 2

expression.**CheckSpelling(*Word, CustomDictionary, IgnoreUppercase, MainDictionary, CustomDictionary2 – CustomDictionary10*)**

expression Syntax 1: Required. An expression that returns a **Document** or **Range** object.

Syntax 2: Optional. An expression that returns an **Application** object.

Word Required **String**. The text whose spelling is to be checked.

CustomDictionary Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of the custom dictionary.

IgnoreUppercase Optional **Variant**. **True** if capitalization is ignored. If this argument is omitted, the current value of the **IgnoreUppercase** property is used.

AlwaysSuggest Optional **Variant**. **True** if Word always suggests alternative spellings. If this argument is omitted, the current value of the **SuggestSpellingCorrections** property is used.

MainDictionary Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of the main dictionary.

CustomDictionary2 – CustomDictionary10 Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of an additional custom dictionary. You can specify as many as nine additional dictionaries.

CheckSpelling Method Example

This example begins a spelling check on all available stories of the active document.

```
ActiveDocument.CheckSpelling
```

This example begins a spelling check on section two of MyDocument.doc. The spelling check includes words in all uppercase letters, and it checks words against two custom dictionaries as well as the main dictionary.

```
Set range2 = Documents("MyDocument.doc").Sections(2).Range
range2.CheckSpelling IgnoreUpperCase:=False, _
    CustomDictionary:="MyWork.Dic", _
    CustomDictionary2:="MyTechnical.Dic"
```

This example begins a spelling check on the selection.

```
Selection.Range.CheckSpelling
```

This example displays the result of a spelling check on the selection.

```
pass = Application.CheckSpelling(Word:=Selection.Text)
MsgBox "Selection has no spelling errors: " & pass
```

CheckSpellingAsYouType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCheckSpellingAsYouTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCheckSpellingAsYouTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCheckSpellingAsYouTypeA "}

True if Word checks spelling and marks errors automatically as you type. Read/write **Boolean**.

Remarks

This property marks spelling errors, but to see them on the screen, you must set the **ShowSpellingErrors** property to **True**.

CheckSpellingAsYouType Property Example

This example turns off automatic spell checking in Word.

```
Options.CheckSpellingAsYouType = False
```

This example sets Word to check for spelling errors as you type and to display any errors found in the active document.

```
Options.CheckSpellingAsYouType = True
ActiveDocument.ShowSpellingErrors = True
```

This example returns the status of the **Check spelling as you type** option on the **Spelling & Grammar** tab in the **Options** dialog box (**Tools** menu).

```
temp = Options.CheckSpellingAsYouType
```

SpellingChecked Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpellingCheckedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpellingCheckedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpellingCheckedA "}

True if spelling has been checked throughout the specified range or document. **False** if all or some of the range or document hasn't been checked for spelling. Read/write **Boolean**.

Remarks

To recheck the spelling in a range or document, set the **SpellingChecked** property to **False**.

To see whether the range or document contains spelling errors, use the **SpellingErrors** property.

SpellingChecked Property Example

This example determines whether spelling in section one of the active document has been checked. If spelling hasn't been checked, the example starts a spelling check.

```
Set myRange = ActiveDocument.Sections(1).Range
isChecked = myRange.SpellingChecked
If isChecked = False Then
    myRange.CheckSpelling
Else
    MsgBox "The range has already been spell checked."
End If
```

This example sets the **SpellingChecked** property to **False** for MyDocument.doc, and then it runs another spelling check on the document.

```
Documents("MyDocument.doc").SpellingChecked = False
Documents("MyDocument.doc").CheckSpelling IgnoreUppercase:=False
```

SpellingErrors Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpellingErrorsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpellingErrorsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpellingErrorsA "}

Returns a **ProofreadingErrors** collection that represents the words identified as spelling errors in the specified document or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

SpellingErrors Property Example

This example checks the active document for spelling errors and displays the number of errors found.

```
myErr = ActiveDocument.SpellingErrors.Count
If myErr = 0 Then
    Msgbox "No spelling errors found."
Else
    Msgbox myErr & " spelling errors found."
End If
```

This example checks the specified range for spelling errors and displays each error found.

```
Set myErrors = ActiveDocument.Paragraphs(3).Range.SpellingErrors
If myErrors.Count = 0 Then
    Msgbox "No spelling errors found."
Else
    For Each myErr in myErrors
        Msgbox myErr.Text
    Next
End If
```

GetSpellingSuggestions Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthGetSpellingSuggestionsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthGetSpellingSuggestionsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthGetSpellingSuggestionsA "}

Syntax 1: Returns a **SpellingSuggestions** collection that represents the words suggested as spelling replacements for the first word in the specified range.

Syntax 2: Returns a **SpellingSuggestions** collection that represents the words suggested as spelling replacements for a given word.

Syntax 1

expression.**GetSpellingSuggestions**(***CustomDictionary, IgnoreUppercase, MainDictionary, SuggestionMode, CustomDictionary2 – CustomDictionary10***)**

Syntax 2

expression.**GetSpellingSuggestions**(***Word, CustomDictionary, IgnoreUppercase, MainDictionary, SuggestionMode, CustomDictionary2 – CustomDictionary10***)**

expression Syntax 1: Required. An expression that returns a **Range** object.

Syntax 2: Optional. An expression that returns an **Application** object.

Word Required **String**. The word whose spelling is to be checked.

CustomDictionary Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of the custom dictionary.

IgnoreUppercase Optional **Variant**. **True** to ignore words in all uppercase letters. If this argument is omitted, the current value of the **IgnoreUppercase** property is used.

MainDictionary Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of the main dictionary. If you don't specify a main dictionary, Word uses the main dictionary that corresponds to the language formatting of ***Word*** or of the first word in the range.

SuggestionMode Optional **Variant**. Specifies the way Word makes spelling suggestions. Can be one of the following **WdSpellingWordType** constants. The default value is **WdSpellword**.

Constant	Description
wdSpellword	Word suggests correct spellings for the word or the first word in the specified range.
wdWildcard	Word suggests replacements that match the search criteria for a word that contains the question mark (?) or asterisk (*) wildcard character.
wdAnagram	Word suggests anagrams for the word. Word doesn't suggest anagrams from a custom dictionary.

Note This parameter may be ignored, depending on the dictionary file that's in use.

CustomDictionary2 – CustomDictionary10 Optional **Variant**. Either an expression that returns a **Dictionary** object or the file name of an additional custom dictionary. You can specify as many as nine additional dictionaries.

Remarks

If the word is spelled correctly, the **Count** property of the **SpellingSuggestions** object returns 0 (zero).

GetSpellingSuggestions Method Example

This example looks for the alternate spelling suggestions for the first word in the selection. If there are suggestions, the example runs a spelling check on the selection.

```
If Selection.Range.GetSpellingSuggestions.Count = 0 Then
    Msgbox "No suggestions."
Else
    Selection.Range.CheckSpelling
End If
```

This example looks for the alternate spelling suggestions for the word "blat?nt." Suggestions include replacements for the ? wildcard character. Any suggested spellings are displayed in messages boxes.

```
Set sugList = GetSpellingSuggestions(Word:="?ook", _
    SuggestionMode:=wdWildcard)
If sugList.Count = 0 Then
    Msgbox "No suggestions."
Else
    For each sug in sugList
        Msgbox sug.Name
    Next sug
End If
```

Activate Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthActivateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthActivateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthActivateA "}

Activates the specified object.

Syntax 1

expression.**Activate(*Wait*)**

Syntax 2

expression.**Activate**

expression Required. An expression that returns a **Task** object (Syntax 1) or another object in the Applies To list (Syntax 2).

Wait Optional **Boolean**. **Task** object only. **True** to immediately activate the task, even if Word isn't active. **False** to not activate the task until the user has activated Word.

Activate Method Example

This example activates the document named "Sales.doc."

```
Documents("Sales.doc").Activate
```

This example activates the next window in the **Windows** collection.

```
ActiveWindow.Next.Activate
```

This example activates the Notepad application if Notepad is in the **Tasks** collection.

```
For Each myTask In Tasks
    If InStr(myTask.Name, "Notepad") > 0 Then
        myTask.Activate
        myTask.WindowState = wdWindowStateNormal
    End If
Next myTask
```

This example splits the active window and then activates the first pane.

```
With ActiveWindow
    .SplitVertical = 50
    .Panes(1).Activate
End With
```

Active Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveA "}

LineNumbering object: **True** if line numbering is active for the specified document, section, or sections. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Selection object: **True** if the selection in the specified window or pane is active. Read-only **Boolean**.

Window object: **True** if the specified window is active. Read-only **Boolean**.

Active Property Example

This example activates the first window in the **Windows** collection, if the window isn't currently active.

```
If Windows(1).Active = False Then Windows(1).Activate
```

This example activates line numbering for the first section in the selection.

```
With Selection.Sections(1).PageSetup.LineNumbering
    .Active = True
    .CountBy = 5
    .StartingNumber = 1
End With
```

This example splits the active window into two panes and activates the selection in the first pane, if it isn't already active.

```
ActiveWindow.Split = True
If ActiveWindow.Panes(1).Selection.Active = False Then
    ActiveWindow.Panes(1).Activate
End If
```

ActiveDocument Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveDocumentA "}

Returns a **Document** object that represents the active document (the document with the focus). If there are no documents open, an error occurs. Read-only.

ActiveDocument Property Example

This example displays the name of the active document, or if there are no documents open, it displays a message.

```
If Documents.Count >= 1 Then
    MsgBox ActiveDocument.Name
Else
    MsgBox "No documents are open"
End If
```

This example collapses the selection to an insertion point and then creates a range for the next five characters in the selection.

```
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.Range Start:=Selection.Start, End:=Selection.Start+5
```

This example inserts texts at the beginning of the active document and then prints the document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
With myRange
    .InsertBefore "Company Report"
    .Font.Name = "Arial"
    .Font.Size = 24
    .InsertParagraphAfter
End With
ActiveDocument.PrintOut
```

ActivePane Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActivePaneC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActivePaneX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActivePaneA "}

Returns a **Pane** object that represents the active pane for the specfied window. Read-only.

ActivePane Property Example

This example splits the active window and then activates the next pane after the active pane.

```
With ActiveWindow
    .Split = True
    .ActivePane.Next.Activate
    MsgBox "Pane " & .ActivePane.Index & " is active"
End With
```

This example activates the first window and displays tabs in the active pane.

```
With Windows(1)
    .Activate
    .ActivePane.View.ShowTabs = True
End With
```

ActiveRecord Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveRecordC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveRecordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveRecordA "}

Returns or sets the active mail merge data record. Can be either a valid data record number in the query result or one of the following **WdMailMergeActiveRecord** constants: **wdFirstRecord**, **wdLastRecord**, **wdNextRecord**, **wdNoActiveRecord**, or **wdPreviousRecord**. Read/write **Long**.

Note The active data record number is the position of the record in the query result produced by the current query options; as such, this number isn't necessarily the position of the record in the data source.

ActiveRecord Property Example

This example hides the mail merge field codes in the active document so that the merge data is visible in the main document. The active record is then advanced to the next record in the data source.

```
If ActiveDocument.MailMerge.MainDocumentType <> _
        wdNotAMergeDocument Then
    With ActiveDocument.MailMerge
        .ViewMailMergeFieldCodes = False
        .DataSource.ActiveRecord = wdNextRecord
    End With
End If
```

This example returns the active data record from Main2.doc.

```
If Documents("Main2.doc").MailMerge.State = wdMainAndDataSource Or _
    wdMainAndSourceAndHeader Then
num = Documents("Main2.doc").MailMerge.DataSource.ActiveRecord
End If
```

ActiveWindow Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveWindowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveWindowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveWindowA "}

Returns a **Window** object that represents the active window (the window with the focus). If there are no windows open, an error occurs. Read-only.

ActiveWindow Property Example

This example displays the caption text for the active window.

```
MsgBox ActiveWindow.Caption
```

This example opens a new window for the active window and then tiles all the windows.

```
Set myWindow = ActiveWindow.NewWindow
Windows.Arrange ArrangeStyle:=wdTiled
```

This example splits the first document window.

```
Documents(1).ActiveWindow.Split= True
```

MoveDown Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveDownC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveDownX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveDownA "}

Moves the selection down and returns the number of units it's been moved.

Note The **wdWindow** constant can be used to move to the top or bottom of the active window. Regardless of the value of ***Count*** (greater than 1 or less than – 1), the **wdWindow** constant moves only one unit. Use the **wdScreen** constant to move more than one screen.

Syntax

expression.**MoveDown(*Unit, Count, Extend*)**

expression Required. An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which the selection is to be moved. Can be one of the following **WdUnits** constants: **wdLine**, **wdParagraph**, **wdWindow**, or **wdScreen**. The default value is **wdLine**.

Count Optional **Variant**. The number of units the selection is to be moved. The default value is 1.

Extend Optional **Variant**. Can be either **wdMove** or **wdExtend**. If **wdMove** is used, the selection is collapsed to the end point and moved down. If **wdExtend** is used, the selection is extended down. The default value is **wdMove**.

MoveDown Method Example

This example extends the selection down one line.

```
Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdExtend
```

This example moves the selection down three paragraphs. If the move is successful, "Company" is inserted at the insertion point.

```
unitsMoved = Selection.MoveDown(Unit:=wdParagraph, Count:=3,
Extend:=wdMove)
If unitsMoved = 3 Then Selection.Text = "Company"
```

This example displays the current line number, moves the selection down three lines, and displays the current line number again.

```
MsgBox "Line " & Selection.Information(wdFirstCharacterLineNumber)
Selection.MoveDown Unit:=wdLine, Count:=3, Extend:=wdMove
MsgBox "Line " & Selection.Information(wdFirstCharacterLineNumber)
```

MoveLeft Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveLeftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveLeftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveLeftA "}

Moves the selection to the left and returns the number of units it's been moved.

Syntax

expression.**MoveLeft(*Unit, Count, Extend*)**

expression Required. An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which the selection is to be moved. Can be one of the following **WdUnits** constants: **wdCell**, **wdCharacter**, **wdWord**, or **wdSentence**. The default value is **wdCharacter**.

Count Optional **Variant**. The number of units the selection is to be moved. The default value is 1.

Extend Optional **Variant**. Can be either **wdMove** or **wdExtend**. If **wdMove** is used, the selection is collapsed to the end point and moved to the left. If **wdExtend** is used, the selection is extended to the left. The default value is **wdMove**.

Remarks

When the *Unit* **is** wdCell**, the** *Extend* **argument will only be** wdMove**.**

MoveLeft Method Example

This example moves the selection one character to the left. If the move is successful, **MoveLeft** returns 1.

```
If Selection.MoveLeft = 1 Then MsgBox "Move was successful"
```

This example enables field shading for the selected field, inserts a DATE field, and then moves the selection left to select the field.

```
ActiveWindow.View.FieldShading = wdFieldShadingWhenSelected
With Selection
    .Fields.Add Range:=Selection.Range, Type:=wdFieldDate
    .MoveLeft Unit:=wdWord, Count:=1
End With
```

This example moves the selection to the previous table cell.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.MoveLeft Unit:=wdCell, Count:=1, Extend:=wdMove
End If
```

MoveRight Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveRightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveRightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveRightA "}

Moves the selection to the right and returns the number of units it's been moved.

Syntax

expression.**MoveRight(*Unit, Count, Extend*)**

expression Required. An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which the selection is to be moved. Can be one of the following **WdUnits** constants: **wdCell**, **wdCharacter**, **wdWord**, or **wdSentence**. The default value is **wdCharacter**.

Count Optional **Variant**. The number of units the selection is to be moved. The default value is 1.

Extend Optional **Variant**. Can be either **wdMove** or **wdExtend**. If **wdMove** is used, the selection is collapsed to the end point and moved to the right. If **wdExtend** is used, the selection is extended to the right. The default value is **wdMove**.

Remarks

When the ***Unit*** is **wdCell**, the ***Extend*** argument will only be **wdMove**.

MoveRight Method Example

This example moves the selection before the previous field and then selects the field.

```
With Selection
    Set MyRange = .GoTo(wdGoToField, wdGoToPrevious)
    .MoveRight Unit:=wdWord, Count:=1, Extend:=wdExtend
    If Selection.Fields.Count = 1 Then Selection.Fields(1).Update
End With
```

This example moves the selection one character to the right. If the move is successful, **MoveRight** returns 1.

```
If Selection.MoveRight = 1 Then MsgBox "Move was successful"
```

This example moves the selection to the next table cell.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.MoveRight Unit:=wdCell, Count:=1, Extend:=wdMove
End If
```

Select Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectA "}

Selects the specified object.

Note After using this method, you can use the **Selection** property to return a **Selection** object. For more informaiton, see Working with the Selection object.

Syntax

expression.**Select**

expression Required. An expression that returns an object in the Applies To list.

Select Method Example

This example selects the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Range.Select
```

This example selects row one in table one in Report.doc.

```
Documents("Report.doc").Tables(1).Rows(1).Select
```

This example updates and selects the first field in the active document.

```
ActiveWindow.View.FieldShading = wdFieldShadingWhenSelected
If ActiveDocument.Fields.Count >= 1 Then
    With ActiveDocument.Fields(1)
       .Update
       .Select
    End With
End If
```

Selection Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSelectionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSelectionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSelectionA "}

Returns the **Selection** object that represents a selected range or the insertion point. Read-only.

Selection Property Example

This example displays the selected text.

```
If Selection.Type = wdSelectionNormal Then MsgBox Selection.Text
```

This example copies the selection from window one to the next window.

```
If Windows.Count >= 2 Then
    Windows(1).Selection.Copy
    Windows(1).Next.Activate
    Selection.Paste
End If
```

This example applies the Arial font and bold formatting to the selection.

```
With Selection.Font
    .Bold = True
    .Italic = False
    .Name = "Arial"
End With
```

If the insertion point isn't located in a table, the selection is moved to the next table.

```
If Selection.Information(wdWithInTable) = False Then
    Selection.GoToNext What:=wdGoToTable
End If
```

Information Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproInformationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproInformationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproInformationA "}

Returns information about the specified selection or range. Read-only **Variant**.

Syntax

expression.**Information(*Type*)**

expression Required. An expression that returns a **Range** or **Selection** object.

Type Required **Long**. The information to return. Can be one of the following **WdInformation** constants:

- **wdActiveEndAdjustedPageNumber** Returns the number of the page that contains the active end of the specified selection or range. If you set a starting page number or make other manual adjustments, returns the adjusted page number (unlike **wdActiveEndPageNumber**).
- **wdActiveEndPageNumber** Returns the number of the page that contains the active end of the specified selection or range, counting from the beginning of the document. Any manual adjustments to page numbering are disregarded (unlike **wdActiveEndAdjustedPageNumber**).
- **wdActiveEndSectionNumber** Returns the number of the section that contains the active end of the specified selection or range.
- **wdAtEndOfRowMarker** Returns **True** if the specified selection or range is at the end-of-row mark in a table.
- **wdCapsLock** Returns **True** if Caps Lock is in effect.
- **wdEndOfRangeColumnNumber** Returns the table column number that contains the end of the specified selection or range.
- **wdEndOfRangeRowNumber** Returns the table row number that contains the end of the specified selection or range.
- **wdFirstCharacterColumnNumber** Returns the character position of the first character in the specified selection or range. If the selection or range is collapsed, the character number immediately to the right of the range or selection is returned (this is the same as the character column number displayed in the status bar after "Col").
- **wdFirstCharacterLineNumber** Returns the line number of the first character in the selection. If the **Pagination** property is **False** or the **Draft** property is **True**, returns – 1.
- **wdFrameIsSelected** Returns **True** if the selection or range is an entire frame or text box.
- **wdHeaderFooterType** Returns a value that indicates the type of header or footer that contains the specified selection or range, as shown in the following table.

Value	Type of header or footer
– 1	None (the selection or range isn't in a header or footer)
0 (zero)	Even page header
1	Odd page header (or the only header, if there aren't odd and even headers)
2	Even page footer
3	Odd page footer (or the only footer, if there aren't odd and even footers)
4	First page header
5	First page footer

- **wdHorizontalPositionRelativeToPage** Returns the horizontal position of the specified selection or range; this is the distance from the left edge of the selection or range to the left edge of the page, in twips (20 twips = 1 point, 72 points = 1 inch). If the selection or range isn't within the screen area, returns – 1.

- **wdHorizontalPositionRelativeToTextBoundary** Returns the horizontal position of the specified selection or range, relative to the left edge of the nearest text boundary enclosing it, in twips (20 twips = 1 point, 72 points = 1 inch). If the selection or range isn't within the screen area, returns – 1.
- **wdInClipboard** Returns **True** if the specified selection or range is on the Macintosh Clipboard.
- **wdInCommentPane** Returns **True** if the specified selection or range is in a comment pane.
- **wdInEndnote** Returns **True** if the specified selection or range is in an endnote area in page layout view or in the endnote pane in normal view.
- **wdInFootnote** Returns **True** if the specified selection or range is in a footnote area in page layout view or in the footnote pane in normal view.
- **wdInFootnoteEndnotePane** Returns **True** if the specified selection or range is in the footnote or endnote pane in normal view or in a footnote or endnote area in page layout view. For more information, see the descriptions of **wdInFootnote** and **wdInEndnote** in the preceding paragraphs.
- **wdInHeaderFooter** Returns **True** if the selection or range is in the header or footer pane or in a header or footer in page layout view.
- **wdInMasterDocument** Returns **True** if the selection or range is in a master document (that is, a document that contains at least one subdocument).
- **wdInWordMail** Returns a value that indicates the WordMail location of the selection or range, as shown in the following table.

Value	WordMail Location
0(zero)	The selection or range isn't in a WordMail message.
1	The selection or range is in a WordMail send note.
2	The selection or range is in a WordMail read note.

- **wdMaximumNumberOfColumns** Returns the greatest number of table columns within any row in the selection or range.
- **wdMaximumNumberOfRows** Returns the greatest number of table rows within the table in the specified selection or range.
- **wdNumberOfPagesInDocument** Returns the number of pages in the document associated with the selection or range.
- **wdNumLock** Returns **True** if Num Lock is in effect.
- **wdOverType** Returns **True** if overtype mode is in effect. The **Overtype** property can be used to change the state of overtype mode.
- **wdReferenceOfType** Returns a value that indicates where the selection is in relation to a footnote, endnote, or comment reference, as shown in the following table.

Value	Description
– 1	The selection or range includes but isn't limited to a footnote, endnote, or comment reference.
0 (zero)	The selection or range isn't before a footnote, endnote, or comment reference.
1	The selection or range is before a footnote reference.
2	The selection or range is before an endnote reference.
3	The selection or range is before a comment reference.

- **wdRevisionMarking** Returns **True** if change tracking is in effect.
- **wdSelectionMode** Returns a value that indicates the current selection mode, as shown in the following table.

Value	Selection mode

0 (zero)	Normal selection
1	Extended selection ("EXT" appears on the status bar)
2	Column selection. ("COL" appears on the status bar)

- **wdStartOfRangeColumnNumber** Returns the table column number that contains the beginning of the selection or range.
- **wdStartOfRangeRowNumber** Returns the table row number that contains the beginning of the selection or range.
- **wdVerticalPositionRelativeToPage** Returns the vertical position of the selection or range; this is the distance from the top edge of the selection to the top edge of the page, in twips (20 twips = 1 point, 72 points = 1 inch). If the selection isn't visible in the document window, returns – 1.
- **wdVerticalPositionRelativeToTextBoundary** Returns the vertical position of the selection or range, relative to the top edge of the nearest text boundary enclosing it, in twips (20 twips = 1 point, 72 points = 1 inch). This is useful for determining the position of the insertion point within a frame or table cell. If the selection isn't visible, returns – 1.
- **wdWithInTable** Returns **True** if the selection is in a table.
- **wdZoomPercentage** Returns the current percentage of magnification as set by the **Percentage** property.

Information Property Example

This example displays the current page number and the total number of pages in the active document.

```
MsgBox "The selection is on page " & _
    Selection.Information(wdActiveEndPageNumber) & " of page " _
    & Selection.Information(wdNumberOfPagesInDocument)
```

If the selection is in a table, this example selects the table.

```
If Selection.Information(wdWithInTable) Then Selection.Tables(1).Select
```

This example displays a message that indicates the current section number.

```
Selection.Collapse Direction:=wdCollapseStart
MsgBox "The insertion point is in section " & _
    Selection.Information(wdActiveEndSectionNumber)
```

TypeText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthTypeTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthTypeTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthTypeTextA "}

Inserts the specified text. If the **ReplaceSelection** property is **True**, the selection is replaced by the specified text. If **ReplaceSelection** is **False**, the specified text is inserted before the selection.

Syntax

expression.**TypeText(*Text*)**

expression Required. An expression that returns a **Selection** object.

Text Required **String**. The text to be inserted.

TypeText Method Example

If **Typing replaces selection** is selected on the **Edit** tab in the **Options** dialog box, this example collapses the selection before inserting "Hello." This technique prevents existing document text from being replaced.

```
If Options.ReplaceSelection = True Then
    Selection.Collapse Direction:=wdCollapseStart
    Selection.TypeText Text:="Hello"
End If
```

This example inserts "Title" followed by a new paragraph.

```
Options.ReplaceSelection = False
With Selection
    .TypeText Text:="Title"
    .TypeParagraph
End With
```

MoveUp Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMoveUpC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMoveUpX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMoveUpA "}

Moves the selection up and returns the number of units it's been moved.

Note The **wdWindow** constant can be used to move to the top or bottom of the active window. Regardless of the value of ***Count*** (greater than 1 or less than – 1), the **wdWindow** constant moves only one unit. Use the **wdScreen** constant to move more than one screen.

Syntax

expression.**MoveUp(*Unit, Count, Extend*)**

expression Required. An expression that returns a **Selection** object.

Unit Optional **Variant**. The unit by which to move the selection. Can be one of the following **WdUnits** constants: **wdLine**, **wdParagraph**, **wdWindow** or **wdScreen**. The default value is **wdLine**.

Count Optional **Variant**. The number of units the selection is to be moved. The default value is 1.

Extend Optional **Variant**. Can be either **wdMove** or **wdExtend**. If **wdMove** is used, the selection is collapsed to the end point and moved up. If **wdExtend** is used, the selection is extended up. The default value is **wdMove**.

MoveUp Method Example

This example moves the selection to the beginning of the previous paragraph.

```
Selection.MoveRight
Selection.MoveUp Unit:=wdParagraph, Count:=2, Extend:=wdMove
```

This example displays the current line number, moves the selection up three lines, and displays the current line number again.

```
MsgBox "Line " & Selection.Information(wdFirstCharacterLineNumber)
Selection.MoveUp Unit:=wdLine, Count:=3, Extend:=wdMove
MsgBox "Line " & Selection.Information(wdFirstCharacterLineNumber)
```

AutomaticallyUpdate Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutomaticallyUpdateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutomaticallyUpdateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutomaticallyUpdateA "}

True if the style is automatically redefined based on the selection. **False** if Word prompts for confirmation before redefining the style based on the selection. A style can be redefined when it's applied to a selection that has the same style but different manual formatting. Read/write **Boolean**.

AutomaticallyUpdate Property Example

This example creates a style named "myStyle" that can be redefined without the need for confirmation.

```
Set myDoc = Documents.Add
Set myStyle = myDoc.Styles.Add("myStyle")
With myStyle
    .BaseStyle = myDoc.Styles(wdStyleNormal)
    .ParagraphFormat.LineSpacingRule = wdLineSpaceDouble
    .AutomaticallyUpdate = True
End With
```

BaseStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBaseStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBaseStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBaseStyleA "}

Returns or sets an existing style on which you can base the formatting of another style. To set this property, specify either the local name of the base style, an integer or a **WdBuiltinStyle** constant, or an object that represents the base style. Read/write **Variant**.

For a list of the **WdBuiltinStyle** constants, see the **Style** property.

BaseStyle Property Example

This example creates a new document and then adds a new paragraph style named "myHeading." It assigns Heading 1 as the base style for the new style . A left indent of 1 inch (72 points) is then specified for the new style.

```
Set myDoc = Documents.Add
Set myHeading = myDoc.Styles.Add("myHeading")
With myHeading
    .BaseStyle = myDoc.Styles(wdStyleHeading1)
    .ParagraphFormat.LeftIndent = 72
End With
```

This example returns the base style that's used for the Body Text paragraph style.

```
base = ActiveDocument.Styles(wdStyleBodyText).BaseStyle
MsgBox base
```

BuiltIn Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBuiltInC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBuiltInX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBuiltInA "}

True if the specified object is one of the built-in styles or caption labels in Word. Read-only **Boolean**.

Remarks

You can specify built-in styles across all languages by using the **WdBuiltinStyle** constants or within a language by using the style name for the language version of Word. For example, in the English (U.S.) version of Word, the following statements are equivalent:

```
ActiveDocument.Styles(wdStyleHeading1)
ActiveDocument.Styles("Heading 1")
```

BuiltIn Property Example

This example checks all the styles in the active document. When it finds a style that isn't built in, it displays the name of the style.

```
For Each sty in ActiveDocument.Styles
    If sty.BuiltIn = False Then
        Msgbox sty.NameLocal
    End If
Next
```

This example checks all the caption labels that have been created in the application. When it finds a caption label that isn't built in, it displays the name of the label.

```
For Each lb in CaptionLabels
    If lb.BuiltIn = False Then
        Msgbox lb.Name
    End If
Next
```

Description Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDescriptionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDescriptionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDescriptionA "}

Returns the description of the specified style. For example, a typical description for the Heading 2 style might be "Normal + Font: Arial, 12 pt, Bold, Italic, Space Before 12 pt After 3 pt, KeepWithNext, Level 2." Read-only **String**.

Description Property Example

This example creates a new document and inserts a tab-delimited list of the active document's styles and their descriptions.

```
Set myDoc = ActiveDocument
Set newDoc = Documents.Add
For Each sty In myDoc.Styles
    With newDoc.Range
        .InsertAfter Text:=sty.NameLocal & Chr(9) & sty.Description
        .InsertParagraphAfter
    End With
Next sty
```

NextParagraphStyle Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNextParagraphStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNextParagraphStyleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNextParagraphStyleA "}

Returns or sets the style to be applied automatically to a new paragraph inserted after a paragraph formatted with the specified style. To set this property, specify either the local name of the next style, an integer or a **WdBuiltinStyle** constant, or an object that represents the next style. Read/write **Variant**.

For a list of the **WdBuiltinStyle** constants, see the **Style** property.

NextParagraphStyle Property Example

This example sets the Heading 1 style to be followed by the Heading 2 style in the active document.

```
ActiveDocument.Styles(wdStyleHeading1).NextParagraphStyle = _
    ActiveDocument.Styles(wdStyleHeading2)
```

This example creates a new document and adds a paragraph style named "MyStyle." The new style is based on the Normal style, is followed by the Heading 3 style, has a left indent of 1 inch (72 points), and is bold.

```
Set myDoc = Documents.Add
Set myStyle = myDoc.Styles.Add(Name:= "MyStyle")
    With myStyle
        .BaseStyle = wdStyleNormal
        .NextParagraphStyle = wdStyleHeading3
        .ParagraphFormat.LeftIndent = 72
        .Font.Bold = True
    End With
```

StyleName Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStyleNameC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStyleNameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStyleNameA "}

Returns the name of the style applied to the specified AutoText entry. Read-only **String**.

StyleName Property Example

This example creates an AutoText entry and then displays the style name of the entry.

```
Set myentry = NormalTemplate.AutoTextEntries.Add(Name:="rsvp", _
    Range:=Selection.Range)
MsgBox myentry.StyleName
```

Styles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStylesA "}

Returns a **Styles** collection for the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Styles Property Example

This example applies the Heading 1 style to each paragraph in the active document that begins with the word "Chapter."

```
For Each para In ActiveDocument.Paragraphs
    If para.Range.Words(1).Text = "Chapter " Then
        para.Style = ActiveDocument.Styles(wdStyleHeading1)
    End If
Next para
```

This example opens the template attached to the active document and modifies the Heading 1 style. The template is saved, and all styles in the active document are updated.

```
Set tempDoc = ActiveDocument.AttachedTemplate.OpenAsDocument
With tempDoc.Styles(wdStyleHeading1).Font
    .Name = "Arial"
    .Size = 16
End With
tempDoc.Close SaveChanges:=wdSaveChanges
ActiveDocument.UpdateStyles
```

AddFromFile Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFromFileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFromFileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFromFileA "}

Adds the specified subdocument to the master document at the start of the selection and returns a **Subdocument** object.

Note If the active view isn't either outline view or master document view, an error occurs.

Syntax

expression.**AddFromFile(*Name, ConfirmConversions, ReadOnly, PasswordDocument, PasswordTemplate, Revert, WritePasswordDocument, WritePasswordTemplate*)**

expression Required. An expression that returns a **Subdocuments** object.

Name Required **String**. The file name of the subdocument to be inserted into the master document.

ConfirmConversions Optional **Variant**. **True** to confirm file conversion in the **Convert File** dialog box if the file isn't in Word format.

ReadOnly Optional **Variant**. **True** to insert the subdocument as a read-only document.

PasswordDocument Optional **Variant**. The password required to open the subdocument if it's password protected.

PasswordTemplate Optional **Variant**. The password required to open the template attached to the subdocument if the template is password protected.

Revert Optional **Variant**. Controls what happens if ***Name*** is the file name of an open document. **True** to insert the saved version of the subdocument. **False** to insert the open version of the subdocument, which may contain unsaved changes.

WritePasswordDocument Optional **Variant**. The password required to save changes to the document file if it's write protected.

WritePasswordTemplate Optional **Variant**. The password required to save changes to the template attached to the subdocument if the template is write protected.

AddFromFile Method Example

This example adds a subdocument named "Subdoc.doc" to the active document.

```
ActiveWindow.View.Type = wdMasterView
ActiveDocument.Subdocuments.AddFromFile Name:="C:\MyFolder\Subdoc.doc"
```

This example adds a password-protected subdocument named "MySubdoc.doc" to the active document on a read-only basis.

```
Selection.Range.Subdocuments.AddFromFile Name:="MySubdoc.doc", _
    ReadOnly:=True, PasswordDocument:="MyPassword"
```

AddFromRange Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAddFromRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAddFromRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAddFromRangeA "}

Creates one or more subdocuments from the text in the specified range and returns a **SubDocument** object.

Note The range must begin with one of the built-in heading level styles (for example, Heading 1). Subdocuments are created at each paragraph formatted with the same heading format used at the beginning of the range. Subdocument files are saved when the master document is saved and are automatically named using text from the first line in the file.

Syntax

expression.**AddFromRange(*Range*)**

expression Required. An expression that returns a **Subdocuments** object.

Range Required **Range** object. The **Range** object used to create one or more subdocuments.

AddFromRange Method Example

This example creates one or more subdocuments from the selection.

```
ActiveWindow.View.Type = wdMasterView
ActiveDocument.SubDocuments.AddFromRange Range:=Selection.Range
```

Expanded Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExpandedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExpandedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExpandedA "}

True if the subdocuments in the specified document are expanded. Read/write **Boolean**.

Expanded Property Example

This example expands all subdocuments in the active master document.

```
If ActiveDocument.Subdocuments.Count >= 1 Then
    ActiveDocument.Subdocuments.Expanded = True
End If
```

This example toggles the **Expanded** property between expanding and collapsing all subdocuments in the active document.

```
ActiveDocument.Subdocuments.Expanded = Not
ActiveDocument.Subdocuments.Expanded
```

This example determines whether the subdocuments in Report.doc are expanded and then displays a message indicating their status.

```
If Documents("Report.doc").Subdocuments.Expanded = True Then
    MsgBox "All available information is displayed."
Else
    MsgBox "Expand subdocuments for more information."
End If
```

HasFile Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHasFileC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHasFileX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHasFileA "}

True if the specified subdocument has been saved to a file. Read-only **Boolean**.

HasFile Property Example

This example displays the file name of each subdocument in the active document. The example also displays a message for each subdocument that hasn't been saved.

```
For Each subdoc In ActiveDocument.Subdocuments
    subdoc.Range.Select
    If subdoc.HasFile = True Then
        MsgBox subdoc.Path & Application.PathSeparator & subdoc.Name
    Else
        MsgBox "This subdocument has not been saved."
    End If
Next subdoc
```

IsMasterDocument Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsMasterDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsMasterDocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsMasterDocumentA "}

True if the specified document is a master document. A master document includes one or more subdocuments. Read-only **Boolean**.

IsMasterDocument Property Example

If the active document is a master document, this example switches to master document view and opens the first subdocument.

```
If ActiveDocument.IsMasterDocument = True Then
    ActiveWindow.View.Type = wdMasterView
    ActiveDocument.Subdocuments(1).Open
Else
    MsgBox "This document is not a master document."
End If
```

IsSubdocument Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsSubdocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsSubdocumentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsSubdocumentA "}

True if the specified document is opened in a separate document window as a subdocument of a master document. Read-only **Boolean**

IsSubdocument Property Example

This example determines whether Sales.doc is a subdocument and then displays a message indicating it's status.

```
If Documents("Sales.doc").IsSubdocument = True Then
    MsgBox "Sales.doc is a subdocument."
Else
    MsgBox "Sales.doc is not a subdocument."
End If
```

Locked Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLockedC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLockedX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLockedA "}

Subdocument object: **True** if a subdocument in a master document is locked. Read/write **Boolean**.

LinkFormat object: **True** if a **Field**, **InlineShape**, or **Shape** object is locked to prevent automatic updating. If you use this property with a **Shape** object that's a floating linked picture (a picture added with the **AddPicture** method of the **Shapes** object), an error occurs. Read/write **Boolean**.

Field or **MailMergeField** object: **True** if the specified field is locked. When a field is locked, you cannot update the field results. Read/write **Boolean**.

Fields object: **True** if all fields in the **Fields** collection are locked. Can be **True**, **False**, or **wdUndefined** (if some of the fields in the collection are locked). Read/write **Long**.

Locked Property Example

This example checks the first subdocument in the specified master document and sets the master document to allow only comments if the subdocument is locked.

```
If ActiveDocument.Subdocuments(1).Locked = True Then
    ActiveDocument.Protect Type:=wdAllowOnlyComments
End If
```

This example inserts a DATE field at the beginning of the selection and then locks the field.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldDate)
myField.Locked = True
```

This example locks all the fields in the selection.

```
Selection.Fields.Locked = True
```

This example displays a message if some of the fields in the active document are locked.

```
Set theFields = ActiveDocument.Fields
If theFields.Locked = wdUndefined Then
    MsgBox "Some fields are locked"
ElseIf theFields.Locked = False Then
    MsgBox "No fields are locked"
ElseIf theFields.Locked = True Then
    MsgBox "All fields are locked"
End If
```

Subdocuments Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSubdocumentsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSubdocumentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSubdocumentsA "}

Returns a **Subdocuments** collection that represents all the subdocuments in the specified range or document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Subdocuments Property Example

This example displays the number of subdocuments embedded in the active document.

```
MsgBox ActiveDocument.Subdocuments.Count
```

This example displays the path and file name of each subdocument in the active document.

```
For Each subdoc In ActiveDocument.Subdocuments
    If subdoc.HasFile = True Then
        MsgBox subdoc.Path & Application.PathSeparator & subdoc.Name
    Else
        MsgBox "This subdocument has not been saved."
    End If
Next subdoc
```

AntonymList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAntonymListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAntonymListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAntonymListA "}

Returns a list of antonyms for the word or phrase. The list is returned as an array of strings. Read-only **Variant**.

Remarks

The **AntonymList** property is a property of the **SynonymInfo** object, which can be returned from either a range or the application. If this object is returned from the application, you specify the word to look up and the language to use. When the object is returned from a range, the range is looked up using the language of the range.

AntonymList Property Example

This example returns a list of antonyms for the word "big" in U.S. English.

```
Alist = SynonymInfo(Word:="big", LanguageID:=wdEnglishUS).AntonymList
For i = 1 To UBound(Alist)
    Msgbox Alist(i)
Next i
```

This example returns a list of antonyms for the word or phrase in the selection and displays them in the **Immediate** window in the Visual Basic Editor.

```
antList = Selection.Range.SynonymInfo.AntonymList
If UBound(antList) <> 0 Then
    For i = 1 To UBound(antList)
        Debug.Print antList(i) & Str(i)
    Next i
Else
    Msgbox "No antonyms were found."
End If
```

This example returns a list of antonyms for the third word in the active document, if there are any.

```
Set myRange = ActiveDocument.Words(3)
myalist = myRange.SynonymInfo.AntonymList
If Ubound(myalist) = 0 Then
    MsgBox "There are no anytonyms for the third word."
Else
    For i = 1 To UBound(myalist)
        Msgbox myalist(i)
    Next I
End If
```

Found Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFoundC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFoundX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFoundA "}

SynonymInfo object: **True** if the thesaurus finds synonyms, antonyms, related words, or related expressions for the word or phrase. Read-only **Boolean**.

Find object: **True** if the search produces a match. Read-only **Boolean**.

Found Property Example

This example checks to see whether the thesaurus contains any synonym suggestions for the word "authorize."

```
Set myword = SynonymInfo(Word:="authorize", LanguageID:=wdEnglishUS)
If myword.Found = True Then
    Msgbox "The thesaurus has suggestions."
Else
    Msgbox "The thesaurus has no suggestions."
End If
```

This example checks to see whether the thesaurus contains any synonym suggestions for the selection. If it does, the example displays the **Thesarus** dialog box with the synonyms listed.

```
Set mySyninfo = Selection.Range.SynonymInfo
If mySyninfo.Found = True Then
    Selection.Range.CheckSynonyms
Else
    Msgbox "The thesaurus has no suggestions."
End If
```

This example removes formatting from the find criteria before searching the selection. If the word "Hello" with bold formatting is found, the entire paragraph is selected and copied to the Clipboard.

```
With Selection.Find
    .ClearFormatting
    .Font.Bold = True
    .Execute FindText:="Hello", Format:=True, Forward:=True
    If .Found = True Then
        .Parent.Expand Unit:=wdParagraph
        .Parent.Copy
    End If
End With
```

MeaningCount Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMeaningCountC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMeaningCountX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMeaningCountA "}

Returns the number of entries in the list of meanings found in the thesaurus for the word or phrase. Returns 0 (zero) if no meanings were found. Read-only **Long**.

Remarks

Each meaning represents a unique list of synonyms for the word or phrase.

The lists of related words, related expressions, and antonyms aren't counted as entries in the list of meanings.

MeaningCount Property Example

This example checks to see whether any meanings were found for the selection. If any were found, the list of meanings is displayed in the **Immediate** window in the Visual Basic Editor.

```
Set mySynInfo = Selection.Range.SynonymInfo
If mySynInfo.MeaningCount <> 0 Then
    myList = mySynInfo.MeaningList
    For i = 1 To Ubound(myList)
        Debug.Print myList(i)
    Next i
Else
    Msgbox "There were no meanings found."
End If
```

MeaningList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMeaningListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMeaningListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMeaningListA "}

Returns the list of meanings for the word or phrase. The list is returned as an array of strings. Read-only **Variant**.

Remarks

The lists of related words, related expressions, and antonyms aren't counted as entries in the list of meanings.

MeaningList Property Example

This example checks to see whether any meanings were found for the third word in MyDoc.doc. If so, the meanings are displayed in a series of message boxes.

```
Set mySyn = Documents("MyDoc.doc").Words(3).SynonymInfo
If mySyn.MeaningCount <> 0 Then
    myList = mySyn.MeaningList
    For i = 1 To UBound(myList)
        Msgbox myList(i)
    Next i
Else
    Msgbox "There were no meanings found."
End If
```

PartOfSpeechList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPartOfSpeechListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPartOfSpeechListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPartOfSpeechListA "}

Returns a list of the parts of speech corresponding to the meanings found for the word or phrase looked up in the thesaurus. The list is returned as an array of integers. Read-only **Variant**.

Remarks

The list of the parts of speech is returned as an array consisting of the following **WdPartOfSpeech** constants: **wdAdjective**, **wdAdverb**, **wdNoun**, and **wdVerb**. The array elements are ordered to correspond to the elements returned by the **MeaningList** property.

PartOfSpeechList Property Example

This example checks to see whether the thesaurus found any meanings for the selection. If so, the meanings and their corresponding parts of speech are displayed in a series of message boxes.

```
Set mySynInfo = Selection.Range.SynonymInfo
If mySynInfo.MeaningCount <> 0 Then
    myList = mySynInfo.MeaningList
    myPos = mySynInfo.PartOfSpeechList
    For i = 1 To UBound(myPos)
        Select Case myPos(i)
            Case wdAdjective
                 pos = "adjective"
            Case wdNoun
                 pos = "noun"
            Case wdAdverb
                 pos = "adverb"
            Case wdVerb
                 pos = "verb"
        End Select
        MsgBox myList(i) & " found as " & pos
    Next i
Else
    MsgBox "There were no meanings found."
End If
```

RelatedExpressionList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRelatedExpressionListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRelatedExpressionListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRelatedExpressionListA "}

Returns a list of expressions related to the specified word or phrase. The list is returned as an array of strings. Read-only **Variant**.

Remarks

Typically, there are very few related expressions found in the thesaurus.

RelatedExpressionList Property Example

This example checks to see whether any related expressions were found for the selection. If so, the meanings are displayed in a series of message boxes. If none were found, this is stated in a message box.

```
Set mySinfo = Selection.Range.SynonymInfo
If mySinfo.Found = True Then
    myRelList = mysInfo.RelatedExpressionList
    If UBound(myRelList) <> 0 Then
        For i = 1 To UBound(myRelList)
            Msgbox myRelList(i)
        Next i
    Else
        Msgbox "There were no related expressions found."
    End If
End If
```

RelatedWordList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRelatedWordListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRelatedWordListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRelatedWordListA "}

Returns a list of words related to the specified word or phrase. The list is returned as an array of strings. Read-only **Variant**.

RelatedWordList Property Example

This example checks to see whether any related words were found for the third word in the active document. If so, the meanings are displayed in a series of message boxes. If there are no related words found, this is stated in a message box.

```
Set mySynon = ActiveDocument.Words(3).SynonymInfo
If mySynon.Found = True Then
    myWdList = mySynon.RelatedWordList
    If UBound(myWdList) <> 0 Then
        For i = 1 To UBound(myWdList)
            Msgbox myWdList(i)
        Next i
    Else
        Msgbox "There were no related words found."
    End If
End If
```

SynonymInfo Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSynonymInfoC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSynonymInfoX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSynonymInfoA "}

Returns a **SynonymInfo** object that contains information from the thesaurus on synonyms, antonyms, or related words and expressions for the specified word or phrase. Read-only.

Syntax 1

expression.**SynonymInfo**

Syntax 2

expression.**SynonymInfo**(***Word, LanguageID***)

expression Syntax 1: Required. An expression that returns a **Range** object.
Syntax 2: Optional. An expression that returns an **Application** object.

Word Required **String**. The word or phrase to look up in the thesaurus.

LanguageID Optional **Variant**. The language used for the thesaurus. Can be one of the following **WdLanguageID** constants:

wdAfrikaans	**wdLanguageNone**
wdArabic	**wdLatvian**
wdBasque	**wdMacedonian**
wdBelgianDutch	**wdMalaysian**
wdBelgianFrench	**wdMexicanSpanish**
wdBrazilianPortuguese	**wdNoProofing**
wdBulgarian	**wdNorwegianBokmol**
wdByelorussian	**wdNorwegianNynorsk**
wdCatalan	**wdPolish**
wdCroation	**wdPortuguese**
wdCzech	**wdRomanian**
wdDanish	**wdRussian**
wdDutch	**wdSerbianCyrillic**
wdEnglishAUS	**wdSerbianLatin**
wdEnglishCanadian	**wdSesotho**
wdEnglishNewZealand	**wdSimplifiedChinese**
wdEnglishSouthAfrica	**wdSlovak**
wdEnglishUK	**wdSlovenian**
wdEnglishUS	**wdSpanish**
wdEstonian	**wdSpanishModernSort**
wdFarsi	**wdSwedish**
wdFinnish	**wdSwissFrench**
wdFrench	**wdSwissGerman**
wdFrenchCanadian	**wdSwissItalian**
wdGerman	**wdTraditionalChinese**
wdGreek	**wdTsonga**
wdHebrew	**wdTswana**
wdHungarian	**wdTurkish**
wdItalian	**wdUkrainian**

wdIcelandic
wdJapanese
wdKorean
wdVenda
wdXhosa
wdZulu

SynonymInfo Property Example

This example returns a list of antonyms for the word "big" in U.S. English.

```
Alist = SynonymInfo(Word:="big", LanguageID:=wdEnglishUS).AntonymList
For i = 1 To UBound(Alist)
    Msgbox Alist(i)
Next i
```

This example returns a list of synonyms for the selection's first meaning.

```
Slist = Selection.Range.SynonymInfo.SynonymList(Meaning:=1)
For i = 1 To UBound(Slist)
    Msgbox Slist(i)
Next i
```

SynonymList Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSynonymListC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSynonymListX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSynonymListA "}

Returns a list of synonyms for a specified meaning of a word or phrase. The list is returned as an array of strings. Read-only **Variant**.

Syntax

expression.**SynonymList(*Meaning*)**

expression Required. An expression that returns a **SynonymInfo** object.

Meaning Required **Variant**. The meaning as a string or an index number in the array of possible meanings.

SynonymList Property Example

This example returns a list of synonyms for the word "big," using the meaning "generous" in U.S. English.

```
Slist = SynonymInfo(Word:="big", LanguageID:=wdEnglishUS) _
    .SynonymList(Meaning:="generous")
For i = 1 To UBound(Slist)
    Msgbox Slist(i)
Next i
```

This example returns a list of synonyms for the second meaning of the selected word or phrase and displays these synonyms in the **Immediate** window in the Visual Basic editor. If there's no second meaning or if there are no synonyms, this is stated in a message box.

```
Set mySi = Selection.Range.SynonymInfo
If mySi.MeaningCount >= 2 Then
    synList = mySi.SynonymList(Meaning:=2)
        For i = 1 To UBound(synList)
            Debug.Print synList(i)
        Next i
Else
    MsgBox "There is no second meaning for this word or phrase."
End If
```

Word Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWordC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWordX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWordA "}

Returns the word or phrase that was looked up by the thesaurus. Read-only **String**.

Remarks

The thesaurus will sometimes look up a shortened version of the string or range used to return the **SynonymInfo** object. The **Word** property allows you to see the exact string that was used.

Word Property Example

This example returns a list of synonyms for the first meaning of the third word in the active document.

```
Set mySynObj = ActiveDocument.Words(3).SynonymInfo
SList = mySynObj.SynonymList(1)
For i = 1 To UBound(SList)
    Msgbox "A synonym for " & mySynObj.Word & " is " & SList(i)
Next i
```

This example checks to make sure that the word or phrase that was looked up isn't empty. If it's not, the example returns a list of synonyms for the first meaning of the word or phrase.

```
Set mySynObj = Selection.Range.SynonymInfo
If mySynObj.Word = "" Then
    Msgbox "Please select a word or phrase"
Else
    SList = mySynObj.SynonymList(1)
    For i = 1 To UBound(SList)
        Msgbox "A synonym for " & mySynObj.Word & " is " & SList(i)
    Next i
End If
```

ActiveThesaurusDictionary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproActiveThesaurusDictionaryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproActiveThesaurusDictionaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproActiveThesaurusDictionaryA "}

Returns a **Dictionary** object that represents the active thesaurus dictionary for the specified language. Read-only.

Remarks

If there's no thesaurus dictionary installed for the specified language, this property returns **Nothing**.

ActiveThesaurusDictionary Property Example

This example returns the full path and file name of the active thesaurus dictionary.

```
myLang = Selection.LanguageID
Set myThes = Languages(myLang).ActiveThesaurusDictionary
MsgBox myThes.Path & Application.PathSeparator & myThes.Name
```

Country Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCountryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCountryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCountryA "}

Returns the country designation of the system. Read-only **Long**.

Can be one of the following **WdCountry** constants:

wdArgentina	**wdKorea**
wdBrazil	**wdLatinAmerica**
wdCanada	**wdMexico**
wdChile	**wdNetherlands**
wdChina	**wdNorway**
wdDenmark	**wdPeru**
wdFinland	**wdSpain**
wdFrance	**wdSweden**
wdGermany	**wdTaiwan**
wdIceland	**wdUK**
wdItaly	**wdUS**
wdJapan	**wdVenezuela**

Country Property Example

If the **Country** property returns **wdUS**, this example converts the top margin value from points to inches.

```
If System.Country = wdUS Then
    topMgn = ActiveDocument.PageSetup.TopMargin
    MsgBox "Top margin is " & PointsToInches(topMgn)
End If
```

Cursor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCursorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCursorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCursorA "}

Returns or sets the state (shape) of the pointer. Can be one of the following **WdCursorType** constants: **wdCursorIBeam**, **wdCursorNormal**, **wdCursorNorthwestArrow**, or **wdCursorWait**. Read/write **Long**.

Cursor Property Example

This example prints a message on the status bar and changes the pointer to a busy pointer.

```
StatusBar = "Please wait..."
For i = 1 To 1000
    System.Cursor = wdCursorWait
Next i
StatusBar = "Task completed"
System.Cursor = wdCursorNormal
```

FreeDiskSpace Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFreeDiskSpaceC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFreeDiskSpaceX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFreeDiskSpaceA "}

Returns the available disk space for the current drive, in bytes. Use the **ChDir** statement to change the current drive. Read-only **Long**.

Note There are 1024 bytes in a kilobyte and 1,048,576 bytes in a megabyte.

FreeDiskSpace Property Example

This example checks the amount of free disk space. If there's less than 10 megabytes of space available, a message is displayed.

```
If (System.FreeDiskSpace \ 1048576) < 10 Then MsgBox "Low disk space"
```

HorizontalResolution Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalResolutionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalResolutionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalResolutionA "}

Returns the horizontal display resolution, in pixels. Read-only **Long**.

HorizontalResolution Property Example

This example displays the current screen resolution (for example, "1024 x 768").

```
horz = System.HorizontalResolution
vert = System.VerticalResolution
MsgBox "Resolution = " & horz & " x " & vert
```

LanguageDesignation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLanguageDesignationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLanguageDesignationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLanguageDesignationA "}

Returns the designated language of the system software. Read-only **String**.

LanguageDesignation Property Example

This example displays "U.S. English" if the **LanguageDesignation** property returns "enu".

```
If System.LanguageDesignation = "enu" Then MsgBox "U.S. English"
```

MathCoprocessorInstalled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMathCoprocessorInstalledC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMathCoprocessorInstalledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMathCoprocessorInstalledA "}

True if a math coprocessor is installed on the system. Read-only **Boolean**.

MathCoprocessorInstalled Property Example

This example displays a message if a math coprocessor is installed on the system.

```
If System.MathCoprocessorInstalled = True Then
    MsgBox "A math coprocessor is installed."
End If
```

MSInfo Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMSInfoC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMSInfoX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMSInfoA "}

Starts the Microsoft System Information application if it's not running, or switches to it if it's already running.

Syntax

expression.**MSInfo**

expression Required. An expression that returns a **System** object.

MSInfo Method Example

This example starts or switches to the Microsoft System Information application.

```
System.MSInfo
```

ProcessorType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProcessorTypeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProcessorTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProcessorTypeA "}

Returns the type of processor that the system is using (for example, i486). Read-only **String**.

ProcessorType Property Example

This example displays a message on the status bar if the processor that the system is using isn't a Pentium processor.

```
If System.ProcessorType <> "Pentium" Then StatusBar = "Please wait..."
```

System Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSystemC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSystemX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSystemA "}

Returns the **System** object, which can be used to return system-related information and perform system-related tasks. Read-only.

System Property Example

This example returns information about the system.

```
processor = System.ProcessorType
enviro = System.OperatingSystem
```

This example establishes a connection to a network drive.

```
System.Connect Path:="\\Project\Info"
```

VerticalResolution Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalResolutionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalResolutionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalResolutionA "}

Returns the vertical screen resolution, in pixels. Read-only **Long**.

VerticalResolution Property Example

This example displays the current screen resolution (for example, "1024 x 768").

```
horz = System.HorizontalResolution
vert = System.VerticalResolution
MsgBox "Resolution = " & horz & " x " & vert
```

MathCoprocessorAvailable Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMathCoprocessorAvailableC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMathCoprocessorAvailableX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMathCoprocessorAvailableA "}

True if a math coprocessor is installed on the system. Read-only **Boolean**.

MathCoprocessorAvailable Property Example

This example displays a message indicating whether a math coprocessor is installed on the system.

```
If Application.MathCoprocessorAvailable = True Then
    MsgBox "A math coprocessor is installed."
Else
    MsgBox "A math coprocessor is not installed."
End If
```

Connect Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConnectC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConnectX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConnectA "}

In Windows, establishes a connection to a network drive. On the Macintosh, **Connect** isn't available and generates an error; use the **MountVolume** method instead.

Syntax

expression.**Connect(*Path, Drive, Password*)**

expression Required. An expression that returns a **System** object.

Path Required **String**. The path for the network drive (for example, "\\Project\Info").

Drive Optional **Variant**. A number corresponding to the letter you want to assign to the network drive, where 0 (zero) corresponds to the first available drive letter, 1 corresponds to the second available drive letter, and so on. If this argument is omitted, the next available letter is used.

Password Optional **Variant**. The password, if the network drive is protected with a password.

Remarks

Use the **Dialogs** property with the **wdDialogConnect** constant to display the **Connect To Network Drive** dialog box. The following example displays the **Connect To Network Drive** dialog box, with a preset path shown.

```
With Dialogs(wdDialogConnect)
    .Path = "\\Marketing\Public"
    .Show
End With
```

Connect Method Example

This example establishes a connection to a network drive (\\Project\Info) protected with the password "smiley" and assigns the network drive to the next available drive letter.

```
System.Connect Path:="\\Project\Info", Password:="smiley"
```

This example establishes a connection to a network drive (\\Team1\Public) and assigns the network drive to the third available drive letter.

```
System.Connect Path:="\\Team1\Public", Drive:=2
```

ProfileString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproProfileStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproProfileStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproProfileStringA "}

Returns or sets a setting in the Windows registry under the following key: HKEY_CURRENT_USER\Software\Microsoft\Office\8.0\Word. On the Macintosh, this property returns or sets a setting in the Word Settings (97) file. Read/write **String**.

Syntax

expression.**ProfileString(*Section, Key*)**

expression Required. An expression that returns a **System** object.

Section Required **String**. A key below the "HKEY_CURRENT_USER\Software\Microsoft\Office\8.0\Word" path in the Windows registry.

Key Required **String**. The value in the key specified by ***Section*** (for example, "BackgroundPrint" in the Options key.

ProfileString Property Example

This example retrieves and displays the startup path stored in the Windows registry.

```
MsgBox System.ProfileString("Options", "Startup-Path")
```

This example sets and returns a setting in the Windows registry (the Settings key is added below HKEY_CURRENT_USER\Software\Microsoft\Office\8.0\Word\Settings).

```
System.ProfileString("Settings", "Test") = "foo"
MsgBox System.ProfileString("Settings", "Test")
```

PrivateProfileString Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPrivateProfileStringC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPrivateProfileStringX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPrivateProfileStringA "}

Returns or sets a string in a settings file or the Windows registry. Read/write **String**.

Syntax

expression.**PrivateProfileString(*Filename, Section, Key*)**

expression Required. An expression that returns a **System** object.

Filename Required **String**. The file name for the settings file. If there's no path specified, the Windows folder (Windows) or the Preferences folder (Macintosh) is assumed. If you're using Windows 95 or Windows NT to return a setting from the registry, ***Filename*** must be an empty string ("").

Section Required **String**. The name of the section in the settings file that contains ***Key***. In a Windows settings file, the section name appears between brackets before the associated keys (don't include the brackets with ***Section***). If you're returning a setting from the Windows registry, ***Section*** should be the complete path to the key, including the root (for example, "HKEY_CURRENT_USER\Software\Microsoft\Office\8.0\Word\Options").

Key Required **String**. The key whose setting you want to retrieve. In a Windows settings file, the key name is followed by an equal sign (=) and the setting. If you're returning a setting from the Windows registry, ***Key*** should be the value in the key specified by ***Section*** (for example, "Tools-Path").

Remarks

You can write macros that use a settings file to store and retrieve settings. For example, you can store the name of the active document when you quit Word so that it can be reopened automatically the next time you start Word. In Windows, a settings file is a text file with information arranged like the information in the Windows 3.x WIN.INI file. On the Macintosh, a settings file is a resource file such as Word Settings (97).

PrivateProfileString Property Example

This example sets the current document name as the LastFile setting under the MacroSettings heading in Settings.txt.

```
System.PrivateProfileString("C:\Settings.txt", "MacroSettings", _
    "LastFile") = ActiveDocument.FullName
```

This example returns the LastFile setting from Settings.txt and then opens the document stored in LastFile.

```
LastFile = System.PrivateProfileString("C:\Settings.Txt", _
    "MacroSettings", "LastFile")
If LastFile <> "" Then Documents.Open FileName:=LastFile
```

This example displays the default user name setting from the Windows registry.

```
aName = System.PrivateProfileString("", _
    "HKEY_CURRENT_USER\Software\Microsoft\MS Setup (ACME)\User Info",
"DefName")
MsgBox aName
```

AllowBreakAcrossPages Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAllowBreakAcrossPagesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAllowBreakAcrossPagesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAllowBreakAcrossPagesA "}

True if the text in a table row or rows are allowed to split across a page break. Can be **True**, **False** or **wdUndefined**. Read/write **Long**.

AllowBreakAcrossPages Property Example

This example creates a new document with a 5x5 table and prevents the third row of the table from being split during pagination.

```
Set newDoc = Documents.Add
Set aTable = newDoc.Tables.Add(Range:=Selection.Range, _
    NumRows:=5, NumColumns:=5)
aTable.Rows(3).AllowBreakAcrossPages = False
```

This example determines whether the rows in the current table can be split across pages. If the insertion point isn't in a table, a message box is displayed.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Tables.Count = 0 Then
    MsgBox "The insertion point is not in a table."
Else
    allowBreak = Selection.Rows.AllowBreakAcrossPages
End If
```

AutoFit Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoFitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoFitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoFitA "}

Changes the width of a table column to accommodate the width of the text without changing the way text wraps in the cells.

Syntax

expression.**AutoFit**

expression Required. An expression that returns a **Column** or **Columns** object.

Remarks

If the table is already as wide as the distance between the left and right margins, this method has no affect.

AutoFit Method Example

This example creates a 3x3 table in a new document and then changes the width of the first column to accommodate the width of the text.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, _
    NumRows:=3, NumColumns:=3)With myTable
    .Cell(1,1).Range.InsertAfter "First cell"
    .Columns(1).AutoFit
End With
```

This example creates a 3x3 table in a new document and then changes the width of all the columns to accommodate the width of the text.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
With myTable
    .Cell(1,1).Range.InsertAfter "First cell"
    .Cell(1,2).Range.InsertAfter "This is cell (1,2)"
    .Cell(1,3).Range.InsertAfter "(1,3)"
    .Columns.AutoFit
End With
```

AutoSum Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoSumC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoSumX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoSumA "}

Inserts an = (Formula) field that calculates and displays the sum of the values in table cells above or to the left of the cell specified in the expression. For information about how Word determines which values to add, see the **Formula** method.

Syntax

expression.**AutoSum**

expression Required. An expression that returns a **Cell** object.

AutoSum Method Example

This example creates a 3x3 table in a new document and sums the numbers in the first column.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
With MyTable
    .Cell(1,1).Range.InsertAfter "10"
    .Cell(2,1).Range.InsertAfter "15"
    .Cell(3,1).AutoSum
End With
```

This example sums the numbers above or to the left of the cell that contains the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).AutoSum
Else
    MsgBox "The insertion point is not in a table."
End If
```

Cell Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCellC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCellX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCellA "}

Returns a **Cell** object that represents a cell in a table.

Syntax

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 cell in the table to return. Can be an integer between 1 and the number of columns in the table.

Cell Method Example

This example creates a 3x3 table in a new document and inserts text into the first and last cells in the table.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
With myTable
    .Cell(1,1).Range.InsertAfter "First cell"
    .Cell(myTable.Rows.Count, _
        myTable.Columns.Count).Range.InsertAfter "Last Cell"
End With
```

This example deletes the first cell from the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    ActiveDocument.Tables(1).Cell(1, 1).Delete
End If
```

Cells Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCellsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCellsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCellsA "}

Returns a **Cells** collection that represents the table cells in a column, row, selection, or range. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Cells Property Example

This example creates a 3x3 table and assigns a sequential cell number to each cell in the table.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
i = 1
For Each c In myTable.Range.Cells
    c.Range.InsertAfter "Cell " & i
    i = i + 1
Next c
```

This example sets the current cell's background color to red.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).Shading.BackgroundPatternColorIndex = wdRed
Else
    MsgBox "The insertion point is not in a table."
End If
```

Column Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColumnC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColumnX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColumnA "}

Bookmark object: **True** if the specified bookmark is a table column. Read-only **Boolean**.

Cell object: Returns a **Column** object that represents the table column containing the specified cell. Read-only.

Column Property Example

This example creates a table with a bookmark and then displays a message box that confirms that the bookmark is a table column.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 5)
Set myCellRange = myTable.Cell(3,5).Range
myCellRange.InsertAfter "Cell(3,5)"
newDoc.Bookmarks.Add Name:="BKMK_Cell35", Range:=myCellRange
MsgBox newDoc.Bookmarks(1).Column
```

This example creates a 3x5 table and applies shading to the even-numbered columns.

```
Selection.Collapse Direction:=wdCollapseStart
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=3, NumColumns:=5)
For Each aCell In myTable.Rows(1).Cells
    If aCell.ColumnIndex Mod 2 = 0 Then
        aCell.Column.Shading.Texture = wdTexture10Percent
    End If
Next aCell
```

ColumnIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColumnIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColumnIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColumnIndexA "}

Returns the number of the table column that contains the specified cell. Read-only **Long**.

ColumnIndex Property Example

This example creates a table in a new document, selects each cell in the first row, and returns the column number that contains the selected cell.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
For Each aCell In myTable.Rows(1).Cells
    aCell.Select
    MsgBox "This is column " & aCell.ColumnIndex
Next aCell
```

This example returns the column number of the cell that contains the insertion point.

```
Msgbox Selection.Cells(1).ColumnIndex
```

Columns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproColumnsA "}

Returns a **Columns** collection that represents all the table columns in the range, selection, or table. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Columns Property Example

This example displays the number of columns in the first table in the active document.

```
If ActiveDocument.Tables.Count >= 1 Then
    MsgBox ActiveDocument.Tables(1).Columns.Count
End If
```

This example sets the width of the current column to 1 inch.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Columns.SetWidth ColumnWidth:=InchesToPoints(1), _
        RulerStyle:=wdAdjustProportional
End If
```

Formula Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthFormulaC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthFormulaX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthFormulaA "}

Inserts an = (Formula) field that contains the specified formula into a table cell.

Syntax

expression.**Formula(*Formula, NumFormat*)**

expression Required. An expression that returns a **Cell** object.

Formula Optional **Variant**. The mathematical formula you want the = (Formula) field to evaluate. Spreadsheet-type references to table cells are valid. For example, "=SUM(A4:C4)" specifies the first three values in the fourth row. For more information about the = (Formula) field, see Field codes:= (Formula) field.

NumFormat Optional **Variant**. A format for the result of the = (Formula) field. For information about the types of formats you can apply, see Numeric Picture (\#) field switch.

Remarks

Formula is optional as long as there is at least one cell that contains a value above or to the left of the cell that contains the insertion point. If the cells above the insertion point contain values, the inserted field is {=SUM(ABOVE)}; if the cells to the left of the insertion point contain values, the inserted field is {=SUM(LEFT)}. If both the cells above the insertion point and the cells to the left of the insertion point contain values, Word uses the following rules to determine which SUM function to insert:

- If the cell immediately above the insertion point contains a value, Word inserts {=SUM(ABOVE)}.
- If the cell immediately above the insertion point doesn't contain a value and the cell immediately to the left of it does, Word inserts {=SUM(LEFT)}.
- If neither adjoining cell contains a value, Word inserts {=SUM(ABOVE)}.
- If you don't specify ***Formula*** and all the cells above and to the left of the insertion point are empty, the result of the field is an error.

Formula Method Example

This example creates a 3x3 table at the beginning of the active document and then averages the numbers in the first column.

```
Set myRange = ActiveDocument.Range(0, 0)
Set myTable = ActiveDocument.Tables.Add(myRange, 3, 3)
With myTable
    .Cell(1,1).Range.InsertAfter "100"
    .Cell(2,1).Range.InsertAfter "50"
    .Cell(3,1).Formula Formula:="=Average(Above)"
End With
```

This example inserts a formula at the insertion point that determines the largest number in the cells above the selected cell.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).Formula Formula:="=Max(Above)"
Else
   MsgBox "The insertion point is not in a table."
End If
```

HeadingFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeadingFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeadingFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeadingFormatA "}

True if the specified row or rows are formatted as a table heading. Rows formatted as table headings are repeated when a table spans more than one page. Can be **True**, **False** or **wdUndefined**. Read/write **Long**.

HeadingFormat Property Example

This example creates a 5x5 table at the beginning of the active document and then adds the table heading format to the first table row.

```
Set myRange = ActiveDocument.Range(0, 0)
Set myTable = ActiveDocument.Tables.Add(myRange, 5, 5)
myTable.Rows(1).HeadingFormat = True
```

This example determines whether the row that contains the insertion point is formatted as a table heading.

```
If Selection.Information(wdWithInTable) = True Then
    If Selection.Rows(1).HeadingFormat = True Then _
        MsgBox "The current row is a table heading"
Else
    MsgBox "The insertion point is not in a table."
End If
```

IsFirst Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsFirstC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsFirstX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsFirstA "}

True if the specified column or row is the first one in the table. Read-only **Boolean**.

IsFirst Property Example

This example determines whether the first row in the selection is the first row in the table.

```
MsgBox Selection.Rows(1).IsFirst
```

IsLast Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIsLastC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIsLastX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIsLastA "}

True if the specified column or row is the last one in the table. Read-only **Boolean**.

IsLast Property Example

This example determines whether the second row is the last row in the table.

```
MsgBox ActiveDocument.Tables(1).Rows(2).IsLast
```

This example determines whether the first column in the selection is the last column in the table.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox Selection.Columns(1).IsLast
End If
```

Row Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRowA "}

Returns a **Row** object that represents the row containing the specified cell. Read-only.

Row Property Example

This example applies shading to the table row that contains the insertion point.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).Row.Shading.Texture = wdTexture10Percent
Else
    MsgBox "The insertion point is not in a table."
End If
```

RowIndex Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRowIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRowIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRowIndexA "}

Returns the number of the row that contains the specified cell. Read-only **Long**.

RowIndex Property Example

This example creates a 3x3 table in a new document, selects each cell in the first column, and displays the row number that contains the selected cell.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, _
    NumRows:=3, NumColumns:=3)
For Each aCell In myTable.Columns(1).Cells
    aCell.Select
    MsgBox "This is row " & aCell.RowIndex
Next aCell
```

This example displays the row number of the first row in the selection.

```
If Selection.Information(wdWithInTable) = True Then
    Msgbox Selection.Cells(1).RowIndex
End If
```

Rows Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRowsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRowsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRowsA "}

Returns a **Rows** collection that represents all the table rows in a range, selection, or table. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Rows Property Example

This example deletes the second row from the first table in the active document.

```
ActiveDocument.Tables(1).Rows(2).Delete
```

This example places a border around the cells in the row that contains the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows(1).Borders.OutsideLineStyle =  wdLineStyleSingle
Else
    MsgBox "The insertion point is not in a table."
End If
```

Sort Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSortC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSortX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSortA "}

Syntax 1: Sorts the specified table column.

Syntax 2: Sorts the specified table.

Syntax 3: Sorts the paragraphs in the specified range or selection.

Syntax 1

expression.**Sort(*ExcludeHeader, SortFieldType, SortOrder, CaseSensitive, LanguageID*)**

Syntax 2

expression.**Sort(*ExcludeHeader, FieldNumber, SortFieldType, SortOrder, FieldNumber2, SortFieldType2, SortOrder2, FieldNumber3, SortFieldType3, SortOrder3, CaseSensitive, LanguageID*)**

Syntax 3

expression.**Sort(*ExcludeHeader, FieldNumber, SortFieldType, SortOrder, FieldNumber2, SortFieldType2, SortOrder2, FieldNumber3, SortFieldType3, SortOrder3, SortColumn, Separator, CaseSensitive, LanguageID*)**

expression Syntax 1: Required. An expression that returns a **Column** object.

Syntax 2: Required. An expression that returns a **Table** object.

Syntax 3: Required. An expression that returns a **Range** or **Selection** object.

ExcludeHeader Optional **Variant**. **True** to exclude the first row or paragraph from the sort operation. The default value is **False**.

FieldNumber, ***FieldNumber2***, ***FieldNumber3*** Optional **Variant**. The fields to sort by. Word sorts by ***FieldNumber***, then by ***FieldNumber2***, and then by ***FieldNumber3***.

SortFieldType, ***SortFieldType2***, ***SortFieldType3*** Optional **Variant**. The column sort type (Syntax 1) or the respective sort types for ***FieldNumber***, ***FieldNumber2***, and ***FieldNumber3*** (Syntax 2 and Syntax 3). Can be one of the following **WdSortFieldType** constants: **wdSortFieldAlphanumeric**, **wdSortFieldDate**, **or wdSortFieldNumeric**. The default value is **wdSortFieldAlphanumeric**.

SortOrder, ***SortOrder2***, ***SortOrder3*** Optional **Variant**. The column sorting order (Syntax 1) or the sorting order to use when sorting ***FieldNumber***, ***FieldNumber2***, and ***FieldNumber3*** (Syntax 2 and Syntax 3). Can be one of the following **WdSortOrder** constants: **wdSortOrderAscending** or **wdSortOrderDescending**. The default value is **wdSortAscending**.

SortColumn Optional **Variant**. **True** to sort only the column specified by the **Range** or **Selection** object.

Separator Optional **Variant**. The type of field separator. Can be one of the following **WdSortSeparator** constants: **wdSortSeparateByCommas**, **wdSortSeparateByDefaultTableSeparator**, or **wdSortSeparateByTabs**. The default value is **wdSortSeparateByCommas**. If the range or selection is in a table, this argument is ignored.

CaseSensitive Optional **Variant**. **True** to sort with case sensitivity. The default value is **False**.

LanguageID Optional **Variant**. Specifies the sorting language. Can be one of the **WdLanguageID** constants.

Remarks

If you want to sort paragraphs within a table cell, include only the paragraphs and not the end-of-cell mark; if you include the end-of-cell mark in a selection or range and then attempt to sort the paragraphs, Word displays a message stating that it found no valid records to sort.

Sort Method Example

This example creates a 5x5 table in a new document, inserts text into each cell, and then sorts the table in descending alphanumeric order.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 5, 5)
For iRow = 1 To myTable.Rows.Count
    For iCol = 1 To myTable.Columns.Count
       Set myRange= myTable.Rows(iRow).Cells(iCol).Range
       myRange.InsertAfter "Cell (" & Str$(iRow) & "," & Str$(iCol) & ")"
    Next iCol
Next iRow
MsgBox "Click OK to sort in descending order."
myTable.Sort SortOrder:=wdSortOrderDescending
```

This example inserts three lines of text into a new document and then sorts the lines in ascending alphanumeric order.

```
Set newDoc = Documents.Add
newDoc.Content.InsertAfter "pear" & Chr(13) & "zucchini" & Chr(13) _
    & "apple" & Chr(13)
newDoc.Content.Sort SortOrder:=wdSortOrderAscending
```

SpaceBetweenColumns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSpaceBetweenColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSpaceBetweenColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSpaceBetweenColumnsA "}

Returns or sets the distance (in points) between text in adjacent columns of the specified row or rows. Read/write **Single**.

SpaceBetweenColumns Property Example

This example creates a 3x3 table in a new document and then sets the distance between columns in the first row to 0.5 inches.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 3, 3)
myTable.Rows(1).SpaceBetweenColumns = InchesToPoints(0.5)
```

This example returns the distance (in points) between columns in the selected table rows.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox Selection.Rows.SpaceBetweenColumns
End If
```

Tables Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTablesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTablesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTablesA "}

Returns a **Tables** collection that represents all the tables in the specified document, range, or selection. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Tables Property Example

This example creates a 5x5 table in the active document and then applies a predefined format to it.

```
Selection.Collapse Direction:=wdCollapseStart
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
NumRows:=5, NumColumns:=5)
myTable.AutoFormat Format:=wdTableFormatClassic2
```

This example inserts numbers and text into the first column of the first table in the active document.

```
num = 90
For Each acell In ActiveDocument.Tables(1).Columns(1).Cells
    acell.Range.Text = num & " Sales"
    num = num + 1
Next acell
```

Uniform Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUniformC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUniformX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUniformA "}

True if all the rows in a table have the same number of columns. Read-only **Boolean**.

Uniform Property Example

This example creates a table that contains a split cell and then displays a message box that confirms that the table doesn't have the same number of columns for each row.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 5, 5)
myTable.Cell(3, 3).Split
If myTable.Uniform = False Then MsgBox "Table is not uniform"
```

This example determines whether the table that contains the selection has the same number of columns for each row.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox Selection.Tables(1).Uniform
End If
```

UpdateAutoFormat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdateAutoFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdateAutoFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdateAutoFormatA "}

Updates the table with the characteristics of a predefined table format. For example, if you apply a table format with **AutoFormat** and then insert rows and columns, the table may no longer match the predefined look. **UpdateAutoFormat** restores the format.

Syntax

expression.**UpdateAutoFormat**

expression Required. An expression that returns a **Table** object.

UpdateAutoFormat Method Example

This example creates a table, applies a predefined format to it, adds a row, and then reapplies the predefined format.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Selection.Range, 5, 5)
With myTable
    .AutoFormat Format:=wdTableFormatColumns1
    .Rows.Add BeforeRow:=myTable.Rows(1)
End With
MsgBox "Click OK to reapply autoformatting."
myTable.UpdateAutoFormat
```

This example restores the predefined format to the table that contains the insertion point.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).UpdateAutoFormat
Else
    MsgBox "The insertion point is not in a table."
End If
```

AutoFormat Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoFormatA "}

Table object: Applies a predefined look to a table. The arguments for this method correspond to the options in the **Table AutoFormat** dialog box (**Table** menu).

Document or **Range** object: Automatically formats a document. Use the **Kind** property to specify a document type.

Syntax 1

expression.**AutoFormat(*Format, ApplyBorders, ApplyShading, ApplyFont, ApplyColor, ApplyHeadingRows, ApplyLastRow, ApplyFirstColumn, ApplyLastColumn, AutoFit*)**

Syntax 2

expression.**AutoFormat**

expression Required. An expression that returns a **Table** object (Syntax 1), or an expression that returns a **Document** or **Range** object (Syntax 2).

Format Optional **Variant**. The predefined table format. Can be any one of the **WdTableFormat** constants. The default value is **wdTableFormatSimple1**.

ApplyBorders Optional **Variant**. **True** to apply the border properties of the specified format. The default value is **True**.

ApplyShading Optional **Variant**. **True** to apply the shading properties of the specified format. The default value is **True**.

ApplyFont Optional **Variant**. **True** to apply the font properties of the specified format. The default value is **True**.

ApplyColor Optional **Variant**. **True** to apply the color properties of the specified format. The default value is **False**.

ApplyHeadingRows Optional **Variant**. **True** to apply the heading-row properties of the specified format. The default value is **True**.

ApplyLastRow Optional **Variant**. **True** to apply the last-row properties of the specified format. The default value is **False**.

ApplyFirstColumn Optional **Variant**. **True** to apply the first-column properties of the specified format. The default value is **True**.

ApplyLastColumn Optional **Variant**. **True** to apply the last-column properties of the specified format. The default value is **False**.

AutoFit Optional **Variant**. **True** to decrease the width of the table columns as much as possible without changing the way text wraps in the cells. The default value is **True**.

AutoFormat Method Example

This example creates a 5x5 table in a new document and applies all the properties of the Colorful 2 format to the table.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, _
    NumRows:=5, NumColumns:=5)
myTable.AutoFormat Format:=wdTableFormatColorful2, ApplyColor:=True
```

This example applies all the properties of the Classic 2 format to the table in which the insertion point is currently located. If the insertion point isn't in a table, a message box is displayed.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).AutoFormat Format:=wdTableFormatClassic2
Else
    MsgBox "The insertion point is not in a table."
End If
```

This example automatically formats the selection.

```
Selection.Range.AutoFormat
```

ConvertToText Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToTextA "}

Converts a table to text and returns a **Range** object that represents the delimited text.

Syntax

expression.**ConvertToText(*Separator*)**

expression Required. An expression that returns a **Row**, **Rows**, or **Table** object.

Separator Optional **Variant**. The character that delimits the converted columns (paragraph marks delimit the converted rows). Can be one of the following **WdTableFieldSeparator** constants: **wdSeparateByCommas**, **wdSeparateByDefaultListSeparator**, **wdSeparateByParagraphs**, or **wdSeparateByTabs**. The default value is **wdSeparateByTabs**.

Remarks

When you apply the **ConvertToText** method to a **Table** object, the object is deleted. To maintain a reference to the converted contents of the table, you must assign the **Range** object returned by the **ConvertToText** method to a new object variable. In the following example, the first table in the active document is converted to text and then formatted as a bulleted list.

```
Set myTable = ActiveDocument.Tables(1)
Set aRange = myTable.ConvertToText(Separator:=wdSeparateByParagraphs)
aRange.ListFormat.ApplyListTemplate ListTemplate:=ListGalleries( _
    wdBulletGallery).ListTemplates(1)
```

ConvertToText Method Example

This example creates a table and then converts it to text by using tabs as separator characters.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, _
    NumRows:=3, NumColumns:=3)
i = 1
For Each aCell In myTable.Range.Cells
    aCell.Range.InsertAfter "Cell " & i
    i = i + 1
Next aCell
MsgBox "Click OK to convert table to text."
Set myRange = myTable.ConvertToText(Separator:=wdSeparateByTabs)
```

This example converts the table that contains the selection to text, with spaces between the columns.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).ConvertToText Separator:=" "
Else
    MsgBox "The insertion point is not in a table."
End If
```

HeightRule Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeightRuleC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeightRuleX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeightRuleA "}

Cell, **Cells**, **Row**, and **Rows** objects: Returns or sets the rule for determining the height of the specified cells or rows. Can be one of the following **WdRowHeightRule** constants: **wdRowHeightAtLeast**, **wdRowHeightAuto**, or **wdRowHeightExactly**. Read/write **Long**.

Frame object: Returns or sets the rule for determining the height of the specified frame. Can be one of the following **WdFrameSizeRule** constants: **wdFrameAtLeast**, **wdFrameAuto**, or **wdFrameExact**. Read/write **Long**.

Remarks

Setting the **HeightRule** property of a **Cell** or **Cells** object automatically sets the property for the entire row.

HeightRule Property Example

This example creates a 3x3 table in a new document and then sets a minimum row height of 24 points for the second row.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, NumRows:=3, _
    NumColumns:=3)
With myTable.Rows(2)
    .Height = 24
    .HeightRule = wdRowHeightAtLeast
End With
```

This example sets the height rule for the selected rows to automatically adjust to the tallest cell in the row.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows.HeightRule = wdRowHeightAuto
Else
    MsgBox "The insertion point is not in a table."
End If
```

This example sets both the height and width of the first frame in the active document to exactly 1 inch.

```
If ActiveDocument.Frames.Count >= 1 Then
    With ActiveDocument.Frames(1)
        .HeightRule = wdFrameExact
        .Height = InchesToPoints(1)
        .WidthRule = wdFrameExact
        .Width = InchesToPoints(1)
    End With
End If
```

SetHeight Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetHeightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetHeightA "}

Sets the height of table rows or cells.

Syntax

expression.**SetHeight(*RowHeight, HeightRule*)**

expression Required. An expression that returns a **Cell**, **Cells**, **Row**, or **Row** object.

RowHeight Required **Single**. The height of the row or rows, in points.

HeightRule Optional **Variant**. The rule for determining the height of the specified cells. Can be one of the following **WdRowHeightRule** constants: **wdRowHeightAtLeast**, **wdRowHeightAuto**, or **wdRowHeightExactly**. The default value is **wdRowHeightAtLeast**.

Remarks

Setting the **SetHeight** property of a **Cell** or **Cells** object automatically sets the property for the entire row.

SetHeight Method Example

This example creates a table and then sets a fixed row height of 0.5 inch (36 points) for the first row.

```
Set newDoc = Documents.Add
Set aTable = newDoc.Tables.Add(Range:=Selection.Range, NumRows:=3, _
    NumColumns:=3)
aTable.Rows(1).SetHeight RowHeight:=InchesToPoints(0.5), _
    HeightRule:=wdRowHeightExactly
```

This example sets the row height of the selected cells to at least 18 points.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells.SetHeight RowHeight:=18, HeightRule:=wdRowHeightAtLeast
Else
    MsgBox "The insertion point is not in a table."
End If
```

SetLeftIndent Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetLeftIndentC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetLeftIndentX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetLeftIndentA "}

Sets the indentation for a row or rows in a table.

Syntax

expression.**SetLeftIndent(*LeftIndent, RulerStyle*)**

expression Required. An expression that returns a **Row** or **Rows** object.

LeftIndent Required **Single**. The distance (in points) between the left page margin and the left edge of the specified row or rows.

RulerStyle Required **Long**. Controls the way Word adjusts the table when the left indent is changed. Can be one of the following **WdRulerStyle** constants.

Constant	Description
wdAdjustFirstColumn	Preserves the position of the right edge of the table by narrowing cells in the first column only.
wdAdjustNone	Moves cells to the right. This is the default value.
wdAdjustProportional	Preserves the position of the right edge of the table by narrowing all the cells in the specified rows in proportion to their widths.
wdAdjustSameWidth	Preserves the position of the right edge of the table by narrowing all the cells in the specified rows, assigning the same width to all of them.

SetLeftIndent Method Example

This example creates a table in a new document and indents the first row 0.5 inch (36 points). When you change the left indent, the cell widths are adjusted to preserve the right edge of the table.

```
Set aDoc = Documents.Add
Set myTable = aDoc.Tables.Add(Range:=Selection.Range, NumRows:=3, _
    NumColumns:=3)
myTable.Rows(1).SetLeftIndent LeftIndent:=InchesToPoints(0.5), _
    RulerStyle:=wdAdjustSameWidth
```

This example indents the first row in table one in the active document 18 points, and it narrows the width of the first column to preserve the position of the right edge of the table.

```
If ActiveDocument.Tables.Count >= 1 Then
    ActiveDocument.Tables(1).Rows.SetLeftIndent LeftIndent:=18, _
        RulerStyle:=wdAdjustFirstColumn
End If
```

SetWidth Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetWidthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetWidthA "}

Sets the width of rows or cells in a table.

Syntax

expression.**SetWidth(*ColumnWidth, RulerStyle*)**

expression Required. An expression that returns a **Cell**, **Cells**, **Column**, or **Columns** object.

ColumnWidth Required **Single**. The width of the specified column or columns, in points.

RulerStyle Required **Long**. Controls the way Word adjusts cell widths. Can be one of the following **WdRulerStyle** constants.

Constant	Description
wdAdjustFirstColumn	Preserves the column width by narrowing cells in the first column only.
wdAdjustNone	Preserves the width of all columns other than the ones that contain the specified cells. This is the default value.
wdAdjustProportional	Preserves the column width by adjusting all cells to the right of the specified column.
wdAdjustSameWidth	Preserves the column width by narrowing all cells in the specified columns, assigning the same width to all of them.

SetWidth Method Example

This example creates a table in a new document and sets the width of the first cell in the second row to 1.5 inches. The example preserves the widths of the other cells in the table.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, NumRows:=3, _
    NumColumns:=3)
myTable.Cell(2,1).SetWidth ColumnWidth:=InchesToPoints(1.5),
RulerStyle:=wdAdjustNone
```

This example sets the width of the cell that contains the insertion point to 36 points. The example also narrows the first column to preserve the position of the right edge of the table.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Cells(1).SetWidth ColumnWidth:=36,
RulerStyle:=wdAdjustFirstColumn
Else
    MsgBox "The insertion point is not in a table."
End If
```

SortAscending Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSortAscendingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSortAscendingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSortAscendingA "}

Sorts paragraphs or table rows in ascending alphanumeric order. The first paragraph or table row is considered a header record and isn't included in the sort. Use the **Sort** method to include the header record in a sort.

Note This method offers a simplified form of sorting intended for mail merge data sources that contain columns of data. For most sorting tasks, use the **Sort** method.

Syntax

expression.**SortAscending**

expression Required. An expression that returns a **Range**, **Selection**, or **Table** object.

SortAscending Method Example

This example sorts the table that contains the selection in ascending order.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).SortAscending
Else
    MsgBox "The insertion point is not in a table."
End If
```

SortDescending Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSortDescendingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSortDescendingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSortDescendingA "}

Sorts paragraphs or table rows in descending alphanumeric order. The first paragraph or table row is considered a header record and isn't included in the sort. Use the **Sort** method to include the header record in a sort.

Note This method offers a simplified form of sorting intended for mail-merge data sources that contain columns of data. For most sorting tasks, use the **Sort** method.

Syntax

expression.**SortDescending**

expression Required. An expression that returns a **Range**, **Selection**, or **Table** object.

SortDescending Method Example

This example creates a 5x5 table in a new document, inserts text into each cell, and then sorts the table in descending alphanumeric order.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, NumRows:=5, _
    NumColumns:=5)
For iRow = 1 To myTable.Rows.Count
    For iCol = 1 To myTable.Columns.Count
        Set MyRange = myTable.Rows(iRow).Cells(iCol).Range
        MyRange.InsertAfter "Cell" & Str$(iRow) & "," & Str$(iCol)
    Next iCol
Next iRow
MsgBox "Click OK to sort in descending order."
myTable.SortDescending
```

This example sorts the table that contains the insertion point in descending alphanumeric order.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).SortDescending
Else
    MsgBox "The insertion point is not in a table."
End If
```

Width Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWidthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWidthA "}

Returns or sets the width of the specified object, in points. Read/write **Long**.

Width Property Example

This example creates a 5x5 table in a new document and then sets the width of the first cell to 1.5 inches.

```
Set newDoc = Documents.Add
Set myTable = newDoc.Tables.Add(Range:=Selection.Range, NumRows:=5, _
    NumColumns:=5)
myTable.Cell(1, 1).Width = InchesToPoints(1.5)
```

This example returns the width (in inches) of the cell that contains the insertion point.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox PointsToInches(Selection.Cells(1).Width)
End If
```

This example formats the section that includes the selection as three columns. The **For Each...Next** loop is used to display the width of each column in the **TextColumns** collection.

```
Selection.PageSetup.TextColumns.SetCount NumColumns:=3
For Each acol In Selection.PageSetup.TextColumns
    MsgBox "Width= " & PointsToInches(acol.Width)
Next acol
```

This example sets the width and height of the Word application window.

```
With Application
    .WindowState = wdWindowStateNormal
    .Width = 500
    .Height = 400
End With
```

AutoFormatType Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAutoFormatTypeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAutoFormatTypeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAutoFormatTypeA"}

Returns the type of automatic formatting that's been applied to the specified table. Read-only **Long**.

Can be one of the following **WdTableFormat** constants:

wdTableFormat3DEffects1	**wdTableFormatGrid4**
wdTableFormat3DEffects2	**wdTableFormatGrid5**
wdTableFormat3DEffects3	**wdTableFormatGrid6**
wdTableFormatClassic1	**wdTableFormatGrid7**
wdTableFormatClassic2	**wdTableFormatGrid8**
wdTableFormatClassic3	**wdTableFormatList1**
wdTableFormatClassic4	**wdTableFormatList2**
wdTableFormatColorful1	**wdTableFormatList3**
wdTableFormatColorful2	**wdTableFormatList4**
wdTableFormatColorful3	**wdTableFormatList5**
wdTableFormatColumns1	**wdTableFormatList6**
wdTableFormatColumns2	**wdTableFormatList7**
wdTableFormatColumns3	**wdTableFormatList8**
wdTableFormatColumns4	**wdTableFormatNone**
wdTableFormatColumns5	**wdTableFormatProfessional**
wdTableFormatContemporary	**wdTableFormatSimple1**
wdTableFormatElegant	**wdTableFormatSimple2**
wdTableFormatGrid1	**wdTableFormatSimple3**
wdTableFormatGrid2	**wdTableFormatSubtle1**
wdTableFormatGrid3	**wdTableFormatSubtle2**

Note Use the **AutoFormat** method to apply automatic formatting to a table.

AutoFormatType Property Example

This example formats the first table in the active document to use the Classic 1 AutoFormat if the current format is Simple 1, Simple 2, or Simple 3.

```
If ActiveDocument.Tables.Count >= 1 Then
    If ActiveDocument.Tables(1).AutoFormatType <= 3 Then
        ActiveDocument.Tables(1).AutoFormat Format:=wdTableFormatClassic1
    End If
End If
```

DistributeHeight Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDistributeHeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDistributeHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDistributeHeightA"}

Adjusts the height of the specified rows or cells so that they're equal.

Syntax

expression.**DistributeHeight**

expression Required. An expression that returns a **Cells** or **Rows** object.

DistributeHeight Method Example

This example adjusts the height of the rows in the first table in the active document so that they're equal.

```
ActiveDocument.Tables(1).Rows.DistributeHeight
```

This example adjusts the height of the first three rows in the first table so that they're equal.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myRange =
ActiveDocument.Range(Start:=ActiveDocument.Tables(1).Rows(1).Range.Start, _
        End:=ActiveDocument.Tables(1).Rows(3).Range.End)
    myRange.Rows.DistributeHeight
End If
```

DistributeWidth Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthDistributeWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthDistributeWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthDistributeWidthA"}

Adjusts the width of the specified columns or cells so that they're equal.

Syntax

expression.**DistributeWidth**

expression Required. An expression that returns a **Cells** or **Columns** object.

DistributeWidth Method Example

This example adjusts the width of the columns in the first table in the active document so that they're equal.

```
ActiveDocument.Tables(1).Columns.DistributeWidth
```

This example adjusts the height of the selected cells.

```
If Selection.Cells.Count >= 2 Then
    Selection.Cells.DistributeWidth
End If
```

InsertCells Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertCellsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertCellsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertCellsA"}

Adds cells to an existing table. The number of cells inserted is equal to the number of cells in the selection.

Note You can also insert cells by using the **Add** method of the **Cells** object.

Syntax

expression.**InsertCells**(***ShiftCells***)

expression Required. An expression that returns a **Selection** object.

ShiftCells Optional **Variant**. Can be one of the following **WdInsertCells** constants.

Constant	Description
wdInsertCellsEntireColumn	Inserts an entire column to the left of the column that contains the selection.
wdInsertCellsEntireRow	Inserts an entire row above the row that contains the selection.
wdInsertCellsShiftDown	Inserts new cells above the selected cells.
wdInsertCellsShiftRight	Insert new cells to the left of the selected cells.

InsertCells Method Example

This example inserts new cells to the left of the selected cells, and then it surrounds the selected cells with a red, single-line border.

```
If Selection.Cells.Count >= 1 Then
    Selection.InsertCells ShiftCells:=wdInsertCellsShiftRight
    For Each aBorder In Selection.Borders
        aBorder.LineStyle = wdLineStyleSingle
        aBorder.ColorIndex = wdRed
    Next aBorder
End If
```

InsertColumns Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertColumnsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertColumnsA"}

Inserts columns to the left of the column that contains the selection. If the selection isn't in a table, an error occurs.

Note The number of columns inserted is equal to the number of columns selected. You can also insert columns by using the **Add** method of the **Columns** object.

Syntax

expression.**InsertColumns**

expression Required. An expression that returns a **Selection** object.

InsertColumns Method Example

This example inserts new columns to the left of the column that contains the selection. The number of columns inserted is equal to the number of columns selected.

```
If Selection.Information(wdWithInTable) = True Then
    With Selection
        .InsertColumns
        .Shading.Texture = wdTexture10Percent
    End With
End If
```

InsertRows Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthInsertRowsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthInsertRowsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthInsertRowsA"}

Inserts the specified number of new rows above the row that contains the selection. If the selection isn't in a table, an error occurs.

Note You can also insert rows by using the **Add** method of the **Rows** object.

Syntax

expression.**InsertRows(*NumRows*)**

expression Required. An expression that returns a **Selection** object.

NumRows Optional **Variant**. The number of rows to be added.

InsertRows Method Example

This example inserts two new rows above the row that contains the selection, and then it removes the borders from the new rows.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.InsertRows NumRows:=2
    Selection.Borders.Enable =False
End If
```

SelectColumn Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectColumnC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectColumnX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectColumnA"}

Selects the column that contains the insertion point, or selects all columns that contain the selection. If the selection isn't in a table, an error occurs.

Syntax

expression.**SelectColumn**

expression Required. An expression that returns a **Selection** object.

SelectColumn Method Example

This example collapses the selection to the ending point and then selects the column that contains the insertion point.

```
Selection.Collapse Direction:=wdCollapseEnd
If Selection.Information(wdWithInTable) = True Then
    Selection.SelectColumn
End If
```

SelectRow Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSelectRowC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSelectRowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSelectRowA"}

Selects the row that contains the insertion point, or selects all rows that contain the selection. If the selection isn't in a table, an error occurs.

Syntax

expression.**SelectRow**

expression Required. An expression that returns a **Selection** object.

SelectRow Method Example

This example collapses the selection to the starting point and then selects the column that contains the insertion point.

```
Selection.Collapse Direction:=wdCollapseStart
If Selection.Information(wdWithInTable) = True Then
    Selection.SelectRow
End If
```

AutoMarkEntries Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoMarkEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoMarkEntriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoMarkEntriesA "}

Automatically adds XE (Index Entry) fields to the specified document, using the entries from a concordance file.

Note A concordance file is a Word document that contains a two-column table, with terms to index in the first column and index entries in the second column.

Syntax

expression.**AutoMarkEntries(*ConcordanceFileName*)**

expression Required. An expression that returns an **Indexes** object.

ConcordanceFileName Required **String**. The concordance file name that includes a list of items to be indexed.

AutoMarkEntries Method Example

This example adds index entries to Thesis.doc based on the entries in C:\Documents\List.doc.

```
Documents("Thesis.doc").Indexes.AutoMarkEntries ConcordanceFileName:="C:\
Documents\List.doc"
```

Caption Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproCaptionC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproCaptionX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproCaptionA "}

TableOfFigures object: Returns or sets the label that identifies the items to be included in a table of figures. Corresponds to the \c switch for a TOC field. Read/write **String**.

Window or **Application** object: Returns or sets the caption text for the specified document or application window. Read/write **String**.

Remarks

To change the caption of the application window to the default text, set this property to an empty string ("").

Caption Property Example

This example displays the caption of each window in the **Windows** collection.

```
Count = 1
For Each win In Windows
    MsgBox Prompt:=win.Caption, Title:="Window" & Str(Count) & _
    " Caption"
    Count = Count + 1
Next win
```

This example resets the caption of the application window.

```
Application.Caption = ""
```

This example sets the caption of the active window to the active document name.

```
ActiveWindow.Caption = ActiveDocument.FullName
```

This example changes the caption of the Word application window to include the user name.

```
Application.Caption = UserName & "'s copy of Word"
```

This example inserts a Table caption and then changes the caption of the first table of figures to "Table."

```
Selection.Collapse Direction:=wdCollapseStart
Selection.Range.InsertCaption "Table"
If ActiveDocument.TablesOfFigures.Count >= 1 Then
    ActiveDocument.TablesOfFigures(1).Caption = "Table"
End If
```

Category Property

This item is not available for this version of Office

Category Property Example

This item is not available for this version of Office.

EntrySeparator Property

This item is not available for this version of Office

EntrySeparator Property Example

This item is not available for this version of Office

HeadingSeparator Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeadingSeparatorC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeadingSeparatorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeadingSeparatorA "}

Returns or sets the text between alphabetic groups (entries that start with the same letter) in the index. Corresponds to the \h switch for an INDEX field. Can be one of the following **WdHeadingSeparator** constants: **wdHeadingSeparatorBlankLine**, **wdHeadingSeparatorLetter**, **wdHeadingSeparatorLetterFull**, **wdHeadingSeparatorLetterLow**, or **wdHeadingSeparatorNone**. Read/write **Long**.

HeadingSeparator Property Example

This example formats the first index for the active document in a single column, with the appropriate letter preceding each alphabetic group.

```
If ActiveDocument.Indexes.Count >= 1 Then
    With ActiveDocument.Indexes(1)
        .HeadingSeparator = wdHeadingSeparatorLetter
        .NumberOfColumns = 1
    End With
End If
```

IncludeCategoryHeader Property

This item is not available for this version of Office

IncludeCategoryHeader Property Example

This item is not available for this version of Office

IncludeLabel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludeLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludeLabelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludeLabelA "}

True if the caption label and caption number are included in a table of figures. Read/write **Boolean**.

IncludeLabel Property Example

This example formats the first table of figures in the active document to exclude caption labels (Figure 1, for example).

```
If ActiveDocument.TablesOfFigures.Count >= 1 Then
    ActiveDocument.TablesOfFigures(1).IncludeLabel = False
End If
```

This example adds a table of figures in place of the selection and then formats the table to include caption labels.

```
Set TOF = ActiveDocument.TablesOfFigures.Add(Range:=Selection.Range, _
    Caption:="Figure")
TOF.IncludeLabel = True
```

IncludePageNumbers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIncludePageNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIncludePageNumbersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIncludePageNumbersA "}

True if page numbers are included in the table of contents or table of figures. Read/write **Boolean**.

IncludePageNumbers Property Example

This example formats the first table of contents in the active document to include right-aligned page numbers.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .IncludePageNumbers = True
        .RightAlignPageNumbers = True
    End With
End If
```

IncludeSequenceName Property

This item is not included for this version of Office

IncludeSequenceName Property Example

This item is not available for this version of Office

Index Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIndexX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIndexA "}

Returns a number that indicates the position of an item in a collection. Read-only **Long**.

Index Property Example

This example adds a document variable to the active document and then returns the position of the variable in the **Variables** collection.

```
Set myVar = ActiveDocument.Variables.Add(Name:="Name", Value:="Joe")
num = myVar.Index
```

This example returns the position of the selected field in the **Fields** collection.

```
num = Selection.Fields(1).Index
```

This example returns the number of the first window in the **Windows** collection. If there are at least two windows in the **Windows** collection, the macro activates the next window, copies the first word, switches back to the original window, and inserts the Clipboard contents there.

```
Set myWindow = Windows(1)
winNum = myWindow.Index
If Windows.Count >= 2 Then
    myWindow.Next.Activate
    ActiveDocument.Words(1).Copy
    Windows(winNum).Activate
    Selection.Range.Paste
End If
```

Indexes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproIndexesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproIndexesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproIndexesA "}

Returns an **Indexes** collection that represents all the indexes in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Indexes Property Example

This example adds an index at the end of the active document.

```
Set MyRange = ActiveDocument.Range(Start:=ActiveDocument.Content.End - 1, _
    End:=ActiveDocument.Content.End - 1)
ActiveDocument.Indexes.Add Range:=MyRange, NumberOfColumns:=1, _
    HeadingSeparator:=False
```

This example sets the number of columns and the page number alignment for the indexes in Report.doc.

```
For Each aIndex In Documents("Report.doc").Indexes
    aIndex.NumberOfColumns = 2
    aIndex.RightAlignPageNumbers = True
Next aIndex
```

This example inserts an index entry for the selected text.

```
If Selection.Type = wdSelectionNormal Then
    ActiveDocument.Indexes.MarkEntry Range:=Selection.Range,
Entry:=Selection.Range.Text
End If
```

KeepEntryFormatting Property

This item is not available for this version of Office

KeepEntryFormatting Property Example

This item is not available for this version of Office

LowerHeadingLevel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproLowerHeadingLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproLowerHeadingLevelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproLowerHeadingLevelA "}

Returns or sets the ending heading level for a table of contents or table of figures. Corresponds to the ending value used with the \o switch for a TOC (Table of Contents) field. Read/write **Long**.

Remarks

Use the **UpperHeadingLevel** property to set the starting heading level. For example, to set the TOC field syntax {TOC \o "1-3"}, set the **LowerHeadingLevel** property to 3 and the **UpperHeadingLevel** property to 1.

LowerHeadingLevel Property Example

This example formats the first table of contents in the active document to show entries formatted with the Heading 2, Heading 3, or Heading 4 styles.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .UseHeadingStyles = True
        .UpperHeadingLevel = 2
        .LowerHeadingLevel = 4
    End With
End If
```

MarkEntry Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMarkEntryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMarkEntryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMarkEntryA "}

Syntax 1: Inserts an XE (Index Entry) field after the specified range. The XE field is returned as a **Field** object.

Syntax 2: Inserts a TC (Table of Contents Entry) field after the specified range. The TC field is returned as a **Field** object.

Syntax 1

expression.**MarkEntry(*Range, Entry, EntryAutoText, CrossReference, CrossReferenceAutoText, BookmarkName, Bold, Italic*)

Syntax 2

expression.**MarkEntry(*Range, Entry, EntryAutoText, TableID, Level*)

expression Syntax 1: Required. An expression that returns an **Indexes** object.

Syntax 2: Required. An expression that returns a **TablesOfContents** or **TablesOfFigures** object.

Range Required **Range** object. The location of the entry. The XE or TC field is inserted after ***Range***.

Entry Optional **Variant**. The text that appears in the index, table of contents, or table of figures. To indicate a subentry, include the main entry text and the subentry text, separated by a colon (:) (for example, "Introduction: The Product").

EntryAutoText Optional **Variant**. The AutoText entry name that includes text for the index, table of figures, or table of contents (***Entry*** is ignored).

CrossReference Optional **Variant**. A cross-reference that will appear in the index (for example, "See Apples").

CrossReferenceAutoText Optional **Variant**. The AutoText entry name that contains the text for a cross-reference (***CrossReference*** is ignored).

BookmarkName Optional **Variant**. The name of the bookmark that marks the range of pages you want to appear in the index. If this argument is omitted, the number of the page containing the XE field appears in the index.

Bold Optional **Variant**. **True** to add bold formatting to the entry page numbers in the index.

Italic Optional **Variant**. **True** to add italic formatting to the entry page numbers in the index.

TableID Optional **Variant**. A one-letter identifier for the table of figures or table of contents item (for example, "i" for an "illustration").

Level Optional **Variant**. A level for the entry in the table of contents or table of figures.

MarkEntry Method Example

This example inserts an index entry after the selection in the active document. The subentry text is the text from the selection.

```
If Selection.Type = wdSelectionNormal Then
    ActiveDocument.Indexes.MarkEntry Range:=Selection.Range, _
        Entry:="Introduction:" & Selection.Range.Text, Italic:=True
End If
```

This example inserts a table of contents entry that references the selected text. The text typed in the input box appears in the table of contents. A table of contents that uses fields is then added at the beginning of the active document.

```
entryText = InputBox("Type entry text")
ActiveDocument.TablesOfContents.MarkEntry Range:=Selection.Range,
Entry:=entryText
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.TablesOfContents.Add Range:=myRange, UseFields:=True,
UseHeadingStyles:=False
```

NumberOfColumns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproNumberOfColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproNumberOfColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproNumberOfColumnsA "}

Sets or returns the number of columns for each page of an index. Read/write **Long**.

Note Specifying 0 (zero) sets the number of columns in the index to the same number as in the document.

NumberOfColumns Property Example

This example sets the number of columns in the first index to the same number as in the active document.

```
ActiveDocument.Indexes(1).NumberOfColumns = 0
```

This example sets a two-column format for each index in the active document.

```
For Each myIndex In ActiveDocument.Indexes
    myIndex.NumberOfColumns = 2
Next myIndex
```

PageNumberSeparator Property

This item is not available for this version of Office

PageNumberSeparator Property Example

This item is not available for this version of Office

PageRangeSeparator Property

This item is not available for this version of Office

PageRangeSeparator Property Example

This item is not available for this version of Office

Passim Property

This item is not available for this version of Office

Passim Property Example

This item is not available for this version of Office

RightAlignPageNumbers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproRightAlignPageNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproRightAlignPageNumbersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproRightAlignPageNumbersA "}

True if page numbers are aligned with the right margin in an index, table of contents, or table of figures. Read/write **Boolean**.

RightAlignPageNumbers Property Example

This example right aligns page numbers for the first table of contents in the active document.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .IncludePageNumbers = True
        .RightAlignPageNumbers = True
    End With
End If
```

TabLeader Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTabLeaderC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTabLeaderX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTabLeaderA "}

Returns or sets the character between entries and their page numbers in an index, table of authorities, table of contents, or table of figures. Can be one of the following **WdTabLeader** constants: **wdTabLeaderDashes**, **wdTabLeaderDots**, **wdTabLeaderHeavy**, **wdTabLeaderLines**, or **wdTabLeaderSpaces**. Read/write **Long**.

TabLeader Property Example

This example formats the tables of contents in Sales.doc to use a dotted tab leader.

```
For Each aTOC In Documents("Sales.doc").TablesOfContents
    aTOC.TabLeader = wdTabLeaderDots
Next aTOC
```

This example adds an index at the end of the active document. The page numbers are right aligned with a dashed-line tab leader.

```
Set myRange = ActiveDocument.Range(Start:=ActiveDocument.Content.End -1, _
    End:=ActiveDocument.Content.End -1)
ActiveDocument.Indexes.Add(Range:=myRange, Type:=wdIndexIndent, _
    RightAlignPageNumbers:=True).TabLeader = wdTabLeaderDashes
```

TableID Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTableIDC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTableIDX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTableIDA "}

Returns or sets a one-letter identifier that's used to build a table of contents or table of figures from TC fields. Corresponds to the \f switch for a TOC field. For example, "T" builds a table of contents from TC fields using the table identifier T. Read/write **String**.

TableID Property Example

This example inserts a TC field with an "A" indentifier after the selection in Sales.doc. The first table of contents is then formatted to compile "A" entries.

```
Documents("Sales.doc").TablesOfContents.MarkEntry Range:=Selection.Range, _
    Entry:="Hello", TableID:="A"
With Documents("Sales.doc").TablesOfContents(1)
    .TableID = "A"
    .UseFields = True
    .UseHeadingStyles = False
    .Update
End With
```

This example adds a table of figures at the beginning of the active document and then formats the table to compile "B" entries.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
Set myTOF = ActiveDocument.TablesOfFigures.Add(Range:=myRange)
With myTOF
    .UseFields = True
    .UseHeadingStyles = False
    .TableID = "B"
    .Caption = ""
End With
```

TablesOfAuthoritiesCategories Property

This item is not available for this version of Office

TablesOfAuthoritiesCategories Property Example

This item is not available for this version of Office

TablesOfAuthorities Property

This item is not available for this version of Office

TablesOfAuthorities Property Example

This item is not available for this version of Office

TablesOfContents Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTablesOfContentsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTablesOfContentsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTablesOfContentsA "}

Returns a **TablesOfContents** collection that represents the tables of contents in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

TablesOfContents Property Example

This example adds a table of contents at the beginning of Sales.doc. The table of contents collects entry text from TC fields.

```
Set myRange = Documents("Sales.doc").Range(Start:=0, End:=0)
Documents("Sales.doc").TablesOfContents.Add Range:=myRange, _
    UseFields:=True, UseHeadingStyles:=False
```

This example updates the page numbers for items in the table of contents in the active document.

```
For Each myTOC In ActiveDocument.TablesOfContents
    myTOC.UpdatePageNumbers
Next myTOC
```

TablesOfFigures Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTablesOfFiguresC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTablesOfFiguresX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTablesOfFiguresA "}

Returns a **TablesOfFigures** collection that represents the tables of figures in the specified document. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

TablesOfFigures Property Example

This example adds a table of figures at the insertion point in the active document.

```
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.TablesOfFigures.Add Range:=Selection.Range, _
    Caption:=wdCaptionFigure
```

This example updates the contents of the first table of figures in Report.doc.

```
Documents("Report.doc").TablesOfFigures(1).Update
```

UpdatePageNumbers Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthUpdatePageNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthUpdatePageNumbersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthUpdatePageNumbersA "}

Updates the page numbers for items in the specified table of contents or table of figures.

Syntax

expression.**UpdatePageNumbers**

expression Required. An expression that returns a **TableOfContents** or **TableOfFigures** object.

UpdatePageNumbers Method Example

This example updates all tables of figures in Sales.doc.

```
For Each figs In Documents("Sales.doc").TablesOfFigures
    figs.UpdatePageNumbers
Next figs
```

This example inserts a page break at the insertion point and then updates the page numbers for the first table of contents in the active document.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.InsertBreak Type:=wdPageBreak
ActiveDocument.TablesOfContents(1).UpdatePageNumbers
```

UpperHeadingLevel Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUpperHeadingLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUpperHeadingLevelX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUpperHeadingLevelA "}

Returns or sets the starting heading level for a table of contents or table of figures. Corresponds to the starting value used with the \o switch for a TOC (Table of Contents) field. Read/write **Long**.

Remarks

Use the **LowerHeadingLevel** property to set the ending heading level. For example, to set the TOC field syntax {TOC \o "1-3"}, set the **LowerHeadingLevel** property to 3 and the **UpperHeadingLevel** property to 1.

UpperHeadingLevel Property Example

This example formats the first table of contents in the active document to compile all headings that are formatted with either the Heading 2 or Heading 3 style.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .UseHeadingStyles = True
        .UseFields = False
        .UpperHeadingLevel = 2
        .LowerHeadingLevel = 3
    End With
End If
```

UseFields Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUseFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUseFieldsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUseFieldsA "}

True if TC (Table of Contents Entry) fields are used to create a table of contents or a table of figures. Read/write **Boolean**.

UseFields Property Example

This example formats the first table of contents in the active document to use heading styles instead of TC fields.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .UseFields = False
        .UseHeadingStyles = True
    End With
End If
```

This example adds a table of figures after the selection and formats the table to compile entries with the "B" identifier.

```
Selection.Collapse Direction:=wdCollapseEnd
Set myTOF = ActiveDocument.TablesOfFigures.Add(Range:=Selection.Range)
With myTOF
    .UseFields = True
    .TableId = "B"
    .Caption = ""
End With
```

UseHeadingStyles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproUseHeadingStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproUseHeadingStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproUseHeadingStylesA "}

True if built-in heading styles are used to create a table of contents or a table of figures. Read/write **Boolean**.

UseHeadingStyles Property Example

This example formats the first table of contents in the active document to compile entries formatted with the Heading 1, Heading 2, or Heading 3 style.

```
If ActiveDocument.TablesOfContents.Count >= 1 Then
    With ActiveDocument.TablesOfContents(1)
        .UseHeadingStyles = True
        .UseFields = False
        .UpperHeadingLevel = 1
        .LowerHeadingLevel = 3
    End With
End If
```

This example adds a table of figures in place of the selection and then formats the table to compile entries from TC fields.

```
With ActiveDocument.TablesOfFigures.Add(Range:=Selection.Range)
    .UseHeadingStyles = False
    .UseFields = True
End With
```

NextCitation Method

This item is not available for this version of Office

NextCitation Method Example

This item is not available for this version of Office

MarkAllCitations Method

This item is not available for this version of Office

MarkAllCitations Method Example

This item is not available for this version of Office

MarkAllEntries Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthMarkAllEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthMarkAllEntriesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthMarkAllEntriesA "}

Inserts an XE (Index Entry) field after all instances of the text in ***Range***.

Syntax

expression.**MarkAllEntries(*Range, Entry, EntryAutoText, CrossReference, CrossReferenceAutoText, BookmarkName, Bold, Italic*)**

expression Required. An expression that returns an **Indexes** object.

Range Required **Range** object. The range whose text is marked with an XE field throughout the document.

Entry Optional **Variant**. The text you want to appear in the index, in the form *MainEntry[:Subentry]*.

EntryAutoText Optional **Variant**. The AutoText entry that contains the text you want to appear in the index (if this argument is specified, ***Entry*** is ignored).

CrossReference Optional **Variant**. A cross-reference that will appear in the index.

CrossReferenceAutoText Optional **Variant**. The name of the AutoText entry that contains the text for a cross-reference (if this argument is specified, ***CrossReference*** is ignored).

BookmarkName Optional **Variant**. The bookmark name that marks the range of pages you want to appear in the index. If this argument is omitted, the number of the page that contains the XE field appears in the index.

Bold Optional **Variant**. **True** to add bold formatting to page numbers for index entries.

Italic Optional **Variant**. **True** to add italic formatting to page numbers for index entries.

MarkAllEntries Method Example

This example marks the selected text with TA fields throughout the active document and then updates the first index in the document. The entry text in the index matches the selected text.

```
If Selection.Type = wdSelectionNormal Then
    ActiveDocument.Indexes.MarkAllEntries Range:=Selection.Range, _
        Entry:=Selection.Range.Text, Italic:=True
    ActiveDocument.Indexes(1).Update
End If
```

HeadingStyles Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHeadingStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHeadingStylesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHeadingStylesA "}

Returns a **HeadingStyles** object that represents additional styles used to compile a table of contents or table of figures (styles other than the Heading 1 – Heading 9 styles). Read-only.

HeadingStyles Property Example

This example adds a style to the **HeadingStyles** collection and then displays the names of all the style in the collection.

```
If ActiveDocument.TablesOfContents.Count >=1 Then
    ActiveDocument.TablesOfContents(1).HeadingStyles.Add _
        Style:="Title", Level:=2
    For Each hStyle In ActiveDocument.TablesOfContents(1).HeadingStyles
        MsgBox hStyle.Style
    Next hStyle
End If
```

This example adds a style named "Blue" to the **HeadingStyles** collection in a table of contents for Sales.doc.

```
With Documents("Sales.doc")
    .Styles.Add Name:="Blue"
    .TablesOfContents(1).UseHeadingStyles = True
    .TablesOfContents(1).HeadingStyles.Add Style:="Blue", Level:=4
End With
```

MarkCitation Method

This item is not available for this version of Office

MarkCitation Method Example

This item is not available for this version of Office

Converting WordBasic macros to Visual Basic

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowConvertingWordBasicMacrosC"}

Word 97 automatically converts the macros in a Word 6.x or Word 95 template the first time you do any of the following:

- Open the template
- Create a new document based on the template
- Attach the template to a document by using the **Templates** command (**Tools** menu).

A message is displayed on the status bar while the macros are being converted. After the conversion is complete, you must save the template to save the converted macros. If you don't save the template, Word converts the macros again the next time you use the template.

Note Word 97 cannot convert Word 2.x macros directly. Instead, you need to open and save your Word 2.x templates in Word 6.x or Word 95 and then open them in Word 97.

The conversion process converts each macro to a Visual Basic module. To see the converted macros, point to **Macro** on the **Tools** menu and click **Macros**. The macro names in the **Macros** dialog box appear as *macroname*.Main, where Main refers to the main subroutine in the converted macro (the subroutine that began with Sub MAIN in earlier versions of Word). To edit the converted macro, select a macro name and click **Edit** to display the Visual Basic module in the Visual Basic Editor.

Each WordBasic statement is modified to work with Visual Basic for Applications. The converted WordBasic macros are functionally equivalent to new Visual Basic for Applications macros you might write or record, but they are not identical. The following example is a WordBasic macro in a Word 95 template.

```
Sub MAIN
FormatFont .Name = "Arial", .Points = 10
Insert "Hello World"
End Sub
```

When the template is opened in Word 97, the macro is converted to the following code.

```
Public Sub Main()
WordBasic.FormatFont Font:="Arial", Points:=10
WordBasic.Insert "Hello World"
End Sub
```

Each statement in the converted macro begins with the **WordBasic** property. **WordBasic** is a property in the Word 97 object model that returns an object with all the WordBasic statements and functions; this object makes it possible to run WordBasic macros in Word 97.

Note If you save the template over the original template, the WordBasic macros will be permanently lost and previous versions of Word will not be able to use the converted macros.

The following Visual Basic macro is functionally the same as the preceding WordBasic macro, but doesn't use the **WordBasic** property.

```
Public Sub Main()
With Selection.Font
    .Name = "Arial"
    .Size = 10
End With
Selection.TypeText Text:="Hello World"
End Sub
```

Conceptual differences between WordBasic and Visual Basic

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWordBasicDifferencesC"}

The primary difference between Visual Basic for Applications and WordBasic is that whereas the WordBasic language consists of a flat list of approximately 900 commands, Visual Basic consists of a hierarchy of objects, each of which exposes a specific set of methods and properties (similar to statements and functions in WordBasic). While most WordBasic commands can be run at any time, Visual Basic only exposes the methods and properties of the available objects at a given time.

Objects are the fundamental building block of Visual Basic; almost everything you do in Visual Basic involves modifying objects. Every element of Word – documents, paragraphs, fields, bookmarks, and so on – can be represented by an object in Visual Basic. Unlike commands in a flat list, there are objects that can only be accessed from other objects. For example, the **Font** object can be accessed from various objects including the **Style**, **Selection**, and **Find** object.

The programming task of applying bold formatting demonstrates the differences between the two programming languages. The following WordBasic instruction applies bold formatting to the selection.

```
Bold 1
```

The following example is the Visual Basic equivalent for applying bold formatting to the selection.

```
Selection.Font.Bold = True
```

Visual Basic doesn't include a **Bold** statement and function. Instead, there's a property named **Bold**. (A property is usually an attribute of an object, such as its size, its color, or whether or not it's bold.) **Bold** is a property of the **Font** object. Likewise, **Font** is a property of the **Selection** object that returns a **Font** object. Following the object hierarchy, you can build the instruction to apply bold formatting to the selection.

The **Bold** property is a read/write Boolean property. This means that the **Bold** property can be set to **True** or **False** (on or off), or the current value can be returned. The following WordBasic instruction returns a value indicating whether bold formatting is applied to the selection.

```
x = Bold()
```

The following example is the Visual Basic equivalent for returning the bold formatting status from the selection.

```
x = Selection.Font.Bold
```

The Visual Basic thought process

To perform a task in Visual Basic, you need to determine the appropriate object. For example, if you want to apply character formatting found in the **Font** dialog box, use the **Font** object. Then you need to determine how to "drill down" through the Word object hierarchy from the **Application** object to the **Font** object, through the objects that contain the **Font** object you want to modify. After you have determined the path to your object (for example, `Selection.Font`), use the Object Browser, Help, or the features such as Auto List Members in the Visual Basic Editor to determine what properties and methods can be applied to the object. For more information about drilling down to objects using properties and methods, see Understanding objects, properties, and methods.

Properties and methods are often available to multiple objects in the Word object hierarchy. For example, the following instruction applies bold formatting to the entire document.

```
ActiveDocument.Content.Bold = True
```

Also, objects themselves often exists in more than one place in the object hierarchy. For an illustration of the Word object model, see Microsoft Word Objects.

If you know the WordBasic command for the task you want to accomplish in Word 97, see Visual Basic Equivalents for WordBasic Commands.

The Selection and Range objects

Most WordBasic commands modify the selection. For example, the **Bold** command formats the selection with bold formatting. The **InsertField** command inserts a field at the insertion point. Anytime you want to work with the selection in Visual Basic, you use the **Selection** property to return the **Selection** object. The selection can be a block of text or just the insertion point.

The following Visual Basic example inserts text and a new paragraph after the selection.

```
Selection.InsertAfter Text:="Hello World"
Selection.InsertParagraphAfter
```

In addition to working with the selection, you can define and work with various ranges of text in a document. A **Range** object refers to a contiguous area in a document with a starting character position and ending character position. Similar to the way bookmarks are used in a document, **Range** objects are used in Visual Basic to identify portions of a document. However, unlike a bookmark, a **Range** object is invisible to the user unless the **Range** has been selected using the **Select** method. For example, you can use Visual Basic to apply bold formatting anywhere in the document without changing the selection. The following example applies bold formatting to the first 10 characters in the active document.

```
ActiveDocument.Range(Start:=0, End:=10).Bold = True
```

The following example applies bold formatting to the first paragraph.

```
ActiveDocument.Paragraphs(1).Range.Bold = True
```

Both of these example change the formatting in the active document without changing the selection. For more information on the **Range** object see Working with Range objects.

Understanding objects, properties, and methods

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowUnderstandingObjectsC"}

Objects are the fundamental building block of Visual Basic; nearly everything you do in Visual Basic involves modifying objects. Every element of Word – documents, tables, paragraphs, bookmarks, fields and so on – can be represented by an object in Visual Basic.

What are objects and collections?

An object represents an element of Word, such as a document, a paragraph, a bookmark, or a single character. A collection is an object that contains several other objects, usually of the same type; for example, all the bookmark objects in a document are contained in a single collection object. Using properties and methods, you can modify a single object or an entire collection of objects.

What is a property?

A property is an attribute of an object or an aspect of its behavior. For example, properties of a document include its name, its content, and its save status, as well as whether change tracking is turned on. To change the characteristics of an object, you change the values of its properties.

To set the value of a property, follow the reference to an object with a period, the property name, an equal sign, and the new property value. The following example turns on change tracking in the document named "MyDoc.doc."

```
Documents("MyDoc.doc").TrackRevisions = True
```

In this example, `Documents` refers to the collection of open documents, and the name "MyDoc.doc" identifies a single document in the collection. The **TrackRevisions** property is set for that single document.

Some properties cannot be set. The Help topic for a property indicates whether that property can be set (read-write) or can only be read (read-only).

You can return information about an object by returning the value of one of its properties. The following example returns the name of the active document.

```
docName = ActiveDocument.Name
```

In this example, `ActiveDocument` refers to the document in the active window in Word. The name of that document is assigned to the variable `docName`.

Note The Help topic for each property indicates whether you can set that property (read-write), only read the property (read-only), or only write the property (write-only). Also the Object Browser in the Visual Basic Editor displays the read-write status at the bottom of the browser window when the property is selected.

What is a method?

A method is an action that an object can perform. For example, just as a document can be printed, the **Document** object has a **PrintOut** method. Methods often have arguments that qualify how the action is performed. The following example prints the first three pages of the active document.

```
ActiveDocument.PrintOut From:=1, To:=3
```

In most cases, methods are actions and properties are qualities. Using a method causes something to happen to an object, while using a property returns information about the object or it causes a quality about the object to change.

Returning an object

Most objects are returned by returning a single object from the collection. For example, the **Documents** collection contains the open Word documents. You use the **Documents** property of the

Application object (the object at the top of the Word object hierarchy) to return the **Documents** collection.

After you've accessed the collection, you can return a single object by using an index value in parentheses (this is similar to how you work with arrays). The index value is usually a number or a name. For more information, see Returning an Object from a Collection.

The following example uses the **Documents** property to access the **Document** collection. The index number is used to return the first document in the **Documents** collection. The **Close** method is then applied to the **Document** object to close the first document in the **Documents** collection.

```
Documents(1).Close
```

The following example uses a name (specified as a string) to identify a **Document** object within the **Documents** collection.

```
Documents("Sales.doc").Close
```

Collection objects often have methods and properties which you can use to modify the entire collection of objects. The **Documents** object has a **Save** method that saves all the documents in the collection. The following example saves the open documents by applying the Save method.

```
Documents.Save
```

The **Document** object also has a **Save** method available for saving a single document. The following example saves the document named Report.doc.

```
Documents("Report.doc").Save
```

To return an object that is further down in the Word object hierarchy, you must "drill down" to it by using properties and methods to return objects.

To see how this is done, open the Visual Basic Editor and click **Object Browser** on the **View** menu. Click **Application** in the **Classes** list on the left. Then click **ActiveDocument** from the list of members on the right. The text at bottom of the Object Browser indicates that **ActiveDocument** is a read-only property that returns a **Document** object. Click **Document** at the bottom of the Object Browser; the **Document** object is automatically selected in the **Classes** list, and the **Members** list displays the members of the **Document** object. Scroll through the list of members until you find **Close**. Click the **Close** method. The text at the bottom of the Object Browser window shows the syntax for the method. For more information about the method, press F1 or click the **Help** button to jump to the **Close** method Help topic.

Given this information, you can write the following instruction to close the active document.

```
ActiveDocument.Close SaveChanges:=wdSaveChanges
```

The following example maximizes the active document window.

```
ActiveWindow.WindowState = wdWindowStateMaximize
```

The **ActiveWindow** property returns a **Window** object that represents the active window. The **WindowState** property is set to the maximize constant (**wdWindowStateMaximize**).

The following example creates a new document and displays the Save As dialog box so that a name can be provided for the document.

```
Documents.Add.Save
```

The **Documents** property returns the **Documents** collection. The **Add** method creates a new document and returns a **Document** object. The **Save** method is then applied to the **Document** object.

As you can see, you use methods or properties to drill down to an object. That is, you return an object by applying a method or property to an object above it in the object hierarchy. After you return the object you want, you can apply the methods and control the properties of that object. To review the hierarchy of objects, see Microsoft Word Objects.

Getting Help on objects, methods, and properties

Until you become familiar with the Word object model, there are a few tools you can use to help you to drill down through the hierarchy.

- Auto List Members. When you type a period (.) after a property or method in the Visual Basic Editor, a list of available properties and methods is displayed. For example, if you type **Application.**, a drop-down list of methods and properties of the **Application** object is displayed.
- Help. You can also use Help to find out which properties and methods can be used with an object. Each object topic in Help includes a Properties and Methods jump that displays a list of properties and methods for the object. Press F1 in the Object Browser or a module to jump the appropriate Help topic.
- Microsoft Word Objects This topic illustrates how Word objects are arranged in the hierarchy. Click an object in the graphic to display the corresponding Help topic.
- Object Browser. The Object Browser in the Visual Basic Editor displays the members (properties and methods) of the Word objects.

Working with Range objects

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWorkingWithRangesC"}

A common task when using Visual Basic is to specify an area in a document and then do something with it, such as insert text or apply formatting. For example, you may want to write a macro that locates a word or phrase within a portion of a document. The portion of the document can be represented by a **Range** object. After the **Range** object is identified, methods and properties of the **Range** object can be applied in order to modify the contents of the range.

A **Range** object refers to a contiguous area in a document. Each **Range** object is defined by a starting and ending character position. Similar to the way bookmarks are used in a document, **Range** objects are used in Visual Basic procedures to identify specific portions of a document. A **Range** object can be as small as the insertion point or as large as the entire document. However, unlike a bookmark, a **Range** object only exists while the procedure that defined it is running.

The **Start**, **End** and **StoryType** properties uniquely identify a **Range** object. The **Start** and **End** properties return or set the starting and ending character positions of the **Range** object. The character position at the beginning of the document is zero, the position after the first character is one, and so on. There are 11 different story types represented by the **WdStoryType** constants of the **StoryType** property.

Note **Range** objects are independent of the selection. That is, you can define and modify a range without changing the current selection. You can also define multiple ranges in a document, while there is only one selection per document pane.

Using the Range method

The **Range** method is used to create a **Range** object in the specified document. The **Range** method (which is available from the **Document** object) returns a **Range** object located in the main story given a start and end point. The following example creates a **Range** object that is assigned to the variable `MyRange`.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=10)
```

`MyRange` refers to the first ten characters in the active document. You can see that the **Range** object has been created when you apply a property or method to the **Range** object stored in the `MyRange` variable. The following example applies bold formatting to the first ten characters in the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=10)
myRange.Bold = True
```

When you need to refer to a **Range** object multiple times, you can use the **Set** statement to set a variable equal to the **Range** object. However, if you only need to perform a single action on a **Range** object, there's no need to store the object in a variable. The same results can be achieved using just one instruction that identifies the range and changes the **Bold** property.

```
ActiveDocument.Range(Start:=0, End:=10).Bold = True
```

Like a bookmark, a range can span a group of characters or mark a location in a document. The **Range** object in the following example has the same starting and ending points. The range does not include any text. The following example inserts text at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
myRange.InsertBefore "Hello "
```

You can define the beginning and end points of a range using the character position numbers as shown above, or use the **Start** and **End** properties with objects such as **Selection**, **Bookmark**, or **Range**. The following example creates a **Range** object beginning at the start of the second paragraph and ending after the third paragraph.

```
Set myDoc = ActiveDocument
Set myRange = myDoc.Range(Start:=myDoc.Paragraphs(2).Range.Start, _
    End:=myDoc.Paragraphs(3).Range.End)
```

For additional information and examples, see the **Range** method.

Using the Range property

The **Range** property appears on multiple objects, such as **Paragraph**, **Bookmark**, and **Cell**, and is used to return a **Range** object. The following example returns a **Range** object that refers to the first paragraph in the active document.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
```

After you have a **Range** object, you can use any of its properties or methods to modify the **Range** object. The following example selects the second paragraph in the active document.

```
ActiveDocument.Paragraphs(2).Range.Select
```

If you need to apply numerous properties or methods to the same **Range** object, you can use the **With…End With** structure. The following example formats the text in the first paragraph of the active document.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
With myRange
    .Bold = True
    .ParagraphFormat.Alignment = wdAlignParagraphCenter
    .Font.Name = "Arial"
End With
```

For additional information and examples, see the **Range** property topic.

Redefining a Range object

Use the **SetRange** method to redefine an existing **Range** object. The following example defines `myRange` to the current selection. The **SetRange** method redefines `myRange` so that it refers to current selection plus the next ten characters.

```
Set myRange = Selection.Range
myRange.SetRange Start:=myRange.Start, End:=myRange.End + 10
```

For additional information and examples, see the **SetRange** method.

Note When debugging your macros, you can use the **Select** method to ensure that a **Range** object is referring to the correct range of text. For example, the following example selects the **Range** object named `aRange`. The `aRange` object refers the second and third paragraphs in the active document.

```
Set aRange = ActiveDocument.Paragraphs(2).Range
aRange.SetRange Start:=aRange.Start, _
    End:=ActiveDocument.Paragraphs(3).Range.End
aRange.Select
```

Working with Document objects

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWorkingWithDocumentsC"}

In Visual Basic, the methods for modifying files are methods of the **Document** object or the **Documents** collection object.

Creating a new document

The **Documents** collection includes all of the open documents. To create a new document, use the **Add** method to add a **Document** object to the **Documents** collection. The following instruction creates a new document.

```
Documents.Add
```

A better way to create a new document is to assign the return value to an object variable. The **Add** method returns a **Document** object that refers to the new document. In the following example, the **Document** object returned by the **Add** method is assigned to an object variable, `newDoc`. Then several properties and methods of the **Document** object are set. You can easily control the new document using the `newDoc` object variable.

```
Set newDoc = Documents.Add
With newDoc
    .Content.Font.Name = "Arial"
    .SaveAs FileName:="Sample.doc"
End With
```

Opening a document

To open an existing document, use the **Open** method with the **Documents** collection. The following instruction opens a document named "MyDocument.doc" located in the folder named "MyFolder."

```
Documents.Open FileName:="C:\MyFolder\MyDocument.doc"
```

Saving an existing document

To save a single document, use the **Save** method with the **Document** object. The following instruction saves the document named Sales.doc.

```
Documents("Sales.doc").Save
```

You can save all open documents by applying the **Save** method to the **Documents** collection. The following instruction saves all open documents.

```
Documents.Save
```

Saving a new document

To save a single document, use the **SaveAs** method with a **Document** object. The following instruction saves the active document as "Temp.doc" in the current folder.

```
ActiveDocument.SaveAs FileName:="Temp.doc"
```

The ***FileName*** argument can include only the file name or the complete path as shown in the following table.

Operating system	FileName syntax
Macintosh	"Hard Drive:Documents:Temporary File"
Windows	"C:\Documents\Temporary File.doc"

Closing documents

To close a single document, use the **Close** method with a **Document** object. The following instruction closes and saves the document named Sales.doc.

```
Documents("Sales.doc").Close SaveChanges:=wdSaveChanges
```

You can close all open documents by applying the **Close** method to the **Documents** collection. The following instruction closes all documents without saving changes.

```
Documents.Close SaveChanges:=wdDoNotSaveChanges
```

The following example prompts the user to save each document before the document is closed.

```
For Each aDoc In Documents
    aDoc.Save NoPrompt:=False
    aDoc.Close
Next
```

Activating a document

To change the active document, use the **Activate** method with a **Document** object. The following instruction activates the open document named MyDocument.doc.

```
Documents("MyDocument.doc").Activate
```

Determining if a document is open

To determine if a document is open, you can enumerate the **Documents** collection by using a **For Each...Next** statement. The following example activates the document named "Sample.doc" if the document is open, or opens Sample.doc if it's not currently open.

```
For Each aDoc In Documents
    If InStr(1, aDoc.Name, "sample.doc", 1) Then
        aDoc.Activate
    Else
        docFound = False
    End If
Next aDoc
If docFound = False Then Documents.Open FileName:="C:\Documents\Sample.doc"
```

Referring to the active document

Instead of referring to a document by name or index number – for example `Documents("Sales.doc")` – the **ActiveDocument** property returns a **Document** object which refers to the active document (the document with the focus). The following example displays the name of the active document, or if there are no documents open, it displays a message.

```
If Documents.Count >= 1 Then
    MsgBox ActiveDocument.Name
Else
    MsgBox "No documents are open"
End If
```

Revising recorded Visual Basic macros

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowRevisingRecordedMacrosC"}

The macro recorder is a great tool for discovering the Visual Basic methods and properties you want to use. If you don't know what properties or methods to use, turn on the macro recorder and manually perform the action. The macro recorder translates your actions into Visual Basic code. There are, however, some limitations to recording macros. You cannot record the following:

- Conditional branches
- Variable assignments
- Looping structures
- Custom user forms
- Error handling
- Text selections made with the mouse (you must use keyboard combinations).

To enhance your macros, you may want to revise the code recorded into your module.

Removing the Selection property

Macros created using the macro recorder depend on the selection. At the beginning of most recorded macro instructions, you'll see "`Selection`." Recorded macros use the **Selection** property to return the **Selection** object. For example, the following example moves the selection to the Temp bookmark and inserts text after the bookmark.

```
Sub Macro1()
Selection.Goto What:=wdGotoBookmark, Name:="Temp"
Selection.MoveRight Unit:=wdCharacter, Count:=1
Selection.TypeText Text:="New text"
End Sub
```

This macro accomplishes the task, but there are a couple of drawbacks. First, if the document doesn't have a bookmark named Temp, the macro posts an error. Second, the macro moves the selection, which may not be appropriate. Both of these issues can be resolved by revising the macro so that it doesn't use the **Selection** object. This is the revised macro.

```
Sub MyMacro()
If ActiveDocument.Bookmarks("Temp").Exists = True Then
    loc = ActiveDocument.Bookmarks("Temp").End
    ActiveDocument.Range(Start:=loc, End:=loc).InsertAfter "New text"
End If
End Sub
```

The **Exists** method is used to check for the existence of the bookmark named Temp. If the bookmark is found, the bookmark's ending character position is returned by the **End** property. Finally, the **Range** method is used to return a **Range** object that refers to the bookmark's ending position, so that text can be inserted using the **InsertAfter** method. For more information on defining **Range** objects, see Working with Range objects.

Using With…End With

Macros instructions that refer to the same object can be simplified using a **With…End With** structure. For example, the following macro was recorded when a title was added at the top of a document.

```
Sub Macro1()
Selection.HomeKey Unit:=wdStory
Selection.TypeText Text:="Title"
Selection.ParagraphAlignment.Alignment = wdAlignParagraphCenter
End Sub
```

The **Selection** property is used with each instruction to return a **Selection** object. The macro can be simplified so that the **Selection** property is only used once.

```
Sub MyMacro()
With Selection
    .HomeKey Unit:=wdStory
    .TypeText Text:="Title"
    .ParagraphAlignment.Alignment = wdAlignParagraphCenter
End With
End Sub
```

The same task can also be performed without using the **Selection** object. The following macro uses a **Range** object at the beginning of the active document to accomplish the same task.

```
Sub MyMacro()
With ActiveDocument.Range(Start:=0, End:=0)
    .InsertAfter "Title"
    .ParagraphFormat.Alignment = wdAlignParagraphCenter
End With
End Sub
```

Removing unnecessary properties

If you record a macro that involves selecting an option in a dialog box, the macro recorder records the settings of all the options in the dialog box, even if you only change one or two options. If you don't need to change all the options, you can remove the unnecessary properties from the recorded macro. The following recorded macro includes a number of options from the **Paragraph** dialog box (**Format** menu).

```
Sub Macro1()
With Selection.ParagraphFormat
    .LeftIndent = InchesToPoints(0)
    .RightIndent = InchesToPoints(0)
    .SpaceBefore = 6
    .SpaceAfter = 6
    .LineSpacingRule = 0
    .Alignment = wdAlignParagraphLeft
    .WidowControl = True
    .KeepWithNext = False
    .KeepTogether = False
    .PageBreakBefore = False
    .NoLineNumber = False
    .Hyphenation = True
    .FirstLineIndent = InchesToPoints(0)
    .OutlineLevel = 10
End With
End Sub
```

However, if you only want to change the spacing before and after the paragraph, you can change the macro to the following.

```
Sub MyMacro()
With Selection.ParagraphFormat
    .SpaceBefore = 6
    .SpaceAfter = 6
End With
End Sub
```

The simplified macro executes faster because it sets fewer properties. Only the spacing before and after are changed; all the other settings for the selected paragraphs are unchanged.

Removing unnecessary arguments

When the macro recorder records a method, the values of all the arguments are included. The following macro was recorded when the document named Test.doc was opened. The resulting macro includes all the arguments for the **Open** method.

```
Sub Macro1()
Documents.Open FileName:="C:\My Documents\Test.doc", ConfirmConversions:= _
    False, ReadOnly:=False, AddToRecentFiles:=False, PasswordDocument:="", _
    PasswordTemplate:="", Revert:=False, WritePasswordDocument:="", _
    WritePasswordTemplate:="", Format:=wdOpenFormatAuto
End Sub
```

The arguments that are not needed can be removed from the recorded macro. For example, you could remove all of arguments set to an empty string (for example, `WritePasswordDocument:=""`), as shown.

```
Sub MyMacro()
Documents.Open FileName:="C:\My Documents\Test.doc", ConfirmConversions:= _
    False, ReadOnly:=False, AddToRecentFiles:=False, _
    Revert:=False, Format:=wdOpenFormatAuto
End Sub
```

Modifying a portion of a document

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowManipulatingADocumentC"}

Visual Basic includes objects which you can use to modify the following document elements: characters, words, sentences, paragraphs and sections. The following table includes the properties that correspond to these document elements and the objects they return.

This expression	Returns this object
Words(*index*)	**Range**
Characters(*index*)	**Range**
Sentences(*index*)	**Range**
Paragraphs(*index*)	**Paragraph**
Sections(*index*)	**Section**

When these properties are used without an index, a collection object with the same name is returned. For example, the **Paragraphs** property returns the **Paragraphs** collection object. However, if you identify an item within these collections by index, the object in the second column of the table is returned. For example, `Words(1)` returns a **Range** object. After you have a **Range** object, you can use any of the range properties or methods to modify the **Range** object. For example, the following instruction copies the first word in the selection to the Clipboard.

```
Selection.Words(1).Copy
```

Note The items in the **Paragraphs** and **Sections** collections are singular forms of the collection rather than **Range** objects. However, the **Range** property (which returns a **Range** object) is available from both the **Paragraph** and **Section** objects. For example, the following instruction copies the first paragraph in the active document to the Clipboard.

```
ActiveDocument.Paragraphs(1).Range.Copy
```

All of the document element properties in the preceding table are available from the **Document**, **Selection**, and **Range** objects. The following examples demonstrate how you can drill down to these properties from **Document**, **Selection**, and **Range** objects.

The following example sets the case of the first word in the active document.

```
ActiveDocument.Words(1).Case = wdUpperCase
```

The following example sets the bottom margin of the current section to 0.5 inch.

```
Selection.Sections(1).PageSetup.BottomMargin = InchesToPoints(0.5)
```

The following example double spaces the text in the active document (the **Content** property returns a **Range** object).

```
ActiveDocument.Content.ParagraphFormat.Space2
```

Modifying a group of document elements

To modify a range of text that consists of a group of document elements (characters, words, sentences, paragraphs or sections), you need to create a **Range** object. The **Range** method creates a **Range** object given a start and end point. For example, the following instruction creates a **Range** object that refers to the first ten characters in the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=10)
```

Using the **Start** and **End** properties with a **Range** object, you can create a new **Range** object that refers to a group of document elements. For example, the following instruction creates a **Range** object (`myRange`) that refers to the first three words in the active document.

```
Set Doc = ActiveDocument
Set myRange = Doc.Range(Start:=Doc.Words(1).Start, _
```

```
    End:=Doc.Words(3).End)
```

The following example creates a **Range** object (`aRange`) beginning at the start of the second paragraph and ending after the third paragraph.

```
Set Doc = ActiveDocument
Set myRange = Doc.Range(Start:=Doc.Paragraphs(2).Range.Start, _
    End:=Doc.Paragraphs(3).Range.End)
```

For more information on defining **Range** objects, see Working with Range objects.

Working with the Selection object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWorkingWithSelectionC"}

When you work on a document in Word, you usually select text and then perform an action, such as formatting the text or typing text. In Visual Basic, it is usually not necessary to select text before modifying the text. Instead, you create a **Range** object that refers to a specific portion of the document. For information on defining **Range** objects, see Working with Range objects. However, when you want your code to respond to or change the selection, you can do so with the **Selection** object.

The **Select** method activates an object. For example, the following instruction selects the first word in the active document.

```
ActiveDocument.Words(1).Select
```

For more **Select** method examples, see the **Select** method or Selecting text in a document.

The **Selection** property returns a **Selection** object that represents the active selection in a document window pane. There can only be one **Selection** object per document window pane and only one **Selection** object can be active. For example, the following example changes the paragraph formatting of the paragraphs in the selection.

```
Selection.Paragraphs.LeftIndent = InchesToPoints(0.5)
```

For example, the following example inserts the word "Hello" after the selection.

```
Selection.InsertAfter Text:="Hello"
```

The following example applies bold formatting to the selected text.

```
Selection.Font.Bold = True
```

The macro recorder will often create a macro that uses the **Selection** property. The following example was created using the macro recorder. The macro applies bold formatting to the first two words in the document.

```
Selection.HomeKey Unit:=wdStory
Selection.MoveRight Unit:=wdWord, Count:=2, Extend:=wdExtend
Selection.Font.Bold = wdToggle
```

The following example accomplishes the same task without using the **Selection** property.

```
ActiveDocument.Range(Start:=0, End:=ActiveDocument.Words(2).End).Bold =
True
```

Returning an Object from a Collection

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowReturningAnObjectC"}

The **Item** method returns a single object from a collection. The following example sets the `firstDoc` variable to a **Document** object that represents the first document in the **Documents** collection.

```
Set firstDoc = Documents.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 firstDoc = Documents(1)
```

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. The following example switches the focus to a document named Sales.doc.

```
Documents("Sales.doc").Activate
```

The following example selects the range of text marked by the bookmark named temp in the active document.

```
ActiveDocument.Bookmarks("temp").Select
```

Not all collections can be indexed by name. To determine the valid collection index values, see the collection object topic.

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 an **WdBorderType** constant with the **Borders** property to return a single **Border** object.

The following example adds a single 0.75 point border below the first paragraph in the selection.

```
With Selection.Paragraphs(1).Borders(wdBorderBottom)
    .LineStyle = wdLineStyleSingle
    .LineWidth = wdLineWidth075pt
End With
```

default properties and methods

Many objects have a default property or method. Visual Basic applies the default property or method to a given object to resolve an expression that wouldn't otherwise be valid. You can write statements that use default properties or methods more concisely by omitting the default keywords. For example, the **Item** method is the default method for most collections in Word, so the following statements are equivalent.

```
ActiveDocument.Paragraphs.Item(1).Alignment = wdAlignParagraphCenter
ActiveDocument.Paragraphs(1).Alignment = wdAlignParagraphCenter
```

Note Default properties and methods are indicated by a blue dot in addition to the property or method graphic that appears before the name in the Object Browser.

Automating common Word tasks

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowAutomatingCommonWordTasksC"}

This topic includes some common Word tasks and the Visual Basic code needed to accomplish the tasks.

Applying formatting to text

This topic includes Visual Basic examples related to the following tasks:

- Applying formatting to the selection
- Applying formatting to a range
- Inserting text and applying character and paragraph formatting
- Toggling the space before a paragraph between 12 points and none
- Toggling bold formatting
- Increasing the left margin by 0.5 inch

Applying formatting to the selection

The following example uses the **Selection** property to apply character and paragraph formatting to the selected text. Use the **Font** property to access character formatting properties and methods and the **ParagraphFormat** property to access paragraph formatting properties and methods.

```
With Selection.Font
    .Name = "Times New Roman"
    .Size = 14
    .AllCaps = True
End With
With Selection.ParagraphFormat
    .LeftIndent = InchesToPoints(0.5)
    .Space1
End With
```

Applying formatting to a range

The following example defines a **Range** object that refers to the first three paragraphs in the active document. The **Range** (`myRange`) is formatted by applying properties of the **Font** and **ParagraphFormat** objects.

```
Set myRange =
ActiveDocument.Range(Start:=ActiveDocument.Paragraphs(1).Range.Start, _
    End:=ActiveDocument.Paragraphs(3).Range.End)
With myRange
    .Font.Name = "Arial"
    .ParagraphFormat.Alignment = wdAlignParagraphJustify
End With
```

Inserting text and applying character and paragraph formatting

The following example adds the word Title at the top of the current document. The first paragraph is center aligned and a half inch space is added after the paragraph. The word Title is formatted with 24 point Arial font.

```
Set oRange = ActiveDocument.Range(Start:=0, End:=0)
With oRange
    .InsertAfter Text:="Title"
    .InsertParagraphAfter
    .Font.Name = "Arial"
    .Font.Size = 24
End With
With ActiveDocument.Paragraphs(1)
    .Alignment = wdAlignParagraphCenter
```

```
    .SpaceAfter = InchesToPoints(.5)
End With
```

Toggling the space before a paragraph between 12 points none

The following example toggles the space before formatting of the first paragraph in the selection. The macro retrieves the current space before value, and if the value is 12 points the space before formatting is removed (the **SpaceBefore** property is set to zero). If the space before value is anything other than 12, then **SpaceBefore** property is set to 12 points.

```
Set oParagraph = Selection.Paragraphs(1)
If oParagraph.SpaceBefore = 12 Then
    oParagraph.SpaceBefore = 0
Else
    oParagraph.SpaceBefore = 12
End If
```

Toggling bold formatting

The following example toggles bold formatting of the selected text.

```
Selection.Font.Bold = wdToggle
```

Increasing the left margin by 0.5 inch

The following example increases the left margin by 0.5 inch. The **PageSetup** object contains all the page setup attributes of a document (left margin, bottom margin, paper size, and so on) as properties. The **LeftMargin** property is used to return and set the left margin setting.

```
iMargin = ActiveDocument.PageSetup.LeftMargin
iMargin = iMargin + InchesToPoints(0.5)
ActiveDocument.PageSetup.LeftMargin = iMargin
```

Editing text

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowEditingTextC"}

This topic includes Visual Basic examples related to the following tasks:

- Determining whether text is selected
- Collapsing a selection or range
- Extending a selection or range
- Redefining a Range object
- Changing text

For information about and examples of other editing tasks, see the following topics:

Returning text from a document

Selecting text in a document

Inserting text in a document

Manipulating a portion of a document

Determining whether text is selected

The **Type** property of the **Selection** object returns information about the type of selection. The following example displays a message if the selection is an insertion point.

```
If Selection.Type = wdSelectionIP Then MsgBox "Nothing is selected"
```

Collapsing a selection or range

Use the **Collapse** method to collapse a **Selection** or **Range** object to it's beginning or ending point. The following example collapses the selection to an insertion point at the beginning of the selection.

```
Selection.Collapse Direction:=wdCollapseStart
```

The following example cancels the `myRange` object to it's ending point (after the first word).

```
Set myRange = ActiveDocument.Words(1)
myRange.Collapse Direction:=wdCollapseEnd
```

Extending a selection or range

The following example uses the **MoveEnd** method to extend the end of the selection to include three additional words. The **MoveLeft**, **MoveRight**, **MoveUp** and **MoveDown** methods can also be used to extend a **Selection** object.

```
Selection.MoveEnd Unit:=wdWord, Count:=3
```

The following example uses the **MoveEnd** method to extend `oRange` to include the first three paragraphs in the active document.

```
Set oRange = ActiveDocument.Paragraphs(1).Range
oRange.MoveEnd Unit:=wdParagraph, Count:=2
```

Redefining a Range object

Use the **SetRange** method to redefine an existing **Range** object. For more information, see Working with Range objects.

Changing text

You can change existing text by changing the contents of a range. The following instruction changes the first word in the active document by setting the **Text** property to "The."

```
ActiveDocument.Words(1).Text = "The "
```

You can also use the **Delete** method to delete existing text and then insert new text using the **InsertAfter** or **InsertBefore** method. The following example deletes the first paragraph in the active document and inserts new text.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
With myRange
    .Delete
    .InsertAfter Text:="New text"
    .InsertParagraphAfter
End With
```

Miscellaneous tasks

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowMiscellaneousC"}

This topic includes Visual Basic examples for the following tasks:

- Changing the view
- Setting text in a header or footer
- Setting options
- Changing the document layout
- Looping through paragraphs in a document
- Customizing menus and toolbars

Changing the view

The **View** object includes properties and methods related to view attributes (show all, field shading, table gridlines, and so on) for a window or pane. The following example changes the view to page layout view.

```
ActiveWindow.View.Type = wdPageView
```

Setting text in a header or footer

The **HeaderFooter** object is returned by the **Headers**, **Footers** and **HeaderFooter** properties. The following example changes the text of current page header.

```
ActiveWindow.View.SeekView = wdSeekCurrentPageHeader
Selection.HeaderFooter.Range.Text = "Header text"
```

This example creates a **Range** object (`oRange`) that references the primary footer for the first section in the active document. After the **Range** object is set, the existing footer text is deleted. The FILENAME field is added to the footer along with two tabs and the AUTHOR field.

```
Set oRange =
ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary).Range
With oRange
    .Delete
    .Fields.Add Range:=oRange, Type:=wdFieldFileName, Text:="\p"
    .InsertAfter Text:=vbTab & vbTab
    .Collapse Direction:=wdCollapseStart
    .Fields.Add Range:=oRange, Type:=wdFieldAuthor
End With
```

Setting options

The **Options** object includes properties that correspond to items in the **Options** dialog box (**Tools** menu). The following example sets three application options for Word.

```
With Options
    .AllowDragAndDrop = True
    .ConfirmConversions = False
    .MeasurementUnit = wdPoints
End With
```

Changing the document layout

The **PageSetup** contains all the page setup attributes of a document (left margin, bottom margin, paper size, and so on) as properties. The following example sets the margin values for the active document.

```
With ActiveDocument.PageSetup
    .LeftMargin = InchesToPoints(0.75)
    .RightMargin = InchesToPoints(0.75)
    .TopMargin = InchesToPoints(1.5)
    .BottomMargin = InchesToPoints(1)
End With
```

Looping through paragraphs in a document

This example loops through all of the paragraphs in the active document. If the space before setting for a paragraph is 6 points, this example changes the spacing to 12 points.

```
For Each aPara In ActiveDocument.Paragraphs
   If aPara.SpaceBefore = 6 Then oPara.SpaceBefore = 12
Next aPara
```

For more information, see Looping through a collection.

Customizing menus and toolbars

The **CommandBar** object represents both menus and toolbars. Use the **CommandBars** property with a menu or toolbar name to return a single **CommandBar** object. The **Controls** property returns a **CommandBarControls** object that refers to the items on the specified command bar. The following example adds the Word Count command to the **Tools** menu.

```
CustomizationContext = NormalTemplate
CommandBars("Tools").Controls.Add Type:=msoControlButton, ID:=792, _
    Before:=6
```

The following example adds the **Double Underline** command to the **Formatting** toolbar.

```
CustomizationContext = NormalTemplate
CommandBars("Formatting").Controls.Add Type:=msoControlButton, ID:=60, _
    Before:=7
```

Turn on the macro recorder and customize a menu or toolbar to determine the **ID** value for a particular command (for example, ID 60 is the **Double Underline** command).

Working with tables

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWorkingWithTablesC"}

This topic includes Visual Basic examples related to the following tasks:

- Inserting text into a table cell
- Creating a table, inserting text, and applying formatting
- Returning text from a table cell without returning the end of cell marker
- Converting text to a table
- Returning the contents of each table cell
- Copying all tables in the active document into a new document

Inserting text into a table cell

The following example inserts text into the first cell of the first table in the active document. The **Cell** method returns a single **Cell** object. The **Range** property returns a **Range** object. The **Delete** method is used to delete the existing text and the **InsertAfter** method inserts the "Cell 1,1" text.

```
If ActiveDocument.Tables.Count >= 1 Then
    With ActiveDocument.Tables(1).Cell(Row:=1, Column:=1).Range
        .Delete
        .InsertAfter Text:="Cell 1,1"
    End With
End If
```

Creating a table, inserting text, and applying formatting

The following example inserts a 4 column, 3 row table at the beginning of the document. The **For Each...Next** structure is used to step through each cell in the table. Within the **For Each...Next** structure, the **InsertAfter** method is used to add text to the table cells (Cell 1, Cell 2, and so on).

```
Set oDoc = ActiveDocument
Set oTable = oDoc.Tables.Add(Range:=oDoc.Range(Start:=0, End:=0),
NumRows:=3, _
    NumColumns:=4)
iCount = 1
For Each oCell In oTable.Range.Cells
    oCell.Range.InsertAfter "Cell " & iCount
    iCount = iCount + 1
Next oCell
oTable.AutoFormat Format:=wdTableFormatColorful2, _
    ApplyBorders:=True, ApplyFont:=True, ApplyColor:=True
```

Returning text from a table cell without returning the end of cell marker

The following examples return and display the contents of each cell in the first row of the first document table.

```
Set oTable = ActiveDocument.Tables(1)
For Each aCell In oTable.Rows(1).Cells
    Set myRange = ActiveDocument.Range(Start:=aCell.Range.Start, _
        End:=aCell.Range.End - 1)
    MsgBox myRange.Text
Next aCell

Set oTable = ActiveDocument.Tables(1)
For Each aCell In oTable.Rows(1).Cells
```

```
    Set myRange = aCell.Range
    myRange.MoveEnd Unit:=wdCharacter, Count:=-1
    MsgBox myRange.Text
Next aCell
```

Converting existing text to a table

The following example inserts tab-delimited text at the beginning of the active document and then converts the text to a table.

```
Set oRange1 = ActiveDocument.Range(Start:=0, End:=0)
oRange1.InsertBefore "one" & vbTab & "two" & vbTab & "three" & vbCr
Set oTable1 = oRange1.ConvertToTable(Separator:=Chr(9), NumRows:=1,
NumColumns:=3)
```

Returning the contents of each table cell

The following example defines an array equal to the number of cells in the first document table (assuming **Option Base 1**). The **For Each...Next** structure is used to return the contents of each table cell and assign the text to the corresponding array element.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set oTable = ActiveDocument.Tables(1)
    iNumCells = oTable.Range.Cells.Count
    ReDim aCells(iNumCells)
    i = 1
    For Each oCell In oTable.Range.Cells
        Set myRange = oCell.Range
        myRange.MoveEnd Unit:=wdCharacter, Count:=-1
        aCells(i) = myRange.Text
        i = i + 1
    Next oCell
End If
```

Copying all tables in the active document into a new document

This example copies the tables from the current document into a new document.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set oDoc1 = ActiveDocument
    Set MyRange = Documents.Add.Range(Start:=0, End:=0)
    For Each oTable In oDoc1.Tables
        oTable.Range.Copy
        With MyRange
            .Paste
            .Collapse Direction:=wdCollapseEnd
            .InsertParagraphAfter
            .Collapse Direction:=wdCollapseEnd
        End With
    Next
End If
```

Finding and replacing text or formatting

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowSearchAndReplaceC"}

Finding and replacing is exposed by the **Find** and **Replacement** objects. The **Find** object is available from the **Selection** and **Range** object. The find action differs slightly depending upon whether you access the **Find** object from the **Selection** or **Range** object.

Finding text and selecting it

If the **Find** object is accessed from the **Selection** object, the selection is changed when the find criteria is found. The following example selects the next occurrence of the word "Hello." If the end of the document is reached before the word "Hello" is found, the search is stopped.

```
With Selection.Find
    .Forward = True
    .Wrap = wdFindStop
    .Text = "Hello"
    .Execute
End With
```

The **Find** object includes properties that relate to the options in the **Find and Replace** dialog box (choose **Find** from the **Edit** menu). You can set the individual properties of the **Find** object or use arguments with the **Execute** method as shown in the following example.

```
Selection.Find.Execute FindText:="Hello", Forward:=True, Wrap:=wdFindStop
```

Finding text without changing the selection

If the **Find** object is accessed from a **Range** object, the selection is not changed but the **Range** is redefined when the find criteria is found. The following example locates the first occurrence of the word "blue" in the active document. If the find operation is successful, the range is redefined and bold formatting is applied to the word "blue."

```
With ActiveDocument.Content.Find
    .Text = "blue"
    .Forward = True
    .Execute
    If .Found = True Then .Parent.Bold = True
End With
```

The following example performs the same result as the previous example using arguments of the **Execute** method.

```
Set myRange = ActiveDocument.Content
myRange.Find.Execute FindText:="blue", Forward:=True
If myRange.Find.Found = True Then myRange.Bold = True
```

Using the Replacement object

The **Replacement** object represents the replace criteria for a find and replace operation. The properties and methods of the **Replacement** object correspond to the options in the **Find and Replace** dialog box (**Edit** menu).

The **Replacement** object is available from the **Find** object. The following example replaces all occurrences of the word "hi" with "hello." The selection changes when the find criteria is found because the **Find** object is accessed from the **Selection** object.

```
With Selection.Find
    .ClearFormatting
    .Text = "hi"
    .Replacement.ClearFormatting
```

```
    .Replacement.Text = "hello"
    .Execute Replace:=wdReplaceAll, Forward:=True, Wrap:=wdFindContinue
End With
```

The following example removes bold formatting in the active document. The **Bold** property is **True** for the **Find** object and **False** for the **Replacement** object. In order to find and replace formatting, set the find and replace text to empty strings ("") and set the ***Format*** argument of the **Execute** method to **True**. The selection remains unchanged because the **Find** object is accessed from a **Range** object (the **Content** property returns a **Range** object).

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Font.Bold = True
    With .Replacement
        .ClearFormatting
        .Font.Bold = False
    End With
    .Execute FindText:="", ReplaceWith:="", Format:=True,
Replace:=wdReplaceAll
End With
```

Frequently asked Visual Basic questions

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowFrequentlyAskedC"}

General questions

Questions about Word features

Referring to the active element

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowActiveParagraphC"}

To refer to the active paragraph, table, field, or other document element, use the **Selection** property to return a **Selection** object. From the **Selection** object, you can access all paragraphs in the selection or the first paragraph in the selection. The following example applies a border around the first paragraph in the selection.

```
Selection.Paragraphs(1).Borders.Enable = True
```

The following example applies a border around all the paragraphs in the selection.

```
Selection.Paragraphs.Borders.Enable = True
```

The following example applies shading to the first row of the first table in the selection.

```
Selection.Tables(1).Rows(1).Shading.Texture = wdTexture10Percent
```

An error occurs if the selection doesn't include a table. Use the **Count** property to determine if the selection includes a table. The following example applies shading to the first row of the first table in the selection.

```
If Selection.Tables.Count >= 1 Then
    Selection.Tables(1).Rows(1).Shading.Texture = wdTexture10Percent
Else
    MsgBox "Selection doesn't include a table"
End If
```

The following example applies shading to the first row of every table in the selection. The **For Each...Next** loop is used to step through the individual tables in the selection.

```
If Selection.Tables.Count >= 1 Then
    For Each aTable In Selection.Tables
        aTable.Rows(1).Shading.Texture = wdTexture10Percent
    Next aTable
End If
```

Selecting text in a document

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowSelectTextC"}

Use the **Select** method to select an item in a document. The **Select** method is available from several objects, such as **Bookmark**, **Field**, **Range**, and **Table**. The following example selects the first table in the active document.

```
ActiveDocument.Tables(1).Select
```

The following example selects the first field in the active document.

```
ActiveDocument.Fields(1).Select
```

The following example selects the first four paragraphs in the active document. The **Range** method is used to create a **Range** object which refers to the first four paragraphs. The **Select** method is then applied to the **Range** object.

```
Set myRange =
ActiveDocument.Range(Start:=ActiveDocument.Paragraphs(1).Range.Start, _
    End:=ActiveDocument.Paragraphs(4).Range.End)
myRange.Select
```

For more information, see Working with the Selection object.

Inserting text in a document

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowInsertTextC"}

Use the **InsertAfter** or **InsertBefore** method to insert text before or after a **Selection** or **Range** object. The following example inserts text at the end of the active document.

```
ActiveDocument.Content.InsertAfter Text:=" the end."
```

The following example inserts text before the selection.

```
Selection.InsertBefore Text:="new text "
```

After using the **InsertBefore** or **InsertAfter** method, the **Range** or **Selection** expands to include the new text. Use the **Collapse** method to collapse a **Selection** or **Range** to the beginning or ending point.

Microsoft Office 97/Visual Basic Programmer's Guide

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowProgrammersGuideC"}

The Microsoft® Office 97/Visual Basic® Programmer's Guide (Microsoft Press, 1996) is available wherever computer books are sold and directly from Microsoft Press. ISBN 1-57231-340-4. A comprehensive guide and reference to programming Word and other Microsoft Office applications. Credit card orders: 1-800-MSPRESS (1-800-677-7377) or FAX 1-206-936-7329. CompuServe: GO MSP.

Information about Microsoft Press books is available at http://www.microsoft.com/mspress/.

The requested member of the collection does not exist

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowMemberExistsC"}

The "requested member of the collection does not exist" error occurs when you try to access an object that doesn't exist. For example, the following instruction may post an error if the active document doesn't contain at least one table.

```
ActiveDocument.Tables(1).Select
```

To avoid this error when accessing a member of a collection, ensure that the member exists prior to accessing the collection member. If you're accessing the member by index number, you can use the **Count** property to determine if the member exists. The following example selects the first table if there is at least one table in the active document.

```
If ActiveDocument.Tables.Count >=1 Then
    ActiveDocument.Tables(1).Select
Else
    MsgBox "Document doesn't contain a table"
End If
```

If you're accessing a collection member by name, you can loop on the elements in a collection using a **For Each...Next** loop to determine if the named member is part of the collection. For example, the following example deletes the AutoCorrect entry named "acheive" if it's part of the **AutoCorrectEntries** collection.

```
For Each aEntry In AutoCorrect.Entries
    If aEntry.Name = "acheive" Then aEntry.Delete
Next aEntry
```

Error accessing a table row of column

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowDrawnTablesC"}

When you try to access an individual row or column in a drawn table, a runtime error may occur if the table is not uniform. For example, the following instruction posts an error if the first table in the active document doesn't have the same number of rows in each column.

```
ActiveDocument.Tables(1).Rows(1).Borders.Enable = False
```

You can avoid this error by first selecting the cells in a column or row using the **SelectColumn** or **SelectRow** method. After the selection is made, use the **Cells** property with the **Selection** object. The following example selects the first row in the first document table. The **Cells** property is used to access the selected cells (all the cells in the first row) so that borders can be removed.

```
ActiveDocument.Tables(1).Cell(1, 1).Select
With Selection
    .SelectRow
    .Cells.Borders.Enable = False
End With
```

The following example selects the first column in the first document table. The **For Each...Next** loop is used to add text to each cell in the selection (all the cells in the first column).

```
ActiveDocument.Tables(1).Cell(1, 1).Select
Selection.SelectColumn
i = 1
For Each oCell In Selection.Cells
    oCell.Range.Text = "Cell " & i
    i = i + 1
Next oCell
```

Returning a single object from a collection

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowReturnSingleItemC"}

Information about returning a single object is available in the object topic itself and in the collection object topic for the collection that contains the object. Most object topics include information about returning a single object as well as adding an object to the collection of those objects. Most collection object topics include information about returning the collection itself and adding an object to the collection.

To browse the Help topics for the objects and collections in Word, see Microsoft Word Objects.

Object doesn't support this property or method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowDoesntSupportC"}

The "object doesn't support this property or method" error occurs when you try to use a method or property that the specified object doesn't support. For example, the following instruction results in an error.

```
ActiveDocument.Copy
```

The **ActiveDocument** property returns a **Document** object. A **Copy** method or property is not available for the **Document** object so an error occurs. To determine what properties and methods are available for an object, do any of the following.

- Use the Object Browser to determine what members (properties and methods) are available for the selected class (object).
- Use the Auto List Members feature in the Visual Basic Editor. When you type a period (.) after a property or method in the Visual Basic Editor, a list of available properties and methods is displayed
- Use Word Visual Basic Help to determine which properties and methods can be used with an object. Each object topic in Help includes a Properties and Methods jump that displays a list of properties and methods for the object. Press F1 in the Object Browser or a module to display the appropriate Help topic.
- Use the **TypeName** function to determine the type of object returned by an expression. The following example displays "Range" because the **Content** property returns a **Range** object.

  ```
  MsgBox TypeName(ActiveDocument.Content)
  ```

Looping through a collection

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowLoopOnCollectionC;vastmForEach"}

There are several different ways you can loop on the elements of a collection. However, the recommended method for looping on a collection is to use the **For Each...Next** loop. In this structure, Visual Basic repeats a block of statements for each object in a collection. The following example displays the name of each document in the **Documents** collection.

```
For Each aDoc In Documents
    MsgBox aDoc.Name
Next aDoc
```

Instead of displaying each element name in a message box, you can use an array to store the information. This example uses the `aMarks()` array to store the name of each bookmark contained in the active document.

```
If ActiveDocument.Bookmarks.Count >= 1 Then
    ReDim aMarks(ActiveDocument.Bookmarks.Count - 1)
    i = 0
    For Each aBookmark In ActiveDocument.Bookmarks
        aMarks(i) = aBookmark.Name
        i = i + 1
    Next aBookmark
End If
```

You can loop through a collection to conditionally perform a task on members of the collection. For example, the following example updates the DATE fields in the active document.

```
For Each aField In ActiveDocument.Fields
    If InStr(1, aField.Code, "Date", 1) Then aField.Update
Next aField
```

You can loop through a collection to determine if an element exists. For example, the following example displays a message if an AutoText entry named "temp" is part of the **AutoTextEntries** collection.

```
For Each aEntry In ActiveDocument.AttachedTemplate.AutoTextEntries
    If aEntry.Name = "temp" Then MsgBox "temp AutoText entry exists"
Next aEntry
```

Prompting for information

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowPromptingForInformationC;vafctInputBox;vafctMsgBox"}

There are several different ways you can prompt for information from a user.

- Use the **InputBox** function to display a dialog box with a message and an edit box. When the user clicks OK, the function returns the text that the user entered.
- Use the **MsgBox** function to display a simple message. You can display several different command buttons and icons. This function returns a number that indicates which button was clicked.
- Use a built-in Word dialog box to get user input for a specific Word feature.
- Use a custom form to get user input. For information on adding controls to a form, see Add a control to a form.

Finding out which property or method to use

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowWhichPropertyC"}

You can use the macro recorder to find out what methods or properties you need to accomplish a task in Word. The macro recorder is a tool that translates your actions into Visual Basic instructions. For example, if you turn on the macro recorder and open a document named Examples.doc, the macro recorder records an instruction similar to the following.

```
Documents.Open FileName:="Examples.doc", ConfirmConversions:=False,
ReadOnly:= _
    False, AddToRecentFiles:=False, PasswordDocument:="",
PasswordTemplate:="", _
    Revert:=False, WritePasswordDocument:="", WritePasswordTemplate:="", _
    Format:=wdOpenFormatAuto
```

The **Documents** property returns the **Documents** collection and the **Open** method opens the specified file name. When you're first learning Visual Basic, using the macro recorder will help you learn which properties and methods you need to use to accomplish a task.

For more information, see the following:

Record a macro in Word

Revising recorded Visual Basic macros

Returning text from a document

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowRetrievingTextC"}

Use the **Text** property to return text from a **Range** or **Selection** object. The following example selects the next paragraph formatted with the Heading 1 style. The contents of the **Text** property are displayed by the **MsgBox** function.

```
With Selection.Find
    .ClearFormatting
    .Style = wdStyleHeading1
    .Execute FindText:="", Format:=True, Forward:=True, Wrap:=wdFindStop
    If .Found = True Then MsgBox Selection.Text
End With
```

The following instruction returns the selected text.

```
strText = Selection.Text
```

The following example returns the first word in the active document. Each item in the **Words** collection is a **Range** object that represents one word.

```
aFirst = ActiveDocument.Words(1).Text
MsgBox aFirst
```

The following example returns the text associated with the first bookmark in the active document.

```
If ActiveDocument.Bookmarks.Count >= 1 Then
    bookText = ActiveDocument.Bookmarks(1).Range.Text
    MsgBox bookText
End If
```

Determining whether the Application property is necessary

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowApplicationNeededC"}

Many of the properties and methods of the **Application** object can be used without the **Application** object qualifier. For example the **ActiveDocument** property can be used without the **Application** object qualifier. Instead of writing `Application.ActiveDocument.PrintOut`, you can write `ActiveDocument.PrintOut`.

Properties and methods that can be used without the **Application** object qualifier are considered "global." To view the global properties and methods in the Object Browser, click **<globals>** at the top of the list in the **Classes** box.

Communicating with other applications

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowCommunicationC;vafctCreateObject;vafctGetObject"}

In addition to working with Word data, you may want your application to exchange data with other applications, such as Microsoft Excel, PowerPoint, or Access. You can communicate with other applications by using Automation (formerly OLE Automation) or dynamic data exchange (DDE).

Tip For detailed information about using Automation to control applications from Word, see the Microsoft Office 97/Visual Basic Programmer's Guide.

Automating Word from another application

Automation allows you to return, edit, and export data by referencing another application's objects, properties, and methods. Application objects that can be referenced by another application are called Automation objects.

The first step toward making Word available to another application for Automation is to make a reference to the Word **Application** object. In Visual Basic, you use the **CreateObject** or **GetObject** function to return a reference to the Word **Application** object. For example, in a Microsoft Excel procedure, you could use the following instruction.

```
Set wrd = CreateObject("Word.Application")
```

This instruction makes the **Application** object in Word available for Automation. Using the objects, properties, and methods of the Word **Application** object, you can control Word. For example, the following instruction creates a new Word document.

```
wrd.Documents.Add
```

The **CreateObject** function starts a Word session that Automation will not close when the object variable that references the **Application** object expires. Setting the object reference to the **Nothing** keyword will not close Word. Instead, use the **Quit** method to close the Word application. The following Microsoft Excel example displays the Word startup path. The **Quit** method is used to close the new instance of Word after the startup path is displayed.

```
Set wrd = CreateObject("Word.Application")
MsgBox wrd.Options.DefaultFilePath(wdStartupPath)
wrd.Quit
```

Automating another application from Word

To exchange data with another application using Automation from Word, you first obtain a reference to the application using the **CreateObject** or **GetObject** function. Then, using the objects, properties, and methods of the other application, you add, change, or delete information. When you finish making your changes, close the application. The following Word example displays the Microsoft Excel startup path. You can use the **Set** statement with the **Nothing** keyword to clear an object variable, which has the same effect as closing the application.

```
Set myobject = CreateObject("Excel.Application")
MsgBox myobject.StartupPath
Set myobject = Nothing
```

For information about referencing an object library and using the objects it exposes, see the Understanding Automation and Set a Reference to a Type Library topics.

Using dynamic data exchange (DDE)

If an application doesn't support Automation, DDE may be an alternative. DDE is a protocol that permits two applications to communicate by continuously and automatically exchanging data through a DDE "channel." To control a DDE conversation between two applications, you establish a channel, select a topic, request and send data, and then close the channel. The following table lists the tasks

that Word performs with DDE and the methods used to control each task in Visual Basic.

Task	Method
Starting DDE	**DDEInitiate**
Getting text from another application	**DDERequest**
Sending text to another application	**DDEPoke**
Carrying out a command in another application	**DDEExecute**
Close DDE channel	**DDETerminate**
Close all DDE channels	**DDETerminateAll**

Visual Basic Equivalents for WordBasic Commands

To find the Visual Basic property or method that's the equivalent of a WordBasic command, click the first letter of the WordBasic command name. Then scroll through the lists of WordBasic commands until you find the appropriate command. The right column includes sample Visual Basic syntax with jumps to topics in the Microsoft Word Visual Basic Help.

For information about converting macros, see Converting WordBasic macros to Visual Basic.

For information about the differences between WordBasic and Visual Basic, see Conceptual differences between WordBasic and Visual Basic.

Visual Basic Equivalents for WordBasic Commands

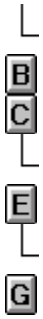
A

Abs(*number*)	Abs(*number*)
Activate *name*	Windows(*name*).Activate ' *or* Documents(*name*).Activate
ActivateObject	ActiveDocument.Shapes(1).OLEFormat.Activate *' or* ActiveDocument.InlineShapes(1).OLEFormat.Activate
AddAddIn	Addins.Add
AddAddress	Application.AddAddress
AddButton	CommandBars(*name*).Controls.Add
AddDropDownItem	ActiveDocument.FormFields(1).DropDown.ListEntries.Add
AddInState	Addins(*name*).Installed = True
state = AddInState(*name*)	state = Addins(*name*).Installed state = Addins(*name*).Compiled state = Addins(*name*).AutoLoad
AllCaps, AllCaps()	Selection.Font.AllCaps = True x = Selection.Font.AllCaps

AnnotationRefFromSel$()	Selection.Comments(1).Reference Text
AOCEAddRecipient	ActiveDocument.Mailer.Recipients = Array(*name*) ActiveDocument.Mailer.CCRecipients = Array(*name*) ActiveDocument.Mailer.BCCRecipients = Array(*name*)
AOCEAuthenticateUser()	x = WordBasic.AOCEAuthenticateUser
AOCEClearMailerField	ActiveDocument.Mailer.Recipients = ""
AOCECountRecipients()	x = Ubound(ActiveDocument.Mailer.Recipients) x = Ubound(ActiveDocument.Mailer.CCRecipients) x = Ubound(ActiveDocument.Mailer.BCCRecipients)
AOCEGetRecipient$()	rec = ActiveDocument.Mailer.Recipients ccRec = ActiveDocument.Mailer.CCRecipients bccRec = ActiveDocument.Mailer.BCCRecipients
AOCEGetSender$()	send = ActiveDocument.Mailer.Sender
AOCEGetSubject$()	sub = ActiveDocument.Mailer.Subject
AOCESendMail	ActiveDocument.SendMailer
AOCESetSubject	ActiveDocument.Mailer.Subject = *text*
AppActivate *name*	Tasks(*name*).Activate
AppClose *name*	Tasks(*name*).Close
AppCount()	Tasks.Count
AppGetNames, AppGetNames()	*' enumerate the Tasks collection* i = 1 For Each aTask In Tasks aArray(i) = aTask.Name i = i + 1 Next aTask
AppHide *name*	Tasks(*name*).Visible = False
AppInfo$(1)	MsgBox System.OperatingSystem & Chr(32) & System.Version
AppInfo$(2)	x = Application.Version
AppInfo$(3)	x = Application.SpecialMode
AppInfo$(4)	x = Application.Left
AppInfo$(5)	x = Application.Top
AppInfo$(6)	x = Application.UsableWidth
AppInfo$(7)	x = Application.UsableHeight
AppInfo$(8)	x = Application.WindowState(wdWindowStateMaximize)
AppInfo$(9)	x = WordBasic.[AppInfo$](9)
AppInfo$(10)	x = WordBasic.[AppInfo$](10)
AppInfo$(13)	x = System.MathCoprocessorInstalled
AppInfo$(14)	x = Application.MouseAvailable
AppInfo$(15)	x = System.FreeDiskSpace
AppInfo$(16)	x = Application.International(wdProductLanguageID)
AppInfo$(17)	x = Application.International(wdListSeparator)
AppInfo$(18)	x = Application.International(wdDecimalSeparator)
AppInfo$(19)	x = Application.International(wdThousandsSeparator)
AppInfo$(20)	x = Application.International(CurrencyCode)
AppInfo$(21)	x = Application.International(wd24HourClock)
AppInfo$(22)	x = Application.International(wdInternationalAM)

AppInfo$(23)	x = Application.International(wdInternationalPM)
AppInfo$(24)	x = Application.International(wdTimeSeparator)
AppInfo$(25)	x = Application.International(wdDateSeparator)
AppInfo$(26)	x = WordBasic.[AppInfo$](26)
AppIsRunning(*name*)	Tasks(*name*).Exists
AppMaximize *name*	Tasks(*name*).WindowState = wdWindowStateMaximize
AppMaximize	Application.WindowState = wdWindowStateMaximize
AppMinimize *name*	Tasks(*name*).WindowState = wdWindowStateMinimize
AppMinimize	Application.WindowState = wdWindowStateMinimize
AppMove *name, horizpos*, v*ertpos*	Tasks(*name*).Move Left:=*horizpos*, Top:=*vertpos*
AppMove *horizpos*, v*ertpos*	Application.Move Left:=*horizpos*, Top:=*vertpos*
AppRestore *name*	Tasks(*name*).WindowState = wdWindowStateNormal
AppRestore	Application.WindowState = wdWindowStateNormal
AppSendMessage	Tasks(*name*).SendWindowMessage
AppShow *name*	Tasks(*name*).Visible = True
AppShow	Application.Visible = True
AppSize *name*, *width*, *height*	Tasks(*name*).Resize Width:=*width*, Height:=*height*
AppSize *width*, *height*	Application.Resize Width:=*width*, Height:=*height*
AppWindowHeight *name*, *height*	Tasks(*name*).Height = *height*
AppWindowHeight *height*	Application.Height = *height*
AppWindowPosLeft *name*, *horizpos*	Tasks(*name*).Left = *horizpos*
AppWindowPosLeft *horizpos*	Application.Left = *horizpos*
AppWindowPosTop *name*, *vertpos*	Tasks(*name*).Top = *vertpos*
AppWindowPosTop *vertpos*	Application.Top = *vertpos*
AppWindowWidth *name*, *width*	Tasks(*name*).Width = *width*
AppWindowWidth *width*	Application.Width = *width*
Asc(*string*)	Asc(*string*)
AtEndOfDocument()	If Selection.Type = wdSelectionIP and Selection.End = ActiveDocument. atEnd = True
AtStartOfDocument()	If Selection.Type = wdSelectionIP and Selection.Start = 0 Then atStart =
AutoMarkIndexEntries	ActiveDocument.Indexes.AutoMarkEntries
AutomaticChange	Application.AutomaticChange
AutoText	Selection.Range.InsertAutoText
AutoTextName$(*num, context*)	x = ActiveDocument.AttachedTemplate.AutoTextEntries(*num*).Name

Visual Basic Equivalents for WordBasic Commands



B

Beep	Beep
Begin Dialog...End Dialog	Create and display a custom form. For information about adding controls control to a form topic.
Bold, Bold()	Selection.Font.Bold = True x = Selection.Font.Bold
name = BookmarkName$(*num*)	*name* = ActiveDocument.Bookmarks(*num*).Name
BorderBottom, BorderBottom()	With ActiveDocument.Paragraphs(1).Borders(wdBorderBottom) .LineStyle = wdLineStyleSingle .LineWidth= wdLineWidth075pt End With x = ActiveDocument.Paragraphs(1).Borders(wdBorderBottom).LineStyle
BorderInside, BorderInside()	With Selection.Paragraphs.Borders .InsideLineStyle = wdLineStyleSingle .InsideLineWidth = wdLineWidth075pt End With x = Selection.Paragraphs.Borders.InsideLineStyle
BorderLeft, BorderLeft()	With ActiveDocument.Paragraphs(1).Borders(wdBorderLeft) .LineStyle = wdLineStyleSingle .LineWidth= wdLineWidth075pt End With x = ActiveDocument.Paragraphs(1).Borders(wdBorderLeft).LineStyle
BorderLineStyle, BorderLineStyle()	Selection.Borders(*wdBorderType*).LineStyle = *wdLineStyle* x = ActiveDocument.Paragraphs(1).Borders(*wdBorderType*).LineStyle
BorderNone, BorderNone()	Selection.Range.Borders.Enable = False *' or* Selection.Range.Borders(wdBorderLeft).LineStyle = wdLineStyleNone x = Selection.Range.Borders.Enable
BorderOutside, BorderOutside()	With Selection.Paragraphs.Borders

	.OutsideLineStyle = wdLineStyleSingle .OutsideLineWidth = wdLineWidth075pt End With x = Selection.Paragraphs.Borders.OutsideLineStyle
BorderRight, BorderRight()	With ActiveDocument.Paragraphs(1).Borders(wdBorderRight) .LineStyle = wdLineStyleSingle .LineWidth= wdLineWidth075pt End With x = ActiveDocument.Paragraphs(1).Borders(wdBorderRight).LineStyle
BorderTop, BorderTop()	With Selection.Paragraphs(1).Borders(wdBorderTop) .LineStyle = wdLineStyleSingle .LineWidth= wdLineWidth075pt End With x = ActiveDocument.Paragraphs(1).Borders(wdBorderTop).LineStyle

Visual Basic Equivalents for WordBasic Commands

C

Call	Call
Cancel	Selection.ColumnSelectMode = False Selection.ExtendMode = False

	Selection.EscapeKey
CancelButton	CommandButton
CenterPara, CenterPara()	Selection.Paragraphs.Alignment = wdAlignParagraphCenter
	x = Selection.Paragraphs.Alignment
ChangeCase, ChangeCase()	Selection.Range.Case = *WdCharacterCase*
	x = Selection.Range.Case
CharColor, CharColor()	Selection.Font.ColorIndex = *WdColorIndex*
	x = Selection.Font.ColorIndex
CharLeft 1	Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdMove
CharLeft 1, 1	Selection.MoveLeft Unit:=wdCharacter, Count:=1, Extend:=wdExtend
num = CharLeft(1)	*num* = Selection.MoveLeft(Unit:=wdCharacter, Count:=1)
CharRight 1	Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdMove
CharRight 1, 1	Selection.MoveRight Unit:=wdCharacter, Count:=1, Extend:=wdExtend
num = CharRight(1)	*num* = Selection.MoveRight(Unit:=wdCharacter, Count:=1)
ChDefaultDir *path*, *wdDefaultFilePath*	Options.DefaultFilePath(*WdDefaultFilePath*) = *path*
ChDir *path*	ChDir *path*
	' or
	Application.ChangeFileOpenDirectory
CheckBox	CheckBox control
CheckBoxFormField	ActiveDocument.Fields.Add Range:=*range*, Type:=wdFieldFormCheckB
ChooseButtonImage	With CommandBars(*name*).Controls(1) .FaceId = *num* .TooltipText = *text* End With
Chr$(*num*)	Chr(*num*)
CleanString$(*string*)	x = CleanString(*string*)
ClearAddIns*t*	Addins.Unload
ClearFormField	ActiveDocument.FormFields(1).TextInput.Clear
Close	Close
ClosePane	ActiveWindow.ActivePane.Close
ClosePreview	ActiveDocument.ClosePrintPreview
CloseUpPara	Selection.Paragraphs.CloseUp
CloseViewHeaderFooter	ActiveWindow.View.SeekView = wdSeekMainDocument
CmpBookmarks()	If ActiveDocument.Bookmarks(*name*) = ActiveDocument.Bookmarks(*nar* same = True End If
	The Start and End properties can be used to compare the starting and e bookmarks.
ColumnSelect	Selection.ColumnSelectMode = True
ComboBox	ComboBox control
CommandValid()	Use the IsObjectValid property to determine if an object variable referenc variable that returns Nothing is not valid.
Connect	System.Connect
ControlRun	WordBasic.ControlRun
	' or
	Shell *appfilename*
Converter$(*num*)	x = FileConverters(*num*).ClassName

ConverterLookup(*name*)	x = FileConverters(*name*).SaveFormat
ConvertObject *IconNumber*, *ActivateAs*, *IconFileName*, *Caption*, Class, *DisplayIcon*	With ActiveDocument.Shapes(1).OLEFormat .IconIndex = *num* .ActivateAs = *text* .IconPath & Application.PathSeparator & .IconName .IconLabel = *text* .ClassType = *text* .DisplayAsIcon = True End With
CopyBookmark	ActiveDocument.Bookmarks(*name*).Copy(*name*)
CopyButtonImage	CommandBars(*name*).Controls(1).CopyFace
CopyFile	FileCopy
CopyFormat	Selection.CopyFormat
CopyText	Application.Run MacroName:="CopyText"
CountAddins()	x = Addins.Count
CountAutoCorrectExceptions(0)	x = AutoCorrect.FirstLetterExceptions.Count
CountAutoCorrectExceptions(1)	x = AutoCorrect.TwoInitialCapsExceptions.Count
CountAutoTextEntries()	x = ActiveDocument.AttachedTemplate.AutoTextEntries.Count
CountBookmarks()	x = ActiveDocument.Bookmarks.Count

CountDirectories()	myPath = "C:\"
	myName = Dir(myPath, vbDirectory)
	Do While myName <> ""
	If myName <> "." And myName <> ".." Then
	If (GetAttr(myPath & myName) And vbDirectory) = vbDirectory T
	count = count + 1
	End If
	End If
	myName = Dir
	Loop
	MsgBox count & " directories"

CountDocumentProperties()	x = ActiveDocument.BuiltInDocumentProperties.Count *' or* x = ActiveDocument.CustomDocumentProperties.Count
CountDocumentVars()	x = ActiveDocument.Variables.Count
CountFiles()	x = RecentFiles.Count
CountFonts()	x = FontNames.Count *' or* x = PortraitFontNames.Count *' or* x = LandscapeFontNames.Count
CountFoundFiles()	x = Application.FileSearch.FoundFiles.Count
CountKeys()	CustomizationContext = *template or document* x = KeyBindings.Count
CountLanguages()	x = Languages.Count
CountMacros()	*' no direct equivalent* *' counts the number of modules associated with the normal template* For Each xItem In NormalTemplate.VBProject.VBComponents If xItem.Type = vbext_ct_StdModule Then Count = Count + 1

	Next x MsgBox Count
CountMenuItems()	x = CommandBars(*name*).Controls.Count
CountMenus()	x = CommandBars.ActiveMenuBar.Controls.Count
CountMergeFields()	x = ActiveDocument.MailMerge.Fields.Count
CountStyles()	x = ActiveDocument.Styles.Count *' or* x = ActiveDocument.AttachedTemplate.OpenAsDocument.Styles.Count *' to exclude built-in styles from the count* For Each xSty In ActiveDocument.Styles If xSty.BuiltIn = False Then aCount = aCount + 1 Next xSty
CountToolbarButtons()	x = CommandBars(*name*).Controls.Count
CountToolbars()	For Each xCB In CommandBars If xCB.Type = msoBarTypeNormal Then aCount = aCount + 1 Next xCB
CountToolsGrammarStatistics()	x = ActiveDocument.Content.ReadabilityStatistics.Count
CountWindows()	x = Windows.Count
CreateSubdocument	ActiveDocument.Subdocuments.AddFromRange

Visual Basic Equivalents for WordBasic Commands

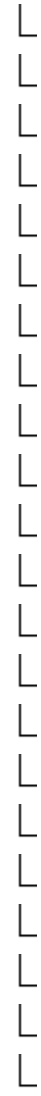
D

Date$()	Date *' or* Date$
DateSerial()	DateSerial
DateValue()	DateValue
Day()	Day
Days360()	DateDiff
DDEExecute *channel*, *command*	DDEExecute *channel*, *command*
chan = DDEInitiate (*application*, *topic*)	*chan* = DDEInitiate(*application*, *topic*)
DDEPoke *channel*, *item, data*	DDEPoke *channel*, *item, data*
data = DDERequest$(*channel*, *item*)	*data* = DDERequest(*channel*, *item*)
DDETerminate *channel*	DDETerminate *channel*
DDETerminateAll	DDETerminateAll
Declare	Declare
DefaultDir$()	x = DefaultFilePath(*WdDefaultFilePath*)
DeleteAddIn *name*	Addins(*name*).Delete
DeleteBackWord	Selection.Delete Unit:=wdWord, Count:=-1
DeleteButton	CommandBars(*name*).Controls(*num*).Delete
DeleteDocumentProperty *name*	ActiveDocument.CustomDocumentProperties(*name*).Delete
DeleteWord	Selection.Words(1).Delete
DemoteList	Selection.Range.ListFormat.ListOutdent
DemoteToBodyList	Selection.Paragraphs(1).OutlineDemoteToBody
Dialog, Dialog()	Dialogs(*WdWordDialog*).Show
DialogEditor	ShowVisualBasicEditor = True
Dim	Dim
DisableAutoMacros	WordBasic.DisableAutoMacros
DisableInput	Application.EnableCancelKey = *WdEnableCancelKey*
DlgControlId()	WordBasic dynamic dialog functionality has been replaced by custom us topics in Microsoft Forms Help (FM20.HLP>LANGREF).
DlgEnable, DlgEnable()	
DlgFilePreview, DlgFilePreview$()	
DlgFocus, DlgFocus$()	
DlgListBoxArray, DlgListBoxArray()	
DlgLoadValues, DlgLoadValues()	
DlgSetPicture	
DlgStoreValues	
DlgText, DlgText$()	
DlgUpdateFilePreview	
DlgValue, DlgValue()	
DlgVisible, DlgVisible()	
DocClose	ActiveWindow.Close
DocMaximize, DocMaximize()	ActiveWindow.WindowState = wdWindowStateMaximize

DocMinimize, DocMinimize()	ActiveWindow.WindowState = wdWindowStateMinimize
DocMove *HorizPos*, *VertPos*	With ActiveWindow .Top = *VertPos* .Left = *HorizPos* End With
DocRestore	ActiveWindow.WindowState = wdWindowStateNormal
DocSize *width*, *height*	With ActiveWindow .Height = *width* .Width = *height* End With
DocSplit, DocSplit()	ActiveWindow.SplitVertical = 50 x = ActiveWindow.SplitVertical
DocumentHasMisspellings()	x = ActiveDocument.SpellingErrors.Count
DocumentPropertyExists()	*' enumerate the DocumentProperties collection* For Each aProp In ActiveDocument.CustomDocumentProperties If aProp.Name = *name* Then itExists = True Next aProp
DocumentPropertyName$()	x = ActiveDocument.CustomDocumentProperties(*num*).Name ' *or* x = ActiveDocument.BuiltInDocumentProperties(*num*).Name
DocumentPropertyType()	x = ActiveDocument.CustomDocumentProperties(*name*).Type
DocumentProtection()	x = ActiveDocument.ProtectionType
DocumentStatistics *FileName*, *Directory*, *Template*, *Title*, *Created*, *LastSaved*, *LastSavedBy*, *Revision*, *Time*, *Printed*, *Pages*, *Words*, *Characters*, *Paragraphs*, *Lines*, *FileSize*	With ActiveDocument var1 = .Name var2 = .Path var3 = .BuiltInDocumentProperties(wdPropertyTemplate) var4 = .BuiltInDocumentProperties(wdPropertyTitle) var5 = .BuiltInDocumentProperties(wdPropertyTimeCreated) var6 = .BuiltInDocumentProperties(wdPropertyTimeLastSaved) var7 = .BuiltInDocumentProperties(wdPropertyLastAuthor) var8 = .BuiltInDocumentProperties(wdPropertyRevision) var9 = .BuiltInDocumentProperties(wdPropertyVBATotalEdit) var10 = .BuiltInDocumentProperties(wdPropertyTimeLastPrinted) var11 = .BuiltInDocumentProperties(wdPropertyPages) var12 = .BuiltInDocumentProperties(wdPropertyWords) var13 = .BuiltInDocumentProperties(wdPropertyCharacters) var14 = .BuiltInDocumentProperties(wdPropertyParas) var15 = .BuiltInDocumentProperties(wdPropertyLines) var16 = .BuiltInDocumentProperties(wdPropertyBytes) End With
DocWindowHeight	ActiveWindow.Height = *height*
DocWindowPosLeft	ActiveWindow.Left = *horizpos*
DocWindowPosTop	ActiveWindow.Top = *vertpos*
DocWindowWidth	ActiveWindow.Width = *width*
DoFieldClick	Selection.Fields(1).DoClick
DOSToWin$()	x = WordBasic.[DOSToWin$](*StringToTranslate*)
DottedUnderline, DottedUnderline()	Selection.Font.UnderLine= wdUnderlineDotted x = Selection.Font.UnderLine
DoubleUnderline, DoubleUnderline()	Selection.Font.UnderLine = wdUnderlineDouble x = Selection.Font.UnderLine

Drawing object statements and functions	Use the properties and methods of the following objects: Shape, Shapes
DropDownFormField	ActiveDocument.Fields.Add Range:=*range*, Type:=wdFieldFormDropDo
DropListBox	ComboBox control

Visual Basic Equivalents for WordBasic Commands



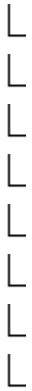
E

EditAutoText .Name= *name*, .Add	ActiveDocument.AttachedTemplate.AutoTextEntries.Add
EditAutoText .Name= *name*, .InsertAs = 0, .Insert	ActiveDocument.AttachedTemplate.AutoTextEntries(*name*).Insert Where
EditAutoText .Name= *name*, .Delete	Templates(*name*).AutoTextEntries(*name*).Delete
EditBookmark .Name = *name*, .Add	ActiveDocument.Bookmarks.Add Name:=*name*, Range:=*range*
EditBookmark .Name = *name*, .Delete	ActiveDocument.Bookmarks(*name*).Delete
EditBookmark .Name = *name*, .Goto	ActiveDocument.Bookmarks(*name*).Select
EditBookmark .Name = *name*, .SortBy	ActiveDocument.Bookmarks.DefaultSorting = wdSortByName

EditButtonImage	WordBasic.EditButtonImage
EditClear	Selection.Range.Delete
EditConvertAllEndnotes	ActiveDocument.Endnotes.Convert
EditConvertAllFootnotes	ActiveDocument.Footnotes.Convert
EditConvertNotes	Selection.Footnotes.Convert *' or* Selection.Endnotes.Convert
EditCopy	Selection.Range.Copy
EditCopyAsPicture	Selection.Range.CopyAsPicture
EditCreatePublisher	Selection.Range.CreatePublisher
EditCut	Selection.Range.Cut
EditFind	Selection.Find
EditFindBorder	Selection.Find.Borders
EditFindClearFormatting	Selection.Find.ClearFormatting
EditFindFont	Selection.Find.Font
EditFindFound()	Selection.Find.Found
EditFindFrame	Selection.Find.Frame
EditFindHighlight	Selection.Find.Highlight = True
EditFindLang	Selection.Find.LanguageID
EditFindNotHighlight	Selection.Find.Highlight = False
EditFindPara	Selection.Find.ParagraphFormat
EditFindStyle	Selection.Find.Style
EditFindTabs	Seleciton.Find.ParagraphFormat.TabStops
EditGoTo	Selection.Goto
EditLinks *UpdateMode*, *Locked*, *SavePictureInDoc*, *UpdateNow*, *OpenSource*, *KillLink*, *Link*, *Application*, *Item*, *FileName*	ActiveDocument.Shapes(1).OLEFormat.Open *' or* ActiveDocument.InlineShapes(1).OLEFormat.Open With ActiveDocument.InlineShapes(1).LinkFormat .AutoUpdate = True .Locked = True .SavePictureWithDocument = True .Update .BreakLink .Application.Name .SourceFullName End With
EditObject	Selection.InlineShapes(1).OLEFormat.Edit
EditPaste	Selection.Range.Paste
EditPasteSpecial	Selection.Range.PasteSpecial
EditPicture	Selection.ShapeRange(1).Activate
EditPublishOptions	ActiveDocument.EditionOptions
EditRedo	ActiveDocument.Redo
EditRepeat	Repeat
EditReplaceBorder	Selection.Find.Replacement.Borders
EditReplaceClearFormatting	Selection.Find.Replacement.ClearFormatting
EditReplaceFont	Selection.Find.Replacement.Font
EditReplaceFrame	Selection.Find.Replacement.Frame

EditReplaceHighlight	Selection.Find.Replacement.Highlight = True
EditReplaceLang	Selection.Find.Replacement.LanguageID
EditReplaceNotHighlight	Selection.Find.Replacement.Highlight = False
EditReplacePara	Selection.Find.Replacement.ParagraphFormat
EditReplaceStyle	Selection.Find.Replacement.Style
EditReplaceTabs	Selection.Find.Replacement.ParagraphFormat.TabStops
EditSelectAll	ActiveDocument.Content.Select
EditSubscribeOptions	ActiveDocument.EditionOptions
EditSubscribeTo	Selection.Range.SubscribeTo
EditSwapAllNotes	ActiveDocument.Endnotes.SwapWithFootnotes ' *or* ActiveDocument.Footnotes.SwapWithEndnotes
EditTOACategory	ActiveDocument.TablesOfAuthoritiesCategories(*num*).Name = *name*
EditUndo	ActiveDocument.Undo
EmptyBookmark(*name*)	x = ActiveDocument.Bookmarks(*name*).Empty
EnableFormField	ActiveDocument.FormFields(1).Enabled = True
EndOfColumn, EndOfColumn()	Selection.EndOf Unit:=wdColumn, Extend:=wdMove
EndOfDocument, EndOfDocument()	Selection.EndKey Unit:=wdStory
EndOfLine, EndOfLine()	Selection.EndKey Unit:=wdLine, Extend:=wdMove
EndOfRow, EndOfRow()	Selection.EndKey Unit:=wdRow, Extend:=wdMove
EndOfWindow, EndOfWindow()	Application.Run MacroName:="EndOfWindow"
Environ$()	Environ$()
Eof()	EOF()
Err	Err
Error	Error
ExistingBookmark(*name*)	x = ActiveDocument.Bookmarks.Exists(*name*)
ExitWindows	Tasks.ExitWindows
ExtendMode()	x= Selection.ExtendMode
ExtendSelection	*' activates extend mode* Selection.ExtendMode = True *' extends the selection* Selection.Expand Unit:=*wdUnits*

Visual Basic Equivalents for WordBasic Commands



F

FieldSeparator$	Application.DefaultTableSeparator
FileAOCEAddMailer	ActiveDocument.HasMailer = True
FileAOCEDeleteMailer	ActiveDocument.HasMailer = False
FileAOCEExpandMailer	Macintosh only
FileAOCEForwardMail	ActiveDocument.ForwardMailer
FileAOCENextLetter	Application.NextLetter
FileAOCEReplyAllMail	Macintosh only
FileAOCEReplyMail	Macintosh only
FileAOCESendMail	ActiveDocument.SendMailer
FileClose	ActiveDocument.Close
FileCloseAll	Documents.Close
FileClosePicture	ActiveDocument.Close
FileConfirmConversions	Options.ConfirmConversions = True
FileCreator$()	Macintosh only
FileDocumentLayout	Macintosh only
FileExit	Application.Quit
FileFind	Application.FileSearch
FileList *number*	RecentFiles(*num*).Open
FileMacCustomPageSetupGX	Macintosh only
FileMacPageSetup	Macintosh only
FileMacPageSetupGX	Macintosh only
FileName$()	x = ActiveDocment.FullName
FileName$(*num*)	x = RecentFiles(*num*).Name & Application.PathSeparator & RecentFiles
FileNameFromWindow$()	x = Windows(1).Document.FullName

FileNameInfo$()	x = WordBasic.[FileNameInfo$]() x = ActiveDocument.Name x = ActiveDocument.Path x = ActiveDocument.FullName
FileNew *Template*	Documents.Add Template:=*filename*
FileNewDefault	Documents.Add
File *num*	RecentFiles(*num*).Open
FileOpen	Documents.Open
FilePost	ActiveDocument.Post
FilePageSetup *Tab*, *TopMargin*, *BottomMargin*, *LeftMargin*, *RightMargin*, *Gutter*, *PageWidth*, *PageHeight*, *Orientation*, *FirstPage*, *OtherPages*, *VertAlign*, *ApplyPropsTo*, *Default*, *FacingPages*, *HeaderDistance*, *FooterDistance*, *SectionStart*, *OddAndEvenPages*, *DifferentFirstPage*, *Endnotes*, *LineNum*, *StartingNum*, *FromText*, *CountBy*, *NumMode*	With ActiveDocument.PageSetup .TopMargin = *num* .BottomMargin = *num* .LeftMargin = *num* .RightMargin = *num* .Gutter = *num* .PageHeight = *num* .PageWidth = *num* .Orientation = *WdOrientation* .FirstPageTray = *WdPaperTray* .OtherPagesTray = *WdPaperTray* .VerticalAlignment = *WdVerticalAlignment* .SetAsTemplateDefault .MirrorMargins = True .HeaderDistance = *num* .FooterDistance = *num* .SectionStart = *WdSectionStart* .OddAndEvenPagesHeaderFooter = True .DifferentFirstPageHeaderFooter = True .SuppressEndnotes = True With LineNumbering .Active =True .StartingNumber = *num* .DistanceFromText = *num* .CountBy = *num* .RestartMode = *WdNumberingRule* End With End With
FilePreview	Image control
FilePrint	ActiveDocument.PrintOut
FilePrintDefault	ActiveDocument.PrintOut
FilePrintOneCopy	Macintosh only
FilePrintPreview, FilePrintPreview()	PrintPreview = True x = PrintPreview
FilePrintPreviewFullScreen	PrintPreview = True ActiveWindow.View.FullScreen = True
FilePrintPreviewPages, FilePrintPreviewPages()	With ActiveWindow.View.Zoom .PageColumns = 2 .PageRows = 1 End With
FilePrintSetup	ActivePrinter
FileProperties	WordBasic.FileProperties

FileQuit	Application.Quit
FileRoutingSlip *Subject*, *Message*, *AllAtOnce*, *ReturnWhenDone*, *TrackStatus*, *Protect*	With ActiveDocument.RoutingSlip .Subject = text .Message = text .Delivery = WdRoutingSlipDelivery .ReturnWhenDone = True .TrackStatus = True .Protect = *WdProtectionType* End With
FileRoutingSlip *AddSlip*	ActiveDocument.HasRoutingSlip = True
FileRoutingSlip *RouteDocument*	ActiveDocument.Route
FileRoutingSlip *AddRecipient*	ActiveDocument.RoutingSlip.AddRecipient
FileRoutingSlip *ResetSlip*	ActiveDocument.RoutingSlip.Reset
FileRoutingSlip *ClearSlip*	ActiveDocument.HasRoutingSlip = False
Files$()	Dir()
FileSave	ActiveDocument.Save
FileSaveAll	Documents.Save
FileSaveAs	ActiveDocument.SaveAs
FileSendMail	ActiveDocument.SendMail
FileSummaryInfo *Title*, *Subject*, *Author*, *Keywords*, *Comments*, *FileName*, Directory, *Template*, *CreateDate*, *LastSavedDate*, *LastSavedBy*, *RevisionNumber*, *EditTime*, LastPrintedDate, *NumPages*, *NumWords*, *NumChars*, *NumParas*, *NumLines*, *FileSize*	With ActiveDocument .BuiltInDocumentProperties(wdPropertyTitle) .BuiltInDocumentProperties(wdPropertySubject) .BuiltInDocumentProperties(wdPropertyLastAuthor) .BuiltInDocumentProperties(wdPropertyKeywords) .BuiltInDocumentProperties(wdPropertyComments) .Name .Path .BuiltInDocumentProperties(wdPropertyTemplate) .BuiltInDocumentProperties(wdPropertyTimeCreated) .BuiltInDocumentProperties(wdPropertyTimeLastSaved) .BuiltInDocumentProperties(wdPropertyLastAuthor) .BuiltInDocumentProperties(wdPropertyRevision) .BuiltInDocumentProperties(wdPropertyVBATotalEdit) .BuiltInDocumentProperties(wdPropertyTimeLastPrinted) .BuiltInDocumentProperties(wdPropertyPages) .BuiltInDocumentProperties(wdPropertyWords) .BuiltInDocumentProperties(wdPropertyCharacters) .BuiltInDocumentProperties(wdPropertyParas) .BuiltInDocumentProperties(wdPropertyLines) .BuiltInDocumentProperties(wdPropertyBytes) End With
FileTemplates	ActiveDocument.AttachedTemplate = *template*
FileType$()	Macintosh only
Font, Font$()	Selection.Font.Name = *text* x = Selection.Font.Name
FontSize, FontSize()	Selection.Font.Size = *num* x = Selection.Font.Size
FontSizeSelect	Application.Run MacroName:="FontSizeSelect"
FontSubstitution	Application.SubstituteFont
For...Next	For...Next
FormatAddrFonts	*' Set properties of the Font object*

	With ActiveDocument.Envelope.Address.Font .Size = *num* .ColorIndex = *WdColorIndex* .Bold = True End With
FormatAutoFormat	ActiveDocument.AutoFormat
FormatBordersAndShading *Shadow*, *TopBorder*, *LeftBorder*, *BottomBorder*, *RightBorder*, *HorizBorder*, *VertBorder*, *TopColor*, *LeftColor*, *BottomColor*, *RightColor*, *HorizColor*, *VertColor*, *FineShading*, *FromText*, *Shading*, *Foreground*, *Background*, *Tab*	With ActiveDocument.Paragraphs(1).Borders .Shadow = True .DistanceFromBottom = *num* .DistanceFromTop = *num* .DistanceFromLeft = *num* .DistanceFromRight = *num* End With With Selection.Paragraphs.Shading .Texture = *WdTextureIndex* .BackgroundPatternColorIndex = *WdColorIndex* .ForegroundPatternColorIndex = *WdColorIndex* End With With ActiveDocument.Paragraphs(1) .Borders(*WdBorderType*).LineStyle = *WdLineStyle* .Borders(*WdBorderType*).LineWidth = *WdLineWidth* .Borders(*WdBorderType*).ColorIndex = *WdColorIndex* End With With Dialogs(wdDialogFormatBordersAndShading) .DefaultTab = *WdWordDialogTab* .Show End With
FormatBullet *Points*, *Color*, *Alignment*, *Indent*, *Space*, *Hang*, *CharNum*, *Font*	With ListGalleries(wdBulletGallery).ListTemplates(1).ListLevels(1) .NumberFormat = ChrW(*num*) .NumberStyle = wdListNumberStyleBullet .NumberPosition = *num* .Alignment = *WdListLevelAlignment* .TextPosition = *num* .TabPosition = *num* With .Font .Size = *num* .Name = *text* .ColorIndex = *WdColorIndex* End With End With
FormatBulletDefault, FormatBulletDefault()	Selection.Range.ListFormat.ApplyBulletDefault Selection.Range.ListFormat.RemoveNumbers
FormatBulletsAndNumbering *Remove*, *Hang*, *Preset*	Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=ListGalleries(*WdListGalleryType*).ListTemplates(*num*) Selection.Range.ListFormat.RemoveNumbers
FormatCallout Type, Gap, Angle, Drop, Length, Border, AutoAttach, Accent	With ActiveDocument.Shapes(1).Callout .Type = *MsoCalloutType* .Gap = *num* .Angle = *MsoCalloutAngleType* .Drop = *num* .DropType = *MsoCalloutDropType* .Length = *num* .Border = *MsoTriState*

	.AutoAttach = *MsoTriState* .Accent = *MsoTriState* End With
FormatChangeCase	Selection.Range.Case = *WdCharacterCase*
FormatColumns *Columns*, *ColumnWidth*, *ColumnSpacing*, *EvenlySpaced*, *ColLine*	With ActiveDocument.TextColumns .SetCount NumColumns:=*num* .Width = *num* .Spacing = *num* .EvenlySpaced = False .LineBetween = False End With
FormatDefineStyleBorders	*' Set properties of the* Borders *object* With ActiveDocument.Styles(*name*).Borders .Enable = True .Shadow = True End With
FormatDefineStyleFont	*' Set properties of the* Font *object* With ActiveDocument.Styles(*name*).Font .Bold = True .Name = "Arial" End With
FormatDefineStyleFrame	*' Set properties of the* Frame *object* With ActiveDocument.Styles(*name*).Frame .Width = *num* .VerticalPosition = *num* End With
FormatDefineStyleLang	ActiveDocument.Styles(*name*).LanguageID = *WdLanguageID*
FormatDefineStyleNumbers	*' Set properties of the* ListLevel *object* With ActiveDocument.Styles(*name*). ListGalleries(*WdListGalleryType*).ListTemplates(*num*).ListLevels(*num*) .NumberFormat = "%1)" .TrailingCharacter = wdTrailingTab .NumberStyle = wdListNumberStyleArabic End With
FormatDefineStylePara	*' Set properties of the* ParagraphFormat *object* With ActiveDocument.Styles(*name*).ParagraphFormat .SpaceAfter = *num* .RightIndent = *num* End With
FormatDefineStyleTabs	*' Set properties of the* TabStops *object* ActiveDocument.Styles(*name*).ParagraphFormat.TabStops(1).Leader =
FormatDrawingObject	Set properties of the Shape object.
FormatDropCap *Position*, *Font*, *DropHeight*, *DistFromText*	With ActiveDocument.Paragraphs(1).DropCap .Position = *WdDropPosition* .FontName = *text* .LinesToDrop = *num* .DistanceFromText = *num* End With
FormatFont *Points*, *Underline*, *Color*, *Strikethrough*, *Superscript*, *Subscript*, *Shadow*, *Hidden*, *SmallCaps*, *AllCaps*, *Outline*, *Spacing*, *Position*, *Kerning*, *KerningMin*, *Default*, *Tab*, *Font*, *Bold*, *Italic*	With Selection.Font .Size = *num* .Underline = True .ColorIndex = *WdColorIndex* .StrikeThrough = True

```
        .Superscript = num
        .Subscript = num
        .Shadow = True
        .Hidden = True
        .SmallCaps = True
        .AllCaps = True
        .Outline = True
        .Spacing = num
        .Position = num
        .Kerning = num
        .SetAsTemplateDefault
        .Font = name
        .Bold = True
        .Italic = True
    End With
```

```
    With Dialogs(wdDialogFormatFont)
        .DefaultTab = WdWordDialogTab
        .Show
    End With
```

FormatFrame *Wrap*, *WidthRule*, *FixedWidth*, *HeightRule*, *FixedHeight*, *PositionHorz*, *PositionHorzRel*, *DistFromText*, *PositionVert*, *PositionVertRel*, *DistVertFromText*, *MoveWithText*, *LockAnchor*, *RemoveFrame*

```
    With ActiveDocument.Frames(1)
        .TextWrap = True
        .WidthRule = WdFrameSizeRule
        .Width = num
        .Height = num
        .HeightRule = WdFrameSizeRule
        .HorizontalPosition = num
        .RelativeHorizontalPosition = WdRelativeHorizontalPosition
        .HorizontalDistanceFromText = num
        .VerticalPosition = num
        .RelativeVerticalPosition = WdRelativeVerticalPosition
        .VerticalDistanceFromText = num
        .LockAnchor = True
        .Delete
    End With
```

FormatHeaderFooterLink

```
    ActiveDocument.Sections(num).Headers(WdHeaderFooterIndex).LinkTo
    ' or
    ActiveDocument.Sections(num).Footers(WdHeaderFooterIndex).LinkToP
```

FormatHeadingNumber

```
    With ListGalleries(WdListGalleryType).ListTemplates(num).ListLevels(nu
        ' Set properties of the ListLevel object and use the ApplyListTemplat
    End With
```

FormatHeadingNumbering

```
    ' Set properties of the ListLevel object and use the ApplyListTemplate me
    Set atemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(num
    With atemp.ListLevels(1)
        .NumberFormat = "Chapter %1"
        .TrailingCharacter = wdTrailingNone
        .NumberStyle = wdListNumberStyleArabic
    End With
    Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=atemp
```

FormatMultilevel

```
    ' Set properties of the ListLevel object and use the ApplyListTemplate me
    Set atemp = ListGalleries(wdOutlineNumberGallery).ListTemplates(num
    atemp.ListLevels(1).NumberStyle = wdListNumberStyleLowercaseLetter
    Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=atemp
```

FormatNumber

```
    ' Set properties of the ListLevel object and use the ApplyListTemplate me
    Set atemp = ListGalleries(wdNumberGallery).ListTemplates(num)
```

	With atemp.ListLevels(1) .NumberFormat = "%1." .TrailingCharacter = wdTrailingTab .NumberStyle = wdListNumberStyleArabic End With Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=atemp
FormatNumberDefault, FormatNumberDefault()	Selection.Range.ListFormat.ApplyNumberDefault Selection.Range.ListFormat.RemoveNumbers
FormatPageNumber *ChapterNumber*, *NumRestart*, *NumFormat*, *StartingNum*, *Level*, *Separator*	With *Section*.Footers(wdHeaderFooterPrimary).PageNumbers .IncludeChapterNumber = True .RestartNumberingAtSection = True .NumberStyle = *WdPageNumberStyle* .StartingNumber = *num* .HeadingLevelForChapter = *num* .ChapterPageSeparator = *WdSeparatorType* End With
FormatParagraph *LeftIndent*, *RightIndent*, *Before*, *After*, *LineSpacingRule*, *LineSpacing*, *Alignment*, *WidowControl*, *KeepWithNext*, *KeepTogether*, *PageBreak*, *NoLineNum*, *DontHyphen*, *Tab*, *FirstIndent*	With ActiveDocument.Paragraphs(1) .LeftIndent = *num* .RightIndent = *num* .SpaceBefore = *num* .SpaceAfter = *num* .LineSpacingRule = *WdLineSpacing* .LineSpacing = *num* .Alignment = *WdParagraphAlignment* .WidowControl = True .KeepWithNext = True .KeepTogether = True .PageBreakBefore = True .NoLineNumber = True .Hyphenation = True .FirstLineIndent = *num* End With With Dialogs(wdDialogFormatParagraph) .DefaultTab = *WdWordDialogTab* .Show End With
FormatPicture *SetSize*, *CropLeft*, *CropRight*, *CropTop*, *CropBottom*, *ScaleX*, *ScaleY*, *SizeX*, *SizeY*	With ActiveDoucment.InlineShapes(1) .Width = *num* .Height = *num* .ScaleHeight = *num* .ScaleWidth = *num* With .PictureFormat .CropBottom = *num* .CropLeft = *num* .CropRight = *num* .CropTop = *num* End With End With
FormatRetAddrFonts	*' Set properties of the* Font *object* With ActiveDocument.Envelope.ReturnAddress.Font .Size = *num* .ColorIndex = *WdColorIndex* .Bold = True End With

FormatSectionLayout *SectionStart*, *VertAlign*, *Endnotes*, *LineNum*, *StartingNum*, *FromText*, *CountBy*, *NumMode*	With ActiveDocument.PageSetup .VerticalAlignment = *WdVerticalAlignment* .SectionStart = *WdSectionStart* .SuppressEndnotes = True With LineNumbering .Active =True .StartingNumber = *num* .DistanceFromText = *num* .CountBy = *num* .RestartMode = *WdNumberingRule* End With End With
FormatStyle *Name*, *Delete*, *Merge*, *NewName*, *BasedOn*, *NextStyle*, *Type*, *FileName*, *Source*, *AddToTemplate*, *Define*, *Rename*, *Apply*	With ActiveDocument.Styles(*name*) .Delete .NameLocal = *name* .BaseStyle = *text* .NextParagraphStyle = *style* x = .Type End With Application.OrganizerCopy With ActiveDocument .UpdateStyles .CopyStylesFromTemplate End With ActiveDocument.Styles.Add Selection.Style = *name*
FormatStyleGallery	ActiveDocument.CopyStylesFromTemplate
FormatTabs *Position*, *DefTabs*, *Align*, *Leader*, *Set*, *Clear*, *ClearAll*	With Selection.Paragraphs.TabStops .ClearAll .Add Position:=*num*, Alignment:= *WdTabAlignment*, Leader:= *WdTab* .Item(1).Clear End With ActiveDocument.DefaultTabStop
FormFieldOptions *Entry*, *Exit*, *Name*, *Enable*, *TextType*, *TextWidth*, *TextDefault*, *TextFormat*, *CheckSize*, *CheckWidth*, *CheckDefault*, *Type*, *OwnHelp*, *HelpText*, *OwnStat*, *StatText*	With ActiveDocument.FormFields(1) .EntryMacro = *text* .ExitMacro = *text* .Name = *text* .Enabled = True .OwnHelp = True .HelpText = *text* .OwnStatus = True .StatusText = *text* .Type = *WdFieldType* End With With ActiveDocument.FormFields(1).TextInput .Width = *num* .Default = *text* .EditType End With With ActiveDocument.FormFields(1).CheckBox .Size = *num* .AutoSize = True .Default = True

	End With
FormShading	ActiveDocument.FormFields.Shaded = True
FoundFileName$()	Application.FileSearch.FoundFiles(*num*)
Function...End Function	Function…End Function

Visual Basic Equivalents for WordBasic Commands

G

GetAddInID(*name*)	x = Addins(*name*).Index
GetAddInName$(*num*)	x = Addins(*num*).Name
GetAddress$()	x = Application.GetAddress
GetAttr(*filename*)	GetAttr(*filename*)
GetAutoCorrect$(*name*)	x = AutoCorrect.Entries(*name*).Value
GetAutoCorrectException$()	x = AutoCorrect.FirstLetterExceptions(*num*).Name
	x = AutoCorrect.TwoInitialCapsExceptions(*num*).Name

GetAutoText$()	x = ActiveDocument.AttachedTemplate.AutoTextEntries(*name*).Value
GetBookmark$(*name*)	x = ActiveDocument.Bookmarks(*name*).Range.Text
GetCurValues	Dialogs(*WdWordDialog*).Update
GetDirectory$()	x = WordBasic.[GetDirectory$]()
GetDocumentProperty(), GetDocumentProperty$()	x = ActiveDocument.CustomDocumentProperties(*name*).Value *' or* x = ActiveDocument.BuiltInDocumentProperties(*name*).Value
GetDocumentVar$(*name*)	x = ActiveDocument.Variables(*name*).Value
GetDocumentVarName$(*num*)	x = ActiveDocument.Variables(*num*).Name
GetFieldData$()	x = Selection.Fields(1).Data
GetFormResult(), GetFormResult$()	x = ActiveDocument.FormFields(1).Result
GetMergeField$()	x = ActiveDocument.MailMerge.DataSource.DataFields(*name*).Value
GetPrivateProfileString$()	x = System.PrivateProfileString(*filename*, s*ection*, *key*)
GetProfileString$()	x = System.ProfileString(*section*, *key*)
GetSelEndPos()	x = Selection.End
GetSelStartPos()	x = Selection.Start
GetSystemInfo$(21)	x = System.OperatingSystem
GetSystemInfo$(22)	x = System.ProcessorType
GetSystemInfo$(23)	*' not available*
GetSystemInfo$(24)	x = System.Version
GetSystemInfo$(25)	*' not available*
GetSystemInfo$(26)	x = System.FreeDiskSpace
GetSystemInfo$(27)	*' not available*
GetSystemInfo$(28)	x = System.MathCoprocessorInstalled
GetSystemInfo$(29)	x = System.Country
GetSystemInfo$(30)	x = System.LanguageDesignation
GetSystemInfo$(31)	x = System.VerticalResolution
GetSystemInfo$(32)	x = System.HorizontalResolution Values 512 to 526 are Macintosh only.
GetText$(*Pos1*, *Pos2*)	x = ActiveDocument.Range(*Pos1*, *Pos2*).Text
GoBack	Application.GoBack
Goto	GoTo
GoToAnnotationScope	Selection.Comments(1).Scope.Select

GoToHeaderFooter	If Selection.HeaderFooter.IsHeader = True Then
	ActiveWindow.ActivePane.View.SeekView = wdSeekCurrentPageFo
	Else
	ActiveWindow.ActivePane.View.SeekView = wdSeekCurrentPageHe
	End If

GoToNextAnnotation	Selection.GoToNext(wdGoToComment)
GoToNextEndnote	Selection.GoToNext(wdGoToEndnote)
GoToNextFootnote	Selection.GoToNext(wdGoToFootnote)
GoToNextPage	Selection.GotoNext(wdGoToPage)
GoToNextSection	Selection.GotoNext(wdGoToSection)
GoToNextSubdocument	Selection.NextSubdocument
GoToPreviousItem	Selection.GoTo What:=*WdGoToItem*, Which:=wdGoToPrevious
GroupBox	Frame control

GrowFont	Selection.Font.Grow
GrowFontOnePoint	Selection.Font.Size = Selection.Font.Size + 1

Visual Basic Equivalents for WordBasic Commands

H

HangingIndent	ActiveDocument.Paragraphs(1).TabHangingIndent
Help	Assistant.Help
HelpAbout	Application.Help HelpType:=wdHelpAbout
HelpActiveWindow	Application.Help HelpType:=wdHelpActiveWindow
HelpContents	Application.Help HelpType:=wdHelpContents
HelpExamplesAndDemos	Not applicable in Word 97
HelpIndex	Application.Help HelpType:=wdHelpIndex
HelpKeyboard	Not applicable in Word 97
HelpMSN	Not applicable in Word 97

HelpPSSHelp	Application.Help HelpType:=wdHelpPSSHelp
HelpQuickPreview	Not applicable in Word 97
HelpSearch	Application.Help HelpType:=wdHelpSearch
HelpTipOfTheDay	Assistant.FeatureTips = True
HelpTool	Application.HelpTool
HelpUsingHelp	Application.Help HelpType:=wdHelpUsingHelp
HelpWordPerfectHelp	Options.WPHelp = True
HelpWordPerfectHelpOptions	Options.SetWPHelpOptions
Hidden	Selection.Font.Hidden = True
Highlight	Selection.Range.HighlightColorIndex = *WdColorIndex*
HighlightColor, HighlightColor()	Selection.Range.HighlightColorIndex = *WdColorIndex* x = Selection.Range.HighlightColorIndex = *WdColorIndex*
HLine	ActiveWindow.SmallScroll ToRight:=*num* *' or* ActiveWindow.SmallScroll ToLeft:=*num*
Hour()	Hour
HPage	ActiveWindow.LargeScroll ToRight:=*num* *' or* ActiveWindow.LargeScroll ToLeft:=*num*
HScroll, HScroll()	ActiveWindow.HorizontalPercentScrolled = *num* num = ActiveWindow.VerticalPercentScrolled

Visual Basic Equivalents for WordBasic Commands

I

If...Then...Else	If...Then...Else
Indent	ActiveDocument.Paragraphs(1).TabIndent
Input	Input
Input$()	Input()
InputBox$()	InputBox
Insert	Selection.InsertAfter Text:=*text* ' *or* Selection.TypeText Text:=*text*
InsertAddCaption	CaptionLabels.Add
InsertAddress	Application.GetAddress
InsertAnnotation	ActiveDocument.Comments.Add
InsertAutoCaption *Clear*, *ClearAll*, *Label*, *Position*	With AutoCaptions(*name*) .AutoInsert = True .CaptionLabel.Name .CaptionLabel.Position End With AutoCaptions.CancelAutoInsert
InsertAutoText	Selection.Range.InsertAutoText
InsertBreak	Selection.InsertBreak Type:=*WdBreakType*
InsertCaption	Selection.InsertCaption
InsertCaptionNumbering *Label*, *FormatNumber*, *ChapterNumber*, *Level*, *Separator*	With CaptionLabels(*name*) .ChapterStyleLevel = *num* .Separator = *WdSeparatorType* .NumberStyle = *WdCaptionNumberStyle* .IncludeChapterNumber = True End With
InsertChart	ActiveDocument.Shapes.AddOLEObject
InsertColumnBreak	Selection.InsertBreak Type:=wdColumnBreak
InsertCrossReference	Selection.InsertCrossReference
InsertDatabase	Selection.Range.InsertDatabase
InsertDateField	Selection.Fields.Add Range:=*range*, Type:=wdFieldDate
InsertDateTime	Selection.InsertDateTime
InsertDrawing	ActiveDocument.Shapes.AddOLEObject
InsertEquation	ActiveDocument.Shapes.AddOLEObject
InsertExcelTable	ActiveDocument.Shapes.AddOLEObject
InsertField *field_type*	ActiveDocument.Fields.Add Range:=*range*, Type:=*field_type*
InsertFieldChars	Selection.Fields.Add Range:=*range*, Type:=wdFieldEmpty, PreserveForm

InsertFile *Name*, *Range*, *ConfirmConversions*, *Link*	Selection.InsertFile
InsertFootnote *Reference*, *NoteType*	ActiveDocument.Footnotes.Add Range:=*range*, Text:=*text* ActiveDocument.Endnotes.Add Range:=*range*, Text:=*text*
InsertFormField *Entry*, *Exit*, *Name*, *Enable*, *TextType*, *TextDefault*, *TextWidth*, *TextFormat*, *CheckSize*, *CheckWidth*, *CheckDefault*, *Type*, *OwnHelp*, *HelpText*, *OwnStat*, *StatText*	Set myField = ActiveDocument.FormFields.Add(Range:=*range*, Type:=*V* With myField .EntryMacro = *text* .ExitMacro = *text* .Name = *text* .Enabled = True .OwnHelp = True .HelpText = *text* .OwnStatus = True .StatusText = *text* End With With myField.TextInput .Width = *num* .Default = *text* .EditType End With With myField.CheckBox .Size = *num* .AutoSize = True .Default = True End With
InsertFrame	Selection.Frames.Add
InsertIndex	ActiveDocument.Indexes.Add
InsertMergeField	ActiveDocument.MailMerge.Fields.Add
InsertObject	ActiveDocument.Shapes.AddOLEObject
InsertPageBreak	Selection.InsertBreak Type:=wdPageBreak
InsertPageField	ActiveDocument.Fields.Add Range:=*range*, Type:=wdFieldPage
InsertPageNumbers	ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary).PageNu
InsertPara	Selection.InsertParagraphAfter ' *or* Selection.TypeParagraph
InsertPicture Name, LinkToFile, New	ActiveDocument.Shapes.AddPicture ActiveDocument.InlineShapes.New Range:=*range*
InsertSectionBreak	Selection.Range.InsertBreak Type:=*WdBreakType*
InsertSound	Selection.InlineShapes.AddOLEObject ClassType:="SoundRec"
InsertSpike	NormalTemplate.AutoTextEntries("Spike").Insert Where:=*range*
InsertSubdocument	ActiveDocument.Subdocuments.AddFromFile Range:=*range*
InsertSymbol	Selection.InsertSymbol
InsertTableOfAuthorities	ActiveDocument.TablesOfAuthorities.Add
InsertTableOfContents	ActiveDocument.TablesOfContents.Add
InsertTableOfFigures	ActiveDocument.TablesOfFigures.Add
InsertTimeField	ActiveDocument.Fields.Add Range:=*range*, Type:=wdFieldTime
InsertWordArt	ActiveDocument.Shapes.AddOLEObject
InStr()	InStr()
Int()	Int()

IsAutoCorrectException()	For Each xItem In AutoCorrect.FirstLetterExceptions If xItem.Name = "apt." Then isFound = True Next xItem For Each aItem In AutoCorrect.TwoInitialCapsExceptions If aItem.Name = "THem" Then aExists = True Next aItem
IsCustomDocumentProperty()	For Each aProp In ActiveDocument.CustomDocumentProperties If aProp.Name = "age" Then isFound = True Next aProp
IsDocumentDirty()	x = Not ActiveDocument.Saved
IsDocumentPropertyReadOnly()	x = WordBasic.IsDocumentPropertyReadOnly(*name*)
IsExecuteOnly()	x = ActiveDocument.VBProject.Protection
IsMacro()	Not applicable in Word 97
IsTemplateDirty()	x = Not ActiveDocument.AttachedTemplate.Saved
Italic, Italic()	Selection.Font.Italic = True

Visual Basic Equivalents for WordBasic Commands

J

JustifyPara, JustifyPara()	Selection.Paragraphs.Alignment = wdAlignParagraphJustify

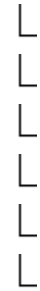
K

KeyCode()	x = KeyBindings(1).KeyCode
KeyMacro$()	x = KeyBindings(1).Command
Kill *filename*	
	Kill *filename*

L

Language, Language$()	Selection.LanguageID
LCase$()	LCase() *' or* LCase$()
Left$()	Left$() *' or* Left()
LeftPara, LeftPara()	Selection.Paragraphs.Alignment = wdAlignParagraphLeft
Len()	Len()
Let	Let
Line Input	Line Input
LineDown, LineDown()	Selection.MoveDown Unit:=wdLine, Count:=1, Extend:=wdMove
LineUp, LineUp()	Selection.MoveUp Unit:= wdLine, Count:=1, Extend:=wdMove
ListBox	ListBox Control
ListCommands	Application.ListCommands
LockDocument, LockDocument()	ActiveDocument.Subdocuments(1).Locked = True state = ActiveDocument.Subdocuments(1).Locked
LockFields	*' You can lock a single field or a group of fields within a range.* Selection.Fields.Locked = True ActiveDocument.Fields(1).Locked =True
Lof()	LOF()
LTrim$()	LTrim()

Visual Basic Equivalents for WordBasic Commands



M

MacID$()	Macintosh only
MacroCopy	Application.OrganizerCopy
MacroDesc$()	x = WordBasic.[MacroDesc$](*name*)
MacroFileName$()	Not applicable in Word 97
MacroName$()	x = WordBasic.[MacroName$](*num*)
MacroNameFromWindow$()	Not applicable in Word 97
MacScript, MacScript$()	Macintosh only
Magnifier, Magnifier()	ActiveWindow.View.Magnifier = True state = ActiveWindow.View.Magnifier
MailCheckNames	Application.MailMessage.CheckName
MailHideMessageHeader	Application.MailMessage.ToggleHeader
MailMerge *CheckErrors*, *Destination*, *MergeRecords*, *From*, *To*, *Suppression*, *MailMerge*, *MailSubject*, *MailAsAttachment*, *MailAddress*	With ActiveDocument.MailMerge .Check .Destination = *WdMailMergeDestination* .DataSource.FirstRecord = *num* .DataSource.LastRecord = *num* .SuppressBlankLines = True .MailSubject = *text* .MailAsAttachment = True .MailAddressFieldName = *text* .Execute End With
MailMergeAskToConvertChevrons, MailMergeAskToConvertChevrons()	FileConverters.ConvertMacWordChevrons = *WdChevronConvertRule* state = FileConverters.ConvertMacWordChevrons

MailMergeCheck	ActiveDocument.MailMerge.Check
MailMergeConvertChevrons, MailMergeConvertChevrons()	FileConverters.ConvertMacWordChevrons = *WdChevronConvertRule* state = FileConverters.ConvertMacWordChevrons
MailMergeCreateDataSource	ActiveDocument.MailMerge.CreateDataSource
MailMergeCreateHeaderSource	Documents(*name*).MailMerge.CreateHeaderSource
MailMergeDataForm	ActiveDocument.DataForm
MailMergeDataSource$(0)	x = ActiveDocument.MailMerge.DataSource.Name
MailMergeDataSource$(1)	x = ActiveDocument.MailMerge.DataSource.HeaderSourceName
MailMergeDataSource$(2)	x = ActiveDocument.MailMerge.DataSource.Type
MailMergeDataSource$(3)	x = ActiveDocument.MailMerge.DataSource.HeaderSourceType
MailMergeEditDataSource	Documents(*name*).MailMerge.EditDataSource
MailMergeEditHeaderSource	Documents(1).MailMerge.EditHeaderSource
MailMergeEditMainDocument	ActiveDocument.MailMerge.EditMainDocument
MailMergeFindRecord	ActiveDocument.MailMerge.DataSource.FindRecord
MailMergeFirstRecord	ActiveDocument.MailMerge.DataSource.ActiveRecord = wdFirstRecord
MailMergeFoundRecord()	x = ActiveDocument.MailMerge.DataSource.FindRecord
MailMergeGotoRecord, MailMergeGotoRecord()	ActiveDocument.MailMerge.DataSource.ActiveRecord = *num* x = ActiveDocument.MailMerge.DataSource.ActiveRecord
MailMergeHelper	Dialogs(wdDialogMailMergeHelper).Show
MailMergeInsertAsk	Documents(*name*).MailMerge.Fields.AddAsk
MailMergeInsertFillIn	Documents(*name*).MailMerge.Fields.AddFillIn
MailMergeInsertIf	ActiveDocument.MailMerge.Fields.AddIf
MailMergeInsertMergeRec	ActiveDocument.MailMerge.Fields.AddMergeRec
MailMergeInsertMergeSeq	ActiveDocument.MailMerge.Fields.AddMergeSeq
MailMergeInsertNext	Documents(1).MailMerge.Fields.AddNext
MailMergeInsertNextIf	ActiveDocument.MailMerge.Fields.AddNextIf
MailMergeInsertSet	ActiveDocument.MailMerge.Fields.AddSet
MailMergeInsertSkipIf	ActiveDocument.MailMerge.Fields.AddSkipIf
MailMergeLastRecord	Documents(*name*).MailMerge.DataSource.ActiveRecord = wdLastRecor
MailMergeMainDocumentType, MailMergeMainDocumentType()	ActiveDocument.MailMerge.MainDocumentType = *WdMailMergeMainDo* state = ActiveDocument.MailMerge.MainDocumentType
MailMergeNextRecord	ActiveDocument.MailMerge.DataSource.ActiveRecord = wdNextRecord
MailMergeOpenDataSource	Documents(1).MailMerge.OpenDataSource
MailMergeOpenHeaderSource	Documents(*name*).MailMerge.OpenHeaderSource
MailMergePrevRecord	ActiveDocument.MailMerge.DataSource.ActiveRecord = wdPreviousRec
MailMergeQueryOptions	ActiveDocument.MailMerge.DataSource.QueryString = *text*
MailMergeReset	ActiveDocument.MailMerge.MainDocumentType = wdNotAMergeDocum
MailMergeState()	theState = ActiveDocument.MailMerge.State
MailMergeToDoc	Documents(*name*).MailMerge.Destination = wdSendToNewDocument
MailMergeToPrinter	ActiveDocument.MailMerge.Destination = wdSendToPrinter
MailMergeUseAddressBook	ActiveDocument.MailMerge.UseAddressBook
MailMergeViewData, MailMergeViewData()	ActiveDocument.MailMerge.ViewMailMergeFieldCodes = True x = ActiveDocument.MailMerge.ViewMailMergeFieldCodes
MailMessageDelete	Application.MailMessage.Delete

MailMessageForward	Application.MailMessage.Forward
MailMessageMove	Application.MailMessage.DisplayMoveDialog
MailMessageNext	Application.MailMessage.GoToNext
MailMessagePrevious	Application.MailMessage.GoToPrevious
MailMessageProperties	Application.MailMessage.DisplayProperties
MailMessageReply	Application.MailMessage.Reply
MailMessageReplyAll	Application.MailMessage.ReplyAll
MailSelectNames	Application.MailMessage.DisplaySelectNamesDialog
MarkCitation	ActiveDocument.TablesOfAuthorities.MarkCitation ActiveDocument.TablesOfAuthorities.MarkAllCitations
MarkIndexEntry	ActiveDocument.Indexes.MarkEntry
MarkTableOfContentsEntry	ActiveDocument.TablesOfContents.MarkEntry
MenuItemMacro$()	x = CommandBars(*name*).Controls(*num*).OnAction
MenuItemText$()	x = CommandBars(*name*).Controls(*num*).Caption
MenuMode	WordBasic.MenuMode
MenuText$()	x = CommandBars.ActiveMenuBar.Controls(*num*).Caption
MergeFieldName$(*num*)	x = ActiveDocument.MailMerge.DataSource.FieldNames(*num*)
MergeSubdocument	ActiveDocument.Subdocuments.Merge
MicrosoftAccess	WordBasic.MicrosoftAccess ' or use the technique shown in Microsoft Excel example

MicrosoftExcel	WordBasic.MicrosoftExcel
	' or
	If Tasks.Exists("Microsoft Excel") = True Then
	  Tasks("Microsoft Excel").Activate
	  Tasks("Microsoft Excel").WindowState = wdWindowStateMaximize
	Else
	  Shell "C:\MSOffice\Excel\Excel.exe"
	End If

MicrosoftFoxPro	WordBasic.MicrosoftFoxPro ' or use the technique shown in Microsoft Excel example
MicrosoftMail	WordBasic.Mail ' or use the technique shown in Microsoft Excel example
MicrosoftPowerPoint	WordBasic.PowerPoint ' or use the technique shown in Microsoft Excel example
MicrosoftProject	WordBasic.Project ' or use the technique shown in Microsoft Excel example
MicrosoftPublisher	WordBasic.Publisher ' or use the technique shown in Microsoft Excel example
MicrosoftSchedule	WordBasic.Schedule ' or use the technique shown in Microsoft Excel example
MicrosoftSystemInfo	System.MSInfo
Mid$()	Mid$() *' or* Mid()
Minute()	Minute()
MkDir *path_name*	MkDir *path_name*

Month()	Month()
MountVolume	Application.MountVolume
MoveButton	CommandBars(*name*).Controls(1).Move
MoveText	WordBasic.MoveText
MoveToolbar	With CommandBars(*name*) .Top = *num* .Left = *num* End With CommandBars(*name*).Position = *MsoBarPosition*
MsgBox, MsgBox()	MsgBox, MsgBox()

Visual Basic Equivalents for WordBasic Commands

N

Name	Name
NewToolbar	CommandBars.Add

NextCell	Selection.Move Unit:=wdCell, Count:=1 ' or Selection.Cells(1).Next.Select
NextField, NextField()	Selection.GoToNext What:=wdGoToField *' or* Selection.NextField
NextMisspelling	Selection.GoToNext What:=wdGoToSpellingError
NextObject	Selection.GoToNext What:=wdGoToObject *' or* Selection.MoveRight Unit:=wdItem
NextPage, NextPage()	Selection.GoToNext What:=wdGoToPage *' or* ActiveWindow.View.Type = wdPageView ActiveWindow.PageScroll Down:=1
NextTab()	x = ActiveDocument.Paragraphs(1).TabStops(1).Next.Position
NextWindow	ActiveWindow.Next.Activate
NormalFontPosition	Selection.Font.Position = 0
NormalFontSpacing	Selection.Font.Spacing = 0
NormalStyle	Selection.Style = wdStyleNormal
NormalViewHeaderArea *Type*, *FirstPage*, *OddAndEvenPages*, *HeaderDistance*, *FooterDistance*	With ActiveDocument.PageSetup .DifferentFirstPageHeaderFooter = True .OddAndEvenPagesHeaderFooter = True .HeaderDistance = *num* .FooterDistance = *num* End With ActiveWindow.View.SeekView = *WdSeekView*
NoteOptions *FootnotesAt*, *FootNumberAs*, *FootStartingNum*, *FootRestartNum*, *EndnotesAt*, *EndNumberAs*, *EndStartingNum*, *EndRestartNum*	With ActiveDocument.Footnotes .Location = *WdFootnoteLocation* .NumberingRule = *WdNumberingRule* .NumberStyle = *WdNoteNumberStyle* .StartingNumber = *num* End With With ActiveDocument.Endnotes .Location = *WdEndnoteLocation* .NumberingRule = *WdNumberingRule* .NumberStyle = *WdNoteNumberStyle* .StartingNumber = *num* End With
Now()	Now

Visual Basic Equivalents for WordBasic Commands

O

OK	WordBasic.OK
OKButton	CommandButton control
On Error	On Error
OnTime	Application.OnTime
Open	Open
OpenSubdocument	ActiveDocument.Subdocuments(*name*).Open
OpenUpPara	Selection.Paragraphs.OpenUp
OptionButton	OptionButton control
OptionGroup	Frame control
Organizer	Application.OrganizerCopy Application.OrganizerDelete Application.OrganizerRename
OtherPane	ActiveWindow.ActivePane.Next.Activate
Outline, Outline()	Selection.Font.Outline = True x = Selection.Font.Outline
OutlineCollapse	ActiveWindow.View.CollapseOutline
OutlineDemote	Selection.Paragraphs.OutlineDemote
OutlineExpand	ActiveWindow.View.ExpandOutline

OutlineLevel()	aLevel = Selection.Paragraphs.OutlineLevel
OutlineMoveDown	Selection.Range.Relocate Direction:=wdRelocateDown
OutlineMoveUp	Selection.Range.Relocate Direction:=wdRelocateUp
OutlinePromote	Selection.Paragraphs.OutlinePromote
OutlineShowFirstLine, OutlineShowFirstLine()	ActiveWindow.View.ShowFirstLineOnly = True x = ActiveWindow.View.ShowFirstLineOnly
OutlineShowFormat	AcitveWindow.View.ShowFormat = True
Overtype	Options.Overtype = True

P

PageDown, PageDown()	Selection.MoveDown Unit:=wdScreen, Count:=1, Extend:=wdMove
PageUp, PageUp()	Selection.MoveUp Unit:=wdScreen, Count:=1, Extend:=wdMove
ParaDown, ParaDown()	Selection.MoveDown Unit:=wdParagraph, Count:=1, Extend:=wdMove
ParaKeepLinesTogether, ParaKeepLinesTogether()	ActiveDocument.Paragraphs(1).KeepTogether = True x = ActiveDocument.Paragraphs(1).KeepTogether
ParaKeepWithNext, ParaKeepWithNext()	ActiveDocument.Paragraphs(1).KeepWithNext = True x = ActiveDocument.Paragraphs(1).KeepWithNext
ParaPageBreakBefore, ParaPageBreakBefore()	ActiveDocument.Paragraphs(1).PageBreakBefore = True x = ActiveDocument.Paragraphs(1).PageBreakBefore
ParaUp, ParaUp()	Selection.MoveUp Unit:=wdParagraph, Count:=1, Extend:=wdMove
ParaWidowOrphanControl, ParaWidowOrphanControl()	ActiveDocument.Paragraphs(1).WidowControl = True x = ActiveDocument.Paragraphs(1).WidowControl
PasteButtonImage	CommandBars(*name*).Controls(1).PasteFace
PasteFormat	Selection.PasteFormat
PathFromMacPath$()	x = WordBasic.[PathFromMacPath$](*path*)
PathFromWinPath$()	x = WordBasic.[PathFromWinPath$](*path*)
PauseRecorder	WordBasic.PauseRecorder
Picture	Image control
PrevCell, PrevCell()	Selection.Move Unit:=wdCell, Count:=-1 *' or* Selection.Cells(1).Previous.Select
PrevField, PrevField()	Selection.GoToPrevious What:=wdGoToField *' or* Selection.PreviousField
PrevObject	Selection.GoToPrevious What:=wdGoToObject *' or* Selection.MoveLeft Unit:=wdItem
PrevPage, PrevPage()	Selection.GoToPrevious What:=wdGoToPage *' or* ActiveWindow.View.Type = wdPageView ActiveWindow.PageScroll Up:=1
PrevTab()	x = ActiveDocument.Paragraphs(1).TabStops(1).Previous.Position
PrevWindow	ActiveWindow.Previous.Activate
Print	Print Print

PromoteList	Selection.Range.ListFormat.ListIndent
PushButton	CommandButton control
PutFieldData	ActiveDocument.Fields(1).Data = *text*

Visual Basic Equivalents for WordBasic Commands

R

Read	Open
Redim	ReDim
REM	REM
RemoveAllDropDownItems	ActiveDocument.FormFields(1).DropDown.ListEntries.Clear
RemoveBulletsNumbers	Selection.Range.ListFormat.RemoveNumbers
RemoveDropDownItem	ActiveDocument.FormFields(1).DropDown.ListEntries(1).Delete
RemoveFrames	Selection.Frames(1).Delete
RemoveSubdocument	ActiveDoucment.Subdocuments(1).Delete

RenameMenu	CommandBars.ActiveMenuBar.Controls(*name*).Caption = *newname*
RepeatFind	Application.Run MacroName:="RepeatFind"
ResetButtonImage	CommandBars(name).Controls(1).Reset
ResetChar, ResetChar()	Selection.Font.Reset
ResetNoteSepOrNotice	ActiveDocument.Endnotes.ResetContinuationNotice ActiveDocument.Footnotes.ResetContinuationNotice ActiveDocument.Endnotes.ResetContinuationSeparator ActiveDocument.Footnotes.ResetContinuationSeparator
ResetPara, ResetPara()	Selection.Paragraphs.Reset
Right$()	Right() ' *or* Right$()
RightPara, RightPara()	Selection.Paragraphs.Alignment = wdAlignParagraphRight
RmDir *path*	RmDir *path*
Rnd(*number*)	Rnd(*number*)
RTrim$()	RTrim() ' *or* RTrim$()
RunPrintManager	Not available

Visual Basic Equivalents for WordBasic Commands

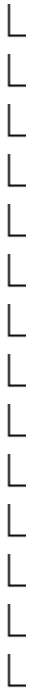
S

SaveTemplate	ActiveDocument.AttachedTemplate.Save *' or* Templates(*name*).Save
ScreenRefresh	Application.ScreenRefresh
ScreenUpdating, ScreenUpdating()	Application.ScreenUpdating = True
Second(*time*)	Second(*time*)
Seek *filenumber*, position	Seek[#]*filenumber*,*position*
Seek(*filenumber*)	Seek(*filenumber*)
Select Case	Select Case
SelectCurAlignment	Selection.SelectCurrentAlignment
SelectCurColor	Selection.SelectCurrentColor
SelectCurFont	Selection.SelectCurrentFont
SelectCurIndent	Selection.SelectCurrentIndent
SelectCurSentence	Selection.Sentences(1).Select
SelectCurSpacing	Selection.SelectCurrentSpacing
SelectCurTabs	Selection.SelectCurrentTabs
SelectCurWord	Selection.Words(1).Select
Selection$()	*text* = Selection.Text
SelectionFileName$()	*aFileName* = Selection.Document.FullName
SelInfo(*Type*)	x = Selection.Information(*WdInformation*)
SelType()	*aType* = Selection.Type(*Type)*
SelType 1	Selection.Collapse Direction:=wdCollapseStart
SendKeys *keys*, *wait*	SendKeys *keys*, *wait*
SentLeft 1,1	Selection.Sentences(1).Previous(Unit:=wdSentence, Count:=1).Select
SentRight 1, 1	Selection.Sentences(1).Next(Unit:=wdSentence, Count:=1).Select
SetAttr *filename*, *attribute*	SetAttr *filename*, *attribute*
SetAutoText	Templates(*name*).AutoTextEntries.Add
SetDocumentDirty 1	ActiveDocument.Saved = False
SetDocumentProperty	ActiveDocument.BuiltInDocumentProperties.Add *' or* ActiveDocument.CustomDocumentProperties.Add
SetDocumentPropertyLink *name*, *source*	ActiveDocument.CustomDocumentProperties(*name*).LinkSource = *sour*
SetDocumentVar *name*, *value*	ActiveDocument.Variables.Add *name*, *value*
SetEndOfBookmark *name*	*range*.Bookmarks(*name*).Start = *range*.Bookmarks(*name*).End

SetEndOfBookmark *name1*, *name2*	ActiveDocument.Bookmarks(*name1*).End = ActiveDocument.Bookmarks
SetFileCreatorAndType	Macintosh only
SetFormResult *name*, "text"	ActiveDocument.FormFields(*name*).Result = "text"
SetFormResult *name*, 1	ActiveDocument.FormFields(*name*).CheckBox.Value = True
SetFormResult *name*, *num*	ActiveDocument.FormFields(*name*).DropDown.Value = *num*
SetFormResult *name*, , default	Use the Default property with a CheckBox, DropDown or TextInput objec
SetPrivateProfileString *section*, *key*, *setting*, *filename*	System.PrivateProfileString(*filename*, *section*, *key*) = *setting*
SetProfileString *section*, *key*, *setting*	System.ProfileString(*section*, *key*) = *setting*
SetSelRange *charpos1*, *charpos2*	ActiveDocument.Range(Start:=*charpos1*, End:=*charpos2*).Select
SetStartOfBookmark *name*	*range*.Bookmarks(*name*).End = *range*.Bookmarks(*name*).Start
SetStartOfBookmark *book1*, *book2*	ActiveDocument.Bookmarks(*book1*).Start = ActiveDocument.Bookmarks
SetTemplateDirty 0	Documents(*name*).AttachedTemplate.Saved = True *' or* Templates(*name*).Saved = True
Sgn()	Sgn()
ShadingPattern, ShadingPattern()	Selection.Shading.Texture = *WdTextureIndex*
Shadow, Shadow()	Selection.Font.Shadow = True x = Selection.Font.Shadow
Shell	Shell
ShowAll, ShowAll()	Windows(1).View.ShowAll = True x = ActiveWindow.View.ShowAll
ShowAllHeadings	ActiveWindow.View.ShowAllHeadings
ShowAnnotationBy *name*	ActiveDocument.Comments.ShowBy = *name*
ShowClipboard	Application.ShowClipboard
ShowHeading*Number*	Windows(*name*).View.ShowHeading Level:=*num*
ShowMe	Application.ShowMe
ShowNextHeaderFooter	ActiveWindow.View.NextHeaderFooter
ShowPrevHeaderFooter	ActiveWindow.View.PreviousHeaderFooter
ShowVars	Add a watch expression in the Visual Basic Editor
ShrinkFont	Selection.Font.Shrink
ShrinkFontOnePoint	Selection.Font.Size = Selection.Font.Size - 1
ShrinkSelection	Selection.Shrink
SizeToolbar *name*, *width*	CommandBars(*name*).Width = *num*
SkipNumbering, SkipNumbering()	Selection.Range.ListFormat.RemoveNumbers
SmallCaps, SmallCaps()	Selection.Font.SmallCaps = True
SortArray	WordBasic.SortArray
SpacePara1, SpacePara1()	Selection.Paragraphs.Space1 x = Selection.Paragraphs.LineSpacing
SpacePara15, SpacePara15()	Selection.Paragraphs.Space15 x = Selection.Paragraphs.LineSpacing
SpacePara2, SpacePara2()	Selection.Paragraphs.Space2 x = Selection.Paragraphs.LineSpacing
SpellChecked, SpellChecked()	ActiveDocument.Content.SpellingChecked = True

	x = ActiveDocument.Content.SpellingChecked
Spike	NormalTemplate.AutoTextEntries.AppendToSpike
SplitSubdocument	ActiveDocument.Subdocuments(1).Split Range:=*range*
StartOfColumn, StartOfColumn()	Selection.StartOf Unit:=wdColumn, Extend:=wdMove
StartOfDocument, StartOfDocument()	Selection.HomeKey Unit:=wdStory, Extend:=wdMove
StartOfLine, StartOfLine()	Selection.HomeKey Unit:=wdLine, Extend:=wdMove
StartOfRow, StartOfRow()	Selection.StartOf Unit:=wdRow, Extend:=wdMove
StartOfWindow, StartOfWindow()	Application.Run MacroName:="StartOfWindow"
Stop	Stop
Str$(*number*)	Str(*number*)
	Str$(*number*)
Strikethrough, Strikethrough()	Selection.Font.StrikeThrough = True
String$(*count*, *character*)	String(*count*, *character*)
	String$(*count*, *character*)
Style	Selection.Style = wdStyleHeading1
StyleDesc$()	x = Selection.Style.Description
StyleName$()	x = Selection.Style.NameLocal
Sub...End Sub	Sub...End Sub
Subscript, Subscript()	Selection.Font.Subscript = True
	x = Selection.Font.Subscript
Superscript, Superscript()	Selection.Font.Superscript = True
	x = Selection.Font.Superscript
SymbolFont	Selection.Font.Name = "Symbol"

Visual Basic Equivalents for WordBasic Commands



T

TabLeader$(*pos*)	aType = Selection.Paragraphs(*num*).TabStops(*pos*).Leader
TableAutoFormat	ActiveDocument.Tables(1).AutoFormat
TableAutoSum	ActiveDocument.Tables(1).Cell(*row*, *column*).AutoSum
TableColumnWidth *ColumnWidth*, *RulerStyle*	ActiveDocument.Tables(1).Columns.SetWidth ColumnWidth:=*num*, Rule
TableColumnWidth *AutoFit*	ActiveDocument.Tables(1).Columns.AutoFit
TableColumnWidth *NextColumn*	Selection.Columns(1).Next.Select
TableColumnWidth *PrevColumn*	Selection.Columns(1).Previous.Select
TableColumnWidth *SpaceBetweenCols*	ActiveDocument.Tables(1).Rows.SpaceBetweenColumns = *num*
TableDeleteCells *ShiftCells*	ActiveDocument.Tables(1).Cell(*row*, *column*).Delete ShiftCells:=*WdDele*
TableDeleteColumn	ActiveDocument.Tables(1).Columns(*num*).Delete ' *or* ActiveDocument.Tables(1).Columns.Delete
TableDeleteRow	ActiveDocument.Tables(1).Rows(*num*).Delete ' *or* ActiveDocument.Tables(1).Rows.Delete
TableFormula	ActiveDocument.Tables(1).Cell(*row*, *column*).Formula
TableGridlines, TableGridlines()	ActiveWindow.View.TableGridlines = True x = ActiveWindow.View.TableGridlines
TableHeadings, TableHeadings()	Selection.Tables(1).Rows(*num*).HeadingFormat = True ' *or* Selection.Tables(1).Rows.HeadingFormat = True x = Selection.Tables(1).Rows.HeadingFormat
TableInsertCells	Selection.Tables(1).Columns(*num*).Cells.Add
TableInsertColumn	Selection.Tables(1).Columns.Add
TableInsertRow	Selection.Tables(1).Rows.Add
TableInsertTable *NumColumns*, *NumRows*	ActiveDocument.Tables.Add Range:=*range*, NumRows:=*num*, NumColu
TableInsertTable *NumColumns*, *NumRows*, *Format*, Apply	ActiveDocument.Tables.Add(Range:=*range*, NumRows:=*num*, NumColu Selection.ConvertToTable Separator:=*WdTableFieldSeparator*, NumRow
TableInsertTable *NumColumns*, *NumRows*, *ConvertFrom*	NumColumns:=*num*
TableMergeCells	Selection.Cells.Merge
TableRowHeight *RulerStyle*, *LineSpacingRule*, *LineSpacing*, *LeftIndent*, *Alignment*,	With ActiveDocument.Tables(*num*).Rows(*num*) .SetHeight RowHeight:= *num*, HeightRule:=*WdRowHeightRule*

AllowRowSplit	.Alignment = *WdRowAlignment* .SetLeftIndent LeftIndent:=*num*, RulerStyle:=*WdRulerStyle* .AllowBreakAcrossPages = True End With
TableRowHeight *NextColumn*	Selection.Rows(1).Next.Select
TableRowHeight *PrevColumn*	Selection.Rows(1).Previous.Select
TableSelectColumn	Selection.Tables(1).Columns(*num*).Select
TableSelectRow	Selection.Tables(1).Rows(*num*).Select
TableSelectTable	ActiveDocument.Tables(1).Select
TableSort	ActiveDocument.Tables(1).Sort
TableSortAToZ	ActiveDocument.Tables(1).SortAscending
TableSortZToA	ActiveDocument.Tables(1).SortDescending
TableSplit	Selection.Tables(1).Split
TableSplitCells	Selection.Tables(1).Cells(*row*, *column*).Split
TableToText	Selection.Tables(1).ConvertToText
TableUpdateAutoFormat	Selection.Tables(1).UpdateAutoFormat
TabType()	x = Selection.ParagraphFormat.TabStops(1).Alignment
Text	Label control
TextBox	TextBox control
TextFormField	ActiveDocument.FormFields.Add Range:=*range*, Type:=wdFieldFormTex
TextToTable]	Selection.ConvertToTable
Time$()	Time() *' or* Time$()
TimeSerial()	TimeSerial
TimeValue()	TimeValue
TipWizard	No direct equivalent *' displays a special tip when Word is launched* Assistant.FeatureTips = True
Today()	Dim x As Long x = DateSerial(Year(Date), Month(Date), Day(Date))
ToggleFieldDisplay	Selection.Fields.ToggleShowCodes
ToggleFull	ActiveWindow.View.FullScreen = Not ActiveWindow.View.FullScreen
ToggleHeaderFooterLink	ActiveDocument.Sections(2).Headers(wdHeaderFooterPrimary).LinkToP
ToggleMainTextLayer	ActiveWindow.View.ShowMainTextLayer = Not ActiveWindow.View.Sho
TogglePortrait	ActiveDocument.PageSetup.TogglePortrait
ToggleScribbleMode	WordBasic.ToggleScribbleMode
ToolbarButtonMacro$()	x = CommandBars(*name*).Controls(1).OnAction
name = ToolbarName$()	*name* = CommandBars(*num*).Name
ToolbarState(*name*)	CommandBars(*name*).Visible = True
ToolsAddRecordDefault	WordBasic.ToolsAddRecordDefault *' or if the data source is a Word table* Selection.Tables(1).Cell(Row:=Selection.Information(wdMaximumNumbe Column:=Selection.Information(wdMaximumNumberOfColumns)).Se Selection.MoveRight Unit:=wdCell
ToolsAdvancedSettings	Not available with Windows 95 and Windows NT
ToolsAutoCorrect *InitialCaps*, *SentenceCaps*,	With AutoCorrect

Days, *CapsLock*, *ReplaceText*	.CorrectInitialCaps = True .CorrectSentenceCaps = True .CorrectDays = True .CorrectCapsLock = True .ReplaceText = True End with
ToolsAutoCorrect *SmartQuotes*	Options.AutoFormatAsYouTypeReplaceQuotes = True
ToolsAutoCorrect *Formatting*, *Replace*, *With*, *Add*	AutoCorrectEntries.AddRichText Name:= *text*, Range:=*range*
ToolsAutoCorrect *Formatting*, *Replace*, *With*, *Add*	AutoCorrectEntries.Add Name:= *text*, Value:= *text*
ToolsAutoCorrect .Replace = *text*, .Delete	AutoCorrectEntries(*name*).Delete
ToolsAutoCorrectCapsLockOff, ToolsAutoCorrectCapsLockOff()	AutoCorrect.CorrectCapsLock = True
ToolsAutoCorrectDays, ToolsAutoCorrectDays()	AutoCorrect.CorrectDays = True
ToolsAutoCorrectExceptions *Tab* = 0, *Name*, *Add*	FirstLetterExceptions.Add *name*
ToolsAutoCorrectExceptions *Tab* = 1, *Name*, *Add*	TwoInitialCapsExceptions.Add *name*
ToolsAutoCorrectExceptions *Tab* = 0, *AutoAdd*	AutoCorrect.FirstLetterAutoAdd = True
ToolsAutoCorrectExceptions *Tab* = 1, *AutoAdd*	AutoCorrect.TwoInitialCapsAutoAdd = True
ToolsAutoCorrectExceptions *Tab* = 0, *Name*, *.Delete*	FirstLetterExceptions(*name*).Delete
ToolsAutoCorrectExceptions *Tab* = 1, *Name*, *Delete*	TwoInitialCapsExceptions(*name*).Delete
ToolsAutoCorrectInitialCaps, ToolsAutoCorrectInitialCaps()	AutoCorrect.CorrectInitialCaps = True
ToolsAutoCorrectReplaceText, ToolsAutoCorrectReplaceText()	AutoCorrect.ReplaceText = True
ToolsAutoCorrectSentenceCaps, ToolsAutoCorrectSentenceCaps()	AutoCorrect.CorrectSentenceCaps = True
ToolsAutoCorrectSmartQuotes, ToolsAutoCorrectSmartQuotes()	Options.AutoFormatAsYouTypeReplaceQuotes = True
ToolsBulletListDefault	Selection.Range.ListFormat.ApplyBulletDefault
ToolsBulletsNumbers *Replace*, *Font*, *CharNum*, *Type*, *FormatOutline*, *AutoUpdate*, *FormatNumber*, *Punctuation*, *StartAt*, *Points*, *Hang*, *Indent*, *Remove*	With ListGalleries(wdNumberGallery).ListTemplates(1).ListLevels(1) .NumberFormat = "%1." .TrailingCharacter = *WdTrailingCharacter* .NumberStyle = *WdListNumberStyle* .Alignment = *WdListLevelAlignment* .TextPosition = InchesToPoints(*num*) .TabPosition = InchesToPoints(*num*) .ResetOnHigher = True .StartAt = *num* .Font.Size = *num* End With Selection.Range.ListFormat.ApplyListTemplate _ ListTemplate:=ListGalleries(wdNumberGallery).ListTemplates(1)
ToolsCalculate, ToolsCalculate()	Selection.Range.Calculate
ToolsCompareVersions	ActiveDocument.Compare
ToolsCreateEnvelope *PrintEnvLabel*	ActiveDocument.Envelope.Insert

ToolsCreateEnvelope *AddToDocument*	
	ActiveDocument.Envelope.PrintOut
ToolsCreateLabels *PrintEnvLabel*	Application.MailingLabel.PrintOut
ToolsCreateLabels *AddToDocument*	Application.MailingLabel.CreateNewDocument
ToolsCustomize *Tab*	With Dialogs(*WdWordDialog*) .DefaultTab = *WdWordDialogTab* .Show End With
ToolsCustomizeKeyboard *KeyCode*, *KeyCode2*, *Category*, *Name*, *Add*, *Remove*, *ResetAll*, *CommandValue*, *Context*	CustomizationContext = *template or document* KeyBindings.Add
	CustomizationContext = *template or document* FindKey(BuildKeyCode(*Wdkey*, *Wdkey*)).Disable
	CustomizationContext = *template or document* KeyBindings.ClearAll
ToolsCustomizeMenuBar *Context*, *Position*, *MenuType*, *MenuText*, *Menu*, *Add*, *Remove*, *Rename*	CustomizationContext = *template or document* CommandBars(*name*).Delete
	CommandBars.Add
	CommandBars(*name*).Name = *text*
ToolsCustomizeMenus *MenuType*, *Position*, *Category*, *Name*, *Menu*, *AddBelow*, *MenuText*, *Rename*, *Add*, *Remove*, *ResetAll*, *CommandValue*, *Context*	CustomizationContext = *template or document* CommandBars(*name*).Controls(*num*).Delete
	CommandBars(*name*).Controls.Add Type:=msoControlButton, ID:=*num*,
	CommandBars(*name*).Controls(*num*).Caption = *text*
ToolsGetSpelling, ToolsGetSpelling()	GetSpellingSuggestions
ToolsGetSynonyms, ToolsGetSynonyms()	SynonymInfo
ToolsGrammar	ActiveDocument.CheckGrammar
ToolsGrammarStatisticsArray	*' enumerate the ReadabilityStatistics collection* i = 1 For Each aStat In ActiveDocument.ReadabilityStatistics aArray(i) = aStat.Value i = i + 1 Next aStat
ToolsHyphenation *AutoHyphenation*, *HyphenateCaps*, *HyphenationZone*, *LimitConsecutiveHyphens*	With ActiveDocument .AutoHyphenation = True .HyphenateCaps = True .HyphenationZone = *num* .ConsecutiveHyphensLimit = *num* End With
ToolsHyphenationManual	ActiveDocument.ManualHyphenation
ToolsLanguage *Language*, *Default*	Selection.Range.LanguageID = *WdLanguageID*
	ActiveDocument.Styles(wdStyleNormal).LanguageID = *WdLanguageID*
ToolsMacro *Name*, *Run*, *Edit*, *Show*, *Delete*, *Rename*, *Description*, *NewName*, *SetDesc*	Application.Run
	Application.OrganizerDelete
	Application.OrganizerRename
	With Dialogs(wdDialogToolsMacro) .Show = "templateName" .Name = "macroName" .Edit = True .Execute End With

	With Dialogs(wdDialogToolsMacro) .Show = "templateName" .Name = "macroName" .Description = "newDescription" .SetDesc = True .Execute End With
ToolsManageFields	Application.Run MacroName:="ToolsManageFields"
ToolsMergeRevisions	ActiveDocument.Merge FileName:=*name*
ToolsNumberListDefault	Selection.Range.ListFormat.ApplyNumberDefault
ToolsOptions	Dialogs(*WdWordDialog*).Show
ToolsOptionsAutoFormat *PreserveStyles*, *ApplyStylesHeadings*, *ApplyStylesLists*, *ApplyStylesOtherParas*, *ReplaceQuotes*, *ReplaceSymbols*, *ApplyBulletedLists*, *ReplaceOrdinals*, *ReplaceFractions*, *ShowOptionsFor*	With Options .AutoFormatPreserveStyles = True .AutoFormatApplyHeadings = True .AutoFormatApplyLists = True .AutoFormatApplyOtherParas = True .AutoFormatReplaceQuotes = True .AutoFormatReplaceSymbols = True .AutoFormatApplyBulletedLists = True .AutoFormatReplaceOrdinals = True .AutoFormatReplaceFractions = True End With
ToolsOptionsAutoFormat *ApplyBorders*, *ApplyBulletedLists*, *ApplyStylesHeadings*, *ApplyNumberedLists*, *ReplaceFractions*, *ReplaceOrdinals*, *ReplaceQuotes*, *ReplaceSymbols*, *ShowOptionFor* There is no Visual Basic equivalent for the following arguments: *AdjustParaMarks*, *AdjustTabsSpaces*, *ReplaceBullets*, *AdjustEmptyParas*.	With Options .AutoFormatAsYouTypeApplyBorders = True .AutoFormatAsYouTypeApplyBulletedLists = True .AutoFormatAsYouTypeApplyHeadings = True .AutoFormatAsYouTypeApplyNumberedLists = True .AutoFormatAsYouTypeReplaceFractions = True .AutoFormatAsYouTypeReplaceOrdinals = True .AutoFormatAsYouTypeReplaceQuotes = True .AutoFormatAsYouTypeReplaceSymbols = True End With
ToolsOptionsCompatibility	ActiveDocument.Compatibility Type:=*WdCompatibility*
ToolsOptionsEdit *ReplaceSelection*, *DragAndDrop*, *AutoWordSelection*, *InsForPaste*, Overtype, *SmartCutPaste*, *AllowAccentedUppercase*, *PictureEditor*, *TabIndent*	With Options .ReplaceSelection = True .AllowDragAndDrop = True .AutoWordSelection = True .INSKeyForPaste = True .Overtype = True .SmartCutPaste = True .AllowAccentedUppercase = True .PictureEditor = *text* .TabIndentKey = True End With
ToolsOptionsFileLocations	Options.DefaultFilePath(*WdDefaultFilePath)* = *text*
ToolsOptionsGeneral *Pagination*, *WPHelp*, *WPDocNavKeys*, *BlueScreen*, *ErrorBeeps*, *UpdateLinks*, *SendMailAttach*, *Units*, *ButtonFieldClicks*, *ShortMenuNames*, *RTFInClipboard*, *ConfirmConversions*, *TipWizardActive*, *RecentFiles*, *RecentFileCount*	With Options .Pagination = True .WPHelp = True .WPDocNavKeys = True .BlueScreen = True .EnableSound = True .UpdateLinksAtOpen = True

	.SendMailAttach = True .MeasurementUnit = *WdUnits* .ButtonFieldClicks = *num* .ShortMenuNames = True .RTFInClipboard = True .ConfirmConversions = True End With Assistant.ActivateWizard With Application .DisplayRecentFiles = True .RecentFiles.Maximum = *num* End With
ToolsOptionsGrammar *Options*, *CheckSpelling*, *ShowStatistics*	With Options .CheckGrammarWithSpelling = True .ShowReadabilityStatistics = True End With ActiveDocument.ActiveWritingStyle(*language*) = *text*
ToolsOptionsPrint *Draft*, *Reverse*, *UpdateFields*, *Summary*, *ShowCodes*, *Annotations*, *ShowHidden*, *EnvFeederInstalled*, *UpdateLinks*, *Background*, *DrawingObjects*, *DefaultTray*, *FormsData*, *FractionalWidths*, *PSOverText*	With Options .PrintDraft = True .PrintReverse = True .UpdateFieldsAtPrint = True .PrintProperties = True .PrintFieldCodes = True .PrintComments = True .PrintHiddenText = True .EnvelopeFeederInstalled = True .UpdateLinksAtPrint = True .PrintBackground = True .PrintDrawingObjects = True .DefaultTray = *text* .DefaultTrayID = *WdPaperTray* End With With ActiveDocument .PrintFormsData = True .PrintFractionalWidths = True .PrintPostScriptOverText = True End With
ToolsOptionsRevisions *InsertedTextMark*, *DeletedTextMark*, *RevisedLinesMark*, *InsertedTextColor*, *DeletedTextColor*, *RevisedLinesColor*, *HighlightColor*	With Options .InsertedTextMark = *WdInsertedTextMark* .DeletedTextMark = *WdDeletedTextMark* .RevisedLinesMark = *WdRevisedLinesMark* .InsertedTextColor = *WdColorIndex* .DeletedTextColor = *WdColorIndex* .RevisedLinesColor = *WdColorIndex* .DefaultHighlightColorIndex = *WdColorIndex* End With
ToolsOptionsSave *CreateBackup*, *FastSaves*, *SummaryPrompt*, *GlobalDotPrompt*, *NativePictureFormat*, *AutoSave*, *SaveInterval*	With Options .CreateBackup = True .AllowFastSave = True .SavePropertiesPrompt = True .SaveNormalPrompt = True .BackgroundSave = True .SaveInterval = *number* End With

ToolsOptionsSave *FormsData*, *Password*, *WritePassword*, *RecommendReadOnly*, *EmbedFonts*	With ActiveDocument .SaveFormsData = True .Password = *text* .WritePassword = *text* .ReadOnlyRecommended = True .EmbedTrueTypeFonts = True End With
ToolsOptionsSpelling *AlwaysSuggest*, *SuggestFromMainDictOnly*, *IgnoreAllCaps*, *IgnoreMixedDigits*, *ResetIgnoreAll*, *Type*, *CustomDictn*, *AutomaticSpellChecking*, *HideSpellingErrors*, *RecheckDocument*	With Options .SuggestSpellingCorrections = True .SuggestFromMainDictionaryOnly = True .IgnoreUppercase = True .IgnoreMixedDigits = True .CheckSpellingAsYouType = True End With With ActiveDocument .SpellingChecked = False .ShowSpellingErrors = True End With Application.ResetIgnoreAll Languages(*wdLanguageID*).SpellingDictionaryType = *wdDictionaryType* CustomDictionaries.Add
ToolsOptionsUserInfo *Name*, *Initials*, *Address*	With Application .UserName = *text* .UserInitials = *text* .UserAddress = *text* End With
ToolsOptionsView *DraftFont*, *WrapToWindow*, *PicturePlaceHolders*, *FieldCodes*, *BookMarks*, *FieldShading*, *Hscroll*, *Vscroll*, *StyleAreaWidth*, *Tabs*, *Spaces*, *Paras*, *Hyphens*, *Hidden*, *ShowAll*, *Drawings*, *Anchors*, *TextBoundaries*, *Vruler*, *Highlight* ToolsOptionsView *StatusBar*	With ActiveWindow.View .Draft = True .WrapToWindow = True .ShowPicturePlaceHolders = True .ShowFieldCodes = True .ShowBookmarks = True .FieldShading = *WdFieldShading* .Parent.DisplayHorizontalScrollBar = True .Parent.DisplayVerticalScrollBar = True .Parent.StyleAreaWidth = *num* .ShowTabs = True .ShowSpaces = True .ShowParagraphs = True .ShowHyphens = True .ShowHiddenText = True .ShowAll = True .ShowDrawings = True .ShowObjectAnchors = True .ShowTextBoundaries = True .Parent.DisplayVerticalRuler = True .ShowHighlight = True End With Application.DisplayStatusBar = True
ToolsProtectDocument	ActiveDocument.Protect
ToolsProtectSection *Protect*, *Section*	ActiveDocument.Sections(*num*).ProtectedForForms = True
ToolsRemoveRecordDefault	WordBasic.ToolsRemoveRecordDefault

	' or if the data source is a Word table Selection.Tables(1).Rows(1).Delete
ToolsRepaginate	ActiveDocument.Repaginate
ToolsReviewRevisions *ShowMarks*, *HideMarks*, *Wrap*, *FindPrevious*, *FindNext*, *AcceptRevisions*, *RejectRevisions*	ActiveDocument.ShowRevisions = True Selection.NextRevision Selection.PreviousRevision Selection.Range.Revisions.AcceptAll Selection.Range.Revisions.RejectAll
ToolsRevisionAuthor$()	anAuthor = Selection.Range.Revisions(1).Author
ToolsRevisionDate$()	aDate = ActiveDocument.Revisions(1).Date
ToolsRevisions *MarkRevisions*, *ViewRevisions*, *PrintRevisions*, *AcceptAll*, *RejectAll*	With ActiveDocument .TrackRevisions = True .PrintRevisions = True .ShowRevisions = True End With With Selection.Range.Revisions .AcceptAll .RejectAll End With
ToolsRevisionType()	aType = ActiveDocument.Revisions(1).Type
ToolsShrinkToFit	ActiveDocument.FitToPages
ToolsSpelling	ActiveDocument.CheckSpelling
ToolsSpellingRecheckDocument	ActiveDocument.SpellingChecked = False
ToolsSpellSelection	Selection.Range.CheckSpelling
ToolsThesaurus	Selection.Range.CheckSynonyms
ToolsUnprotectDocument	ActiveDocument.UnProtect(Password:=*text*)
ToolsWordCount *CountFootnotes*, *Pages*, *Words*, *Characters*, *Paragraphs*, *Lines*	ActiveDocument.ComputeStatistics Statistic:=*WdStatistic*, IncludeFootnc

Visual Basic Equivalents for WordBasic Commands

U

UCase$(*string*)	UCase(*string*) *' or* UCase$(*string*)
Underline, Underline()	ActiveDocument.Words(1).Underline = True status = Selection.Font.Underline
UnHang	ActiveDocument.Paragraphs(1).TabHangingIndent Count:=-1
UnIndent	Selection.Paragraphs.TabIndent Count:=-1
UnlinkFields	Selection.Range.Fields.Unlink ActiveDocument.Fields(*num*).Unlink
UnlockFields	Selection.Paragraphs(1).Range.Fields.Locked = False ActiveDocument.Fields(*num*).Locked = False
UpdateFields	ActiveDocument.Sections(1).Range.Fields.Update ActiveDocument.Fields(*num*).Update
UpdateSource	ActiveDocument.Paragraphs(1).Range.Fields.UpdateSource ActiveDocument.Fields(*num*).UpdateSource

V

Val(*text*)	Val(*text*)
ViewAnnotations	ActiveWindow.View.SplitSpecial = wdPaneComments
ViewBorderToolbar	CommandBars("Borders").Visible = True
ViewDraft, ViewDraft()	ActiveWindow.View.Draft = True x = ActiveWindow.View.Draft
ViewDrawingToolbar	CommandBars("Drawing").Visible = True
ViewEndnoteArea, ViewEndnoteArea()	ActiveWindow.View.SplitSpecial = wdPaneEndnotes x = ActiveWindow.View.SplitSpecial
ViewEndnoteContNotice	ActiveWindow.View.SplitSpecial = wdPaneEndnoteContinuationNotice
ViewEndnoteContSeparator	ActiveWindow.View.SplitSpecial = wdPaneEndnoteContinuationSeparat
ViewEndnoteSeparator	ActiveWindow.View.SplitSpecial = wdPaneEndnoteSeparator
ViewFieldCodes	ActiveWindow.View.ShowFieldCodes = True

ViewFooter, ViewFooter()	ActiveWindow.View.SplitSpecial = wdPaneCurrentPageFooter ' *or* With ActiveWindow.View .Type = wdPageView .SeekView = wdSeekCurrentPageFooter End With *' use the* StoryType *property to return the active story/pane* aPane = Selection.StoryType
ViewFootnoteArea, ViewFootnoteArea()	ActiveWindow.View.SplitSpecial = wdPaneFootnotes x = ActiveWindow.View.SplitSpecial
ViewFootnoteContNotice	ActiveWindow.View.SplitSpecial = wdPaneFootnoteContinuationNotice
ViewFootnoteContSeparator	ActiveWindow.View.SplitSpecial = wdPaneFootnoteContinuationSeparat

ViewFootnotes, ViewFootnotes()	If ActiveDocument.Footnotes.Count >= 1 Then
	ActiveWindow.View.SplitSpecial = wdPaneFootnotes
	ElseIf ActiveDocument.Endnotes.Count >= 1 Then
	ActiveWindow.View.SplitSpecial = wdPaneEndnotes
	End If
	' Use the Information *property to determine if the selection is in a footnot*
	x = Selection.Information(wdInFootnoteEndnotePane)

ViewFootnoteSeparator	ActiveWindow.View.SplitSpecial = wdPaneFootnoteSeparator
ViewHeader, ViewHeader()	ActiveWindow.View.SplitSpecial = wdPaneCurrentPageHeader ' *or* With ActiveWindow.View .Type = wdPageView .SeekView = wdSeekCurrentPageHeader End With *' use the* StoryType *property to return the active story/pane* aPane = Selection.StoryType
ViewMasterDocument, ViewMasterDocument()	ActiveWindow.View.Type = wdMasterView aView = ActiveWindow.View.Type
ViewMenus()	Not applicable in Word 97
ViewNormal, ViewNormal()	ActiveWindow.View.Type =wdNormalView x = ActiveWindow.View.Type
ViewOutline, ViewOutline()	Windows(1).View.Type =wdOutlineView x = Windows(1).View.Type
ViewPage, ViewPage()	Windows(*name*).View.Type =wdPageView x = Windows(*name*).View.Type
ViewRibbon, ViewRibbon()	CommandBars("Formatting").Visible = True x = CommandBars("Formatting").Visible
ViewRuler, ViewRuler()	ActiveWindow.DisplayRulers = True x = ActiveWindow.DisplayRulers
ViewStatusBar, ViewStatusBar()	Application.DisplayStatusBar = True x = Application.DisplayStatusBar

ViewToggleMasterDocument	If ActiveWindow.View.Type = wdOutlineView Then
	ActiveWindow.View.Type = wdMasterView
	ElseIf ActiveWindow.View.Type = wdMasterView Then
	ActiveWindow.View.Type = wdOutlineView
	End If

ViewToolbars *LargeButtons*, *ToolTips*,	With CommandBars

ToolTipsKey, *Reset*, *Delete*, *Show*	.LargeButtons = True .DisplayToolTips = True .DisplayKeysInToolTips = True End With CommandBars(*name*).Reset CommandBars(*name*).Delete CommandBars(*name*).Visible = True
ViewZoom *AutoFit*	Windows(*name*).View.Zoom.PageFit = wdPageFitBestFit
ViewZoom *TwoPages*	With ActiveWindow.View.Zoom .PageColumns = 2 .PageRows = 1 End With
ViewZoom *FullPage*	ActiveWindow.View.Zoom.PageFit = wdPageFitFullPage
ViewZoom *NumColumns*, *NumRows*	With ActiveWindow.View.Zoom .PageColumns = *num* .PageRows = *num* End With
ViewZoom *ZoomPercent*	ActiveWindow.View.Zoom.Percentage = *num*
ViewZoom100	Windows(1).View.Zoom.Percentage = 100
ViewZoom200	ActiveWindow.View.Zoom.Percentage = 200
ViewZoom75	ActiveWindow.View.Zoom.Percentage = 75
ViewZoomPageWidth	Windows(*name*).View.Zoom.PageFit = wdPageFitBestFit
ViewZoomWholePage	ActiveWindow.View.Zoom.PageFit = wdPageFitFullPage
VLine	ActiveWindow.SmallScroll Down:=*num* *' or* ActiveWindow.SmallScroll Up:=*num*
VPage	ActiveWindow.LargeScroll Down:=*num* *' or* ActiveWindow.LargeScroll Up:=*num*
VScroll, VScroll()	ActiveWindow.VerticalPercentScrolled = *num* num = ActiveWindow.VerticalPercentScrolled

Visual Basic Equivalents for WordBasic Commands

W

WaitCursor	System.Cursor = *WdCursorType*
Weekday(*date*)	Weekday(*date*)
While...Wend	While...Wend
num = Window()	num = ActiveWindow.Index
WindowArrangeAll	Windows.Arrange ArrangeStyle:=wdTiled
WindowList *num*	Window(*num*).Activate
WindowName$()	aCap = ActiveWindow.Caption
WindowNewWindow	Windows.Add *' or* ActiveWindow.NewWindow
Window *num*	Window(*num*).Activate
WindowPane()	Use the Split property to determine if a Window is split. Use StoryType property with the Selection object to determine the pane/
WinToDOS$()	x = WordBasic.[WinToDOS$](*StringToTranslate*)
WordLeft *count*	Selection.MoveLeft Unit:=wdWord, Count:=1, Extend:=wdMove
WordLeft *count*, *select*	Selection.MoveStart Unit:=wdWord, Count:=-1 *' or* Selection.MoveLeft Unit:=WdWord, Count:=1, Extend:=wdExtend
WordRight 1	Selection.MoveRight Unit:=wdWord, Count:=1, Extend:=wdMove
WordRight 1, 1	Selection.MoveEnd Unit:=wdWord, Count:=1 *' or* Selection.MoveRight Unit:=WdWord, Count:=1, Extend:=wdExtend
WordUnderline, WordUnderline()	Selection.Range.Underline = wdUnderlineWords status = Selection.Range.Underline
Write	Write

Y
Year Year()

Built-in dialog box argument lists

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowDialogArgumentsC"}

Many of the built-in Word dialog boxes have arguments that you can use to set or get values from a dialog. For more information, see Displaying built-in Word dialog boxes.

WdWordDialog constant	Argument list(s)
wdDialogConnect	Drive, Path, Password
wdDialogControlRun	Application
wdDialogConvertObject	IconNumber, ActivateAs, IconFileName, Caption, Class, DisplayIcon, Floating
wdDialogCopyFile	FileName, Directory
wdDialogDocumentStatistics	FileName, Directory, Template, Title, Created, LastSaved, LastSavedBy, Revision, Time, Printed, Pages, Words, Characters, Paragraphs, Lines, FileSize
wdDialogDrawAlign	Horizontal, Vertical, RelativeTo
wdDialogDrawSnapToGrid	SnapToGrid, XGrid, YGrid, XOrigin, YOrigin, SnapToShapes
wdDialogEditAutoText	Name, Context, InsertAs, Insert, Add, Define, InsertAsText, Delete, CompleteAT
wdDialogEditAutoText	Name, Context, InsertAs, Insert, Add, Define, InsertAsText, Delete, CompleteAT
wdDialogEditFind	Replace, Direction, MatchCase, WholeWord, PatternMatch, SoundsLike, FindNext, ReplaceOne, ReplaceAll, Format, Wrap, FindAllWordForms, MatchByte, FuzzyFind, Destination
wdDialogEditGoTo	Destination
wdDialogEditGoTo	Find, Replace, Direction, MatchCase, WholeWord, PatternMatch, SoundsLike, FindNext, ReplaceOne, ReplaceAll, Format, Wrap, FindAllWordForms, MatchByte, FuzzyFind, Destination
wdDialogEditLinks	UpdateMode, Locked, SavePictureInDoc, UpdateNow, OpenSource, KillLink, Link, Application, Item, FileName
wdDialogEditObject	Verb
wdDialogEditPasteSpecial	IconNumber, Link, DisplayIcon, Class, DataType, IconFileName, Caption, Floating
wdDialogEditReplace	Find, Replace, Direction, MatchCase, WholeWord, PatternMatch, SoundsLike, FindNext, ReplaceOne, ReplaceAll, Format, Wrap, FindAllWordForms, MatchByte, FuzzyFind, Destination
wdDialogEditTOACategory	Category, CategoryName
wdDialogFileFind	SearchName, SearchPath, Name, SubDir, Title, Author, Keywords, Subject, Options, MatchCase, Text, PatternMatch, DateSavedFrom, DateSavedTo, SavedBy, DateCreatedFrom, DateCreatedTo, View, SortBy, ListBy, SelectedFile, Add, Delete, ShowFolders, MatchByte
wdDialogFileNew	Template, NewTemplate
wdDialogFileOpen	Name, ConfirmConversions, ReadOnly, LinkToSource, AddToMru, PasswordDoc, PasswordDot, Revert, WritePasswordDoc, WritePasswordDot, Connection, SQLStatement, SQLStatement1, Format
wdDialogFilePageSetup	Tab, PaperSize, TopMargin, BottomMargin,

	LeftMargin, RightMargin, Gutter, PageWidth, PageHeight, Orientation, FirstPage, OtherPages, VertAlign, ApplyPropsTo, Default, FacingPages, HeaderDistance, FooterDistance, SectionStart, OddAndEvenPages, DifferentFirstPage, Endnotes, LineNum, StartingNum, FromText, CountBy, NumMode, TwoOnOne, GutterPosition, CharsLine, LinesPage, LinePitch, CharPitch, DocFontName, DocFontSize, PageColumns, TextFlow, FirstPageOnLeft
wdDialogFilePrint	Background, AppendPrFile, Range, PrToFileName, From, To, Type, NumCopies, Pages, Order, PrintToFile, Collate, FileName, Printer, OutputPrinter, DuplexPrint
wdDialogFilePrintSetup	Printer, Options, Network, DoNotSetAsSysDefault
wdDialogFileRoutingSlip	Subject, Message, AllAtOnce, ReturnWhenDone, TrackStatus, Protect, AddSlip, RouteDocument, AddRecipient, OldRecipient, ResetSlip, ClearSlip, ClearRecipients, Address
wdDialogFileSaveAs	Name, Format, LockAnnot, Password, AddToMru, WritePassword, RecommendReadOnly, EmbedFonts, NativePictureFormat, FormsData, SaveAsAOCELetter, WriteVersion, VersionDesc
wdDialogFileSummaryInfo	Title, Subject, Author, Keywords, Comments, FileName, Directory, Template, CreateDate, LastSavedDate, LastSavedBy, RevisionNumber, EditTime, LastPrintedDate, NumPages, NumWords, NumChars, NumParas, NumLines, Update, FileSize
wdDialogFileVersions	AutoVersion, VersionDesc
wdDialogFontSubstitution	UnavailableFont, SubstituteFont
wdDialogFormatAddrFonts	Points, Underline, Color, StrikeThrough, Superscript, Subscript, Hidden, SmallCaps, AllCaps, Spacing, Position, Kerning, KerningMin, Default, Tab, Font, Bold, Italic, DoubleStrikeThrough, Shadow, Outline, Emboss, Engrave, Scale, Animations, CharAccent, FontMajor, FontLowAnsi, FontHighAnsi, CharacterWidthGrid
wdDialogFormatBordersAndShading	ApplyTo, Shadow, TopBorder, LeftBorder, BottomBorder, RightBorder, HorizBorder, VertBorder, TopColor, LeftColor, BottomColor, RightColor, HorizColor, VertColor, FromText, Shading, Foreground, Background, Tab, FineShading, TopStyle, LeftStyle, BottomStyle, RightStyle, HorizStyle, VertStyle, TopWeight, LeftWeight, BottomWeight, RightWeight, HorizWeight, VertWeight, BorderObjectType, BorderArtWeight, BorderArt, FromTextTop, FromTextBottom, FromTextLeft, FromTextRight, OffsetFrom, WhichPages, InFront, SurroundHeader, SurroundFooter, JoinBorder
wdDialogFormatCallout	Type, Gap, Angle, Drop, Length, Border, AutoAttach, Accent
wdDialogFormatChangeCase	Type
wdDialogFormatColumns	Columns, ColumnNo, ColumnWidth, ColumnSpacing, EvenlySpaced, ApplyColsTo, ColLine, StartNewCol
wdDialogFormatDrawingObject	Left, PositionHorzRel, Top, PositionVertRel, LockAnchor, FloatOverText, Wrap, WrapSide, TopDistanceFromText, BottomDistanceFromText, LeftDistanceFromText, RightDistanceFromText
Use the **DefaultTab** property to control which tab is displayed	

wdDialogFormatDropCap	Position, Font, DropHeight, DistFromText
wdDialogFormatFont	Points, Underline, Color, StrikeThrough, Superscript, Subscript, Hidden, SmallCaps, AllCaps, Spacing, Position, Kerning, KerningMin, Default, Tab, Font, Bold, Italic, DoubleStrikeThrough, Shadow, Outline, Emboss, Engrave, Scale, Animations, CharAccent, FontMajor, FontLowAnsi, FontHighAnsi, CharacterWidthGrid
wdDialogFormatFrame	Wrap, WidthRule, FixedWidth, HeightRule, FixedHeight, PositionHorz, PositionHorzRel, DistFromText, PositionVert, PositionVertRel, DistVertFromText, MoveWithText, LockAnchor, RemoveFrame
wdDialogFormatPageNumber	ChapterNumber, NumRestart, NumFormat, StartingNum, Level, Separator
wdDialogFormatParagraph	LeftIndent, RightIndent, Before, After, LineSpacingRule, LineSpacing, Alignment, WidowControl, KeepWithNext, KeepTogether, PageBreak, NoLineNum, DontHyphen, Tab, FirstIndent, OutlineLevel
wdDialogFormatPicture	SetSize, CropLeft, CropRight, CropTop, CropBottom, ScaleX, ScaleY, SizeX, SizeY
wdDialogFormatRetAddrFonts	Points, Underline, Color, StrikeThrough, Superscript, Subscript, Hidden, SmallCaps, AllCaps, Spacing, Position, Kerning, KerningMin, Default, Tab, Font, Bold, Italic, DoubleStrikeThrough, Shadow, Outline, Emboss, Engrave, Scale, Animations, CharAccent, FontMajor, FontLowAnsi, FontHighAnsi, CharacterWidthGrid
wdDialogFormatSectionLayout	SectionStart, VertAlign, Endnotes, LineNum, StartingNum, FromText, CountBy, NumMode
wdDialogFormatStyle	Name, Delete, Merge, NewName, BasedOn, NextStyle, Type, FileName, Source, AddToTemplate, Define, Rename, Apply, New
wdDialogFormatStyleGallery	Template, Preview
wdDialogFormatTabs	Position, DefTabs, Align, Leader, Set, Clear, ClearAll
wdDialogFormFieldOptions	Entry, Exit, Name, Enable, TextType, TextWidth, TextDefault, TextFormat, CheckSize, CheckWidth, CheckDefault, Type, OwnHelp, HelpText, OwnStat, StatText, Calculate
wdDialogHelpWordPerfectHelp	WPCommand, HelpText, DemoGuidance
wdDialogHelpWordPerfectHelpOptions	CommandKeyHelp, DocNavKeys, MouseSimulation, DemoGuidance, DemoSpeed, HelpType
wdDialogInsertAddCaption	Name
wdDialogInsertAutoCaption	Clear, ClearAll, Object, Label, Position
wdDialogInsertBookmark	Name, SortBy, Add, Delete, Goto, Hidden
wdDialogInsertBreak	Type
wdDialogInsertCaption	Label, TitleAutoText, Title, Delete, Position, AutoCaption
wdDialogInsertCaptionNumbering	Label, FormatNumber, ChapterNumber, Level, Separator
wdDialogInsertCrossReference	ReferenceType, ReferenceKind, ReferenceItem, InsertAsHyperLink, InsertPosition
wdDialogInsertDatabase	Format, Style, LinkToSource, Connection, SQLStatement, SQLStatement1, PasswordDoc, PasswordDot, DataSource, From, To, IncludeFields,

	WritePasswordDoc, WritePasswordDot
wdDialogInsertDateTime	DateTimePic, DbCharField, InsertAsField
wdDialogInsertField	Field
wdDialogInsertFile	Name, Range, ConfirmConversions, Link, Attachment
wdDialogInsertFootnote	Reference, NoteType, Symbol
wdDialogInsertFootnote	Reference, NoteType, Symbol
wdDialogInsertFormField	Entry, Exit, Name, Enable, TextType, TextWidth, TextDefault, TextFormat, CheckSize, CheckWidth, CheckDefault, Type, OwnHelp, HelpText, OwnStat, StatText, Calculate
wdDialogInsertIndex	Outline, Fields, From, To, TableId, AddedStyles, Caption, HeadingSeparator, Replace, MarkEntry, AutoMark, MarkCitation, Type, RightAlignPageNumbers, Passim, KeepFormatting, Columns, Category, Label, ShowPageNumbers, AccentedLetters, Filter, SortBy, Leader
wdDialogInsertIndexAndTables	Outline, Fields, From, To, TableId, AddedStyles, Caption, HeadingSeparator, Replace, MarkEntry, AutoMark, MarkCitation, Type, RightAlignPageNumbers, Passim, KeepFormatting, Columns, Category, Label, ShowPageNumbers, AccentedLetters, Filter, SortBy, Leader
wdDialogInsertMergeField	MergeField, WordField
wdDialogInsertObject	IconNumber, FileName, Link, DisplayIcon, Tab, Class, IconFileName, Caption, Floating
wdDialogInsertPageNumbers	Type, Position, FirstPage
wdDialogInsertPicture	Name, LinkToFile, New, FloatOverText
wdDialogInsertSubdocument	Name, ConfirmConversions, ReadOnly, LinkToSource, AddToMru, PasswordDoc, PasswordDot, Revert, WritePasswordDoc, WritePasswordDot, Connection, SQLStatement, SQLStatement1, Format
wdDialogInsertSymbol	Font, Tab, CharNum, Unicode
wdDialogInsertTableOfAuthorities	Outline, Fields, From, To, TableId, AddedStyles, Caption, HeadingSeparator, Replace, MarkEntry, AutoMark, MarkCitation, Type, RightAlignPageNumbers, Passim, KeepFormatting, Columns, Category, Label, ShowPageNumbers, AccentedLetters, Filter, SortBy, Leader
wdDialogInsertTableOfContents	Outline, Fields, From, To, TableId, AddedStyles, Caption, HeadingSeparator, Replace, MarkEntry, AutoMark, MarkCitation, Type, RightAlignPageNumbers, Passim, KeepFormatting, Columns, Category, Label, ShowPageNumbers, AccentedLetters, Filter, SortBy, Leader
wdDialogInsertTableOfContents	Source, From, To, Replace
wdDialogInsertTableOfFigures	Outline, Fields, From, To, TableId, AddedStyles, Caption, HeadingSeparator, Replace, MarkEntry, AutoMark, MarkCitation, Type, RightAlignPageNumbers, Passim, KeepFormatting, Columns, Category, Label, ShowPageNumbers, AccentedLetters, Filter, SortBy, Leader
wdDialogLetterWizard	DateFormat, IncludeHeaderFooter, LetterStyle, Letterhead, LetterheadLocation, LetterheadSize, RecipientName, RecipientAddress, Salutation, SalutationType, RecipientReference, MailingInstructions, AttentionLine, LetterSubject, CCList, SenderName, ReturnAddress, Closing,

	SenderJobTitle, SenderCompany, SenderInitials, EnclosureNumber, InfoBlock, RecipientCode, RecipientGender, ReturnAddressSF, SenderCity, SenderCode, SenderGender, SenderReference, PageDesign
wdDialogListCommands	ListType
wdDialogMailMerge	CheckErrors, Destination, MergeRecords, From, To, Suppression, MailMerge, QueryOptions, MailSubject, MailAsAttachment, MailAddress
wdDialogMailMergeCheck	CheckErrors
wdDialogMailMergeCreateDataSource	FileName, PasswordDoc, PasswordDot, HeaderRecord, MSQuery, SQLStatement, SQLStatement1, Connection, LinkToSource, WritePasswordDoc
wdDialogMailMergeCreateDataSource	FileName, PasswordDoc, PasswordDot, HeaderRecord, MSQuery, SQLStatement, SQLStatement1, Connection, LinkToSource, WritePasswordDoc
wdDialogMailMergeCreateHeaderSource	FileName, PasswordDoc, PasswordDot, HeaderRecord, MSQuery, SQLStatement, SQLStatement1, Connection, LinkToSource, WritePasswordDoc
wdDialogMailMergeCreateHeaderSource	FileName, PasswordDoc, PasswordDot, HeaderRecord, MSQuery, SQLStatement, SQLStatement1, Connection, LinkToSource, WritePasswordDoc
wdDialogMailMergeFindRecord	Find, Field
wdDialogMailMergeHelper	Merge, Options
wdDialogMailMergeInsertAsk	Name, Prompt, DefaultBookmarkText, AskOnce
wdDialogMailMergeInsertFillIn	Prompt, DefaultFillInText, AskOnce
wdDialogMailMergeInsertIf	MergeField, Comparison, CompareTo, TrueAutoText, TrueText, FalseAutoText, FalseText
wdDialogMailMergeInsertSet	Name, ValueText, ValueAutoText
wdDialogMailMergeInsertSkipIf	MergeField, Comparison, CompareTo
wdDialogMailMergeNextIf	MergeField, Comparison, CompareTo
wdDialogMailMergeOpenDataSource	Name, ConfirmConversions, ReadOnly, LinkToSource, AddToMru, PasswordDoc, PasswordDot, Revert, WritePasswordDoc, WritePasswordDot, Connection, SQLStatement, SQLStatement1, Format
wdDialogMailMergeOpenHeaderSource	Name, ConfirmConversions, ReadOnly, LinkToSource, AddToMru, PasswordDoc, PasswordDot, Revert, WritePasswordDoc, WritePasswordDot, Connection, SQLStatement, SQLStatement1, Format
wdDialogMailMergeQueryOptions	SQLStatement, SQLStatement1
wdDialogMailMergeUseAddressBook	AddressBookType
wdDialogMarkCitation	LongCitation, LongCitationAutoText, Category, ShortCitation, NextCitation, Mark, MarkAll
wdDialogMarkIndexEntry	MarkAll, Entry, Range, Bold, Italic, CrossReference, EntryAutoText, CrossReferenceAutoText
wdDialogMarkIndexEntry	Entry, EntryAutoText, TableId, Level
wdDialogNewToolbar	Name, Context
wdDialogNoteOptions	FootnotesAt, StartingNum, RestartNum, Separator, ContSeparator, ContNotice
wdDialogNoteOptions	FootnotesAt, FootNumberAs, FootStartingNum, FootRestartNum, EndnotesAt, EndNumberAs,

	EndStartingNum, EndRestartNum
wdDialogOrganizer	Copy, Delete, Rename, Source, Destination, Name, NewName, Tab
wdDialogTableAutoFormat	HideAutoFit, Preview, Format, Borders, Shading, Font, Color, AutoFit, HeadingRows, FirstColumn, LastRow, LastColumn
wdDialogTableDeleteCells	ShiftCells
wdDialogTableFormatCell	Category
wdDialogTableFormula	Formula, NumFormat
wdDialogTableInsertCells	ShiftCells
wdDialogTableInsertRows	NumRows
wdDialogTableInsertTable	ConvertFrom, NumColumns, NumRows, InitialColWidth, Wizard, Format, Apply
wdDialogTableSort	DontSortHdr, FieldNum, Type, Order, FieldNum2, Type2, Order2, FieldNum3, Type3, Order3, Separator, SortColumn, CaseSensitive, Language
wdDialogTableSplitCells	NumColumns, NumRows, MergeBeforeSplit
wdDialogTableToText	ConvertTo
wdDialogTextToTable	ConvertFrom, NumColumns, NumRows, InitialColWidth, Wizard, Format, Apply
wdDialogToolsAcceptRejectChanges	ShowMarks, HideMarks, Wrap, FindPrevious, FindNext, AcceptRevisions, RejectRevisions, AcceptAll, RejectAll
wdDialogToolsAdvancedSettings	Application, Option, Setting, Delete, Set
wdDialogToolsAutoCorrect	InitialCaps, SentenceCaps, Days, CapsLock, ReplaceText, Formatting, Replace, With, Add, Delete, SmartQuotes, ConvBrackets, ConvQuotes, ConvPunct
wdDialogToolsAutoCorrectExceptions	Tab, Name, AutoAdd, Add, Delete
wdDialogToolsAutoSummarize	TextSize, Show, Update
wdDialogToolsBulletsNumbers	Replace, Font, CharNum, Type, FormatOutline, AutoUpdate, FormatNumber, Punctuation, StartAt, Points, Hang, Indent, Remove
wdDialogToolsCompareDocuments	Name
wdDialogToolsCreateEnvelope	ExtractAddress, LabelListIndex, LabelIndex, LabelDotMatrix, LabelTray, LabelAcross, LabelDown, EnvAddress, EnvOmitReturn, EnvReturn, PrintBarCode, SingleLabel, LabelRow, LabelColumn, PrintEnvLabel, AddToDocument, EnvWidth, EnvHeight, EnvPaperSize, PrintFIMA, UseEnvFeeder, Tab, AddrAutoText, AddrText, AddrFromLeft, AddrFromTop, RetAddrFromLeft, RetAddrFromTop, LabelTopMargin, LabelSideMargin, LabelVertPitch, LabelHorPitch, LabelHeight, LabelWidth, CustomName, RetAddrText, EnvPaperName, DefaultFaceUp, DefaultOrientation, RetAddrAutoText
wdDialogToolsCreateLabels	ExtractAddress, LabelListIndex, LabelIndex, LabelDotMatrix, LabelTray, LabelAcross, LabelDown, EnvAddress, EnvOmitReturn, EnvReturn, PrintBarCode, SingleLabel, LabelRow, LabelColumn, PrintEnvLabel, AddToDocument, EnvWidth, EnvHeight, EnvPaperSize, PrintFIMA, UseEnvFeeder, Tab, AddrAutoText, AddrText, AddrFromLeft, AddrFromTop, RetAddrFromLeft, RetAddrFromTop, LabelTopMargin, LabelSideMargin, LabelVertPitch, LabelHorPitch, LabelHeight, LabelWidth,

	CustomName, RetAddrText, EnvPaperName, DefaultFaceUp, DefaultOrientation, RetAddrAutoText
wdDialogToolsCustomize	KeyCode, KeyCode2, MenuType, Position, AddAll, Category, Name, Menu, AddBelow, MenuText, Rename, Add, Remove, ResetAll, CommandValue, Context, Tab
wdDialogToolsCustomize	KeyCode, KeyCode2, MenuType, Position, AddAll, Category, Name, Menu, AddBelow, MenuText, Rename, Add, Remove, ResetAll, CommandValue, Context, Tab
wdDialogToolsCustomizeKeyboard	KeyCode, KeyCode2, MenuType, Position, AddAll, Category, Name, Menu, AddBelow, MenuText, Rename, Add, Remove, ResetAll, CommandValue, Context, Tab
wdDialogToolsCustomizeMenuBar	Context, Position, MenuType, MenuText, Menu, Add, Remove, Rename
wdDialogToolsCustomizeMenus	KeyCode, KeyCode2, MenuType, Position, AddAll, Category, Name, Menu, AddBelow, MenuText, Rename, Add, Remove, ResetAll, CommandValue, Context, Tab
wdDialogToolsEnvelopesAndLabels	ExtractAddress, LabelListIndex, LabelIndex, LabelDotMatrix, LabelTray, LabelAcross, LabelDown, EnvAddress, EnvOmitReturn, EnvReturn, PrintBarCode, SingleLabel, LabelRow, LabelColumn, PrintEnvLabel, AddToDocument, EnvWidth, EnvHeight, EnvPaperSize, PrintFIMA, UseEnvFeeder, Tab, AddrAutoText, AddrText, AddrFromLeft, AddrFromTop, RetAddrFromLeft, RetAddrFromTop, LabelTopMargin, LabelSideMargin, LabelVertPitch, LabelHorPitch, LabelHeight, LabelWidth, CustomName, RetAddrText, EnvPaperName, DefaultFaceUp, DefaultOrientation, RetAddrAutoText
wdDialogToolsHyphenation	AutoHyphenation, HyphenateCaps, HyphenationZone, LimitConsecutiveHyphens
wdDialogToolsHyphenation	HyphenateCaps, Confirm, HotZone
wdDialogToolsLanguage	Language, Default
wdDialogToolsMacro	Name, Run, Edit, Show, Delete, Rename, Description, NewName, SetDesc
wdDialogToolsManageFields	FieldName, Add, Remove, Rename, NewName
wdDialogToolsMergeDocuments	Name
wdDialogToolsOptionPrint	Draft, Reverse, UpdateFields, Summary, ShowCodes, Annotations, ShowHidden, EnvFeederInstalled, WidowControl, DfltTrueType, UpdateLinks, Background, DrawingObjects, FormsData, DefaultTray, PSOverText, MapPaperSize, FractionalWidths, PrOrder1, PrOrder2
wdDialogToolsOptions	Tab
wdDialogToolsOptions	Tab
Use the **DefaultTab** property to control which tab is displayed	
wdDialogToolsOptionsAutoFormat	ShowOptionsFor, ApplyStylesHeadings, ApplyBorders, ApplyStylesLists, ApplyBulletedLists, ApplyNumberedLists, ApplyStylesOtherParas, ReplaceQuotes, ReplaceOrdinals, ReplaceFractions, ReplaceSymbols, ReplaceBullets, AdjustParaMarks, AdjustTabsSpaces, AdjustEmptyParas, PreserveStyles
wdDialogToolsOptionsCompatibility	Product, Default, NoTabHangIndent,

	NoSpaceRaiseLower, PrintColBlack, WrapTrailSpaces, NoColumnBalance, ConvMailMergeEsc, SuppressSpBfAfterPgBrk, SuppressTopSpacing, OrigWordTableRules, TransparentMetafiles, ShowBreaksInFrames, SwapBordersFacingPages, LeaveBackslashAlone, ExpandShiftReturn, DontULTrailSpace, DontBalanceSbDbWidth, SuppressTopSpacingMac5, SpacingInWholePoints, PrintBodyTextBeforeHeader, NoLeading, NoSpaceForUL, MWSmallCaps, NoExtraLineSpacing, TruncateFontHeight, SubFontBySize, UsePrinterMetrics, WW6BorderRules, ExactOnTop, SuppressBottomSpacing, WPSpaceWidth, WPJustification, LineWrapLikeWord6
wdDialogToolsOptionsEdit	ReplaceSelection, DragAndDrop, AutoWordSelection, InsForPaste, Overtype, SmartCutPaste, AllowAccentedUppercase, PictureEditor, TabIndent
wdDialogToolsOptionsFileLocations	Path, Setting
wdDialogToolsOptionsGeneral	TipWizardActive, Pagination, WPHelp, WPDocNavKeys, BlueScreen, ErrorBeeps, Effects3d, UpdateLinks, SendMailAttach, RecentFiles, RecentFileCount, Units, ButtonFieldClicks, AnimatedCursors, ConfirmConversions, VirusProtection, ShortMenuNames, RTFInClipboard
wdDialogToolsOptionsPrint	Draft, Reverse, UpdateFields, Summary, ShowCodes, Annotations, ShowHidden, EnvFeederInstalled, WidowControl, DfltTrueType, UpdateLinks, Background, DrawingObjects, FormsData, DefaultTray, PSOverText, MapPaperSize, FractionalWidths, PrOrder1, PrOrder2
wdDialogToolsOptionsSave	CreateBackup, FastSaves, SummaryPrompt, GlobalDotPrompt, NativePictureFormat, EmbedFonts, FormsData, AutoSave, SaveInterval, Password, WritePassword, RecommendReadOnly, SubsetFonts, BackgroundSave, DefaultSaveFormat
wdDialogToolsOptionsSpellingAndGrammar	AlwaysSuggest, SuggestFromMainDictOnly, IgnoreAllCaps, IgnoreMixedDigits, ResetIgnoreAll, Type, CustomDict1, CustomDict2, CustomDict3, CustomDict4, CustomDict5, CustomDict6, CustomDict7, CustomDict8, CustomDict9, CustomDict10, AutomaticSpellChecking, FilenamesEmailAliases, UserDict1, AutomaticGrammarChecking, ForegroundGrammar, ShowStatistics, Options, RecheckDocument, IgnoreAuxFind, IgnoreMissDictSearch, HideGrammarErrors, CheckSpelling, GrLidUI, SpLidUI, DictLang1, DictLang2, DictLang3, DictLang4, DictLang5, DictLang6, DictLang7, DictLang8, DictLang9, DictLang10, HideSpellingErrors
wdDialogToolsOptionsTrackChanges	InsertedTextMark, InsertedTextColor, DeletedTextMark, DeletedTextColor, RevisedLinesMark, RevisedLinesColor, HighlightColor, RevisedPropertiesMark, RevisedPropertiesColor
wdDialogToolsOptionsUserInfo	Name, Initials, Address
wdDialogToolsOptionsView	DraftFont, WrapToWindow, PicturePlaceHolders, FieldCodes, BookMarks, FieldShading, StatusBar, HScroll, VScroll, StyleAreaWidth, Tabs, Spaces, Paras, Hyphens, Hidden, ShowAll, Drawings, Anchors, TextBoundaries, VRuler, Highlight, ShowAnimation,

	ScrnTp, EnlargeFontsLessThan, BrowseToWindow, OptionalBreak
wdDialogToolsProtectDocument	DocumentPassword, NoReset, Type
wdDialogToolsProtectSection	Protect, Section
wdDialogToolsRevisions	MarkRevisions, ViewRevisions, PrintRevisions, AcceptAll, RejectAll
wdDialogToolsSpellingAndGrammar	ForegroundGrammar, SuggestionListBox
wdDialogToolsSpellingAndGrammar	ForegroundGrammar, SuggestionListBox
wdDialogToolsTemplates	Store, Template, LinkStyles
wdDialogToolsUnprotectDocument	DocumentPassword
wdDialogToolsWordCount	CountFootnotes, Pages, Words, Characters, CharactersIncludingSpaces, Paragraphs, Lines
wdDialogViewZoom	CustomZoomPercent, ZoomPercent, AutoFit, FullPage
wdDialogViewZoom	AutoFit, TwoPages, FullPage, NumColumns, NumRows, ZoomPercent
With Dialogs(wdDialogFormatBulletsAndNumbering) .DefaultTab = wdDialogFormatBulletsAndNumberingTabBulleted .Show End With	Points, Color, Before, Type, After, StartAt, Include, Alignment, Indent, Space, Hang, RestartNum, Level, CharNum, Font, StrikeThrough, Bold, Italic, Underline, Remove
With Dialogs(wdDialogFormatBulletsAndNumbering) .DefaultTab = wdDialogFormatBulletsAndNumberingTabNumbered .Show End With	Points, Color, Before, Type, After, StartAt, Include, Alignment, Indent, Space, Hang, RestartNum, Level, CharNum, Font, StrikeThrough, Bold, Italic, Underline, Remove
With Dialogs(wdDialogFormatBulletsAndNumbering) .DefaultTab = wdDialogFormatBulletsAndNumberingTabOutlineNumbered .Show End With	Points, Color, Before, Type, After, StartAt, Include, Alignment, Indent, Space, Hang, RestartNum, Level, CharNum, Font, StrikeThrough, Bold, Italic, Underline, Remove
With Dialogs(wdDialogFormatDrawingObject) .DefaultTab = wdDialogFormatDrawingObjectTabWrapping .Show End With	Left, PositionHorzRel, Top, PositionVertRel, LockAnchor, FloatOverText, Wrap, WrapSide, TopDistanceFromText, BottomDistanceFromText, LeftDistanceFromText, RightDistanceFromText
With Dialogs(wdDialogToolsAutoManager) .DefaultTab = wdDialogToolsAutoManagerTabAutoFormatAsYouType .Show End With	ApplyStylesHeadings, ApplyBorders, ApplyBulletedLists, ApplyNumberedLists, ApplyTables, ReplaceQuotes, ReplaceOrdinals, ReplaceFractions, ReplaceSymbols, ReplacePlainTextEmphasis, ReplaceHyperlinks, FormatListItemBeginning, DefineStyles, ApplyFirstIndent, ApplyDates, ApplyClosings, MatchParentheses, ReplaceDbDashes, ReplaceAutoSpaces, InsertClosings, AutoLetterWizard, InsertOvers, ShowOptionsFor, ApplyStylesLists, ApplySkipList, ApplyStylesOtherParas, ReplaceBullets, AdjustParaMarks, AdjustTabsSpaces, AdjustEmptyParas, PreserveStyles
With Dialogs(wdDialogToolsAutoManager) .DefaultTab = wdDialogToolsAutoManagerTabAutoFormat .Show End With	ApplyStylesHeadings, ApplyStylesLists, ApplyBulletedLists, ApplyStylesOtherParas, ReplaceQuotes, ReplaceOrdinals, ReplaceFractions, ReplaceSymbols, ReplacePlainTextEmphasis, ReplaceHyperlinks, PreserveStyles, PlainTextWordMail, ApplyFirstIndent, MatchParentheses, ReplaceDbDashes,

ReplaceAutoSpaces

Displaying built-in Word dialog boxes

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowDisplayBuiltinDialogsC"}

You can display a built-in dialog box to get user input or to control Word by using Visual Basic. The **Show** method of the **Dialog** object displays and executes any action taken in a built-in Word dialog box. To access a particular built-in Word dialog box, you specify a **WdWordDialog** constant with the **Dialogs** property. For example, the following macro instruction displays the **Open** dialog box (**wdDialogFileOpen**).

```
Dialogs(wdDialogFileOpen).Show
```

If a file is selected and **OK** is clicked, the file is opened (the action is executed). The following example displays the **Print** dialog box (**wdDialogFilePrint**).

```
Dialogs(wdDialogFilePrint).Show
```

Set the **DefaultTab** property to access a particular tab in a Word dialog box. The following example displays the **Page Border** tab in the **Borders and Shading** dialog box (**Format** menu).

```
With Dialogs(wdDialogFormatBordersAndShading)
    .DefaultTab = wdDialogFormatBordersAndShadingTabPageBorder
    .Show
End With
```

The **Display** method displays a dialog box without executing the actions taken in the dialog box. This can be useful if you want to prompt the user with a built-in dialog box and return the settings. For example, the following macro instruction displays the **User Information** tab from the **Options** dialog box and then returns and displays the user name.

```
With Dialogs(wdDialogToolsOptionsUserInfo)
    .Display
    MsgBox .Name
End With
```

If the user name is changed in the previous example, the change is not set in the dialog box. Use the **Execute** method to execute the settings in a dialog box without displaying the dialog box. The following example displays the User Information dialog box, and if the name is not an empty string, the settings are set in the dialog box using the **Execute** method.

```
With Dialogs(wdDialogToolsOptionsUserInfo)
    .Display
    If .Name <> "" Then .Execute
End With
```

Returning and changing dialog box settings

Prior to returning or changing a dialog box setting, you need to identify the individual dialog box. This is done using the **Dialogs** property with a **WdWordDialog** constant. The following example returns a **Dialog** object that refers to the **Paragraph** dialog box (**Format** menu).

```
Set myDialog = Dialogs(wdDialogFormatParagraph)
```

After you have a **Dialog** object you can return or sets options in the dialog box. The following example displays the right indent from the **Paragraphs** dialog box.

```
Set myDialog = Dialogs(wdDialogFormatParagraph)
Msgbox "Right indent = " & myDialog.RightIndent
```

Many of the built-in Word dialog boxes have arguments that you can use to set or get values from a dialog box (for example, RightIndent in the previous example). To determine which arguments you can use, see Built-in dialog box argument lists.

Just as you can return dialog box settings, you can also set dialog box settings. The following

example marks the **Keep with next** check box in the **Paragraph** dialog box.

```
With Dialogs(wdDialogFormatParagraph)
    .KeepWithNext = 1
    .Execute
End With
```

The **Keep with next** check box is enabled and the **Execute** method sets the changed settings value in the dialog box. The following Visual Basic instructions is the equivalent of the four instructions in the previous example.

```
Selection.Paragraphs(1).KeepWithNext = True
```

It's not very efficient to use a **Dialog** object to change a value that you can set with a property or method.

Note Use the **Update** method to ensure that the dialog box values reflect the current values. It may be necessary to use the **Update** method if you define a dialog box variable early in your macro and later want to return or change the current settings.

Presetting dialog box settings

The previous examples have return and set dialog box values without displaying the dialog box. You can also can change settings in a built-in Word dialog box prior to using the **Show** method. For example, you can change the find text before displaying the **Find and Replace** dialog box (**Edit** menu). The following example displays the **Find and Replace** dialog box with the word "Blue" preset in the **Find what** edit box.

```
With  Dialogs(wdDialogEditFind)
    .Find = "Blue"
    .Show
End With
```

The following example displays the **Open** dialog box with all file names displayed.

```
With Dialogs(wdDialogFileOpen)
    .Name = "*.*"
    .Show
End With
```

Checking how a dialog box was closed

The value returned by the **Show** and **Display** methods indicates which button was clicked to close the dialog box. The following example displays the **Break** dialog box, and if **OK** is clicked, a message is displayed on the status bar.

```
If Dialogs(wdDialogInsertBreak).Show = -1 Then
    StatusBar = "Break inserted"
End If
```

Return value	Description
-2	The **Close** button.
-1	The **OK** button.
0 (zero)	The **Cancel** button.
> 0 (zero)	A command button: 1 is the first button, 2 is the second button, and so on.

Assigning ranges

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowAssigning RangesC"}

There are several ways to assign an existing **Range** object to a variable. This topic explains the results of two different techniques. In the following examples, the `Range1` and `Range2` variables refer to **Range** objects. For example, the following instructions assign the first and second words in the active document to the `Range1` and `Range2` variables.

```
Set Range1 = ActiveDocument.Words(1)
Set Range2 = ActiveDocument.Words(2)
```

Setting a Range object variable equal to another Range object variable

This following instruction assigns a range variable named `Range2` to represent to the same location as `Range1`.

```
Set Range2 = Range1
```

You now have two variables that represent to the same range. When you manipulate the start or end point or the text of `Range2`, it affects `Range1` and vice versa.

Note that the following instruction is the same as `Range2.Text = Range1.Text`. This instruction assigns the default property of `Range1`, which is the **Text** property, to the default property of `Range2`. It doesn't change what the objects actually refer to.

```
Range2 = Range1
```

The ranges (`Range2` and `Range1`) have the same contents, but they may point to different locations in the document or even different documents.

Using the Duplicate property

The following instruction creates a new duplicated **Range** object, `Range2`, which has the same start and end points and text as `Range1`.

```
Set Range2 = Range1.Duplicate
```

If you change the start or end point of `Range1`, it doesn't affect `Range2`, and vice versa. Because these two ranges point to the same location in the document, changing the text in one range affects the text in the other range.

Recording a macro to generate code

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowGenerateCodeC"}

If you are unsure of which Visual Basic method or property to use, you can turn on the macro recorder and manually perform the action. The macro recorder translates your actions into Visual Basic code. After you've recorded your actions, you can modify the code to do exactly what you want. For example if you don't know what property or method to use to indent a paragraph, do the following:

1. On the **Tools** menu, point to **Macro**, and then click **Record New Macro**.
2. Change the default macro name if you'd like and click **OK** to start the recorder.
3. On the **Format** menu, choose **Paragraph**.
4. Change the left paragraph indent value and click **OK**.
5. Click the **Stop Recording** button on the **Stop Recording** toolbar.
6. On the **Tools** menu, point to **Macro**, and then click **Macros**.
7. Select the macro name from Step 2 and click the **Edit** button.

View the Visual Basic code to determine the property that corresponds to the left paragraph indent (the **LeftIndent** property). Position the insertion point within `LeftIndent` and press `F1` or click the **Help** button. Within the topic, you can view examples and review the objects that support the **LeftIndent** property (click Applies To).

Note Recorded macros use the **Selection** property to return the **Selection** object. For example, the following instruction indents the selected paragraphs by a half inch.

```
Selection.ParagraphFormat.LeftIndent = InchesToPoints(0.5)
```

You can, however, modify the recorded macro to work with Range objects. For information, see Revising recorded Visual Basic macros.

For complete steps on recording macros, see Record a macro in Word.

Using Events with the Application Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowApplicationEventsC"}

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 Word.Application
```

Write the Event Procedures

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

```
Private Sub App_DocumentChange()

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 Register_Event_Handler()
    Set X.App = Word.Application
End Sub
```

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

Using Events with the Document Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowDocumentEventsC"}

The **Document** object supports three events: **Close**, **New** and **Open**. You write procedures to respond to these events in the class module named "ThisDocument." Use the following steps to create an event procedure.

1. Under your Normal project or document project in the Project Explorer window, double-click **ThisDocument**. (In Folder view, **ThisDocument** is located in the **Microsoft Word Objects** folder.)
2. Select **Document** from the **Object** drop-down list box.
3. Select an event from the **Procedure** drop-down list box.

   An empty subroutine is added to the class module.
4. Add the Visual Basic instructions you want to run when the event occurs.

The following example shows a **New** event procedure in the Normal project that will run when a new document based on the Normal template is created.

```
Private Sub Document_New()
    MsgBox "New document was created"
End Sub
```

The following example shows a **Close** event procedure in a document project that runs only when that document is closed.

```
Private Sub Document_Close()
    MsgBox "Closing the document"
End Sub
```

Unlike auto macros, event procedures in the Normal template don't have a global scope. For example, event procedures in the Normal template only occur if the attached template is the Normal template.

If an auto macro exists in a document and the attached template, only the auto macro stored in the document will execute. If an event procedure for a **Document** event exists in a document and its attached template, both event procedures will run.

Note For information on creating event procedures for the **Application** object, see Using Events with the Application Object.

Auto Macros

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowAutoMacrosC"}

By giving a macro a special name, you can run it automatically when you perform an operation such as starting Word or opening a document. Word recognizes the following names as automatic macros, or "auto" macros.

Macro name	When it runs
AutoExec	When you start Word or load a global template
AutoNew	Each time you create a new document
AutoOpen	Each time you open an existing document
AutoClose	Each time you close a document
AutoExit	When you quit Word or unload a global template

Auto macros in code modules are recognized if either of the following conditions are true.

- The module is named after the auto macro (for example, AutoExec) and it contains a procedure named "Main."
- A procedure in any module is named after the auto macro.

Just like other macros, auto macros can be stored in the Normal template, another template, or a document. The only exception is the AutoExec macro, which will not run automatically unless it is stored in the Normal template or a global template stored in the folder specified as the Startup folder.

In the case of a naming conflict (multiple auto macros with the same name), Word runs the auto macro stored in the closest context. For example, if you create an AutoClose macro in a document and the attached template, only the auto macro stored in the document will execute. If you create an AutoNew macro in the normal template, the macro will run if a macro named AutoNew doesn't exist in the document or the attached template.

Note You can hold down the SHIFT key to prevent auto macros from running. For example, if you create a new document based on a template that contains an AutoNew macro, you can prevent the AutoNew macro from running by holding down SHIFT when you click **OK** in the **New** dialog box (**File** menu) and continuing to hold down SHIFT until the new document is displayed. In a macro that might trigger an auto macro, you can use the following instruction to prevent auto macros from running.

```
WordBasic.DisableAutoMacros
```

Predefined Bookmarks

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowPredefinedBookmarksC"}

Word sets and automatically updates a number of reserved bookmarks. You can use these predefined bookmarks just as you use the ones that you place in documents, except that you don't have to set them and they are not listed on the **Go To** tab in the **Find and Replace** dialog box (**Edit** menu).

You can use predefined bookmarks with the **Bookmarks** property. The following example sets the bookmark named "currpara" to the location marked by the predefined bookmark named "\Para."

```
ActiveDocument.Bookmarks("\Para").Copy "currpara"
```

The following table describes the predefined bookmarks available in Word.

Bookmark	Description
\Sel	Current selection or the insertion point.
\PrevSel1	Most recent selection where editing occurred; going to this bookmark is equivalent to running the **GoBack** method once.
\PrevSel2	Second most recent selection where editing occurred; going to this bookmark is equivalent to running the **GoBack** method twice.
\StartOfSel	Start of the current selection.
\EndOfSel	End of the current selection.
\Line	Current line or the first line of the current selection. If the insertion point is at the end of a line that is not the last line in the paragraph, the bookmark includes the entire next line.
\Char	Current character, which is the character following the insertion point if there is no selection, or the first character of the selection.
\Para	Current paragraph, which is the paragraph containing the insertion point or, if more than one paragraph is selected, the first paragraph of the selection. Note that if the insertion point or selection is in the last paragraph of the document, the "\Para" bookmark does not include the paragraph mark.
\Section	Current section, including the break at the end of the section, if any. The current section contains the insertion point or selection. If the selection contains more than one section, the "\Section" bookmark is the first section in the selection.
\Doc	Entire contents of the active document, with the exception of the final paragraph mark.
\Page	Current page, including the break at the end of the page, if any. The current page contains the insertion point. If the current selection contains more than one page, the "\Page" bookmark is the first page of the selection. Note that if the insertion point or selection is in the last page of the document, the "\Page" bookmark does not include the final paragraph mark.
\StartOfDoc	Beginning of the document.
\EndOfDoc	End of the document.
\Cell	Current cell in a table, which is the cell containing the insertion point. If one or more cells of a table are included in the current selection, the "\Cell" bookmark is the first cell in the selection.
\Table	Current table, which is the table containing the insertion point

	or selection. If the selection includes more than one table, the "\Table" bookmark is the entire first table of the selection, even if the entire table is not selected.
\HeadingLevel	The heading that contains the insertion point or selection, plus any subordinate headings and text. If the current selection is body text, the "\HeadingLevel" bookmark includes the preceding heading, plus any headings and text subordinate to that heading.

Using DAO from Microsoft Word

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowUsingDAOWithWordC"}

You can use Data Access Objects (DAO) properties, objects, and methods the same way you reference and use Word properties, objects, and methods. After you establish a reference to the DAO object library, you can open databases, design and run queries to extract a set of records, and bring the results back to Word.

Referencing DAO

Before you can use DAO, you must establish a reference to the DAO object library. Use the following steps to establish a reference to the DAO object library.

1. Switch to the Visual Basic Editor.
2. On the **Tools** menu, click **References**.
3. In the **Available References** box, click **Microsoft DAO 3.5 Object Library**.

If you don't see **Microsoft DAO 3.5 Object Library** in the **Available References** box, run Office Professional Setup to install the Data Access Objects for Visual Basic. The Microsoft DAO 3.5 Object Library is not included with the standalone version of Word or the Standard edition of Microsoft Office.

The following example opens the Northwind database and inserts the items from the Shippers table into the active document.

```
Set Db = OpenDatabase(Name:="C:\MSOffice\Access\Samples\Northwind.mdb")
Set Rs = Db.OpenRecordset(Name:="Shippers")
For I = 0 To Rs.RecordCount - 1
    Selection.InsertAfter Text:=Rs.Fields(1).Value
    Rs.MoveNext
    Selection.Collapse Direction:=wdCollapseEnd
    Selection.InsertParagraphAfter
Next I
Rs.Close
Db.Close
```

Use the **OpenDatabase** method to connect to a database and open it. After opening the database, use the **OpenRecordset** method to access a table or query for results. To navigate through the recordset, use the **Move** method. To find a specific record, use the **Seek** method. If you need only a subset of records instead of the entire recordset, use the **CreateQueryDef** method to design a customized query to select records that meet your criteria. When you finish working with a database, it's a good idea to close it using the **Close** method, to save memory.

Note For more information about a specific DAO object, method, or property, see Data Access Objects Help.

Modifying a Word Command

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowModifyWordCommandC"}

You can modify most Word commands by turning them into macros. For example, you can modify the **Open** command on the **File** menu so that instead of displaying a list of Word document files (in Windows, files ending with the .DOC file name extension), Word displays every file in the current folder.

To display the list of built-in Word commands in the **Macro** dialog box, you select **Word Commands** in the **Macros In** box. Every menu command and every command available on a toolbar or through shortcut keys is listed. Menu commands begin with the menu name associated with the command. For example, the **Save** command on the **File** menu is listed as **FileSave**.

You can replace a Word command with a macro by giving a macro the same name as a Word command. For example, if you create a macro named "FileSave," Word runs the macro when you choose **Save** from the **File** menu, click the **Save** toolbar button, or press the **FileSave** shortcut key combination.

This example takes you through the steps needed to modify the FileSave command.

1. On the **Tools** menu, point to **Macro**, and then click **Macros**.
2. In the **Macros** In box, select **Word Commands**.
3. In the **Macro Name** box, select **FileSave**.
4. In the **Macros In** box, select a template or document location to store the macro. For example, select **Normal.dot (Global Template)** to create a global macro (the **FileSave** command will be modified for all documents).
5. Click the **Create** button.

The FileSave macro appears as shown below.

```
Sub FileSave()
'
' FileSave Macro
' Saves the active document or template
'
    ActiveDocument.Save

End Sub
```

You can add additional instructions or remove the existing `ActiveDocument.Save` instruction. Now every time the **FileSave** command runs, your FileSave macro runs instead of the word command. To restore the original **FileSave** functionality, you need to rename or delete your FileSave macro.

Note You can also replace a Word command by creating a code module named after a Word command (for example, FileSave) with a subroutine named Main.

Using Events with ActiveX Controls

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wohowControlEventsC"}

Word documents can contain ActiveX controls. Use the **Control Toolbox** to insert ActiveX controls such as command buttons, check boxes and list boxes. Use the following steps to add an ActiveX check box control with a **LostFocus** event.

1. Right-click a toolbar in Word and click **Control Toolbox**.
2. Click the **Check Box** control.

   A check box control is inserted in the active document.
3. Right-click the check box control and click **View Code**.

   Word switches to the Visual Basic Editor and displays the ThisDocument class module with the check box selected in the **Object** drop-down list box.
4. Select the **LostFocus** event from the **Procedure** drop-down list box.

   An empty procedure is added to the class module.
5. Add the Visual Basic instructions you want to run when the event occurs.

The following example shows a **LostFocus** event procedure that runs when the focus is moved away from CheckBox1. The macro displays the state of CheckBox1 using the **Value** property (**True** for selected and **False** for clear).

```
Private Sub CheckBox1_LostFocus()
    MsgBox CheckBox1.Value
End Sub
```

To see your event procedure run, switch back to Word with the document that includes the check box displayed. Click the **Exit Design Mode** button on the **Control Toolbox**. Select or clear the check box and then click on another element in the document. The check box control loses the focus and your LostFocus procedure runs; a message box is displayed with either "True" or "False."

Word implements the **LostFocus** and **GotFocus** events for ActiveX controls in a Word document. The other events listed in the **Procedure** drop-down list box in are documented in Microsoft Forms Help.

AddIn Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAddInC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAddInP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAddInM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAddInE "}

AddIns (AddIn)

Represents a single add-in, either installed or not installed. The **AddIn** object is a member of the **AddIns** collection. The **AddIns** collection contains all the add-ins available to Word, regardless of whether or not they're currently loaded. The **AddIns** collection includes global templates or Word add-in libraries (WLLs) displayed in the **Templates and Add-ins** dialog box (**Tools** menu).

Using the AddIn Object

Use **AddIns**(*index*), where *index* is the add-in name or index number, to return a single **AddIn** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown in the **Templates and Add-Ins** dialog box. The following example loads the Letter.dot template as a global template.

```
AddIns("Letter.dot").Installed = True
```

The index number represents the position of the add-in in the list of add-ins in the **Templates and Add-ins** dialog box. The following instruction displays the path of the first available add-in.

```
If Addins.Count >= 1 Then MsgBox Addins(1).Path
```

The following example creates a list of add-ins at the beginning of the active document. The list contains the name, path, and installed state of each available add-in.

```
With ActiveDocument.Range(Start:=0, End:=0)
    .InsertAfter "Name" & vbTab & "Path" & vbTab & "Installed"
    .InsertParagraphAfter
    For Each oAddIn In AddIns
        .InsertAfter oAddIn.Name & vbTab & oAddIn.Path & vbTab _
            & oAddIn.Installed
        .InsertParagraphAfter
    Next oAddIn
    .ConvertToTable
End With
```

Use the **Add** method to add an add-in to the list of available add-ins and (optionally) install it using the ***Install*** argument.

```
AddIns.Add FileName:="C:\Templates\Other\Letter.dot", Install:=True
```

To install an add-in shown in the list of available add-ins, use the **Installed** property.

```
AddIns("Letter.dot").Installed = True
```

Note Use the **Compiled** property to determine whether an **AddIn** object is a template or a WLL.

AddIns Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAddInsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAddInsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAddInsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAddInsE "}

AddIns (AddIn)

A collection of **AddIn** objects that represents all the add-ins available to Word, regardless of whether or not they're currently loaded. The **AddIns** collection includes global templates or Word add-in libraries (WLLs) displayed in the **Templates and Add-ins** dialog box (**Tools** menu).

Using the AddIns Collection

Use the **AddIns** property to return the **AddIns** collection. The following example displays the name and the installed state of each available add-in.

```
For Each ad In AddIns
    If ad.Installed = True Then
        MsgBox ad.Name & " is installed"
    Else
        MsgBox ad.Name & " is available but not installed"
    End If
Next ad
```

Use the **Add** method to add an add-in to the list of available add-ins and (optionally) install it using the ***Install*** argument.

```
AddIns.Add FileName:="C:\Templates\Other\Letter.dot", Install:=True
```

To install an add-in shown in the list of available add-ins, use the **Installed** property.

```
AddIns("Letter.dot").Installed = True
```

Use **AddIns**(*index*), where *index* is the add-in name or index number, to return a single **AddIn** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown in the **Templates and Add-ins** dialog box. To install an add-in shown in the list of available add-ins, use the **Installed** property. The following example loads the Letter.dot template as a global template.

```
AddIns("Letter.dot").Installed = True
```

Note If the add-in is not located in the User Templates, Workgroup Examples, or Startup folder, you must specify the full path and file name when indexing an add-in by name.

Remarks

Use the **Compiled** property to determine whether an **AddIn** object is a template or a WLL.

Application Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjApplicationC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjApplicationP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjApplicationM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjApplicationE "}

Represents the Word application. The **Application** object includes properties and methods that return top-level objects. For example, the **ActiveDocument** property returns a **Document** object.

Using the Application Object

Use the **Application** property to return the **Application** object. The following example displays the user name for Word.

```
MsgBox Application.UserName
```

Many of the properties and methods that return the most common user-interface objects − such as the active document (**ActiveDocument** property) − can be used without the **Application** object qualifier. For example, instead of writing `Application.ActiveDocument.PrintOut`, you can write `ActiveDocument.PrintOut`. Properties and methods that can be used without the **Application** object qualifier are considered "global." To view the global properties and methods in the **Object Browser**, click **<globals>** at the top of the list in the **Classes** box.

Remarks

To use Automation (formerly OLE Automation) to control Word from another application, use the **CreateObject** or **GetObject** function to return a Word **Application** object. The following Microsoft Excel example starts Word (if it's not already running) and opens an existing document.

```
Set wrd = GetObject(, "Word.Application")
wrd.Visible = True
wrd.Documents.Open "C:\My Documents\Temp.doc"
Set wrd = Nothing
```

Addins, **Assistant**, **AutoCaptions**, **AutoCorrect**, **Browser**, **CaptionLabels**, **CommandBars**, **Dictionaries**, **Dialogs**, **Documents**, **FileConverters**, **FileSearch**, **FontNames**, **KeyBindings**, **KeysBoundTo**, **Languages**, **ListGalleries**, **MailingLabel**, **MailMessage**, **Options**, **RecentFiles**, **Selection**, **SpellingSuggestions**, **SynonymInfo**, **System**, **Tasks**, **Templates**, **VBE**, **Windows**

AutoCaption Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoCaptionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoCaptionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoCaptionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoCaptionE "}

AutoCaptions (AutoCaption)

Represents a single caption that can be automatically added when items such as tables, pictures, or OLE objects are inserted into a document. The **AutoCaption** object is a member of the **AutoCaptions** collection. The **AutoCaptions** collection contains all the captions listed in the **AutoCaption** dialog box (**Insert** menu).

Using the AutoCaption Object

Use **AutoCaptions**(*index*), where *index* is the caption name or index number, to return a single **AutoCaption** object. The caption names correspond to the items listed in the **AutoCaption** dialog box (**Insert** menu). You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown in the **AutoCaption** dialog box. The following example enables autocaptions for Word tables.

```
AutoCaptions("Microsoft Word Table").AutoInsert = True
```

The index number represents the position of the **AutoCaption** object in the list of items in the **AutoCaption** dialog box. The following example displays the name of the first item listed in the **AutoCaption** dialog box.

```
MsgBox AutoCaptions(1).Name
```

AutoCaption objects cannot be programmatically added to or deleted from the **AutoCaptions** collection.

AutoCaptions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoCaptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoCaptionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoCaptionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoCaptionsE "}

AutoCaptions (AutoCaption)

A collection of **AutoCaption** objects that represent the captions that can be automatically added when items such as tables, pictures, or OLE objects are inserted into a document.

Using the AutoCaptions Collection

Use the **AutoCaptions** property to return the **AutoCaptions** collection. The following example displays the names of the selected items in the **AutoCaption** dialog box.

```
For Each autoCap In AutoCaptions
    If autoCap.AutoInsert = True Then
        MsgBox autoCap.Name & " is configured for auto insert"
    End If
Next autoCap
```

The **AutoCaptions** collection contains all the captions listed in the **AutoCaption** dialog box (**Insert** menu). **AutoCaption** objects cannot be programmatically added to or deleted from the **AutoCaptions** collection.

Use **AutoCaptions**(*index*), where *index* is the caption name or index number, to return a single **AutoCaption** object. The caption names correspond to the items listed in the **AutoCaption** dialog box (**Insert** menu). You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown in the **AutoCaption** dialog box. The following example displays the caption text "Microsoft Word Table."

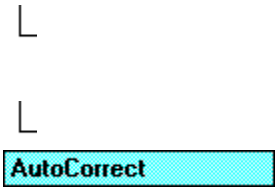
```
MsgBox AutoCaptions("Microsoft Word Table").CaptionLabel.Name
```

The index number represents the position of the **AutoCaption** object in the list of captions in the **AutoCaption** dialog box. The following example displays the name of the first item selected in the **AutoCaption** dialog box.

```
MsgBox AutoCaptions(1).Name
```

AutoCorrect Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoCorrectC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoCorrectP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoCorrectM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoCorrectE "}



Represents the AutoCorrect functionality in Word.

Using the AutoCorrect Object

Use the **AutoCorrect** property to return the **AutoCorrect** object. The following example enables the AutoCorrect options and creates an AutoCorrect entry.

```
With AutoCorrect
    .CorrectCapsLock = True
    .CorrectDays = True
    .Entries.Add Name:="usualy", Value:="usually"
End With
```

The **Entries** property returns the **AutoCorrectEntries** object that represents the AutoCorrect entries in the **AutoCorrect** dialog box (**Tools** menu).

AutoCorrectEntries, **FirstLetterExceptions**, **TwoInitialCapsExceptions**

AutoCorrectEntries Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoCorrectEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoCorrectEntriesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoCorrectEntriesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoCorrectEntriesE "}

AutoCorrect

AutoCorrectEntries (AutoCorrectEntry)

A collection of **AutoCorrectEntry** objects that represent all the AutoCorrect entries available to Word. The **AutoCorrectEntries** collection includes all the entries in the **AutoCorrect** dialog box (**Tools** menu).

Using the AutoCorrectEntries Collection

Use the **Entries** property to return the **AutoCorrectEntries** collection. The following example displays the number of **AutoCorrectEntry** objects in the **AutoCorrectEntries** collection.

```
MsgBox AutoCorrect.Entries.Count
```

Use the **Add** or the **AddRichText** method to add an AutoCorrect entry to the list of available entries. The following example adds a plain-text AutoCorrect entry for the misspelling of the word "their."

```
AutoCorrect.Entries.Add Name:="thier", Value:="their"
```

The following example creates an AutoCorrect entry named "PMO" based on the text and formatting of the selection.

```
AutoCorrect.Entries.AddRichText Name:="PMO", Range:=Selection.Range
```

Use **Entries**(*index*), where *index* is the AutoCorrect entry name or index number, to return a single **AutoCorrectEntry** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown under **Replace** in the **AutoCorrect** dialog box. The following example sets the value of an existing AutoCorrect entry named "teh."

```
AutoCorrect.Entries("teh").Value = "the"
```

The following example displays the name and value of the first AutoCorrent entry.

```
MsgBox "Name = " & AutoCorrect.Entries(1).Name & vbCr & _
    "Value " & AutoCorrect.Entries(1).Value
```

AutoCorrectEntry Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoCorrectEntryC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoCorrectEntryP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoCorrectEntryM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoCorrectEntryE "}

AutoCorrect

AutoCorrectEntries (AutoCorrectEntry)

Represents a single AutoCorrect entry. The **AutoCorrectEntry** object is a member of the **AutoCorrectEntries** collection. The **AutoCorrectEntries** collection includes the entries in the **AutoCorrect** dialog box (**Tools** menu).

Using the AutoCorrectEntry Object

Use **Entries**(*index*), where *index* is the AutoCorrect entry name or index number, to return a single **AutoCorrectEntry** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown under **Replace** in the **AutoCorrect** dialog box. The following example sets the value of the AutoCorrect entry named "teh."

```
AutoCorrect.Entries("teh").Value = "the"
```

Use the **Apply** method to insert an AutoCorrect entry at the specified range. The following example adds an AutoCorrect entry and then inserts it in place of the selection.

```
AutoCorrect.Entries.Add Name:="hellp", Value:="hello"
AutoCorrect.Entries("hellp").Apply Range:=Selection.Range
```

Use either the **Add** or **AddRichText** method to add an AutoCorrect entry to the list of available entries. The following example adds a plain-text AutoCorrect entry for the misspelling of the word "their.'

```
AutoCorrect.Entries.Add Name:="thier", Value:="their"
```

The following example creates an AutoCorrect entry named "PMO" based on the text and formatting of the selection.

```
AutoCorrect.Entries.AddRichText Name:="PMO", Range:=Selection.Range
```

AutoTextEntries Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoTextEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoTextEntriesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoTextEntriesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoTextEntriesE "}

└

└

Templates (Template)

└

└

AutoTextEntries (AutoTextEntry)

A collection of **AutoTextEntry** objects that represent the AutoText entries in a template. The **AutoTextEntries** collection includes all the entries listed on the **AutoText** tab in the **AutoCorrect** dialog box (**Tools** menu).

Using the AutoTextEntries Object

Use the **AutoTextEntries** property to return the **AutoTextEntries** collection. The following example determines whether an **AutoTextEntry** object named "test" is in the **AutoTextEntries** collection.

```
For Each i In NormalTemplate.AutoTextEntries
    If LCase(i.Name) = "test" Then MsgBox "AutoText entry exists"
Next i
```

Use the **Add** method to add an AutoText entry to the **AutoTextEntries** collection. The following example adds an AutoText entry named "Blue" based on the text of the selection.

```
NormalTemplate.AutoTextEntries.Add Name:="Blue", _
    Range:=Selection.Range
```

Use **AutoTextEntries**(*index*), where *index* is the AutoText entry name or index number, to return a single **AutoTextEntry** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown on the **AutoText** tab in the **AutoCorrect** dialog box. The following example sets the value of an existing AutoText entry named "cName."

```
NormalTemplate.AutoTextEntries("cName").Value = "The Johnson Company"
```

The following example displays the name and value of the first AutoText entry in the template attached to the active document.

```
Set myTemplate = ActiveDocument.AttachedTemplate
MsgBox "Name = " & myTemplate.AutoTextEntries(1).Name & vbCr & _
    "Value " & myTemplate.AutoTextEntries(1).Value
```

AutoTextEntry Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjAutoTextEntryC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjAutoTextEntryP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjAutoTextEntryM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjAutoTextEntryE "}

Templates (Template)

AutoTextEntries (AutoTextEntry)

Represents a single AutoText entry. The **AutoTextEntry** object is a member of the **AutoTextEntries** collection. The **AutoTextEntries** collection contains all the AutoText entries in the specified template. The entries are listed listed on the **AutoText** tab in the **AutoCorrect** dialog box (**Tools** menu).

Using the AutoTextEntry Object

Use **AutoTextEntries**(*index*), where *index* is the AutoText entry name or index number, to return a single **AutoTextEntry** object. You must exactly match the spelling (but not necessarily the capitalization) of the name, as it's shown on the **AutoText** tab in the **AutoCorrect** dialog box. The following example sets the value of an existing AutoText entry named "cName."

```
NormalTemplate.AutoTextEntries("cName").Value = "The Johnson Company"
```

The following example displays the name and value of the first AutoText entry in the template attached to the active document.

```
Set myTemplate = ActiveDocument.AttachedTemplate
MsgBox "Name = " & myTemplate.AutoTextEntries(1).Name & vbCr & _
    "Value " & myTemplate.AutoTextEntries(1).Value
```

The following example inserts the global AutoText entry named "TheWorld" at the insertion point.

```
Selection.Collapse Direction:=wdCollapseEnd
NormalTemplate.AutoTextEntries("TheWorld").Insert _
    Where:=Selection.Range
```

Use the **Add** method to add an **AutoTextEntry** object to the **AutoTextEntries** collection. The following example adds an AutoText entry named "Blue" based on the text of the selection.

```
NormalTemplate.AutoTextEntries.Add Name:="Blue", _
    Range:=Selection.Range
```

Bookmark Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjBookmarkC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjBookmarkP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjBookmarkM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjBookmarkE "}

Bookmarks (Bookmark)

Represents a single bookmark. The **Bookmark** object is a member of the **Bookmarks** collection. The **Bookmarks** collection includes all the bookmarks listed in the **Bookmark** dialog box (**Insert** menu).

Using the Bookmark Object

Use **Bookmarks**(*index*), where *index* is the bookmark name or index number, to return a single **Bookmark** object. You must exactly match the spelling (but not necessarily the capitalization) of the bookmark name. The following example selects the bookmark named "temp" in the active document.

```
ActiveDocument.Bookmarks("temp").Select
```

The index number represents the position of the bookmark in the **Selection** or **Range** object. For the **Document** object, the index number represents the position of the bookmark in the alphabetic list of bookmarks in the **Bookmarks** dialog box (click **Name** to sort the list of bookmarks alphabetically). The following example displays the name of the second bookmark in the **Bookmarks** collection.

```
MsgBox ActiveDocument.Bookmarks(2).Name
```

Use the **Add** method to add a bookmark to a document range. The following example marks the selection by adding a bookmark named "temp."

```
ActiveDocument.Bookmarks.Add Name:="temp", Range:=Selection.Range
```

Remarks

Use the **BookmarkID** property with a range or selection object to return the index number of the **Bookmark** object in the **Bookmarks** collection. The following example displays the index number of the bookmark named "temp" in the active document.

```
MsgBox ActiveDocument.Bookmarks("temp").Range.BookmarkID
```

You can use predefined bookmarks with the **Bookmarks** property. The following example sets the bookmark named "currpara" to the location marked by the predefined bookmark named "\Para".

```
ActiveDocument.Bookmarks("\Para").Copy "currpara"
```

Use the **Exists** method to determine whether a bookmark already exists in the selection, range, or document. The following example ensures that the bookmark named "temp" exists in the active document before selecting the bookmark.

```
If ActiveDocument.Bookmarks.Exists("temp") = True Then
    ActiveDocument.Bookmarks("temp").Select
End If
```

Bookmarks that Word automatically sets in each document. For example, the "\Sel" predefined bookmark refers to the current selection or the insertion point.

Document, **Range**, **Selection**

Bookmarks Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjBookmarksC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjBookmarksP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjBookmarksM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjBookmarksE "}

A collection of **Bookmark** objects that represent the bookmarks in the specified selection, range, or document.

Using the Bookmarks Collection

Use the **Bookmarks** property to return the **Bookmarks** collection. The following example ensures that the bookmark named "temp" exists in the active document before selecting the bookmark.

```
If ActiveDocument.Bookmarks.Exists("temp") = True Then
    ActiveDocument.Bookmarks("temp").Select
End If
```

Use the **Add** method to set a bookmark for a range in a document. The following example marks the selection by adding a bookmark named "temp".

```
ActiveDocument.Bookmarks.Add Name:="temp", Range:=Selection.Range
```

Use **Bookmarks**(*index*), where *index* is the bookmark name or index number, to return a single **Bookmark** object. You must exactly match the spelling (but not necessarily the capitalization) of the bookmark name. The following example selects the bookmark named "temp" in the active document.

```
ActiveDocument.Bookmarks("temp").Select
```

The index number represents the position of the bookmark in the **Selection** or **Range** object. For the **Document** object, the index number represents the position of the bookmark in the alphabetic list of bookmarks in the **Bookmarks** dialog box (click **Name** to sort the list of bookmarks alphabetically). The following example displays the name of the second bookmark in the **Bookmarks** collection.

```
MsgBox ActiveDocument.Bookmarks(2).Name
```

Remarks

The **ShowHidden** property effects the number of elements in the **Bookmarks** collection. If **ShowHidden** is **True**, hidden bookmarks are included in the **Bookmarks** collection.

Border Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjBorderC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjBorderP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjBorderM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjBorderE "}

Borders (Border)

Represents a border of an object. The **Border** object is a member of the **Borders** collection.

Using the Border Object

Use **Borders**(*index*), where *index* identifies the border, to return a single **Border** object. *Index* can be one of the following **WdBorderType** constants: **wdBorderBottom**, **wdBorderHorizontal**, **wdBorderLeft**, **wdBorderRight**, **wdBorderTop**, or **wdBorderVertical**. Use the **LineStyle** property to apply a border line to a **Border** object. The following example applies a double-line border below the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).Borders(wdBorderBottom)
    .LineStyle = wdLineStyleDouble
    .LineWidth = wdLineWidth025pt
End With
```

The following example applies a single-line border around the first character in the selection.

```
With Selection.Characters(1)
    .Font.Size = 36
    .Borders.Enable = True
End With
```

The following example adds an art border around each page in the first section.

```
For Each aBorder In ActiveDocument.Sections(1).Borders
    With aBorder
        .ArtStyle = wdArtSeattle
        .ArtWidth = 20
    End With
Next aBorder
```

Border objects cannot be added to the **Borders** collection. The number of members in the **Borders** collection is finite and varies depending on the type of object. For example, a table has six elements in the **Borders** collection, whereas a paragraph has four.

Cell, **Cells**, **Column**, **Columns**, **Font**, **Frame**, **InlineShape**, **Paragraph**, **ParagraphFormat**, **Paragraphs**, **Range**, **Row**, **Rows**, **Section**, **Selection**, **Style**, **Table**

Borders Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjBordersC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjBordersP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjBordersM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjBordersE "}

Borders (Border)

A collection of **Border** objects that represent the borders of an object.

Using the Borders Collection

Use the **Borders** property to return the **Borders** collection. The following example applies the default border around the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Borders.Enable = True
```

Border objects cannot be added to the **Borders** collection. The number of members in the **Borders** collection is finite and varies depending on the type of object. For example, a table has six elements in the **Borders** collection, whereas a paragraph has four.

Use **Borders**(*index*), where *index* identifies the border, to return a single **Border** object. *Index* can be one of the following **WdBorderType** constants: **wdBorderBottom**, **wdBorderHorizontal**, **wdBorderLeft**, **wdBorderRight**, **wdBorderTop**, or **wdBorderVertical**. (The constants **wdBorderDiagonalDown** and **wdBorderDiagonalUp** aren't used in U.S. English Word.) Use the **LineStyle** property to apply a border line to a **Border** object. The following example applies a double-line border below the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).Borders(wdBorderBottom)
    .LineStyle = wdLineStyleDouble
    .LineWidth = wdLineWidth025pt
End With
```

The following example applies a single-line border around the first character in the selection.

```
With Selection.Characters(1)
    .Font.Size = 36
    .Borders.Enable = True
End With
```

The following example adds an art border around each page in the first section.

```
For Each aBorder In ActiveDocument.Sections(1).Borders
    With aBorder
        .ArtStyle = wdArtSeattle
        .ArtWidth = 20
    End With
Next aBorder
```

Browser Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjBrowserC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjBrowserP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjBrowserM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjBrowserE "}

Browser

Represents the browser tool used to move the insertion point to objects in a document. This tool is comprised of the three buttons at the bottom of the vertical scroll bar.

Using the Browser Object

Use the **Browser** property to return the **Browser** object. The following example moves the insertion point just before the next field in the active document.

```
With Application.Browser
    .Target = wdBrowseField
    .Next
End With
```

The following example moves the insertion point to the previous table and selects it.

```
With Application.Browser
    .Target = wdBrowseTable
    .Previous
End With
If Selection.Information(wdWithInTable) = True Then
    Selection.Tables(1).Select
End If
```

CaptionLabel Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCaptionLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCaptionLabelP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCaptionLabelM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCaptionLabelE "}

CaptionLabels (CaptionLabel)

Represents a single caption label. The **CaptionLabel** object is a member of the **CaptionLabels** collection. The items in the **CaptionLabels** collection are listed in the **Label** box in the **Caption** dialog box (**Insert** menu).

Using the CaptionLabel Object

Use **CaptionLabels**(*index*), where *index* is the caption label name or index number, to return a single **CaptionLabel** object. The following example sets the numbering style for the Figure caption label.

```
CaptionLabels("Figure").NumberStyle = wdCaptionNumberStyleLowercaseLetter
```

The index number represents the position of the caption label in the **CaptionLabels** collection. The following example displays the first caption label.

```
MsgBox CaptionLabels(1).Name
```

Use the **Add** method to add a custom caption label. The following example adds a caption label named "Photo."

```
CaptionLabels.Add Name:="Photo"
```

CaptionLabels Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCaptionLabelsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCaptionLabelsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCaptionLabelsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCaptionLabelsE "}

CaptionLabels (CaptionLabel)

A collection of **CaptionLabel** objects that represent the available caption labels. The items in the **CaptionLabels** collection are listed in the **Label** box in the **Caption** dialog box (**Insert** menu).

Using the CaptionLabels Collection

Use the **CaptionLabels** property to return the **CaptionLabels** collection. By default, the **CaptionLabels** collection includes the three built-in caption labels: Figure, Table, and Equation.

Use the **Add** method to add a custom caption label. The following example adds a caption label named "Photo."

```
CaptionLabels.Add Name:="Photo"
```

Use **CaptionLabels**(*index*), where *index* is the caption label name or index number, to return a single **CaptionLabel** object. The following example sets the numbering style for the Figure caption label.

```
CaptionLabels("Figure").NumberStyle = wdCaptionNumberStyleLowercaseLetter
```

The index number represents the position of the caption label in the **CaptionLabels** collection. The following example displays the first caption label.

```
MsgBox CaptionLabels(1).Name
```

Cell Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCellC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCellP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCellM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCellE "}

Cells (Cell)

Represents a single table cell. The **Cell** object is a member of the **Cells** collection. The **Cells** collection represents all the cells in the specified object.

Using the Cell Object

Use **Cell**(*row*, *column*), where *row* is the row number and *column* is the column number, to return the **Cell** object. The following example applies shading to the second cell in the first row.

```
Set myCell = ActiveDocument.Tables(1).Cell(Row:=1, Column:=2)
myCell.Shading.Texture = wdTexture20Percent
```

Use the **Add** method to add a **Cell** object to the **Cells** collection. You can also use the **InsertCells** method of the **Selection** object to insert new cells. The following example adds a cell before the first cell in `myTable`.

```
Set myTable = ActiveDocument.Tables(1)
myTable.Range.Cells.Add BeforeCell:=myTable.Cell(1, 1)
```

The following example sets a range (`myRange`) that references the first two cells in the first table. After the range is set, the cells are combined by the **Merge** method.

```
Set myTable = ActiveDocument.Tables(1)
Set myRange = ActiveDocument.Range(myTable.Cell(1, 1).Range.Start, _
    myTable.Cell(1, 2).Range.End)
myRange.Cells.Merge
```

Remarks

Use the **Add** method with the **Rows** or **Columns** collection to add a row or column of cells.

Use the **Information** property with a **Selection** object to return the current row and column number. The following example changes the width of the first cell in the selection and then displays the cell's row number and column number.

```
If Selection.Information(wdWithInTable) = True Then
    With Selection
        .Cells(1).Width = 22
        MsgBox "Cell " & .Information(wdStartOfRangeRowNumber) & "," _
            & .Information(wdStartOfRangeColumnNumber)
    End With
End If
```

Cells Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCellsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCellsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCellsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCellsE "}

Cells (Cell)

A collection of **Cell** objects in a table column, table row, selection, or range.

Using the Cells Object

Use the **Cells** property to return the **Cells** collection. The following example formats the cells in the first row in table one in the active document to be 30 points wide.

```
ActiveDocument.Tables(1).Rows(1).Cells.Width = 30
```

The following example returns the number of cells in the current row.

```
num = Selection.Rows(1).Cells.Count
```

Use the **Add** method to add a **Cell** object to the **Cells** collection. You can also use the **InsertCells** method of the **Selection** object to insert new cells. The following example adds a cell before the first cell in myTable.

```
Set myTable = ActiveDocument.Tables(1)
myTable.Range.Cells.Add BeforeCell:=myTable.Cell(1, 1)
```

Use **Cell**(*row*, *column*), where *row* is the row number and *column* is the column number, to return the **Cell** object. The following example applies shading to the second cell in the first row in table one.

```
Set myCell = ActiveDocument.Tables(1).Cell(Row:=1, Column:=2)
myCell.Shading.Texture = wdTexture20Percent
```

Remarks

Use the **Add** method with the **Rows** or **Columns** collection to add a row or column of cells. The following example adds a column to the first table in the active document and then inserts numbers into the first column.

```
Set myTable = ActiveDocument.Tables(1)
Set aColumn = myTable.Columns.Add(BeforeColumn:=myTable.Columns(1))
For Each aCell In aColumn.Cells
    aCell.Range.Delete
    aCell.Range.InsertAfter num + 1
    num = num + 1
Next aCell
```

Column, **Range**, **Row**, **Selection**

Borders, **Column**, **Range**, **Row**, **Shading**

Characters Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCharactersC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCharactersP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCharactersM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCharactersE "}

Characters (Range)

A collection of characters in a selection, range, or document. There is no Character object; instead, each item in the **Characters** collection is a **Range** object that represents one character.

Using the Characters Collection

Use the **Characters** property to return the **Characters** collection. The following example displays how many characters are selected.

```
MsgBox Selection.Characters.Count & " characters are selected"
```

Use **Characters**(*index*), where *index* is the index number, to return a **Range** object that represents one character. The index number represents the position of a character in the **Characters** collection. The following example formats the first letter in the selection as 24-point bold.

```
With Selection.Characters(1)
    .Bold = True
    .Font.Size = 24
End With
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

An **Add** method isn't available for the **Characters** collection. Instead, use the **InsertAfter** or **InsertBefore** method to add characters to a **Range** object. The following example inserts a new paragraph after the first paragraph in the active document.

```
With ActiveDocument
    .Paragraphs(1).Range.InsertParagraphAfter
    .Paragraphs(2).Range.InsertBefore "New Text"
End With
```

CheckBox Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCheckBoxC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCheckBoxP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCheckBoxM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCheckBoxE "}

FormFields (FormField)

└

CheckBox

Represents a single check box form field.

Using the CheckBox Object

Use **FormFields**(*index*), where *index* is index number or the bookmark name associated with the check box, to return a single **FormField** object. Use the **CheckBox** property with the **FormField** object to return a **CheckBox** object. The following example selects the check box form field named "Check1" in the active document.

```
ActiveDocument.FormFields("Check1").CheckBox.Value = True
```

The index number represents the position of the form field in the **FormFields** collection. The following example checks the type of the first form field; if it's a check box, the check box is selected.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormCheckBox Then
    ActiveDocument.FormFields(1).CheckBox.Value = True
End If
```

The following example determines whether the `ffield` object is valid before changing the check box size to 14 points.

```
Set ffield = ActiveDocument.FormFields(1).CheckBox
If ffield.Valid = True Then
    ffield.AutoSize = False
    ffield.Size = 14
Else
    MsgBox "First field is not a check box"
End If
```

Use the **Add** method with the **FormFields** object to add a check box form field. The following example adds a check box at the beginning of the active document, sets the name to "Color", and then selects the check box.

```
With ActiveDocument.FormFields.Add(Range:=ActiveDocument.Range _
    (Start:=0,End:=0), Type:=wdFieldFormCheckBox)
    .Name = "Color"
    .CheckBox.Value = True
End With
```

Column Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjColumnC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjColumnP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjColumnM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjColumnE "}

Columns (Column)

Represents a single table column. The **Column** object is a member of the **Columns** collection. The **Columns** collection includes all the columns in a table, selection, or range.

Using the Column Object

Use **Columns**(*index*), where *index* is the index number, o return a single **Column** object. The index number represents the position of the column in the **Columns** collection (counting from left to right).

The following example selects column one in table one in the active document.

```
ActiveDocument.Tables(1).Columns(1).Select
```

Use the **Column** property with a **Cell** object to return a **Column** object.The following example deletes the text in cell one, inserts new text, and then sorts the entire column.

```
With ActiveDocument.Tables(1).Cell(1, 1)
    .Range.Delete
    .Range.InsertBefore "Sales"
    .Column.Sort
End With
```

Use the **Add** method to add a column to a table. The following example adds a column to the first table in the active document, and then it makes the column widths equal.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    myTable.Columns.Add BeforeColumn:=myTable.Columns(1)
    myTable.Columns.DistributeWidth
End If
```

Remarks

Use the **Information** property with a **Selection** object to return the current column number. The following example selects the current column and then displays the column number in a message box.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Columns(1).Select
    MsgBox "Column " & Selection.Information(wdStartOfRangeColumnNumber)
End If
```

Columns Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjColumnsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjColumnsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjColumnsE "}

Columns (Column)

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. The following example displays the number of **Column** objects in the **Columns** collection for the first table in the active document.

```
MsgBox ActiveDocument.Tables(1).Columns.Count
```

The following example creates a table with six columns and three rows and then formats each column with a progressively larger (darker) shading percentage.

```
Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
    NumRows:=3, NumColumns:=6)
For Each col In myTable.Columns
    col.Shading.Texture = 2 + i
    i = i + 1
Next col
```

Use the **Add** method to add a column to a table. The following example adds a column to the first table in the active document, and then it makes the column widths equal.

```
If ActiveDocument.Tables.Count >= 1 Then
    Set myTable = ActiveDocument.Tables(1)
    myTable.Columns.Add BeforeColumn:=myTable.Columns(1)
    myTable.Columns.DistributeWidth
End If
```

Use **Columns**(*index*), where *index* is the index number, to return a single **Column** object. The index number represents the position of the column in the **Columns** collection (counting from left to right). The following example selects the first column in the first table.

```
ActiveDocument.Tables(1).Columns(1).Select
```

Range, **Selection**, **Table**

Borders, **Cells**, **Shading**

Comment Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCommentC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCommentP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCommentM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCommentE "}

Comments (Comment)

Represents a single comment. The **Comment** object is a member of the **Comments** collection. The **Comments** collection includes comments in a selection, range or document.

Using the Comment Object

Use **Comments**(*index*), where *index* is the index number, to return a single **Comment** object. The index number represents the position of the comment in the specified selection, range, or document. The following example displays the author of the first comment in the active document.

```
MsgBox ActiveDocument.Comments(1).Author
```

Use the **Add** method to add a comment at the specified range. The following example adds a comment immediately after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Comments.Add Range:=Selection.Range, Text:="review this"
```

Use the **Reference** property to return the reference mark associated with the specified comment. Use the **Range** property to return the text associated with the specified comment. The following example displays the text associated with the first comment in the active document.

```
MsgBox ActiveDocument.Comments(1).Range.Text
```

Comments Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCommentsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCommentsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCommentsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCommentsE "}

Comments (Comment)

A collection of **Comment** objects that represent the comments in a selection, range, or document.

Using the Comments Collection

Use the **Comments** property to return the **Comments** collection. The following example displays comments made by Don Funk in the active document.

```
ActiveWindow.View.SplitSpecial = wdPaneComments
ActiveDocument.Comments.ShowBy = "Don Funk"
```

Use the **Add** method to add a comment at the specified range. The following example adds a comment immediately after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Comments.Add Range:=Selection.Range, Text:="review this"
```

Use **Comments**(*index*), where *index* is the index number, to return a single **Comment** object. The index number represents the position of the comment in the specified selection, range, or document. The following example displays the author of the first comment in the active document.

```
MsgBox ActiveDocument.Comments(1).Author
```

The following example displays the initials of the author of the first comment in the selection.

```
If Selection.Comments.Count >= 1 Then MsgBox Selection.Comments(1).Initial
```

Document, **Range**, **Selection**

CustomLabel Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCustomLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCustomLabelP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCustomLabelM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCustomLabelE "}

MailingLabel

CustomLabels (CustomLabel)

Represents a custom mailing label. The **CustomLabel** object is a member of the **CustomLabels** collection. The **CustomLabels** collection contains all the custom mailing labels listed in the **Label Options** dialog box.

Using the CustomLabel Object

Use **CustomLabels**(*index*), where *index* is the custom label name or index number, to return a single **CustomLabel** object. The following example creates a new document with the custom label layout named "My Labels."

```
Set ML = Application.MailingLabel
If ML.CustomLabels("My Labels").Valid = True Then
    ML.CreateNewDocument Name:="My Labels"
Else
    MsgBox "The My Labels custom label is not available"
End If
```

The index number represents the position of the custom mailing label in the **CustomLabels** collection. The following example displays the name of the first custom mailing label.

```
If Application.MailingLabel.CustomLabels.Count >= 1 Then
    MsgBox Application.MailingLabel.CustomLabels(1).Name
End If
```

Use the **Add** method to create a custom label. The following example adds a custom mailing label named "My Label" and sets the page size.

```
Set ML = Application.MailingLabel.CustomLabels.Add(Name:="My Labels", _
    DotMatrix:=False)
ML.PageSize = wdCustomLabelA4
```

CustomLabels Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjCustomLabelsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjCustomLabelsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjCustomLabelsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjCustomLabelsE "}

MailingLabel

CustomLabels (CustomLabel)

A collection of **CustomLabel** objects available in the **Label Options** dialog box. This collection includes custom labels of all printer types (dot-matrix, laser, and ink-jet printers).

Using the CustomLabels Collection

Use the **CustomLabels** property to return the **CustomLabels** collection. The following example displays the number of available custom labels.

```
MsgBox Application.MailingLabel.CustomLabels.Count
```

Use the **Add** method to create a custom label. The following example adds a custom mailing label named "My Label" and sets the page size.

```
Set ML = Application.MailingLabel.CustomLabels.Add(Name:="My Labels", _
    DotMatrix:=False)
ML.PageSize = wdCustomLabelA4
```

Use **CustomLabels**(*index*), where *index* is the custom label name or index number, to return a single **CustomLabel** object. The following example creates a new document with the custom label layout named "My Labels."

```
Set ML = Application.MailingLabel
If ML.CustomLabels("My Labels").Valid = True Then
    ML.CreateNewDocument Name:="My Labels"
Else
    MsgBox "The My Labels custom label is not available"
End If
```

The index number represents the position of the custom mailing label in the **CustomLabels** collection. The following example displays the name of the first custom mailing label.

```
If Application.MailingLabel.CustomLabels.Count >= 1 Then
    MsgBox Application.MailingLabel.CustomLabels(1).Name
End If
```

Dialogs Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDialogsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDialogsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDialogsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDialogsE "}

Dialogs (Dialog)

A collection of **Dialog** objects in Word. Each **Dialog** object represents a built-in Word dialog box.

Using the Dialogs Collection

Use the **Dialogs** property to return the **Dialogs** collection. The following example displays the number of available built-in dialog boxes.

```
MsgBox Dialogs.Count
```

You cannot create a new built-in dialog box or add one to the **Dialogs** collection. Use **Dialogs**(*index*), where *index* is the **WdWordDialog** constant that identifies the dialog box, to return a single **Dialog** object. The following example displays the built-in **Open** dialog box.

```
dlgAnswer = Dialogs(wdDialogFileOpen).Show
```

For more information, see Displaying built-in Word dialog boxes.

Document Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDocumentP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDocumentM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDocumentE "}

Documents (Document)

Represents a document. The **Document** object is a member of the **Documents** collection. The **Documents** collection contains all the **Document** objects that are currently open in Word.

Using the Document Object

Use **Documents**(*index*), where *index* is the document name or index number to return a single **Document** object. The following example closes the document named "Report.doc" without saving changes.

```
Documents("Report.doc").Close SaveChanges:=wdDoNotSaveChanges
```

The index number represents the position of the document in the **Documents** collection. The following example activates the first document in the **Documents** collection.

```
Documents(1).Activate
```

Using ActiveDocument

You can use the **ActiveDocument** property to refer to the document with the focus. The following example uses the **Activate** method to activate the document named "Document 1." The example also sets the page orientation to landscape mode and then prints the document.

```
Documents("Document1").Activate
ActiveDocument.PageSetup.Orientation = wdOrientLandscape
ActiveDocument.PrintOut
```

Application, **Pane**, **Selection**, **Window**

Documents Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDocumentsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDocumentsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDocumentsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDocumentsE "}

Documents (Document)

A collection of all the **Document** objects that are currently open in Word.

Using the Documents Collection

Use the **Documents** property to return the **Documents** collection. The following example displays the names of the open documents.

```
For Each aDoc In Documents
    aName = aName & aDoc.Name & vbCr
Next aDoc
MsgBox aName
```

Use the **Add** method to create a new empty document and add it to the **Documents** collection. The following example creates a new document based on the Normal template.

```
Documents.Add
```

Use the **Open** method to open a file. The following example opens the document named "Sales.doc."

```
Documents.Open FileName:="C:\My Documents\Sales.doc"
```

Use **Documents**(*index*), where *index* is the document name or index number to return a single **Document** object. The following instruction closes the document named "Report.doc" without saving changes.

```
Documents("Report.doc").Close SaveChanges:=wdDoNotSaveChanges
```

The index number represents the position of the document in the **Documents** collection. The following example activates the first document in the **Documents** collection.

```
Documents(1).Activate
```

Remarks

The following example enumerates the **Documents** collection to determine whether the document named "Report.doc" is open. If this document is contained in the **Documents** collection, the document is activated; otherwise, it's opened.

```
For Each doc In Documents
    If doc.Name = "Report.doc" Then found = True
Next doc
If found <> True Then
    Documents.Open FileName:="C:\Documents\Report.doc"
Else
```

```
    Documents("Report.doc").Activate
End If
```

Bookmarks, **Characters**, **CommandBars**, **Comments**, **DocumentProperties**, **Endnotes**, **Envelope**, **Fields**, **Footnotes**, **FormFields**, **Frames**, **HyperLinks**, **Indexes**, **InlineShapes**, **LetterContent**, **ListParagraphs**, **Lists**, **ListTemplates**, **Mailer**, **MailMerge**, **PageSetup**, **Paragraphs**, **ProofreadingErrors**, **Range**, **ReadabilityStatistics**, **Revisions**, **RoutingSlip**, **Sections**, **Sentences**, **Shapes**, **StoryRanges**, **Styles**, **Subdocuments**, **Tables**, **TablesOfAuthorities**, **TablesOfAuthoritiesCategories**, **TablesOfContents**, **TablesOfFigures**, **Template**, **Variables**, **VBProject**, **Versions**, **Windows**, **Words**

DropCap Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDropCapC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDropCapP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDropCapM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDropCapE "}

Paragraphs (Paragraph)

DropCap

Represents a dropped capital letter at the beginning of a paragraph. There is no DropCaps collection; each **Paragraph** object contains only one **DropCap** object.

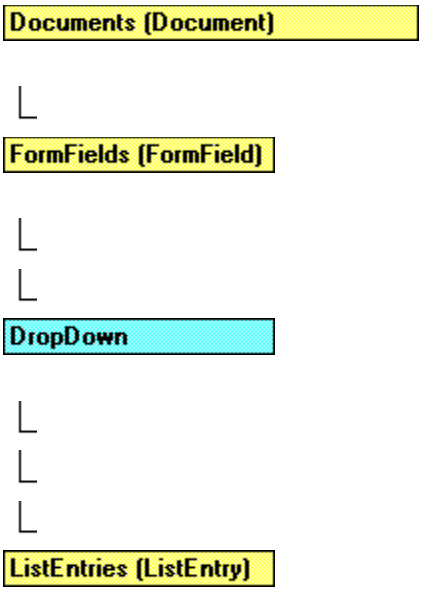
Using the DropCap Object

Use the **DropCap** property to return a **DropCap** object. The following example sets a dropped capital letter for the first letter in the first paragraph in the active document.

```
With ActiveDocument.Paragraphs(1).DropCap
    .Enable
    .Position = wdDropNormal
End With
```

DropDown Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDropDownC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDropDownP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDropDownM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDropDownE "}



Represents a drop-down form field that contains a list of items in a form.

Using the DropDown Object

Use **FormFields**(*index*), where *index* is the index number or the bookmark name associated with the drop-down form field, to return a single **FormField** object. Use the **DropDown** property with the **FormField** object to return a **DropDown** object. The following example selects the first item in the drop-down form field named "DropDown" in the active document.

```
ActiveDocument.FormFields("DropDown1").DropDown.Value = 1
```

The index number represents the position of the form field in the **FormFields** collection. The following example checks the type of the first form field in the active document. If it's a drop-down form field, the second item is selected.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormDropDown Then
    ActiveDocument.FormFields(1).DropDown.Value = 2
End If
```

The following example determines whether form field represented by `ffield` is a valid drop-down form field before adding an item to it.

```
Set ffield = ActiveDocument.FormFields(1).DropDown
If ffield.Valid = True Then
    ffield.ListEntries.Add Name:="Hello"
Else
    MsgBox "First field is not a drop down"
End If
```

Use the **Add** method with the **FormFields** collection to add a drop-down form field. The following example adds a drop-down form field at the beginning of the active document and then adds items to the form field.

```
Set ffield = ActiveDocument.FormFields.Add( _
    Range:=ActiveDocument.Range(Start:=0, End:=0),
Type:=wdFieldFormDropDown)
```

```
With ffield
    .Name = "Colors"
    With .DropDown.ListEntries
        .Add Name:="Blue"
        .Add Name:="Green"
        .Add Name:="Red"
    End With
End With
```

Endnote Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjEndnoteC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjEndnoteP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjEndnoteM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjEndnoteE "}

Endnotes (Endnote)

Represents an endnote. The **Endnote** object is a member of the **Endnotes** collection. The **Endnotes** collection represents the endnotes in a selection, range, or document.

Using the Endnote Object

Use **Endnotes**(*index*), where *index* is the index number, to return a single **Endnote** object. The index number represents the position of the endnote in the selection, range, or document. The following example applies red formatting to the first endnote in the selection.

```
If Selection.Endnotes.Count >= 1 Then
    Selection.Endnotes(1).Reference.Font.ColorIndex = wdRed
End If
```

Use the **Add** method to add an endnote to the **Endnotes** collection. The following example adds an endnote immediately after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Endnotes.Add Range:=Selection.Range , _
    Text:="The Willow Tree, (Lone Creek Press, 1996)."
```

Endnotes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjEndnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjEndnotesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjEndnotesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjEndnotesE "}

Endnotes (Endnote)

A collection of **Endnote** objects that represents all the endnotes in a selection, range, or document.

Using the Endnotes Collection

Use the **Endnotes** property to return the **Endnotes** collection. The following example sets the location of endnotes in the active document.

```
ActiveDocument.Endnotes.Location = wdEndOfSection
```

Use the **Add** method to add an endnote to the **Endnotes** collection. The following example adds an endnote immediately after the selection.

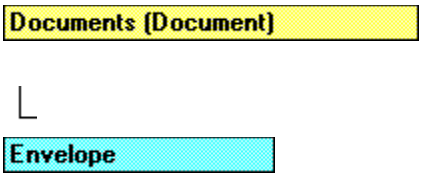
```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Endnotes.Add Range:=Selection.Range ,
    Text:="The Willow Tree, (Lone Creek Press, 1996)."
```

Use **Endnotes**(*index*), where *index* is the index number, to return a single **Endnote** object. The index number represents the position of the endnote in a selection, range, or document. The following example applies red formatting to the first endnote in the selection.

```
If Selection.Endnotes.Count >= 1 Then
    Selection.Endnotes(1).Reference.Font.ColorIndex = wdRed
End If
```

Envelope Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjEnvelopeC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjEnvelopeP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjEnvelopeM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjEnvelopeE "}



Represents an envelope. There is no Envelopes collection; each **Document** object contains only one **Envelope** object.

Using the Envelope Object

Use the **Envelope** property to return the **Envelope** object. The following example adds an envelope to a new document and sets the distance between the top of the envelope and the address to 2.25 inches.

```
Set myDoc = Documents.Add
addr = "Michael Matey" & vbCr & "123 Skye St." & vbCr & "Redmond, WA 98107"
retaddr = "Cora Edmonds" & vbCr & "456 Erde Lane" & vbCr & _
    "Redmond, WA 98107"
With myDoc.Envelope
    .Insert Address:=addr, ReturnAddress:=retaddr
    .AddressFromTop = InchesToPoints(2.25)
End With
```

Remarks

The **Envelope** object is available regardless of whether an envelope has been added to the specified document. However, an error occurs if you use one of the following properties when an envelope hasn't been added to the document: **Address**, **AddressFromleft**, **AddressFromTop**, **FeedSource**, **ReturnAddress**, **ReturnAddressFromLeft**, **ReturnAddressFromTop**, and **UpdateDocument**.

The following example demonstrates how to use the **On Error GoTo** statement to trap the error that occurs if an envelope hasn't been added to the active document. If, however, an envelope has been added to the document, the recipient address is displayed.

```
On Error GoTo ErrorHandler
MsgBox ActiveDocument.Envelope.Address
ErrorHandler:
If Err = 5852 Then MsgBox "Envelope is not in the specified document"
```

Use the **Insert** method to add an envelope to the specified document. Use the **PrintOut** method to set the properties of an envelope and print it without adding it to the document.

Range, **Style**

Field Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFieldC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFieldP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFieldM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFieldE "}

Fields (Field)

Represents a field. The **Field** object is a member of the **Fields** collection. The **Fields** collection represents the fields in a selection, range, or document.

Using the Field Object

Use **Fields**(*index*), where *index* is the index number, to return a single **Field** object. The index number represents the position of the field in the selection, range, or document. The following example displays the field code and the result of the first field in the active document.

```
If ActiveDocument.Fields.Count >= 1 Then
    MsgBox "Code =  " & ActiveDocument.Fields(1).Code & vbCr _
        & "Result =  " & ActiveDocument.Fields(1).Result & vbCr
End If
```

Use the **Add** method to add a field to the **Fields** collection. The following example inserts a DATE field at the beginning of the selection and then displays the result.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldDate)
MsgBox myField.Result
```

The **wdFieldDate** constant is part of the **WdFieldType** group of constants, which includes all the various field types.

Fields Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFieldsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFieldsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFieldsE "}

Fields (Field)

A collection of **Field** objects that represent all the fields in a selection, range, or document.

Using the Fields Collection

Use the **Fields** property to return the **Fields** collection. The following example updates all the fields in the selection.

```
Selection.Fields.Update
```

Use the **Add** method to add a field to the **Fields** collection. The following example inserts a DATE field at the beginning of the selection and then displays the result.

```
Selection.Collapse Direction:=wdCollapseStart
Set myField = ActiveDocument.Fields.Add(Range:=Selection.Range, _
    Type:=wdFieldDate)
MsgBox myField.Result
```

Use **Fields**(*index*), where *index* is the index number, to return a single **Field** object. The index number represents the position of the field in the selection, range, or document. The following example displays the field code and the result of the first field in the active document.

```
If ActiveDocument.Fields.Count >= 1 Then
    MsgBox "Code =  " & ActiveDocument.Fields(1).Code & vbCr _
        & "Result =  " & ActiveDocument.Fields(1).Result & vbCr
End If
```

Remarks

Use the **Fields** property with a **MailMerge** object to return the **MailMergeFields** collection.

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

InlineShape, **LinkFormat**, **OLEFormat**,**Range**

FileConverter Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFileConverterC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFileConverterP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFileConverterM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFileConverterE "}

FileConverters (FileConverter)

Represents a file converter that's used to open or save files. The **FileConverter** object is a member of the **FileConverters** collection. The **FileConverters** collection contains all the installed file converters for opening and saving files.

Using the FileConverter Object

Use **FileConverters**(*index*), where *index* is a class name or index number, to return a single **FileConverter** object. The following example displays the path of the Microsoft Excel worksheet converter.

```
MsgBox FileConverters("MSBiff").Path
```

The index number represents the position of the file converter in the **FileConverters** collection. The following example displays the format name of the first file converter.

```
MsgBox FileConverters(1).FormatName
```

You cannot create a new file converter or add one to the **FileConverters** collection. **FileConverter** objects are added during setup of Microsoft Office or by installing supplemental file converters. Use either the **CanSave** or **CanOpen** property to determine whether a **FileConverter** object can be used to open or save document.

Remarks

File converters for saving documents are listed in the **Save As** dialog box. File converters for opening documents appear in a dialog box if the **Confirm conversion at Open** check box is selected on the **General** tab in the **Options** dialog box (**Tools** menu).

FileConverters Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFileConvertersC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFileConvertersP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFileConvertersM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFileConvertersE "}

FileConverters (FileConverter)

A collection of **FileConverter** objects that represent all the file converters available for opening and saving files.

Using the FileConverters Collection

Use the **FileConverters** property to return the **FileConverters** collection. The following example determines whether a WordPerfect 6.0 converter is available.

```
For Each conv In FileConverters
    If conv.FormatName = "WordPerfect 6.x" Then
        MsgBox "WordPerfect 6.0 converter is installed"
    End if
Next conv
```

The **Add** method isn't available for the **FileConverters** collection. **FileConverter** objects are added during setup of Microsoft Office or by installing supplemental converters.

Use **FileConverters**(*index*), where *index* is a class name or index number, to return a single **FileConverter** object. The following example displays the path of the Microsoft Excel worksheet converter.

```
MsgBox FileConverters("MSBiff").Path
```

The index number represents the position of the file converter in the **FileConverters** collection. The following example displays the format name of the first file converter.

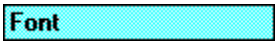
```
MsgBox FileConverters(1).FormatName
```

Remarks

File converters for saving documents are listed in the **Save As** dialog box. File converters for opening documents appear in a dialog box if the **Confirm conversion at Open** check box is selected on the **General** tab in the **Options** dialog box (**Tools** menu).

Font Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFontC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFontP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFontM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFontE "}



Contains font attributes (font name, font size, color, and so on) for an object.

Using the Font Object

Use the **Font** property to return the **Font** object. The following instruction applies bold formatting to the selection.

```
Selection.Font.Bold = True
```

The following example formats the first paragraph in the active document as 24point Arial and italic.

```
Set myRange = ActiveDocument.Paragraphs(1).Range
With myRange.Font
    .Bold = True
    .Name = "Arial"
    .Size = 24
End With
```

The following example changes the formatting of the Heading 2 style in the active document to Arial and bold.

```
With ActiveDocument.Styles(wdStyleHeading2).Font
    .Name = "Arial"
    .Italic = True
End With
```

Remarks

You can use the **New** keyword to create a new, stand-alone **Font** object. The following example creates a **Font** object, sets some formatting properties, and then applies the **Font** object to the first paragraph in the active document.

```
Set myFont = New Font
myFont.Bold = True
myFont.Name = "Arial"
ActiveDocument.Paragraphs(1).Range.Font = myFont
```

You can also duplicate a **Font** object by using the **Duplicate** property. The following example creates a new character style with the character formatting from the selection as well as italic formatting. The formatting of the selection isn't changed.

```
Set aFont = Selection.Font.Duplicate
aFont.Italic = True
```

```
ActiveDocument.Styles.Add(Name:="Italics", Type:=wdStyleTypeCharacter).Font
= aFont
```

Find, **ListLevel**, **Range**, **Replacement**, **Selection**, **Style**

Borders, **Shading**

FontNames Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFontNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFontNamesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFontNamesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFontNamesE "}

FontNames

Represents a list of the names of all the available fonts.

Using the FontNames Object

Use the **FontNames**, **LandscapeFontNames**, or **PortraitFontNames** property to return the **FontNames** object. The following example displays the number of portrait fonts available.

```
MsgBox PortraitFontNames.Count & " fonts available"
```

This example lists all the font names in the **FontNames** object at the end of the active document.

```
For Each aFont In FontNames
    ActiveDocument.Range.InsertAfter aFont & vbCr
Next aFont
```

Use **FontNames**(*index*), where *index* is the index number, to return the name of a font. The following example displays the first font name in the **FontNames** object.

```
MsgBox FontNames(1)
```

Remarks

You cannot add names to or remove names from the list of available font names.

Footnote Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFootnoteC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFootnoteP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFootnoteM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFootnoteE "}

Footnotes (Footnote)

Represents a footnote positioned at the bottom of the page or beneath text. The **Footnote** object is a member of the **Footnotes** collection. The **Footnotes** collection represents the footnotes in a selection, range, or document.

Using the Footnote Object

Use **Footnotes**(*index*), where *index* is the index number, to return a single **Footnote** object. The index number represents the position of the footnote in the selection, range, or document. The following example applies red formatting to the first footnote in the selection.

```
If Selection.Footnotes.Count >= 1 Then
    Selection.Footnotes(1).Reference.Font.ColorIndex = wdRed
End If
```

Use the **Add** method to add a footnote to the **Footnotes** collection. The following example inserts an automatically numbered footnote immediately after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Footnotes.Add Range:=Selection.Range , _
    Text:="The Willow Tree, (Lone Creek Press, 1996)."
```

Remarks

Footnotes positioned at the end of a document or section are considered endnotes and are included in the **Endnotes** collection.

Footnotes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFootnotesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFootnotesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFootnotesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFootnotesE "}

Footnotes (Footnote)

A collection of **Footnote** objects that represent all the footnotes in a selection, range, or document.

Using the Footnotes Collection

Use the **Footnotes** property to return the **Footnotes** collection. The following example changes all of the footnotes in the active document to endnotes.

```
ActiveDocument.Footnotes.SwapWithEndnotes
```

Use the **Add** method to add a footnote to the **Footnotes** collection. The following example adds a footnote immediately after the selection.

```
Selection.Collapse Direction:=wdCollapseEnd
ActiveDocument.Footnotes.Add Range:=Selection.Range , _
    Text:="The Willow Tree, (Lone Creek Press, 1996)."
```

Use **Footnotes**(*index*), where *index* is the index number, to return a single **Footnote** object. The index number represents the position of the footnote in the selection, range, or document. The following example applies red formatting to the first footnote in the selection.

```
If Selection.Footnotes.Count >= 1 Then
    Selection.Footnotes(1).Reference.Font.ColorIndex = wdRed
End If
```

Remarks

Footnotes positioned at the end of a document or section are considered endnotes and are included in the **Endnotes** collection.

FormField Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFormFieldC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFormFieldP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFormFieldM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFormFieldE "}

Represents a single form field. The **FormField** object is a member of the **FormFields** collection.

Using the FormField Object

Use **FormFields**(*index*), where *index* is a bookmark name or index number, to return a single **FormField** object. The following example sets the result of the Text1 form field to "Don Funk."

```
ActiveDocument.FormFields("Text1").Result = "Don Funk"
```

The index number represents the position of the form field in the selection, range, or document. The following example displays the name of the first form field in the selection.

```
If Selection.FormFields.Count >= 1 Then
    MsgBox Selection.FormFields(1).Name
End If
```

Use the **Add** method with the **FormFields** object to add a form field. The following example adds a check box at the beginning of the active document and then selects the check box.

```
Set ffield = ActiveDocument.FormFields.Add( _
    Range:=ActiveDocument.Range(Start:=0, End:=0),
Type:=wdFieldFormCheckBox)
ffield.CheckBox.Value = True
```

Remarks

Use the **CheckBox**, **DropDown**, and **TextInput** properties with the **FormField** object to return the **CheckDown**, **DropDown**, and **TextInput** objects. The following example selects the check box named "Check1."

```
ActiveDocument.FormFields("Check1").CheckBox.Value = True
```

FormFields Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFormFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFormFieldsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFormFieldsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFormFieldsE "}

A collection of **FormField** objects that represent all the form fields in a selection, range, or document.

Using the FormFields Collection

Use the **FormFields** property to return the **FormFields** collection. The following example counts the number of text box form fields in the active document.

```
For Each aField In ActiveDocument.FormFields
    If aField.Type = wdFieldFormTextInput Then count = count + 1
Next aField
MsgBox "There are " & count & " text boxes in this document"
```

Use the **Add** method with the **FormFields** object to add a form field. The following example adds a check box at the beginning of the active document and then selects the check box.

```
Set ffield = ActiveDocument.FormFields.Add( _
    Range:=ActiveDocument.Range(Start:=0,End:=0),
Type:=wdFieldFormCheckBox)
ffield.CheckBox.Value = True
```

Use **FormFields**(*index*), where *index* is a bookmark name or index number, to return a single **FormField** object. The following example sets the result of the Text1 form field to "Don Funk."

```
ActiveDocument.FormFields("Text1").Result = "Don Funk"
```

The index number represents the position of the form field in the selection, range, or document. The following example displays the name of the first form field in the selection.

```
If Selection.FormFields.Count >= 1 Then
    MsgBox Selection.FormFields(1).Name
End If
```

CheckBox, **DropDown**, **Range**, **TextInput**

Frame Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFrameC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFrameP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFrameM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFrameE "}

Frames (Frame)

Represents a frame. The **Frame** object is a member of the **Frames** collection. The **Frames** collection includes all frames in a selection, range, or document.

Using the Frame Object

Use **Frames**(*index*), where *index* is the index number, to return a single **Frame** object. The index number represents the position of the frame in the selection, range, or document. The following example allows text to wrap around the first frame in the active document.

```
ActiveDocument.Frames(1).TextWrap = True
```

Use the **Add** method to add a frame around a range. The following example adds a frame around the first paragraph in the active document.

```
ActiveDocument.Frames.Add Range:=ActiveDocument.Paragraphs(1).Range
```

Remarks

You can wrap text around **Shape** or **ShapeRange** objects by using the **WrapFormat** property.You can position a **Shape** or **ShapeRange** object by using the **Top** and **Left** properties.

Frames Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFramesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFramesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFramesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFramesE "}

Frames (Frame)

A collection of **Frame** objects in a selection, range, or document.

Using the Frames Collection

Use the **Frames** property to return the **Frames** collection. The following example removes borders from all frames in the active document.

```
For Each aFrame In ActiveDocument.Frames
    aFrame.Borders.Enable = False
Next aFrame
```

Use the **Add** method to add a frame around a range. The following example adds a frame around the first paragraph in the active document.

```
ActiveDocument.Frames.Add Range:=ActiveDocument.Paragraphs(1).Range
```

Use **Frames**(*index*), where *index* is the index number, to return a single **Frame** object. The index number represents the position of the frame in the selection, range, or document. The following example causes text to wrap around the first frame in the first section of the active document.

```
ActiveDocument.Sections(1).Range.Frames(1).TextWrap = True
```

Remarks

You can wrap text around **Shape** or **ShapeRange** objects by using the **WrapFormat** property.You can position a **Shape** or **ShapeRange** object by using the **Top** and **Left** properties.

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

Borders, **Range**, **Shading**

HeaderFooter Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHeaderFooterC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHeaderFooterP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHeaderFooterM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHeaderFooterE "}

Represents a single header or footer. The **HeaderFooter** object is a member of the **HeaderFooters** collection. The **HeaderFooters** collection includes all headers and footers in the specified document section.

Using the HeaderFooter Object

Use **Headers**(*index*) or **Footers**(*index*), where *index* is one of the **WdHeaderFooterIndex** constants (**wdHeaderFooterEvenPages**, **wdHeaderFooterFirstPage**, or **wdHeaderFooterPrimary**), to return a single **HeaderFooter** object. The following example changes the text of both the primary header and the primary footer the first section of the active document.

```
With ActiveDocument.Sections(1)
    .Headers(wdHeaderFooterPrimary).Range.Text = "Header text"
    .Footers(wdHeaderFooterPrimary).Range.Text = "Footer text"
End With
```

You can also return a single **HeaderFooter** object by using the **HeaderFooter** property with a **Selection** object.

Note You cannot add **HeaderFooter** objects to the **HeaderFooters** collection.

Remarks

Use the **DifferentFirstPageHeaderFooter** property with the **PageSetup** object to specify a different first page. The following example inserts text into the first page footer in the active document.

```
With ActiveDocument
    .PageSetup.DifferentFirstPageHeaderFooter = True
    .Sections(1).Footers(wdHeaderFooterFirstPage).Range.InsertBefore _
        "Written by Joe Smith"
End With
```

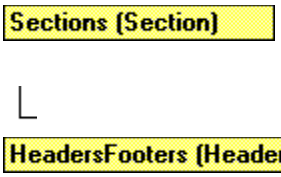
Use the **OddAndEvenPagesHeaderFooter** property with the **PageSetup** object to specify different odd and even page headers and footers. If the **OddAndEvenPagesHeaderFooter** property is **True**, you can return an odd header or footer by using **wdHeaderFooterPrimary**, and you can return an even header or footer by using **wdHeaderFooterEvenPages**.

Use the **Add** method with the **PageNumbers** object to add a page number to a header or footer. The following example adds page numbers to the primary footer the first section of the active document.

```
With ActiveDocument.Sections(1)
    .Footers(wdHeaderFooterPrimary).PageNumbers.Add
End With
```

HeadersFooters Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHeadersFootersC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHeadersFootersP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHeadersFootersM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHeadersFootersE "}



A collection of **HeaderFooter** objects that represent the headers or footers in the specified section of a document.

Using the HeaderFooters Collection

Use the **Headers** or **Footers** property to return the **HeaderFooters** collection. The following example displays the text from the primary footer in the first section of the active document.

```
With ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary)
    If .Range.Text <> vbCr Then
        MsgBox .Range.Text
    Else
        MsgBox "Footer is empty"
    End If
End With
```

Note You cannot add **HeaderFooter** objects to the **HeaderFooters** collection.

Use **Headers**(*index*) or **Footers**(*index*), where *index* is one of the **WdHeaderFooterIndex** constants (**wdHeaderFooterEvenPages**, **wdHeaderFooterFirstPage**, or **wdHeaderFooterPrimary**), to return a single **HeaderFooter** object. The following example changes the text of both the primary header and the primary footer the first section of the active document.

```
With ActiveDocument.Sections(1)
    .Headers(wdHeaderFooterPrimary).Range.Text = "Header text"
    .Footers(wdHeaderFooterPrimary).Range.Text = "Footer text"
End With
```

You can also return a single **HeaderFooter** object by using the **HeaderFooter** property with a **Selection** object.

Remarks

Use the **DifferentFirstPageHeaderFooter** property with the **PageSetup** object to specify a different first page. The following example inserts text into the first page footer in the active document.

```
With ActiveDocument
    .PageSetup.DifferentFirstPageHeaderFooter = True
    .Sections(1).Footers(wdHeaderFooterFirstPage).Range.InsertBefore _
        "Written by Kate Edson"
End With
```

Use the **OddAndEvenPagesHeaderFooter** property with the **PageSetup** object to specify different odd and even page headers and footers. If the **OddAndEvenPagesHeaderFooter** property is **True**,

you can return an odd header or footer by using **wdHeaderFooterPrimary**, and you can return an even header or footer by using **wdHeaderFooterEvenPages**.

Use the **Add** method with the **PageNumbers** object to add a page number to a header or footer. The following example adds page numbers to the first page footer in the first section in the active document.

```
With ActiveDocument.Sections(1)
    .PageSetup.DifferentFirstPageHeaderFooter = True
    .Footers(wdHeaderFooterFirstPage).PageNumbers.Add FirstPage:=True
End With
```

PageNumbers, **Range**, **Shapes**

HeadingStyle Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHeadingStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHeadingStyleP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHeadingStyleM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHeadingStyleE "}

HeadingStyles (HeadingStyle)

Represents a style used to build a table of contents or figures. The **HeadingStyle** object is a member of the **HeadingStyles** collection.

Using the HeadingStyle Object

Use **HeadingStyles**(*index*), where *index* is the index number, to return a single **HeadingStyle** object. The index number represents the position of the style in the **HeadingStyles** collection. The following example adds (at the beginning of the active document) a table of figures built from the Title style, and then displays the name of the first style in the **HeadingStyles** collection.

```
Set myTOF = ActiveDocument.TablesOfFigures.Add _
    (Range:=ActiveDocument.Range(0, 0), AddedStyles:="Title")
MsgBox myTOF.HeadingStyles(1).Style
```

Use the **Add** method to add a style to the **HeadingStyles** collection. The following example adds a table of contents at the beginning of the active document and then adds the Title style to the list of styles used to build a table of contents.

```
Set myToc = ActiveDocument.TablesOfContents.Add _
    (Range:=ActiveDocument.Range(0, 0), UseHeadingStyles:=True, _
     LowerHeadingLevel:=3, UpperHeadingLevel:=1)
myToc.HeadingStyles.Add Style:="Title", Level:=2
```

HeadingStyles Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHeadingStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHeadingStylesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHeadingStylesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHeadingStylesE "}

HeadingStyles (HeadingStyle)

A collection of **HeadingStyle** objects that represent the styles used to compile a table of figures or table of contents.

Using the HeadingStyles Collection

Use the **HeadingStyles** property to return the **HeadingStyles** collection. The following example displays the number of items in the **HeadingStyles** collection for the first table of contents in the active document.

```
MsgBox ActiveDocument.TablesOfContents(1).HeadingStyles.Count
```

Use the **Add** method to add a style to the **HeadingStyles** collection. The following example adds a table of contents at the beginning of the active document and then adds the Title style to the list of styles used to build a table of contents.

```
Set myToc = ActiveDocument.TablesOfContents.Add _
    (Range:=ActiveDocument.Range(0, 0), UseHeadingStyles:=True, _
     LowerHeadingLevel:=3, UpperHeadingLevel:=1)
myToc.HeadingStyles.Add Style:="Title", Level:=2
```

Use **HeadingStyles**(*index*), where *index* is the index number, to return a single **HeadingStyle** object. The index number represents the position of the style in the **HeadingStyles** collection. The following example adds (at the beginning of the active document) a table of figures built from the Title style, and then displays the name of the first style in the **HeadingStyles** collection.

```
Set myTOF = ActiveDocument.TablesOfFigures.Add _
    (Range:=ActiveDocument.Range(0, 0), AddedStyles:="Title")
MsgBox myTOF.HeadingStyles(1).Style
```

TableOfContents, **TableOfFigures**

Index Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjIndexC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjIndexP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjIndexM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjIndexE "}

Indexes (Index)

Represents a single index. The **Index** object is a member of the **Indexes** collection. The **Indexes** collection includes all the indexes in the specified document.

Using the Index Object

Use **Indexes**(*index*), where *index* is the index number, to return a single **Index** object. The index number represents the position of the **Index** object in the document. The following example updates the first index in the active document.

```
If ActiveDocument.Indexes.Count >= 1 Then
    ActiveDocument.Indexes(1).Update
End If
```

Use the **Add** method to create an index and add it to the **Indexes** collection. The following example creates an index at the end of the active document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.Indexes.Add Range:=myRange, Type:=wdIndexRunin
```

Indexes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjIndexesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjIndexesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjIndexesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjIndexesE "}

Indexes (Index)

A collection of **Index** objects that represent all the indexes in the specified document.

Using the Indexes Collection

Use the **Indexes** property to return the **Indexes** collection. The following example formats indexes in the active document with the classic format.

```
ActiveDocument.Indexes.Format = wdIndexClassic
```

Use the **Add** method to create an index and add it to the **Indexes** collection. The following example creates an index at the end of the active document.

```
Set myRange = ActiveDocument.Content
myRange.Collapse Direction:=wdCollapseEnd
ActiveDocument.Indexes.Add Range:=myRange, Type:=wdIndexRunin
```

Use **Indexes**(*index*), where *index* is the index number, to return a single **Index** object. The index number represents the position of the **Index** object in the document. The following example updates the first index in the active document.

```
If ActiveDocument.Indexes.Count >= 1 Then
    ActiveDocument.Indexes(1).Update
End If
```

MailingLabel Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailingLabelC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailingLabelP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailingLabelM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailingLabelE "}

Represents a mailing label.

Using the MailingLabel Object

Use the **MailingLabel** property to return the **MailingLabel** object. The following example sets default mailing label options.

```
With Application.MailingLabel
    .DefaultLaserTray = wdPrinterLowerBin
    .DefaultPrintBarCode = True
End With
```

Use the **PrintOut** method to print a mailing label listed in the **Product Number** box in the **Label Options** dialog box. The following example prints a page of Avery 5162 standard address labels using the specified address.

```
addr = "Katie Jordan" & vbCr & "123 Skye St." & vbCr & "OurTown, WA 98107"
Application.MailingLabel.PrintOut Name:="5162", Address:=addr
```

Remarks

Use the **CustomLabels** property to format or print a custom mailing label. The following example sets the number of labels across and down for the custom label named "MyLabel."

```
With Application.MailingLabel.CustomLabels("MyLabel")
    .NumberAcross = 2
    .NumberDown = 5
End With
```

LineNumbering Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLineNumberingC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLineNumberingP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLineNumberingM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjLineNumberingE "}

LineNumbering

Represents line numbers in the left margin or to the left of each newspaper-style column.

Using the LineNumbering Object

Use the **LineNumbering** property to return the **LineNumbering** object. The following example applies line numbering to the text in the first section of the active document.

```
With ActiveDocument.Sections(1).PageSetup.LineNumbering
    .Active = True
    .CountBy = 5
    .RestartMode = wdRestartPage
End With
```

The following example applies line numbering to the pages in the current section.

```
Selection.PageSetup.LineNumbering.Active = True
```

Document, **Range**, **Section**, **Sections**, **Selection**

ListEntries Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListEntriesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListEntriesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListEntriesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListEntriesE "}

A collection of **ListEntry** objects that represent all the items in a drop-down form field.

Using the ListEntries Collection

Use the **ListEntries** property to return the **ListEntries** collection. The following example displays the items that appear in the form field named "Drop1."

```
For Each le In ActiveDocument.FormFields("Drop1").DropDown.ListEntries
    MsgBox le.Name
Next le
```

Use the **Add** method to add an item to a drop-down form field. The following example inserts a drop-down form field and then adds "red," "blue," and "green" to the form field.

```
Set myField = ActiveDocument.FormFields.Add(Range:=Selection.Range, _
     Type:=wdFieldFormDropDown)
With myField.DropDown.ListEntries
    .Add Name:="Red"
    .Add Name:="Blue"
    .Add Name:="Green"
End With
```

Use **ListEntries**(*index*), where *index* is the list entry name or the index number, to return a single **ListEntry** object. The index number represents the position of the entry in the drop-down form field (the first item is index number 1). The following example deletes the "Blue" entry from the drop-down form field named "Color."

```
ActiveDocument.FormFields("Color").DropDown.ListEntries("Blue").Delete
```

The following example displays the first item in the drop-down form field named "Color."

```
MsgBox ActiveDocument.FormFields("Color").DropDown.ListEntries(1).Name
```

ListEntry Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListEntryC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListEntryP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListEntryM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListEntryE "}

Represents an item in a drop-down form field. The **ListEntry** object is a member of the **ListEntries** collection. The **ListEntries** collection includes all the items in a drop-down form field.

Using the ListEntry Object

Use **ListEntries**(*index*), where *index* is the list entry name or the index number, to return a single **ListEntry** object. The index number represents the position of the entry in the drop-down form field (the first item is index number 1). The following example deletes the "Blue" entry from the drop-down form field named "Color."

```
ActiveDocument.FormFields("Color").DropDown.ListEntries("Blue").Delete
```

The following example displays the first item in the drop-down form field named "Color."

```
MsgBox ActiveDocument.FormFields("Color").DropDown.ListEntries(1).Name
```

Use the **Add** method to add an item to a drop-down form field. The following example inserts a drop-down form field and then adds "red," "blue," and "green" to the form field.

```
Set myField = ActiveDocument.FormFields.Add(Range:=Selection.Range, _
     Type:=wdFieldFormDropDown)
With myField.DropDown.ListEntries
    .Add Name:="Red"
    .Add Name:="Blue"
    .Add Name:="Green"
End With
```

MailMerge Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeE "}

Represents the mail merge functionality in Word.

Using the MailMerge Object

Use the **MailMerge** property to return the **MailMerge** object. The **MailMerge** object is always available regardless of whether the mail merge operation has begun. Use the **State** property to determine the status of the mail merge operation. The following example executes a mail merge if the active document is a main document with an attached data source.

```
If ActiveDocument.MailMerge.State = wdMainAndDataSource Then
    ActiveDocument.MailMerge.Execute
End If
```

The following example merges the main document with the first three data records in the attached data source and then sends the results to the printer.

```
Set myMerge = ActiveDocument.MailMerge
If myMerge.State = wdMainAndSourceAndHeader Or _
    myMerge.State = wdMainAndDataSource Then
    With myMerge.DataSource
        .FirstRecord = 1
        .LastRecord = 3
    End With
End If
With myMerge
    .Destination = wdSendToPrinter
    .Execute
End With
```

MailMergeDataSource, **MailMergeFields**

MailMergeDataField Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeDataFieldC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeDataFieldP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeDataFieldM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeDataFieldE "}

Represents a single mail merge field in a data source. The **MailMergeDataField** object is a member of the **MailMergeDataFields** collection. The **MailMergeDataFields** collection includes all the data fields in a mail merge data source (for example, Name, Address, and City).

Using the MailMergeDataField Object

Use **DataFields**(*index*), where *index* is the data field name or the index number, to return a single **MailMergeDataField** object. The index number represents the position of the data field in the mail merge data source. The following example retrieves the first value from the FName field in the data source attached to the active document.

```
first = ActiveDocument.MailMerge.DataSource.DataFields("FName").Value
```

The following example displays the name of first field in the data source attached to the active document.

```
MsgBox ActiveDocument.MailMerge.DataSource.DataFields(1).Name
```

You cannot add fields to the **MailMergeDataFields** collection. All data fields in a data source are automatically included in the **MailMergeDataFields** collection.

MailMergeDataFields Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeDataFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeDataFieldsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeDataFieldsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeDataFieldsE "}

A collection of **MailMergeDataField** objects that represent the data fields in a mail merge data source.

Using the MailMergeDataFields Collection

Use the **DataFields** property to return the **MailMergeDataFields** collection. The following example displays the names of all the fields in the attached data source.

```
For Each afield In ActiveDocument.MailMerge.DataSource.DataFields
    MsgBox afield.Name
Next afield
```

You cannot add fields to the **MailMergeDataFields** collection. When a data field is added to a data source, the field is automatically included in the **MailMergeDataFields** collection. Use the **EditDataSource** method to edit the contents of a data source. The following example adds a data field named "Author" to a table in the attached data source.

```
If ActiveDocument.MailMerge.DataSource.Type = wdMergeInfoFromWord
    ActiveDocument.MailMerge.EditDataSource
    With ActiveDocument.Tables(1)
        .Columns.Add
        .Cell(Row:=1, Column:=.Columns.Count).Range.Text = "Author"
    End With
End If
```

Use **DataFields**(*index*), where *index* is the data field name or the index number, to return a single **MailMergeDataField** object. The index number represents the position of the data field in the mail merge data source. The following example retrieves the first value from the FName field in the data source attached to the active document.

```
first = ActiveDocument.MailMerge.DataSource.DataFields("FName").Value
```

The following example displays the name of first data field in the data source attached to the active document.

```
MsgBox ActiveDocument.MailMerge.DataSource.DataFields(1).Name
```

MailMergeFieldName Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeFieldNameC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeFieldNameP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeFieldNameM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeFieldNameE "}

Represents a mail merge field name in a data source. The **MailMergeFieldName** object is a member of the **MailMergeFieldNames**collection. The **MailMergeFieldNames** collection includes all the data field names in a mail merge data source.

Using the MailMergeFieldName Object

Use **FieldNames**(*index*), where *index* is the index number, to return a single **MailMergeFieldName** object. The index number represents the position of the field in the mail merge data source. The following example retrieves the name of the last field in the data source attached to the active document.

```
alast = ActiveDocument.MailMerge.DataSource.FieldNames.Count
afirst = ActiveDocument.MailMerge.DataSource.FieldNames(alast).Name
MsgBox afirst
```

You cannot add fields to the **MailMergeFieldNames** collection. Field names in a data source are automatically included in the **MailMergeFieldNames** collection.

MailMergeFieldNames Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeFieldNamesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeFieldNamesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeFieldNamesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeFieldNamesE "}

A collection of **MailMergeFieldName** objects that represent the field names in a mail merge data source.

Using the MailMergeFieldNames Collection

Use the **FieldNames** property to return the **MailMergeFieldNames** collection. The following example displays the names of the fields in the data source attached to the active document.

```
For Each afield In ActiveDocument.MailMerge.DataSource.FieldNames
    MsgBox afield.Name
Next afield
```

You cannot add names to the **MailMergeFieldNames** collection. When a field is added to a data source, the field name is automatically included in the **MailMergeFieldNames** collection. Use the **EditDataSource** method to edit the contents of a data source. The following example adds a data field named "Author" to a table in the data source attached to the active document.

```
If ActiveDocument.MailMerge.DataSource.Type = wdMergeInfoFromWord
    ActiveDocument.MailMerge.EditDataSource
    With ActiveDocument.Tables(1)
        .Columns.Add
        .Cell(Row:=1, Column:=.Columns.Count).Range.Text = "Author"
    End With
End If
```

RecentFile Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRecentFileC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRecentFileP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRecentFileM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRecentFileE "}

RecentFiles (RecentFile)

Represents a recently used file. The **RecentFile** object is a member of the **RecentFiles** collection. The **RecentFiles** collection includes all the files that have been used recently. The items in the **RecentFiles** collection are displayed at the bottom of the **File** menu.

Using the RecentFile Object

Use **RecentFiles**(*index*), where *index* is the index number, to return a single **RecentFile** object. The index number represents the position of the file on the **File** menu. The following example opens the first document in the **RecentFiles** collection.

```
If RecentFiles.Count >= 1 Then RecentFiles(1).Open
```

Use the **Add** method to add a file to the **RecentFiles** collection. The following example adds the active document to the list of recently-used files.

```
If ActiveDocument.Saved = True Then
    RecentFiles.Add Document:=ActiveDocument.FullName, ReadOnly:=True
End If
```

Remarks

The **SaveAs** and **Open** methods include an ***AddToRecentFiles*** argument that controls whether or not a file is added to the recently-used-files list when the file is opened or saved.

RecentFiles Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRecentFilesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRecentFilesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRecentFilesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRecentFilesE "}

RecentFiles (RecentFile)

A collection of **RecentFile** objects that represents the files that have been used recently. The items in the **RecentFiles** collection are displayed at the bottom of the **File** menu.

Using the RecentFiles Collection

Use the **RecentFiles** property to return the **RecentFiles** collection. The following example sets five as the maximum number of files that the **RecentFiles** collection can contain.

```
RecentFiles.Maximum = 5
```

Use the **Add** method to add a file to the **RecentFiles** collection. The following example adds the active document to the list of recently-used files.

```
If ActiveDocument.Saved = True Then
    RecentFiles.Add Document:=ActiveDocument.FullName, ReadOnly:=True
End If
```

Use **RecentFiles**(*index*), where *index* is the index number, to return a single **RecentFile** object. The index number represents the position of the file on the **File** menu. The following example opens the first document in the **RecentFiles** collection.

```
If RecentFiles.Count >= 1 Then RecentFiles(1).Open
```

Remarks

The **SaveAs** and **Open** methods include an ***AddToRecentFiles*** argument that controls whether or not a file is added to the recently-used-files list when the file is opened or saved.

Shading Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjShadingC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjShadingP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjShadingM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjShadingE "}

Shading

Contains shading attributes for an object.

Using the Shading Object

Use the **Shading** property to return the **Shading** object. The following example applies fine gray shading to the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Shading.Texture = wdTexture10Percent
```

The following example applies shading with different foreground and background colors to the selection.

```
With Selection.Shading
    .Texture = wdTexture20Percent
    .ForegroundPatternColorIndex = wdBlue
    .BackgroundPatternColorIndex = wdYellow
End With
```

The following example applies a vertical line texture to the first row in the first table in the active document.

```
ActiveDocument.Tables(1).Rows(1).Shading.Texture = wdTextureVertical
```

Cell, **Cells**, **Column**, **Columns**, **Font**, **Frame**, **Paragraph**, **ParagraphFormat**, **Paragraphs**, **Range**, **Row**, **Rows**, **Selection**, **Style**, **Table**

Row Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRowC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRowP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRowM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRowE "}

Rows (Row)

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 selection, range, or table.

Using the Row Object

Use **Rows**(*index*), where *index* is the index number, to return a single **Row** object. The index number represents the position of the row in the selection, range, or table. The following example deletes the first row in the first table in the active document.

```
ActiveDocument.Tables(1).Rows(1).Delete
```

Use the **Add** method to add a row to a table. The following example inserts a row before the first row in the selection.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows.Add BeforeRow:=Selection.Rows(1)
End If
```

Remarks

Use the **Cells** property to modify the individual cells in a **Row** object. The following example adds a table to the selection and then inserts numbers into each cell in the second row of the table.

```
Selection.Collapse Direction:=wdCollapseEnd
If Selection.Information(wdWithInTable) = False Then
    Set myTable = ActiveDocument.Tables.Add(Range:=Selection.Range, _
        NumRows:=3, NumColumns:=5)
    For Each aCell In myTable.Rows(2).Cells
        i = i + 1
        aCell.Range.Text = i
    Next aCell
End If
```

Rows Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRowsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRowsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRowsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRowsE "}

Rows (Row)

A collection of **Row** objects that represent the table rows in the specified selection, range, or table.

Using the Rows Collection

Use the **Rows** property to return the **Rows** collection. The following example centers rows in the first table in the active document between the left and right margins.

```
ActiveDocument.Tables(1).Rows.Alignment = wdAlignRowCenter
```

Use the **Add** method to add a row to a table. The following example inserts a row before the first row in the selection.

```
If Selection.Information(wdWithInTable) = True Then
    Selection.Rows.Add BeforeRow:=Selection.Rows(1)
End If
```

Use **Rows**(*index*), where *index* is the index number, to return a single **Row** object. The index number represents the position of the row in the selection, range, or table. The following example deletes the first row in the first table in the active document.

```
ActiveDocument.Tables(1).Rows(1).Delete
```

Borders, **Cells**, **Range**, **Shading**

Sentences Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSentencesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSentencesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSentencesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSentencesE "}

Sentences (Range)

A collection of **Range** objects that represent all the sentences in a selection, range, or document. There is no Sentence object.

Using the Sentences Collection

Use the **Sentences** property to return the **Sentences** collection. The following example displays the number of sentences selected.

```
MsgBox Selection.Sentences.Count & " sentences are selected"
```

Use **Sentences**(*index*), where *index* is the index number, to return a **Range** object that represents a sentence. The index number represents the position of a sentence in the **Sentences** collection. The following example formats the first sentence in the active document.

```
With ActiveDocument.Sentences(1)
    .Bold = True
    .Font.Size = 24
End With
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

The **Add** method isn't available for the **Sentences** collection. Instead, use the **InsertAfter** or **InsertBefore** method to add a sentence to a **Range** object. The following example inserts a sentence after the first paragraph in the active document.

```
With ActiveDocument
    MsgBox .Sentences.Count & " sentences"
    .Paragraphs(1).Range.InsertParagraphAfter
    .Paragraphs(2).Range.InsertBefore "The house is blue."
    MsgBox .Sentences.Count & " sentences"
End With
```

FirstLetterException Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFirstLetterExceptionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFirstLetterExceptionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFirstLetterExceptionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFirstLetterExceptionE "}

FirstLetterExceptions (FirstLetterException)

Represents an abbreviation excluded from automatic correction. The **FirstLetterException** object is a member of the **FirstLetterExceptions** collection. The **FirstLetterExceptions** collection includes all the excluded abbreviations.

Note The first character following a period is automatically capitialized when the **CorrectSentenceCaps** property is set to **True**. The character you type following an item in the **FirstLetterExceptions** collection isn't capitialized.

Using the FirstLetterException Object

Use **FirstLetterExceptions**(*index*), where *index* is the abbreviation or the index number, to return a single **FirstLetterException** object. The following example deletes the abbreviation "appt." from the **FirstLetterExceptions** collection.

```
AutoCorrect.FirstLetterExceptions("appt.").Delete
```

The following example displays the name of the first item in the **FirstLetterExceptions** collection.

```
MsgBox AutoCorrect.FirstLetterExceptions(1).Name
```

Use the **Add** method to add an abbreviation to the list of first-letter exceptions. The following example adds the abbreviation "addr." to this list.

```
AutoCorrect.FirstLetterExceptions.Add Name:="addr."
```

FirstLetterExceptions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFirstLetterExceptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFirstLetterExceptionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFirstLetterExceptionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjFirstLetterExceptionsE "}

FirstLetterExceptions (FirstLetterException)

A collection of **FirstLetterException** objects that represent the abbreviations excluded from automatic correction.

Note The first character following a period is automatically capitialized when the **CorrectSentenceCaps** property is set to **True**. The **FirstLetterExceptions** collection includes exceptions to this behavior (for example, abbreviations such as "addr." and "apt.").

Using the FirstLetterExceptions Collection

Use the **FirstLetterExceptions** property to return the **FirstLetterExceptions** collection. The following example deletes the abbreviation "addr." if it's included in the **FirstLetterExceptions** collection.

```
For Each aExcept In AutoCorrect.FirstLetterExceptions
    If aExcept.Name = "addr." Then aExcept.Delete
Next aExcept
```

The following example creates a new document and inserts all the AutoCorrect first-letter exceptions into it.

```
Documents.Add
For Each aExcept In AutoCorrect.FirstLetterExceptions
    With Selection
        .InsertAfter aExcept.Name
        .InsertParagraphAfter
        .Collapse Direction:=wdCollapseEnd
    End With
Next aExcept
```

Use the **Add** method to add an abbreviation to the list of first-letter exceptions. The following example adds the abbreviation "addr." to this list.

```
AutoCorrect.FirstLetterExceptions.Add Name:="addr."
```

Use **FirstLetterExceptions**(*index*), where *index* is the abbreviation or the index number, to return a single **FirstLetterException** object. The following example deletes the abbreviation "appt." from the **FirstLetterExceptions** collection.

```
AutoCorrect.FirstLetterExceptions("appt.").Delete
```

The following example displays the name of the first item in the **FirstLetterExceptions** collection.

```
MsgBox AutoCorrect.FirstLetterExceptions(1).Name
```

Language Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLanguageC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLanguageP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLanguageM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjLanguageE "}

Languages (Language)

Dictionaries (Dictionary)

Represents a language used for proofing or formatting in Word. The **Language** object is a member of the **Languages** collection.

Using the Language Object

Use **Languages**(*index*), where *index* is either a name or one of the **WdLanguageID** constants, to return a single **Language** object.

The **Name** property returns the name of a language, whereas the **NameLocal** property returns the name of a language in the language of the user. The following example returns the string "Italiano" for **Name** and "Italian (Standard)" for **NameLocal** when it's run in the U.S. English version of Word.

```
Msgbox Languages(wdItalian).Name
Msgbox Languages(wdItalian).NameLocal
```

Returning the Active Proofing Dictionaries

For each language for which proofing tools are installed, you can use the **ActiveGrammarDictionary**, **ActiveHyphenationDictionary**, **ActiveSpellingDictionary**, and **ActiveThesaurusDictionary** properties to return the corresponding **Dictionary** objects. The following example returns the full path for the active spelling dictionary used in the U.S. English version of Word.

```
Set myspell = Languages(wdEnglishUS).ActiveSpellingDictionary
MsgBox mySpell.Path & Application.PathSeparator & mySpell.Name
```

Setting the Writing Style

The writing style is the set of rules used by the grammar checker. The **WritingStyleList** property returns an array of strings that represent the available writing styles for the specified language. The following example returns the list of writing styles for U.S. English.

```
WrStyles = Languages(wdEnglishUS).WritingStyleList
For i = 1 To UBound(WrStyles)
    MsgBox WrStyles(i)
Next i
```

Use the **DefaultWritingStyle** property to set the default writing style you want Word to use.

```
Languages(wdEnglishUS).DefaultWritingStyle = "Casual"
```

You can override the default writing style with the **ActiveWritingStyle** property. This property is applied to a specified document for text marked in a specified language. The following example sets the writing style to be used for checking U.S. English, French, and German for the active document.

```
With ActiveDocument
    .ActiveWritingStyle(wdEnglishUS) = "Technical"
    .ActiveWritingStyle(wdFrench) = "Commercial"
    .ActiveWritingStyle(wdGerman) = "Technisch/Wiss"
End With
```

Remarks

You must have the proofing tools installed for each language you intend to check. You need both a .dll file and an .lex file for each of the following: the thesaurus, spelling checker, grammar checker, and hyphenation tools.

If you mark text as **wdNoProofing**, Word skips the marked text when running a spelling or grammar check. To mark text for a specified language or for no proofing, use the **Set Language** command (**Tools** menu, **Language** submenu).

Languages Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLanguagesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLanguagesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLanguagesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjLanguagesE "}

Languages (Language)

Dictionaries (Dictionary)

A collection of **Language** objects that representa languages used for proofing or formatting in Word.

Using the Languages Collection

Use the **Languages** property to return the **Languages** collection. The following example displays the localized name for each language.

```
For Each la In Languages
    Msgbox la.NameLocal
Next la
```

Use **Languages**(*index*), where *index* is either a name or one of the **WdLanguageID** constants, to return a single **Language** object.

Remarks

The **Count** property returns the number of languages for which you can mark text (languages for which proofing tools are available). To check proofing, you must install the appropriate tools for each language you intend to check. You need both a .dll file and an .lex file for each of the following: the thesaurus, spelling checker, grammar checker, and hyphenation tools.

If you mark text as **wdNoProofing**, Word skips the marked text when running a spelling or grammar check. To mark text for a specified language or for no proofing, use the **Set Language** command (**Tools** menu, **Language** sub menu).

SynonymInfo Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSynonymInfoC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSynonymInfoP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSynonymInfoM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSynonymInfoE "}

SynonymInfo

Represents the information about synonyms, antonyms, related words, or related expressions for the specified range or a given string.

Using the SynonymInfo Object

Use the **SynonymInfo** property to return a **SynonymInfo** object. The **SynonymInfo** object can be returned either from a range or from Word. If it's returned from Word, you specify the lookup word or phrase and a proofing language ID. If it's returned from a range, Word uses the specified range as the lookup word. The following example returns a **SynonymInfo** object from Word.

```
temp = SynonymInfo(Word:="meant", LanguageID:=wdEnglishUS).Found
```

The following example returns a **SynonymInfo** object from a range.

```
temp = Selection.Range.SynonymInfo.Found
```

The **Found** property, used in the precedinge examples, returns **True** if any information is found in the thesaurus for the specified range or for ***Word***. Note, however, that this property returns **True** not only if synonyms are found but also if related words, related expressions, or antonyms are found.

Many of the properties of the **SynonymInfo** object return a **Variant** that contains an array of strings. When working with these properties, you can assign the returned array to a variable and then index the variable to see the elements in the array. In the following example, **Slist** is assigned the synonym list for the first meaning of the selected word or phrase. The **UBound** function finds the upper bound of the array, and then each element is displayed in a message box.

```
Slist = Selection.Range.SynonymInfo.SynonymList(1)
For i = 1 To UBound(Slist)
    Msgbox Slist(i)
Next i
```

You can check the value of the **MeaningCount** property to prevent potential errors in your code. The following example returns a list of synonyms for the second meaning for the word or phrase in the selection and displays these synonyms in the **Immediate** pane.

```
Set synInfo = Selection.Range.SynonymInfo
If synInfo.MeaningCount >= 2 Then
    synList = synInfo.SynonymList(2)
    For i = 1 To UBound(synList)
        Debug.Print synList(i)
    Next i
Else
    MsgBox "There is no second meaning for the selection."
End If
```

Application, **Range**

SpellingSuggestion Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSpellingSuggestionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSpellingSuggestionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSpellingSuggestionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSpellingSuggestionE "}

Represents a single spelling suggestion for a misspelled word. The **SpellingSuggestion** object is a member of the **SpellingSuggestions** collection. The **SpellingSuggestions** collection includes all the suggestions for a specified word or for the first word in the specified range.

Using the SpellingSuggestion Object

Use **GetSpellingSuggestions**(*index*), where *index* is the index number, to return a single **SpellingSuggestion** object. The following example checks to see whether there are any spelling suggestions for the first word in the active document. If there are, the first suggestion is displayed in a message box.

```
If ActiveDocument.Words(1).GetSpellingSuggestions.Count <> 0 Then
    MsgBox ActiveDocument.Words(1).GetSpellingSuggestions(1).Name
EndIf
```

Remarks

The **Count** property for the **SpellingSuggestions** object returns 0 (zero) if the word is spelled correctly or if there are no suggestions.

SpellingSuggestions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSpellingSuggestionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSpellingSuggestionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSpellingSuggestionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSpellingSuggestionsE "}

A collection of **SpellingSuggestion** objects that represent all the suggestions for a specified word or for the first word in the specified range.

Using the SpellingSuggestions Collection

Use the **GetSpellingSuggestions** method to return the **SpellingSuggestions** collection. The **SpellingSuggestions** method, when applied to the **Application** object, must specify the word to be checked. When the **GetSpellingSuggestions** method is applied to a range, the first word in the range is checked. The following example checks to see whether there are any spelling suggestions for any of the words in the active document. If there are, the suggestions are displayed in message boxes.

```
For Each wd In ActiveDocument.Words
    Set sugg = wd.GetSpellingSuggestions
    If sugg.Count <> 0 Then
        For Each ss In sugg
            MsgBox ss.Name
        Next ss
    End If
Next wd
```

Remarks

You cannot add suggestions to or remove suggestions from the collection of spelling suggestions. Spelling suggestions are derived from main and custom dictionary files.

Dictionaries Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDictionariesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDictionariesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDictionariesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDictionariesE "}

Dictionaries (Dictionary)

A collection of **Dictionary** objects that includes the active custom spelling dictionaries.

Using the Dictionaries Collection

Use the **CustomDictionaries** property to return the collection of currently active custom dictionaries. The following example displays the names of all the active custom dictionaries.

```
For Each d In CustomDictionaries
    Msgbox d.Name
Next d
```

Use the **Add** method to add a new custom dictionary to the collection of active custom dictionaries. If there isn't a file with the name specified by ***FileName***, Word creates it. The following example adds "MyCustom.dic" to the collection of custom dictionaries.

```
new = CustomDictionaries.Add(FileName:="MyCustom.dic")
```

Use the **ClearAll** method to unload all custom dictionaries. Note, however, that this method doesn't delete the dictionary files. After you use this method, the number of custom dictionaries in the collection is 0 (zero). The following example clears the custom dictionaries and creates a new custom dictionary file. The new dictionary is set as the active custom dictionary, to which Word will automatically add any new words it encounters.

```
With CustomDictionaries
    .ClearAll
    .Add FileName:= "MyCustom.dic"
    .ActiveCustomDictionary = CustomDictionaries(1)
End With
```

Remarks

You set the custom dictionary to which new words are added by using the **ActiveCustomDictionary** property. If you try to set this property to a dictionary that isn't a custom dictionary, an error occurs.

The **Maximum** property returns the maximum number of simultaneous custom spelling dictionaries that the application can support. For Word, this mamimum is 10.

Dictionary Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDictionaryC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDictionaryP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDictionaryM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjDictionaryE "}

Dictionaries (Dictionary)

Represents a dictionary. **Dictionary** objects that represent custom dictionaries are members of the **Dictionaries** collection. Other dictionary objects are returned by properties of the **Languages** collection; these include the **ActiveSpellingDictionary**, **ActiveGrammarDictionary**, **ActiveThesaurusDictionary**, and **ActiveHyphenationDictionary** properties.

Using the Dictionary Object

Use **CustomDictionaries**(*index*), where *index* is an index number or the string name for the dictionary, to return a single **Dictionary** object that represents a custom dictionary. The following example returns the first dictionary in the collection.

```
CustomDictionaries(1)
```

The following example returns the dictionary named "MyDictionary."

```
CustomDictionaries("MyDictionary")
```

Use the **ActiveCustomDictionary** property to set the custom spelling dictionary in the collection to which new words are added. If you try to set this property to a dictionary that's not a custom dictionary, an error occurs.

Use the **Add** method to add a new dictionary to the collection of active custom dictionaries. If there's no file with the name specified by ***FileName***, Word creates it. The following example adds "MyCustom.dic" to the collection of custom dictionaries.

```
new = CustomDictionaries.Add FileName:="MyCustom.dic"
```

Remarks

Use the **Name** and **Path** properties to locate any of the dictionaries. The following example displays a message box that contains the full path for each dictionary.

```
For Each d in CustomDictionaries
    Msgbox d.Path & Application.PathSeparator & d.Name
Next d
```

Use the **LanguageSpecific** property to determine whether the specified custom dictionary can have a specific language assigned to it with the **LanguageID** property. If the dictionary is language specific, it will verify only text that's formatted for the specified language.

For each language for which proofing tools are installed, you can use the **ActiveGrammarDictionary**, **ActiveHyphenationDictionary**, **ActiveSpellingDictionary**, and **ActiveThesaurusDictionary** properties to return the corresponding **Dictionary** objects. The following example returns the full path for the active spelling dictionary used in the U.S. English version of Word.

```
myspell = Languages(wdEnglishUS).ActiveSpellingDictionary
MsgBox mySpell.Path & Application.PathSeparator & mySpell.Name
```

The **ReadOnly** property returns **True** for .lex files (built-in proofing dictionaries) and **False** for .dic files (custom spelling dictionaries).

Application, **Language**

ReadabilityStatistic Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjReadabilityStatisticC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjReadabilityStatisticP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjReadabilityStatisticM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjReadabilityStatisticE "}

ReadabilityStatistics (ReadabilityStatistic)

Represents one of the readability statistics for a document or range. The **ReadabilityStatistic** object is a member of the **ReadabilityStatistics**collection.

Using the ReadabilityStatistic Object

Use **ReadabilityStatistics**(*index*), where *index* is the index number, to return a single **ReadabilityStatistic** object. The statistics are ordered as follows: Words, Characters, Paragraphs, Sentences, Sentences per Paragraph, Words per Sentence, Characters per Word, Passive Sentences, Flesch Reading Ease, and Flesch-Kincaid Grade Level. The following example returns the character count for the active document.

```
Msgbox ActiveDocument.Content.ReadabilityStatistics(2).Value
```

ReadabilityStatistics Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjReadabilityStatisticsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjReadabilityStatisticsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjReadabilityStatisticsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjReadabilityStatisticsE "}

ReadabilityStatistics (ReadabilityStatistic)

A collection of **ReadabilityStatistic** objects for a document or range.

Using the ReadabilityStatistics Collection

Use the **ReadabilityStatistics** property to return the **ReadabilityStatistics** collection. The following example enumerates the readability statistics for the selection and displays each one in a message box.

```
For each rs in Selection.Range.ReadabilityStatistics
    Msgbox rs.Name & " - " & rs.Value
Next rs
```

Use **ReadabilityStatistics**(*index*), where *index* is the index number, to return a single **ReadabilityStatistic** object. The statistics are ordered as follows: Words, Characters, Paragraphs, Sentences, Sentences per Paragraph, Words per Sentence, Characters per Word, Passive Sentences, Flesch Reading Ease, and Flesch-Kincaid Grade Level. The following example returns the word count for the active document.

```
Set myRange = ActiveDocument.Content
wordval = myRange.ReadabilityStatistics(1).Value
Msgbox wordval
```

Document, **Range**

ProofreadingErrors Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjProofreadingErrorsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjProofreadingErrorsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjProofreadingErrorsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjProofreadingErrorsE "}

ProofreadingErrors (Range)

A collection of spelling and grammatical errors for the specified document or range. There is no ProofreadingError object; instead, each item in the **ProofreadingErrors** collection is a **Range** object that represents one spelling or grammatical error.

Using the ProofreadingErrors Collection

Use the **SpellingErrors** or **GrammaticalErrors** property to return the **ProofreadingErrors** collection. The following example counts the spelling and grammatical errors in the selection and displays the results in a message box.

```
Set pr1 = Selection.Range.SpellingErrors
   sc = pr1.Count
Set pr2 = Selection.Range.GrammaticalErrors
   gc = pr2.Count
Msgbox "Spelling errors: " & sc & vbCr & "Grammatical errors: " & gc
```

Use **SpellingErrors**(*index*), where *index* is the index number, to return a single spelling error (represented by a **Range** object). The following example finds the second spelling error in the selection and then selects it.

```
Set myRange = Selection.Range.SpellingErrors(2)
myRange.Select
```

Use **GrammarErrors**(*index*), where *index* is the index number, to return a single grammatical error (represented by a **Range** object). The following example returns the sentence that contains the first grammatical error in the selection.

```
Set myRange = Selection.Range.GrammaticalErrors(1)
Msgbox myRange.Text
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object. If all the words in the document or range are spelled correctly and are grammatically correct, the **Count** property for the **ProofreadingErrors** object returns 0 (zero) and the **SpellingChecked** and **GrammarChecked** properties return **True**.

Revision Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRevisionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRevisionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRevisionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRevisionE "}

Revisions (Revision)

Represents a change marked with a revision mark. The **Revision** object is a member of the **Revisions** collection. The **Revisions** collection includes all the revision marks in a range or document.

Using the Revision Object

Use **Revisions**(*index*), where *index* is the index number, to return a single **Revision** object. The index number represents the position of the revision in the range or document. The following example displays the author name for the first revision in section one in the active document.

```
MsgBox ActiveDocument.Sections(1).Range.Revisions(1).Author
```

The **Add** method isn't available for the **Revisions** collection. **Revision** objects are added when change tracking is enabled. Set the **TrackRevisions** property to **True** to track revisions made to the document text. The following example enables revision tracking and then inserts "Action " before the selection.

```
ActiveDocument.TrackRevisions = True
Selection.InsertBefore "Action "
```

Document, **Range**

Revisions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRevisionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRevisionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRevisionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRevisionsE "}

Revisions (Revision)

A collection of **Revision** objects that represent the changes marked with revision marks in a range or document.

Using the Revisions Collection

Use the **Revisions** property to return the **Revisions** collection. The following example displays the number of revisions in the main text story.

```
MsgBox ActiveDocument.Revisions.Count
```

The following example accepts all the revisions in the selection.

```
For Each myRev In Selection.Range.Revisions
    myRev.Accept
Next myRev
```

The following example accepts all the revisions in the first paragraph in the selection.

```
Set myRange = Selection.Paragraphs(1).Range
myRange.Revisions.AcceptAll
```

The **Add** method isn't available for the **Revisions** collection. **Revision** objects are added when change tracking is enabled. Set the **TrackRevisions** property to **True** to track revisions made to the document text. The following example enables revision tracking in the active document and then inserts "The " before the selection.

```
ActiveDocument.TrackRevisions = True
Selection.InsertBefore "The "
```

Use **Revisions**(*index*), where *index* is the index number, to return a single **Revision** object. The index number represents the position of the revision in the range or document. The following example displays the author name for the first revision in the first section.

```
MsgBox ActiveDocument.Sections(1).Range.Revisions(1).Author
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

MailMergeField Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeFieldC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeFieldP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeFieldM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeFieldE "}

MailMergeFields (MailMergeField)

Represents a single mail merge field in a document. The **MailMergeDataField** object is a member of the **MailMergeDataFields** collection. The **MailMergeDataFields** collection includes all the mail merge related fields in a document.

Using the MailMergeField Object

Use **Fields**(*index*), where *index* is the index number, to return a single **MailMergeField** object. The following example displays the field code of the first mail merge field in the active document.

```
MsgBox ActiveDocument.MailMerge.Fields(1).Code
```

Use the **Add** method to add a merge field to the **MailMergeFields** collection. The following example replaces the selection with a MiddleInitial merge field.

```
ActiveDocument.MailMerge.Fields.Add Range:=Selection.Range, _
    Name:="MiddleInitial"
```

Remarks

The **MailMergeFields** collection has additional methods, such as **AddAsk** and **AddFillIn**, for adding fields related to a mail merge operation.

MailMergeFields Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeFieldsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeFieldsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeFieldsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeFieldsE "}

MailMergeFields (MailMergeField)

A collection of **MailMergeField** objects that represent the mail merge related fields in a document.

Using the MailMergeFields Collection

Use the **Fields** property to return the **MailMergeFields** collection. The following example adds an ASK field after the last mail merge field in the active document.

```
Set myMMFields = ActiveDocument.MailMerge.Fields
myMMFields(myMMFields.Count).Select
Selection.MoveRight Unit:=wdWord, Count:=1, Extend:=wdMove
ActiveDocument.MailMerge.Fields.AddAsk Range:=Selection.Range, _
    Name:="Name", Prompt:="Type your name", AskOnce:=True
```

Use the **Add** method to add a merge field to the **MailMergeFields** collection. The following example replaces the selection with a **MiddleInitial** merge field.

```
ActiveDocument.MailMerge.Fields.Add Range:=Selection.Range, _
    Name:="MiddleInitial"
```

Use **Fields**(*index*), where *index* is the index number, to return a single **MailMergeField** object. The following example displays the field code of the first mail merge field in the active document.

```
MsgBox ActiveDocument.MailMerge.Fields(1).Code
```

Remarks

The **MailMergeFields** collection has additional methods, such as **AddAsk** and **AddFillIn**, for adding fields related to a mail merge operation.

Options Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjOptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjOptionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjOptionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjOptionsE "}

Options

Represents application and document options in Word. Many of the properties for the **Options** object correspond to items in the **Options** dialog box (**Tools** menu).

Using the Options Object

Use the **Options** property to return the **Options** object. The following example sets three application options for Word.

```
With Options
    .AllowDragAndDrop = True
    .ConfirmConversions = False
    .MeasurementUnit = wdPoints
End With
```

PageNumber Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjPageNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjPageNumberP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjPageNumberM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjPageNumberE "}

PageNumbers (PageNumber)

Represents a page number in a header or footer. The **PageNumber** object is a member of the **PageNumbers** collection. The **PageNumbers** collection includes all the page numbers in a single header or footer.

Using the PageNumber Object

Use **PageNumbers**(*index*), where *index* is the index number, to return a single **PageNumber** object. In most cases, a header or footer will contain only one page number, which is index number 1. The following example centers the first page number in the primary header in section one in the active document.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).PageNumbers(1) _
.Alignment = wdAlignPageNumberCenter
```

Use the **Add** method to add a page number (a PAGE field) to a header or footer. The following example adds a page number to the primary footer in the first section. The page number doesn't appear on the first page.

```
With ActiveDocument.Sections(1)
    .Footers(wdHeaderFooterPrimary).PageNumbers.Add _
        PageNumberAlignment:=wdAlignPageNumberLeft, FirstPage:=False
End With
```

PageNumbers Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjPageNumbersC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjPageNumbersP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjPageNumbersM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjPageNumbersE "}

PageNumbers (PageNumber)

A collection of **PageNumber** objects that represent the page numbers in a single header or footer.

Using the PageNumbers Collection

Use the **PageNumbers** property to return the **PageNumbers** collection. The following example starts page numbering at 3 for the first section in the active document.

```
ActiveDocument.Sections(1).Footers(wdHeaderFooterPrimary).PageNumbers _
    .StartingNumber = 3
```

Use the **Add** method to add page numbers to a header or footer. The following example adds a page number to the primary footer in the first section.

```
With ActiveDocument.Sections(1)
    .Footers(wdHeaderFooterPrimary).PageNumbers.Add _
        PageNumberAlignment:=wdAlignPageNumberLeft, FirstPage:=False
End With
```

To add or change page numbers in a document with multiple sections, modify the page numbers in each section or set the **LinkToPrevious** property to **True**.

Use **PageNumbers**(*index*), where *index* is the index number, to return a single **PageNumber** object. In most cases, a header or footer contains only one page number, which is index number 1. The following example centers the first page number in the primary header in the first section.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).PageNumbers(1) _
.Alignment = wdAlignPageNumberCenter
```

Pane Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjPaneC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjPaneP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjPaneM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjPaneE "}

Windows (Window)

└

Panes (Pane)

└

└

└

Represents a window pane. The **Pane** object is a member of the **Panes** collection. The **Panes** collection includes all the window panes for a single window.

Using the Pane Object

Use **Panes**(*index*), where *index* is the index number, to return a single **Pane** object. The following example closes the active pane.

```
If ActiveWindow.Panes.Count >= 2 Then ActiveWindow.ActivePane.Close
```

Use the **Add** method or the **Split** property to add a window pane. The following example splits the active window at 20 percent of the current window size.

```
ActiveWindow.Panes.Add SplitVertical:=20
```

The following example splits the active window in half.

```
ActiveWindow.Split = True
```

You can use the **SplitSpecial** property to show comments, footnotes, or endnotes in a separate pane.

Remarks

A window has more than one pane if the window is split or the view is not page layout view and information such as footnotes or comments are displayed. The following example displays the comments pane in normal view and then prompts to close the pane.

```
ActiveWindow.View.Type = wdNormalView
If ActiveDocument.Comments.Count >= 1 Then
    ActiveWindow.View.SplitSpecial = wdPaneComments
    response = MsgBox("Do you want to close the comments pane?", vbYesNo)
    If response = vbYes Then ActiveWindow.ActivePane.Close
End If
```

Selection, **View**, **Zooms**

Panes Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjPanesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjPanesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjPanesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjPanesE "}

Windows (Window)

└

Panes (Pane)

└

└

└

A collection of **Pane** objects that represent the window panes for a single window.

Using the Panes Collection

Use the **Panes** property to return the **Panes** collection. The following example splits the active window and hides the ruler for each pane.

```
ActiveWindow.Split = True
For Each aPane In ActiveWindow.Panes
    aPane.DisplayRulers = False
Next aPane
```

Use the **Add** method or the **Split** property to add a window pane. The following example splits the active window at 20 percent of the current window size.

```
ActiveWindow.Panes.Add SplitVertical:=20
```

The following example splits the active window in half.

```
ActiveWindow.Split = True
```

You can use the **SplitSpecial** property to show comments, footnotes, or endnotes in a separate pane.

Remarks

A window has more than one pane if it's split, or if the active view isn't page layout view and information such as footnotes or comments is displayed. The following example displays the footnote pane in normal view and then prompts the user to close the pane.

```
ActiveWindow.View.Type = wdNormalView
If ActiveDocument.Footnotes.Count >= 1 Then
    ActiveWindow.View.SplitSpecial = wdPaneFootnotes
    response = MsgBox("Do you want to close the footnotes pane?", vbYesNo)
    If response = vbYes Then ActiveWindow.ActivePane.Close
End If
```

Paragraph Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjParagraphC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjParagraphP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjParagraphM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjParagraphE "}

Represents a single paragraph in a selection, range, or document. The **Paragraph** object is a member of the **Paragraphs** collection. The **Paragraphs** collection includes all the paragraphs in a selection, range, or document.

Using the Paragraph Object

Use **Paragraphs**(*index*), where *index* is the index number, to return a single **Paragraph** object. The following example right aligns the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Alignment = wdAlignParagraphRight
```

Use the **Add**, **InsertParagraph**, **InsertParagraphAfter**, or **InsertParagraphBefore** method to add a new, blank paragraph to a document. The following example adds a paragraph mark before the first paragraph in the selection.

```
Selection.Paragraphs.Add Range:=Selection.Paragraphs(1).Range
```

The following example also adds a paragraph mark before the first paragraph in the selection.

```
Selection.Paragraphs(1).Range.InsertParagraphBefore
```

Document, **ListParagraphs**, **Range**, **Selection**

Paragraphs Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjParagraphsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjParagraphsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjParagraphsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjParagraphsE "}

A collection of **Paragraph** objects in a selection, range, or document.

Using the Paragraphs Collection

Use the **Paragraphs** property to return the **Paragraphs** collection. The following example formats the selected paragraphs to be double-spaced and right-aligned.

```
With Selection.Paragraphs
    .Alignment = wdAlignParagraphRight
    .LineSpacingRule = wdLineSpaceDouble
End With
```

Use the **Add**, **InsertParagraph**, **InsertParagraphAfter**, or **InsertParagraphBefore** method to add a new paragraph to a document. The following example adds a new paragraph before the first paragraph in the selection.

```
Selection.Paragraphs.Add Range:=Selection.Paragraphs(1).Range
```

The following example also adds a paragraph before the first paragraph in the selection.

```
Selection.Paragraphs(1).Range.InsertParagraphBefore
```

Use **Paragraphs**(*index*), where *index* is the index number, to return a single **Paragraph** object. The following example right aligns the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).Alignment = wdAlignParagraphRight
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

Borders, **ParagraphFormat**, **Shading**, **TabStops**

Section Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSectionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSectionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSectionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSectionE "}

Represents a single section in a selection, range, or document. The **Section** object is a member of the **Sections** collection. The **Sections** collection includes all the sections in a selection, range, or document.

Using the Section Object

Use **Sections**(*index*), where *index* is the index number, to return a single **Section** object. The following example changes the left and right page margins for the first section in the active document.

```
With ActiveDocument.Sections(1).PageSetup
    .LeftMargin = InchesToPoints(0.5)
    .RightMargin = InchesToPoints(0.5)
End With
```

Use the **Add** method or the **InsertBreak** method to add a new section to a document. The following example adds a new section at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.Sections.Add Range:=myRange
myRange.InsertParagraphAfter
```

The following example adds a section break above the first paragraph in the selection.

```
Selection.Paragraphs(1).Range.InsertBreak Type:=wdSectionBreakContinuous
```

Note The **Headers** and **Footers** properties of the specified **Section** object return a **HeadersFooters** object.

Borders, **HeadersFooters**, **PageSetup**, **Range**

Sections Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSectionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSectionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSectionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSectionsE "}

A collection of **Section** objects in a selection, range, or document.

Using the Sections Collection

Use the **Sections** property to return the **Sections** collection. The following example inserts text at the end of the last section in the active document.

```
With ActiveDocument.Sections.Last.Range
    .Collapse Direction:=wdCollapseEnd
    .InsertAfter "end of document"
End With
```

Use the **Add** method or the **InsertBreak** method to add a new section to a document. The following example adds a new section at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.Sections.Add Range:=myRange
myRange.InsertParagraphAfter
```

The following example displays the number of sections in the active document, adds a section break above the first paragraph in the selection, and then displays the number of sections again.

```
MsgBox ActiveDocument.Sections.Count & " sections"
Selection.Paragraphs(1).Range.InsertBreak Type:=wdSectionBreakContinuous
MsgBox ActiveDocument.Sections.Count & " sections"
```

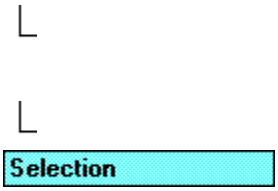
Use **Sections**(*index*), where *index* is the index number, to return a single **Section** object. The following example changes the left and right page margins for the first section in the active document.

```
With ActiveDocument.Sections(1).PageSetup
    .LeftMargin = InchesToPoints(0.5)
    .RightMargin = InchesToPoints(0.5)
End With
```

PageSetup

Selection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSelectionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSelectionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSelectionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSelectionE "}



Represents the selection in a window pane. The selection can either encompass an area in the document or be collapsed to an insertion point.

Note There can only be one **Selection** object per document window pane and only one **Selection** object can be active.

Using the Selection Object

Use the **Selection** property to return the **Selection** object. The following example collapses the selection (if necessary) and moves the insertion point to the end of the current line.

```
Selection.EndKey Unit:=wdLine, Extend:=wdMove
```

The following example updates the results of the fields in the selection.

```
If Selection.Fields.Count >= 1 Then Selection.Fields.Update
```

Use the **Type** property to return the selection type (for example, a block selection or an insertion point). The following example selects the current paragraph if the selection is an insertion point.

```
If Selection.Type = wdSelectionIP Then
    Selection.Paragraphs(1).Range.Select
End If
```

Use the **Information** property to return information about the selection. If the selection is in a table, the following example displays the number or rows and columns in the table.

```
If Selection.Information(wdWithInTable) = True Then
    MsgBox "Columns = " & Selection.Information(wdMaximumNumberOfColumns) _
        & vbCr & "Rows = " & Selection.Information(wdMaximumNumberOfRows)
End If
```

Use the **Select** method to select an item in a document. The following example selects the first bookmark in the active document and formats it to appear in red.

```
If ActiveDocument.Bookmarks.Count >= 1 Then
    ActiveDocument.Bookmarks(1).Select
    Selection.Font.ColorIndex = wdRed
End If
```

The **Selection** object also includes various methods you can use to expand or move an existing selection. For example, the **MoveDown** method has an ***Extend*** argument that you can set to **wdExtend**. The following example selects the next three paragraphs in the active window.

```
With Selection
```

```
    .StartOf Unit:=wdParagraph, Extend:=wdMove
    .MoveDown Unit:=wdParagraph, Count:=3, Extend:=wdExtend
End With
```

Remarks

Use the **Range** property to return a **Range** object from the **Selection** object. The following example defines the variable `myRange` as the selected range.

```
Set myRange = Selection.Range
```

When you record macros, the macro recorder will often record changes to the **Selection** object. The following recorded macro applies bold formatting to the first two words in the document, and then inserts a new paragraph.

```
Selection.HomeKey Unit:=wdStory
Selection.MoveRight Unit:=wdWord, Count:=2, Extend:=wdExtend
Selection.Font.Bold = wdToggle
Selection.MoveRight Unit:=wdCharacter, Count:=1
Selection.TypeParagraph
```

The following example accomplishes the same task as the preceding example, but without using the **Selection** object.

```
Set myRange = ActiveDocument.Range(Start:=0, _
    End:=ActiveDocument.Words(2).End)
myRange.Bold = True
myRange.InsertParagraphAfter
```

There can be only one **Selection** object per window pane; however, you can have multiple **Range** objects defined in a single document. A **Range** object represents an document area that may or may not be selected. Working with **Range** objects, you can manipulate a document with minimal screen updates. For more information, see Working with Range objects.

Application, **Pane**, **Window**

Bookmarks, **Borders**, **Cells**, **Characters**, **Columns**, **Comments**, **Document**, **Endnotes**, **Fields**, **Find**, **Font**, **Footnotes**, **FormFields**, **Frames**, **HeaderFooter**, **Hyperlinks**, **InlineShapes**, **PageSetup**, **ParagraphFormat**, **Paragraphs**, **Revision**, **Range**, **Rows**, **Sections**, **Sentences**, **Shading**, **ShapeRange**, **Tables**, **Words**

PageSetup Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjPageSetupC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjPageSetupP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjPageSetupM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjPageSetupE "}

PageSetup

Represents the page setup description. The **PageSetup** object contains all the page setup attributes of a document (left margin, bottom margin, paper size, and so on) as properties.

Using the PageSetup Object

Use the **PageSetup** property to return the **PageSetup** object. The following example sets the first section in the active document to landscape orientation and then prints the document.

```
ActiveDocument.Sections(1).PageSetup.Orientation = wdOrientLandscape
ActiveDocument.PrintOut
```

The following example sets all the margins for the document named "Sales.doc."

```
With Documents("Sales.doc").PageSetup
    .LeftMargin = InchesToPoints(0.75)
    .RightMargin = InchesToPoints(0.75)
    .TopMargin = InchesToPoints(1.5)
    .BottomMargin = InchesToPoints(1)
End With
```

Document, **Range**, **Section**, **Sections**, **Selection**

LineNumbering, **TextColumns**

RoutingSlip Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRoutingSlipC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRoutingSlipP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRoutingSlipM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRoutingSlipE "}

RoutingSlip

Represents the routing slip associated with a document. You use a routing slip to send a document through an electronic mail system.

Using the RoutingSlip Object

Use the **RoutingSlip** property to return the **RoutingSlip** object. The following example routes the active document to the specified recipients, one after another.

```
ActiveDocument.HasRoutingSlip = True
With ActiveDocument.RoutingSlip
    .Subject = "Project Documentation"
    .AddRecipient "Don Funk"
    .AddRecipient "Dave Edson"
    .Delivery = wdOneAfterAnother
End With
ActiveDocument.Route
```

Remarks

The **RoutingSlip** object cannot be used (doesn't exist) unless the **HasRoutingSlip** property for the document is set to **True**.

TabStop Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTabStopC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTabStopP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTabStopM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTabStopE "}

TabStops (TabStop)

Represents a single tab stop. The **TabStop** object is a member of the **TabStops** collection. The **TabStops** collection represents all the custom and default tab stops in a paragraph or group of paragraphs.

Using the TabStop Object

Use **TabStops**(*index*), where *index* is the location of the tab stop (in points) or the index number, to return a single **TabStop** object. Tab stops are indexed numerically from left to right along the ruler. The following example removes the first custom tab stop from the selected paragraphs.

```
Selection.Paragraphs.TabStops(1).Clear
```

The following example adds a right-aligned tab stop positioned at 2 inches to the selected paragraphs.

```
Selection.Paragraphs.TabStops(InchesToPoints(2)).Alignment =
wdAlignTabRight
```

Use the **Add** method to add a tab stop. The following example adds two tab stops to the selected paragraphs. The first tab stop is a left-aligned tab with a dotted tab leader positioned at 1 inch (72 points). The second tab stop is centered and is positioned at 2 inches.

```
With Selection.Paragraphs.TabStops
    .Add Position:=InchesToPoints(1), Leader:=wdTabLeaderDots,
Alignment:=wdAlignTabLeft
    .Add Position:=InchesToPoints(2), Alignment:=wdAlignTabCenter
End With
```

You can also add a tab stop by specifying a location with the **TabStops** property. The following example adds a right-aligned tab stop positioned at 2 inches to the selected paragraphs.

```
Selection.Paragraphs.TabStops(InchesToPoints(2)).Alignment =
wdAlignTabRight
```

Note Set the **DefaultTabStop** property to adjust the spacing of default tab stops.

Paragraph, **ParagraphFormat**, **Paragraphs**

TabStops Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTabStopsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTabStopsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTabStopsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTabStopsE "}

TabStops (TabStop)

A collection of **TabStop** objects that represent the custom and default tabs for a paragraph or group of paragraphs.

Using the TabStops Collection

Use the **TabStops** property to return the **TabStops** collection. The following example clears all the custom tab stops from the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).TabStops.ClearAll
```

The following example adds a tab stop positioned at 2.5 inches to the selected paragraphs and then displays the position of each item in the **TabStops** collection.

```
Selection.Paragraphs.TabStops.Add Position:=InchesToPoints(2.5)
For Each aTab In Selection.Paragraphs.TabStops
    MsgBox "Position = " & PointsToInches(aTab.Position) & " inches"
Next aTab
```

Use the **Add** method to add a tab stop. The following example adds two tab stops to the selected paragraphs. The first tab stop is a left-aligned tab with a dotted tab leader positioned at 1 inch (72 points). The second tab stop is centered and is positioned at 2 inches.

```
With Selection.Paragraphs.TabStops
    .Add Position:=InchesToPoints(1), Leader:=wdTabLeaderDots,
Alignment:=wdAlignTabLeft
    .Add Position:=InchesToPoints(2), Alignment:=wdAlignTabCenter
End With
```

You can also add a tab stop by specifying a location with the **TabStops** property. The following example adds a right-aligned tab stop positioned at 2 inches to the selected paragraphs.

```
Selection.Paragraphs.TabStops(InchesToPoints(2)).Alignment =
wdAlignTabRight
```

Use **TabStops**(*index*), where *index* is the location of the tab stop (in points) or the index number, to return a single **TabStop** object. Tab stops are indexed numerically from left to right along the ruler. The following example removes the first custom tab stop from the first paragraph in the active document.

```
ActiveDocument.Paragraphs(1).TabStops(1).Clear
```

The following example adds a right-aligned tab stop positioned at 2 inches to the selected paragraphs.

```
Selection.Paragraphs.TabStops(InchesToPoints(2)).Alignment =
wdAlignTabRight
```

Remarks

When working with the **Paragraphs** collection (or a range with several paragraphs), you must modify each paragraph in the collection individually if the tab stops aren't identical in all the paragraphs. The

following example removes the tab positioned at 1 inch from every paragraph in the active document.

```
For Each para In ActiveDocument.Content.Paragraphs
    para.TabStops(InchesToPoints(1)).Clear
Next para
```

Task Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTaskC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTaskP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTaskM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTaskE "}

Tasks (Task)

Represents a single task running on the system. The **Task** object is a member of the **Tasks** collection. The **Tasks** collection includes all the applications that are currently running on the system.

Using the Task Object

Use **Tasks**(*index*), where *index* is the application name or the index number, to return a single **Task** object. The following example switches to and resizes the application window for the first visible task in the **Tasks** collection.

```
With Tasks(1)
    If .Visible = True Then
        .Activate
        .Width = 400
        .Height = 200
    End If
End With
```

The following example restores the Calculator application window if Calculator is in the **Tasks** collection.

```
If Tasks.Exists("Calculator") = True Then
    Tasks("Calculator").WindowState = wdWindowStateNormal
End If
```

Use the **Shell** function to run an executable program and add the program to the **Tasks** collection.

Tasks Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTasksC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTasksP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTasksM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTasksE "}

Tasks (Task)

A collection of **Task** objects that represent all the tasks currently running on the system.

Using the Tasks Collection

Use the **Tasks** property to return the **Tasks** collection. The following example determines whether Microsoft Excel is running. If Microsoft Excel is already running, this example switches to it and maximizes it; otherwise, the example starts it.

```
If Tasks.Exists("Microsoft Excel") = True Then
    Tasks("Microsoft Excel").Activate
    Tasks("Microsoft Excel").WindowState = wdWindowStateMaximize
Else
    Shell "C:\MSOffice\excel\Excel.exe"
End If
```

Use the **Shell** function to run an executable program and add the program to the **Tasks** collection.

Use **Tasks**(*index*), where *index* is the application name or the index number, to return a single **Task** object. The following example opens and resizes the application window for the first visible task in the **Tasks** collection.

```
With Tasks(1)
    If .Visible = True Then
        .Activate
        .Width = 400
        .Height = 200
    End If
End With
```

The following example restores the Calculator application window if the application is in the **Tasks** collection.

```
If Tasks.Exists("Calculator") = True Then
    Tasks("Calculator").WindowState = wdWindowStateNormal
End If
```

Template Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTemplateP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTemplateM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTemplateE "}

Represents a document template. The **Template** object is a member of the **Templates** collection. The **Templates** collection includes all the available **Template** objects.

Using the Template Object

Use **Templates**(*index*), where *index* is the template name or the index number, to return a single **Template** object. The following example saves the Memo2.dot template if it's in the **Templates** collection.

```
For Each aTemp In Templates
    If LCase(aTemp.Name) = "memo2.dot" Then aTemp.Save
Next aTemp
```

The index number represents the position of the template in the **Templates** collection. The following example opens the first template in the **Templates** collection.

```
Templates(1).OpenAsDocument
```

The **Add** method isn't available for the **Templates** collection. Instead, you can add a template to the **Templates** collection by doing any of the following:

- Using the **Open** method with the **Documents** collection to open a document based on a template or a template
- Using the **Add** method with the **Documents** collection to open a new document based on a template
- Using the **Add** method with the **Addins** collection to load a global template
- Using the **AttachedTemplate** property with the **Document** object to attach a template to a document

Remarks

Use the **NormalTemplate** property to return a template object that refers to the Normal template. Use the **AttachedTemplate** property to return the template attached to the specified document.

Use the **DefaultFilePath** property to return or set the location of user or workgroup templates (that is, the folder where you want to store these templates). The following example displays the user template folder from the **File Locations** tab in the **Options** dialog box (**Tools** menu).

```
MsgBox Options.DefaultFilePath(wdUserTemplatesPath)
```

Application, **Document**

AutoTextEntries, **Document**, **DocumentProperties**, **ListTemplates**, **VBProject**

Templates Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTemplatesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTemplatesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTemplatesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTemplatesE "}

A collection of **Template** objects that represent all the templates that are currently available. This collection includes open templates, templates attached to open documents, and global templates loaded in the **Templates and Add-ins** dialog box (**Tools** menu).

Using the Templates Collection

Use the **Templates** property to return the **Templates** collection. The following example displays the path and file name of each template in the **Templates** collection.

```
For Each aTemp In Templates
    MsgBox aTemp.FullName
Next aTemp
```

The **Add** method isn't available for the **Templates** collection. Instead, you can add a template to the **Templates** collection by doing any of the following:

- Using the **Open** method with the **Documents** collection to open a document based on a template or a template
- Using the **Add** method with the **Documents** collection to open a new document based on a template
- Using the **Add** method with the **Addins** collection to load a global template
- Using the **AttachedTemplate** property with the **Document** object to attach a template to a document

Use **Templates**(*index*), where *index* is the template name or the index number, to return a single **Template** object. The following example saves the Dot1.dot template.

```
Templates("C:\MSOffice\WinWord\Templates\Dot1.dot").Save
```

The index number represents the position of the template in the **Templates** collection. The following example displays the file name of the first template in the **Templates** collection.

```
MsgBox Templates(1).FullName
```

Remarks

Use the **NormalTemplate** property to return a template object that refers to the Normal template. Use the **AttachedTemplate** property to return the template attached to the specified document.

Use the **DefaultFilePath** property to determine the location of user or workgroup templates (that is, the folder where you want to store these templates). The following example displays the user template folder from the **File Locations** tab in the **Options** dialog box (**Tools** menu).

```
MsgBox Options.DefaultFilePath(wdUserTemplatePath)
```

TextColumn Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextColumnC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextColumnP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextColumnM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTextColumnE "}

PageSetup

└

TextColumns (TextColumn)

Represents a single text column. The **TextColumn** object is a member of the **TextColumns** collection. The **TextColumns** collection includes all the columns in a document or section of a document.

Using the TextColumn Object

Use **TextColumns**(*index*), where *index* is the index number, to return a single **TextColumn** object. The index number represents the position of the column in the **TextColumns** collection (counting from left to right).

The following example sets the space after the first text column in the active document to 0.5 inch.

```
ActiveDocument.PageSetup.TextColumns(1).SpaceAfter = InchesToPoints(0.5)
```

Use the **Add** method to add a column to the collection of columns. By default, there's one text column in the **TextColumns** collection. The following example adds a 2.5-inch-widecolumn to the active document.

```
ActiveDocument.PageSetup.TextColumns.Add Width:=InchesToPoints(2.5), _
    Spacing:=InchesToPoints(0.5), EvenlySpaced:=False
```

Remarks

Use the **SetCount** method to arrange text into columns. The following example arranges the text in the active document into three columns.

```
ActiveDocument.PageSetup.TextColumns.SetCount NumColumns:=3
```

TextColumns Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextColumnsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextColumnsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTextColumnsE "}

PageSetup

└

TextColumns (TextColumn)

A collection of **TextColumn** objects that represent all the columns of text in a document or a section of a document.

Using the TextColumns Collection

Use the **TextColumns** property to return the **TextColumns** collection. The following example formats the columns in the first section in the active document to be evenly spaced, with a line between the columns.

```
With ActiveDocument.Sections(1).PageSetup.TextColumns
    .EvenlySpaced = True
    .LineBetween = True
End With
```

Use the **Add** method to add a column to the collection of columns. By default, there's one text column in the **TextColumns** collection. The following example adds a 2.5-inch-wide column to the active document.

```
ActiveDocument.PageSetup.TextColumns.Add Width:=InchesToPoints(2.5), _
    Spacing:=InchesToPoints(0.5), EvenlySpaced:=False
```

Remarks

Use the **SetCount** method to arrange text into columns. The following example arranges the text in the active document into three columns.

```
ActiveDocument.PageSetup.TextColumns.SetCount NumColumns:=3
```

TextInput Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextInputC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextInputP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextInputM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTextInputE "}

TextInput

Represents a single text form field.

Using the TextInput Object

Use **FormFields**(*index*), where *index* is either the bookmark name associated with the text form field or the index number, to return a **FormField** object. Use the **TextInput** property with the **FormField** object to return a **TextInput** object. The following example deletes the contents of the text form field named "Text1" in the active document.

```
ActiveDocument.FormFields("Text1").TextInput.Clear
```

The index number represents the position of the form field in the **FormFields** collection. The following example checks the type of the first form field in the active document. If the form field is a text form field, the example sets "Mission Critical" as the value of the field.

```
If ActiveDocument.FormFields(1).Type = wdFieldFormTextInput Then
    ActiveDocument.FormFields(1).Result = "Mission Critical"
End If
```

The following example determines whether the `ffield` variable represents a valid text form field in the active document before it sets the default text.

```
Set ffield = ActiveDocument.FormFields(1).TextInput
If ffield.Valid = True Then
    ffield.Default = "Type your name here"
Else
    MsgBox "First field is not a text box"
End If
```

Use the **Add** method with the **FormFields** object to add a text form field. The following example adds a text form field at the beginning of the active document and then sets the name of the form field to "FirstName."

```
Set ffield = ActiveDocument.FormFields.Add( _
    Range:=ActiveDocument.Range(Start:=0, End:=0),
Type:=wdFieldFormTextInput)
ffield.Name = "FirstName"
```

Variable Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjVariableC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjVariableP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjVariableM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjVariableE "}

Variables (Variable)

Represents a variable stored as part of a document. Document variables are used to preserve macro settings in between macro sessions. The **Variable** object is a member of the **Variables** collection. The **Variables** collection includes all the document variables in a document or template.

Using the Variable Object

Use **Variables**(*index*), where *index* is the document variable name or the index number, to return a single **Variable** object. The following example displays the value of the Temp document variable in the active document.

```
MsgBox ActiveDocument.Variables("Temp").Value
```

The index number represents the position of the document variable in the **Variables** collection. The last variable added to the **Variables** collection is index number 1; the second-to-last variable added to the collection is index number 2, and so on. The following example displays the name of the first document variable in the active document.

```
MsgBox ActiveDocument.Variables(1).Name
```

Use the **Add** method to add a variable to a document. The following example adds a document variable named "Temp" with a value of 12 to the active document.

```
ActiveDocument.Variables.Add Name:="Temp", Value:="12"
```

If you try to add a document variable with a name that already exists in the **Variables** collection, an error occurs. To avoid this error, you can enumerate the collection before adding any new variables. If the Blue document variable already exists in the active document, the following example sets its value to 6. If this variable doesn't already exist, this example adds it to the document and sets it to 6.

```
For Each aVar In ActiveDocument.Variables
    If aVar.Name = "Blue" Then num = aVar.Index
Next aVar
If num = 0 Then
    ActiveDocument.Variables.Add Name:="Blue", Value:=6
Else
    ActiveDocument.Variables(num).Value = 6
End If
```

Remarks

Document variables are invisible to the user unless a DOCVARIABLE field is inserted with the appropriate variable name. The following example adds a document variable named "Temp" to the active document and then inserts a DOCVARIABLE field to display the value in the variable.

```
With ActiveDocument
    .Variables.Add Name:="Temp", Value:="12"
    .Fields.Add Range:=Selection.Range, Type:=wdFieldDocVariable,
Text:="Temp"
End With
ActiveWindow.View.ShowFieldCodes = False
```

To add a document variable to a template, open the template as a document by using the **OpenAsDocument** method. The following example stores the user name (from the **Options** dialog box) in the template attached to the active document.

```
ScreenUpdating = False
With ActiveDocument.AttachedTemplate.OpenAsDocument
    .Variables.Add Name:="UserName", Value:=Application.UserName
    .Close SaveChanges:=wdSaveChanges
End With
```

Variables Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjVariablesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjVariablesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjVariablesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjVariablesE "}

Variables (Variable)

A collection of **Variable** objects that represent the variables added to a document or template. Document variables are used to preserve macro settings in between macro sessions.

Using the Variables Collection

Use the **Variables** property to return the **Variables** collection. The following example displays the number of variables in the document named "Sales.doc."

```
MsgBox Documents("Sales.doc").Variables.Count & " variables"
```

Use the **Add** method to add a variable to a document. The following example adds a document variable named "Temp" with a value of 12 to the active document.

```
ActiveDocument.Variables.Add Name:="Temp", Value:="12"
```

If you try to add a document variable with a name that already exists in the **Variables** collection, an error occurs. To avoid this error, you can enumerate the collection before adding any new variables. If the Blue document variable already exists in the active document, the following example sets its value to 6. If this variable doesn't already exist, this example adds it to the document and sets it to 6.

```
For Each aVar In ActiveDocument.Variables
    If aVar.Name = "Blue" Then num = aVar.Index
Next aVar
If num = 0 Then
    ActiveDocument.Variables.Add Name:="Blue", Value:=6
Else
    ActiveDocument.Variables(num).Value = 6
End If
```

Use **Variables**(*index*), where *index* is the document variable name or the index number, to return a single **Variable** object. The following example displays the value of the Temp document variable in the active document.

```
MsgBox ActiveDocument.Variables("Temp").Value
```

The index number represents the position of the document variable in the **Variables** collection. The last variable added to the **Variables** collection is index number 1; the second-to-last variable added to the collection is index number 2, and so on. The following example displays the name of the first document variable in the active document.

```
MsgBox ActiveDocument.Variables(1).Name
```

To add a variable to a template, open the template as a document by using the **OpenAsDocument** method. The following example stores the user name (from the **Options** dialog box) in the template attached to the active document.

```
ScreenUpdating = False
With ActiveDocument.AttachedTemplate.OpenAsDocument
    .Variables.Add Name:="UserName", Value:= Application.UserName
    .Close SaveChanges:=wdSaveChanges
End With
```

Window Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjWindowC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjWindowP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjWindowM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjWindowE "}

Windows (Window)

Represents a window. Many document characteristics, such as scroll bars and rulers, are actually properties of the window. The **Window** object is a member of the **Windows** collection. The **Windows** collection for the **Application** object contains all the windows in the application, whereas the **Windows** collection for the **Document** object contains only the windows that display the specified document.

Using the Window Object

Use **Windows**(*index*), where *index* is the window name or the index number, to return a single **Window** object. The following example maximizes the Document1 window.

```
Windows("Document1").WindowState = wdWindowStateMaximize
```

The index number is the number to the left of the window name on the **Window** menu. The following example displays the caption of the first window in the **Windows** collection.

```
MsgBox Windows(1).Caption
```

Use the **Add** method or the **NewWindow** method to add a new window to the **Windows** collection. Each of the following statements creates a new window for the document in the active window.

```
ActiveWindow.NewWindow
NewWindow
Windows.Add
```

Remarks

A colon (:) and a number appear in the window caption when more than one window is open for a document.

When you switch the view to print preview, a new window is created. This window is removed from the **Windows** collection when you close print preview.

Windows Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjWindowsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjWindowsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjWindowsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjWindowsE "}

A collection of **Window** objects that represent all the available windows. The **Windows** collection for the **Application** object contains all the windows in the application, whereas the **Windows** collection for the **Document** object contains only the windows that display the specified document.

Using the Windows Collection

Use the **Windows** property to return the **Windows** collection. The following example tiles all the windows so that they don't overlap one another.

```
Windows.Arrange ArrangeStyle:=wdTiled
```

Use the **Add** method or the **NewWindow** method to add a new window to the **Windows** collection. Each of the following statements creates a new window for the document in the active window.

```
ActiveWindow.NewWindow
NewWindow
Windows.Add
```

Use **Windows**(*index*), where *index* is the window name or the index number, to return a single **Window** object. The following example maximizes the Document1 window.

```
Windows("Document1").WindowState = wdWindowStateMaximize
```

The index number is the number to the left of the window name on the **Window** menu. The following example displays the caption of the first window in the **Windows** collection.

```
MsgBox Windows(1).Caption
```

Remarks

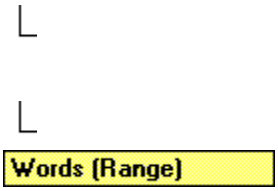
A colon (:) and a number appear in the window caption when more than one window is open for a document.

When you switch the view to print preview, a new window is created. This window is removed from the **Windows** collection when you close print preview.

Document, **Panes**, **Selection**, **View**

Words Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjWordsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjWordsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjWordsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjWordsE "}



A collection of words in a selection, range, or document. Each item in the **Words** collection is a **Range** object that represents one word. There is no Word object.

Using the Words Collection

Use the **Words** property to return the **Words** object. The following example displays how many words are currently selected.

```
MsgBox Selection.Words.Count & " words are selected"
```

Use **Words**(*index*), where *index* is the index number, to return a **Range** object that represents one word. The index number represents the position of the word in the **Words** collection. The following example formats the first word in the selection as 24-point italic.

```
With Selection.Words(1)
    .Italic = True
    .Font.Size = 24
End With
```

The item in the **Words** collection includes both the word and the spaces after the word. To remove the trailing spaces, use the **RTrim** function – for example, `RTrim(ActiveDocument.Words(1)`. The following example selects the first word (and its trailing spaces) in the active document.

```
ActiveDocument.Words(1).Select
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

The **Add** method isn't available for the **Words** collection. Instead, use the **InsertAfter** method or the **InsertBefore** method to add text to a **Range** object. The following example inserts text after the first word in the active document.

```
ActiveDocument.Words(1).InsertAfter "New text "
```

TwoInitialCapsException Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTwoInitialCapsExceptionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTwoInitialCapsExceptionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTwoInitialCapsExceptionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTwoInitialCapsExceptionE "}

TwoInitialCapsExceptions (TwoInitialCapsException)

Represents a single initial-capital AutoCorrect exception. The **TwoInitialCapsException** object is a member of the **TwoInitialCapsExceptions** collection. The **TwoInitialCapsExceptions** collection includes all the items listed in the **Don't correct** box on the **INitial CAps** tab in the **AutoCorrect Exceptions** dialog box.

Using the TwoInitialCapsException Object

Use **TwoInitialCapsExceptions**(*index*), where *index* is the initial capital exception name or the index number, to return a single **TwoInitialCapsException** object. The following example deletes the initial-capital exception named "KMenu."

```
AutoCorrect.TwoInitialCapsExceptions("KMenu").Delete
```

The index number represents the position of the initial-capital exception in the **TwoInitialCapsExceptions** collection. The last exception added to this collection is index number 1. The following example displays the name of the first item in the **TwoInitialCapsExceptions** collection.

```
MsgBox AutoCorrect.TwoInitialCapsExceptions(1).Name
```

If the **TwoInitialCapsAutoAdd** property is **True**, words are automatically added to the list of initial-capital exceptions. Use the **Add** method to add an item to the **TwoInitialCapsExceptions** collection. The following example adds "Industry" to the list of initial-capital exceptions.

```
AutoCorrect.TwoInitialCapsExceptions.Add Name:="INdustry"
```

TwoInitialCapsExceptions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTwoInitialCapsExceptionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTwoInitialCapsExceptionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTwoInitialCapsExceptionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTwoInitialCapsExceptionsE "}

TwoInitialCapsExceptions (TwoInitialCapsException)

A collection of **TwoInitialCapsException** objects that represent all the items listed in the **Don't correct** box on the **INitial CAps** tab in the **AutoCorrect Exceptions** dialog box.

Using the TwoInitialCapsExceptions Collection

Use the **TwoInitialCapsExceptions** property to return the **TwoInitialCapsExceptions** collection. The following example displays the items in this collection.

```
For Each aCap In AutoCorrect.TwoInitialCapsExceptions
    MsgBox aCap.Name
Next aCap
```

If the **TwoInitialCapsAutoAdd** property is **True**, words are automatically added to the list of initial-capital exceptions. Use the **Add** method to add an item to the **TwoInitialCapsExceptions** collection. The following example adds "Industry" to the list of initial-capital exceptions.

```
AutoCorrect.TwoInitialCapsExceptions.Add Name:="INdustry"
```

Use **TwoInitialCapsExceptions**(*index*), where *index* is the initial cap name or the index number, to return a single **TwoInitialCapsException** object. The following example deletes the initial-capital item named "KMenu."

```
AutoCorrect.TwoInitialCapsExceptions("KMenu").Delete
```

The index number represents the position of the initial-capital exception in the **TwoInitialCapsExceptions** collection. The last exception added to this collection is index number 1. The following example displays the name of the first item in the **TwoInitialCapsExceptions** collection.

```
MsgBox AutoCorrect.TwoInitialCapsExceptions(1).Name
```

Version Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjVersionC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjVersionP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjVersionM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjVersionE "}

Versions (Version)

Represents a single version of a document. The **Version** object is a member of the **Versions** collection. The **Versions** collection includes all the versions of the specified document.

Using the Version Object

Use **Versions**(*index*), where *index* is the index number, to return a single **Version** object. The index number represents the position of the version in the **Versions** collection. The first version added to the **Versions** collection is index number 1. The following example displays the comment, author, and date of the first version of the active document.

```
If ActiveDocument.Versions.Count >= 1 Then
    With ActiveDocument.Versions(1)
        MsgBox "Comment = " & .Comment & vbCr & "Author = " & _
            .SavedBy & vbCr & "Date = " & .Date
    End With
End If
```

Use the **Save** method to add an item to the **Versions** collection. The following example adds a version of the active document with the specified comment.

```
ActiveDocument.Versions.Save Comment:="incorporated Judy's revisions"
```

Versions Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjVersionsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjVersionsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjVersionsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjVersionsE "}

Versions (Version)

A collection of **Version** objects that represent all the versions of a document. Corresponds to the items listed in the **Versions** dialog box (**File** menu).

Using the Versions Collection

Use the **Versions** property to return the **Versions** collection. The following example turns off the option that automatically creates new document versions.

```
ActiveDocument.Versions.AutoVersion = wdAutoVersionOff
```

Use the **Save** method to add an item to the **Versions** collection. The following example adds a version with the specified comment.

```
ActiveDocument.Versions.Save Comment:="incorporated Judy's revisions"
```

Use **Versions**(*index*), where *index* is the index number, to return a single **Version** object. The index number represents the position of the version in the **Versions** collection. The first version added to the **Versions** collection is index number 1. The following example displays the comment, author, and date of the first version of the active document.

```
If ActiveDocument.Versions.Count >= 1 Then
    With ActiveDocument.Versions(1)
        MsgBox "Comment = " & .Comment & vbCr & "Author = " & _
            .SavedBy & vbCr & "Date = " & .Date
    End With
End If
```

View Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjViewC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjViewP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjViewM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjViewE "}

View

Zooms (Zoom)

Contains the view attributes (show all, field shading, table gridlines, and so on) for a window or pane.

Using the View Object

Use the **View** property to return the **View** object. The following example sets view options for the active window.

```
With ActiveWindow.View
    .ShowAll = True
    .TableGridlines = True
    .WrapToWindow = False
End With
```

Remarks

Use the **Type** property to change the view. The following example switches the active window to normal view.

```
ActiveWindow.View.Type = wdNormalView
```

Use the **Percentage** property to change the size of the text on-screen. The following example enlarges the on-screen text to 120 percent.

```
ActiveWindow.View.Zoom.Percentage = 120
```

Use the **SeekView** property to view comments, endnotes, footnotes, or the document header or footer. The following example displays the current footer in the active window in page layout view.

```
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageFooter
End With
```

Pane, **Window**

Zoom Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjZoomC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjZoomP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjZoomM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjZoomE "}

Zooms (Zoom)

Contains magnification options (for example, the zoom percentage) for a window or pane. The **Zoom** object is a member of the **Zooms** collection.

Using the Zoom Object

Use the **Zoom** property of the **View** object to return a single **Zoom** object. The following example sets the zoom percentage for the active window to 110 percent.

```
ActiveWindow.View.Zoom.Percentage = 110
```

Use **Zooms**(*index*), where *index* identifies the view type, to return a single **Zoom** object. The view type specified by *index* can be one of the following **WdViewType** constants: **wdMasterView**, **wdNormalView**, **wdOnlineView**, **wdOutlineView**, **wdPageView**, or **wdPrintPreview**. The following example sets the page layout magnification for active window so that an entire page is visible.

```
ActiveWindow.ActivePane.Zooms(wdPageView).PageFit = wdPageFitFullPage
```

The **Add** method isn't available for the **Zooms** collection. The **Zooms** collection includes a single **Zoom** object for each of the various view types (outline, normal, page layout, and so on).

Pane, **View**

Zooms Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjZoomsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjZoomsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjZoomsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjZoomsE "}

Zooms (Zoom)

A collection of **Zoom** objects that represent the magnification options for each view (outline, normal, page layout, and so on).

Using the Zooms Collection

Use the **Zooms** property to return the **Zooms** collection. The following example sets the zoom percentage for the active window to 100 percent in Normal view.

```
ActiveWindow.ActivePane.Zooms(wdNormalView).Percentage = 100
```

The **Add** method isn't available for the **Zooms** collection. The **Zooms** collection includes a single **Zoom** object for each of the various view types (outline, normal, page layout, and so on). You cannot enumerate the **Zooms** collection by using a **For Each...Next** loop.

Use **Zooms**(*index*), where *index* identifies the view type, to return a single **Zoom** object. The view type specified by *index* can be one of the following **WdViewType** constants: **wdMasterView**, **wdNormalView**, **wdOnlineView**, **wdOutlineView**, **wdPageView**, or **wdPrintPreview**. The following example sets the page layout magnification for the active window so that an entire page is visible.

```
ActiveWindow.ActivePane.Zooms(wdPageView).PageFit = wdPageFitFullPage
```

You can also use the **Zoom** property of the **View** object to return a single **Zoom** object. The following example sets the zoom percentage for the active window to 110 percent.

```
ActiveWindow.View.Zoom.Percentage = 110
```

ListGalleries Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListGalleriesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListGalleriesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListGalleriesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListGalleriesE "}

ListGalleries (ListGallery)

ListTemplates (ListTemplate)

A collection of **ListGallery** objects that represent the three tabs in the **Bullets and Numbering** dialog box.

Using the ListGalleries Collection

Use the **ListGalleries** property to return the **ListGalleries** collection. The following example enumerates the collection of list galleries and sets each of the seven list templates (formats) back to the list template format built into Word.

```
For Each lg In ListGalleries
    For x = 1 To 7
        lg.Reset(x)
    Next x
Next lg
```

Use **ListGalleries**(*index*), where *index* is **wdBulletGallery**, **wdNumberGallery**, or **wdOutlineNumberGallery**, to return a single **ListGallery** object.

The following example returns the third list format (excluding **None**) on the **Bulleted** tab in the **Bullets and Numbering** dialog box and then applies it to the selection.

```
Set temp3 = ListGalleries(wdBulletGallery).ListTemplates(3)
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:= temp3
```

Resetting a List Template in the Gallery

To see whether the specified list template contains the formatting built into Word, use the **Modified** property with the **ListGallery** object. To reset formatting to the original list format, use the **Reset** method for the **ListGallery** object.

ListGallery Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListGalleryC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListGalleryP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListGalleryM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListGalleryE "}

ListGalleries (ListGallery)

ListTemplates (ListTemplate)

Represents a single gallery of list formats. The **ListGallery** object is a member of the **ListGalleries** collection. Each **ListGallery** object represents one of the three tabs in the **Bullets and Numbering** dialog box.

Using the ListGallery Object

Use **ListGalleries**(*index*), where *index* is **wdBulletGallery**, **wdNumberGallery**, or **wdOutlineNumberGallery**, to return a single **ListGallery** object.

The following example returns the third list format (excluding **None**) on the **Bulleted** tab in the **Bullets and Numbering** dialog box and then applies it to the selection.

```
Set temp3 = ListGalleries(wdBulletGallery).ListTemplates(3)
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:= temp3
```

Resetting a List Template in the Gallery

To see whether the specified list template contains the formatting built into Word, use the **Modified** property for the **ListGallery** object. To reset formatting to the original list format, use the **Reset** method for the **ListGallery** object.

ListLevel Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListLevelC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListLevelP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListLevelM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListLevelE "}

ListTemplates (ListTemplate)

ListLevels (ListLevel)

Represents a single list level – either the only level for a bulleted or numbered list, or one of the nine levels of an outline numbered list. The **ListLevel** object is a member of the **ListLevels** collection.

Using the ListLevel Object

Use **ListLevels**(*index*), where *index* is a number from 1 through 9, to return a single **ListLevel** object. The following example sets list level one of list template one in the active document to start at 4.

```
ActiveDocument.ListTemplates(1).ListLevels(1).StartAt = 4
```

Remarks

The **ListLevel** object gives you access to all the formatting properties for the specified list level, such as the **Alignment**, **Font**, **NumberFormat**, **NumberPosition**, **NumberStyle**, and **TrailingCharacter** properties.

To apply a list level, first identify the range or list, and then use the **ApplyListTemplate** method. Each tab at the beginning of the paragraph is translated into a list level. For example, a paragraph that begins with three tabs will become a level three list paragraph after the **ApplyListTemplate** method is used.

ListLevels Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListLevelsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListLevelsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListLevelsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListLevelsE "}

ListTemplates (ListTemplate)

ListLevels (ListLevel)

A collection of **ListLevel** objects that represent all the list levels of a list template – either the only level for a bulleted or numbered list, or one of the nine levels of an outline numbered list.

Using the ListLevels Collection

Use the **ListLevels** property to return the **ListLevels** collection. The following example sets the variable `mytemp` to the first list template in the active document, and then modifies each level to use lowercase letters for its number style.

```
Set mytemp = ActiveDocument.ListTemplates(1)
For Each lev In mytemp.ListLevels
    lev.NumberStyle = wdListNumberStyleLowercaseLetter
Next lev
```

Use **ListLevels**(*index*), where *index* is a number from 1 through 9, to return a single **ListLevel** object. The following example sets list level one of list template one in the active document to start at 4.

```
ActiveDocument.ListTemplates(1).ListLevels(1).StartAt = 4
```

Note You cannot add new levels to a list template.

Remarks

To apply a list level, first identify the range or list, and then use the **ApplyListTemplate** method. Each tab at the beginning of the paragraph is translated into a list level. For example, a paragraph that begins with three tabs will become a level three list paragraph after the **ApplyListTemplate** method is used.

Lists Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListsE "}

Lists (List)

A collection of **List** objects that represent all the lists in the specified document.

Using the Lists Collection

Use the **Lists** property to return the **Lists** collection. The following example displays the number of items in each list in the active document.

```
For Each li In ActiveDocument.Lists
    MsgBox li.CountNumberedItems
Next li
```

Use **Lists**(*index*), where *index* is the index number, to return a single **List** object. The following example applies the first list format (excluding **None**) on the **Numbered** tab in the **Bullets and Numbering** dialog box to the second list in the active document.

```
Set temp1 = ListGalleries(wdNumberGallery).ListTemplates(1)
ActiveDocument.Lists(2).ApplyListTemplate ListTemplate:=temp1
```

Remarks

When you use a **For Each...Next** loop to enumerate the **Lists** collection, the lists in a document are returned in reverse order. The following example counts the items for each list in the active document, from the bottom of the document upward.

```
For Each li In ActiveDocument.Lists
   MsgBox li.CountNumberedItems
Next li
```

To add a new list to a document, use the **ApplyListTemplate** method with the **ListFormat** object for a specified range.

You can manipulate the individual **List** objects within a document, but for more precise control you should work with the **ListFormat** object.

List Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListE "}

Lists (List)

Represents a single list format that's been applied to specified paragraphs in a document. The **List** object is a member of the **Lists** collection.

Using the List Object

Use **Lists**(*index*), where *index* is the index number, to return a single **List** object. The following example returns the number of items in list one in the active document.

```
mycount = ActiveDocument.Lists(1).CountNumberedItems
```

To return all the paragraphs that have list formatting, use the **ListParagraphs** property. To return them as a range, use the **Range** property.

Remarks

To apply a different list format to an existing list, use the **ApplyListTemplate** method with the **List** object. To add a new list to a document, use the **ApplyListTemplate** method with the **ListFormat** object for a specified range.

Use the **CanContinuePreviousList** method to determine whether you can continue the list formatting from a list that was previously applied to the document.

Use the **CountNumberedItems** method to return the number of items in a numbered or bulleted list, including LISTNUM fields.

To determine whether a list contains more than one list template, use the **SingleListTemplate** property.

You can manipulate the individual **List** objects within a document, but for more precise control you should work with the **ListFormat** object.

ListParagraphs, **Range**

ListFormat, **Document**

ListTemplate Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListTemplateC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListTemplateP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListTemplateM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListTemplateE "}

ListTemplates (ListTemplate)

ListLevels (ListLevel)

Represents a single list template that includes all the formatting that defines a list. The **ListTemplate** object is a member of the **ListTemplates** collection. Each of the seven formats (excluding **None**) found on each of the three tabs in the **Bullets and Numbering** dialog box corresponds to a list template object. These predefined list templates can be accessed from the three **ListGallery** objects in the **ListGalleries** collection. Documents and templates can also contain collections of list templates.

Using the ListTemplate Object

Use **ListTemplates***(index)*, where *index* is a number from 1 through 7, to return a single list template from a list gallery. The following example returns the third list format (excluding **None**) on the **Numbered** tab in the **Bullets and Numbering** dialog box.

```
Set temp3 = ListGalleries(2).ListTemplates(3)
```

Note Some properties and methods – **Convert** and **Add**, for example – won't work with list templates that are accessed from a list gallery. You can modify these list templates, but you cannot change their list gallery type (**wdBulletGallery**, **wdNumberGallery**, or **wdOutlineNumberGallery**).

The following example sets an object variable equal to the list template used in the third list in the active document, and then it applies that list template to the selection.

```
Set myLt = ActiveDocument.ListTemplates(3)
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=myLt
```

Use the **Add** method to add a list template to the collection of list templates in a document or template.

Resetting a List Template in the Gallery

To see whether the specified list template contains the formatting built into Word, use the **Modified** property with the **ListGallery** object. To reset formatting to the original list format, use the **Reset** method for the **ListGallery** object.

Remarks

After you have returned a **ListTemplate** object, use **ListLevels**(*index*), where *index* is a number from 1 through 9, to return a single **ListLevel** object. With a **ListLevel** object, you have access to all the formatting properties for the specified list level, such as **Alignment**, **Font**, **NumberFormat**, **NumberPosition**, **NumberStyle**, and **TrailingCharacter**.

Use the **Convert** method to convert a multiple-level list template to a single-level template.

ListTemplates Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListTemplatesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListTemplatesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListTemplatesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListTemplatesE "}

ListTemplates (ListTemplate)

ListLevels (ListLevel)

A collection of **ListTemplate** objects that represent the seven predefined list formats on each tab in the **Bullets and Numbering** dialog box.

Using the ListTemplates Collection

Use the **ListTemplates** property to return the **ListTemplates** collection.The following example displays a message with the level status (single or multiple-level) for each list template in the active document.

```
For Each lt In ActiveDocument.ListTemplates
    MsgBox "This is a multiple-level list template - " & lt.OutlineNumbered
Next LT
```

Use the **Add** method to add a list template to the collection in the specified document or template. The following example adds a new list template to the active document and applies it to the selection.

```
Set myLT = ActiveDocument.ListTemplates.Add
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=myLT
```

Use **ListTemplates***(index)*, where *index* is a number 1 through 7, to return a single list template from a list gallery. The following example sets an object variable equal to the list template used in the third list in the active document, and then it applies that list template to the selection.

```
Set mylt = ActiveDocument.ListTemplates(3)
Selection.Range.ListFormat.ApplyListTemplate ListTemplate:=mylt
```

Note Some properties and methods − **Convert** and **Add**, for example − won't work with list templates that are accessed from a list gallery. You can modify these list templates, but you cannot change their list gallery type (**wdBulletGallery**, **wdNumberGallery**, or **wdOutlineNumberGallery**).

Resetting a List Template in the Gallery

To see whether the specified list template contains the formatting built into Word, use the **Modified** property with the **ListGallery** object. To reset formatting to the original list format, use the **Reset** method for the **ListGallery** object.

Remarks

After you have returned a **ListTemplate** object, use **ListLevels**(*index*), where *index* is a number from 1 through 9, to return a single **ListLevel** object. With a **ListLevel** object, you have access to all the formatting properties for the specified list level, such as **Alignment**, **Font**, **NumberFormat**, **NumberPosition**, **NumberStyle**, and **TrailingCharacter**.

Use the **Convert** method to convert a multiple-level list template to a single-level template.

Document, **ListFormat**, **ListGallery**, **Style**, **Template**,

ListParagraphs Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListParagraphsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListParagraphsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListParagraphsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListParagraphsE "}

A collection of **Paragraph** objects that represent the paragraphs of the specified document, list, or range that have list formatting applied.

Using the ListParagraphs Collection

Use the **ListParagraphs** property to return the **ListParagraphs** collection. The following example applies highlighting to the collection of paragraphs with list formatting in the active document.

```
For Each para in ActiveDocument.ListParagraphs
    para.Range.HighlightColorIndex = wdTurquoise
Next para
```

Use **ListParagraphs**(*index*), where *index* is the index number, to return a single **Paragraph** object with list formatting.

Remarks

Paragraphs can have two types of list formatting. The first type includes an automatically added number or bullet at the beginning of each paragraph in the list. The second type includes LISTNUM fields, which can be placed anywhere inside a paragraph. There can be more than one LISTNUM field per paragraph.

To add list formatting to paragraphs, you can use the **ApplyListTemplate**, **ApplyBulletDefault**, **ApplyNumberDefault**, or **ApplyOutlineNumberDefault** method. You access these methods from the **ListFormat** object for a specified range.

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

Document, **List**, **Range**

Borders, **DropCap**, **ParagraphFormat**, **Range**, **Shading**, **TabStops**

List, **ListTemplate**

ListFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjListFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjListFormatP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjListFormatM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjListFormatE "}

Represents the list formatting attributes that can be applied to the paragraphs in a range.

Using the ListFormat Object

Use the **ListFormat** property to return the **ListFormat** object for a range. The following example applies the default bulleted list format to the selection.

```
Selection.Range.ListFormat.ApplyBulletDefault
```

Applying a List Template

An easy way to apply list formatting is to use the **ApplyBulletDefault**, **ApplyNumberDefault**, and **ApplyOutlineNumberDefault** methods, which correspond, respectively, to the first list format (excluding **None**) on each tab in the **Bullets and Numbering** dialog box.

To apply a format other than the default format, use the **ApplyListTemplate** method, which allows you to specify the list format (list template) you want to apply.

Returning the List or List Template

Use the **List** or **ListTemplate** property to return the list or list template from the first paragraph in the specified range.

Remarks

Use the **ListFormat** property with a **Range** object to access the list formatting properties and methods available for the specified range. The following example applies the default bullet list format to the second paragraph in the active document.

```
ActiveDocument.Paragraphs(2).Range.ListFormat.ApplyBulletDefault
```

However, if there's already a list defined in your document, you can access a **List** object by using the **Lists** property. The following example changes the format of the list created in the preceding example to the first number format on the **Numbered** tab in the **Bullets and Numbering** dialog box.

```
ActiveDocument.Lists(1).ApplyListTemplate
ListTemplate:=ListGalleries(2).ListTemplates(1)
```

LetterContent Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjLetterContentC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjLetterContentP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjLetterContentM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjLetterContentE "}

LetterContent

Represents the elements of a letter created by the Letter Wizard.

Using the LetterContent Object

Use the **GetLetterContent** method or the **CreateLetterContent** method to return a **LetterContent** object. The following example retrieves and displays the letter recipient's name from the active document.

```
Set myLetterContent = ActiveDocument.GetLetterContent
MsgBox myLetterContent.RecipientName
```

The following example uses the **CreateLetterContent** method to create a new **LetterContent** object, which is then used with the **RunLetterWizard** method.

```
Set myLetter = ActiveDocument.CreateLetterContent(DateFormat:="July 11,
1996", _
    IncludeHeaderFooter:=False, _
    PageDesign:="C:\MSOffice\Templates\Letters & Faxes\Contemporary
Letter.dot", _
    LetterStyle:=wdFullBlock, Letterhead:=True,
LetterheadLocation:=wdLetterTop, _
    LetterheadSize:=InchesToPoints(1.5), RecipientName:="Dave Edson", _
    RecipientAddress:="100 Main St." & vbCr & "Bellevue, WA 98004", _
    Salutation:="Dear Dave,", SalutationType:=wdSalutationInformal, _
    RecipientReference:="", MailingInstructions:="", AttentionLine:="", _
    Subject:="End of year report", CCList:="", ReturnAddress:="", _
    SenderName:="", Closing:="Sincerely yours,", SenderCompany:="", _
    SenderJobTitle:="", SenderInitials:="", EnclosureNumber:=0)
ActiveDocument.RunLetterWizard LetterContent:=myLetter, WizardMode:=True
```

Remarks

The **CreateLetterContent** method creates a **LetterContent** object; however, there are numerous required arguments. If you want to set only a few properties, use the **New** keyword to create a new, stand-alone **LetterContent** object. The following example creates a **LetterContent** object, sets some of its properties, and then uses the **LetterContent** object with the **RunLetterWizard** method to run the Letter Wizard, using the preset values as the default settings.

```
Set myLetter = New LetterContent
With myLetter
    .AttentionLine = "Read this"
    .EnclosureNumber = 1
    .Letterhead = True
    .LetterheadLocation = wdLetterTop
    .LetterheadSize = InchesToPoints(2)
End With
Documents.Add.RunLetterWizard LetterContent:=myLetter, WizardMode:=True
```

You can duplicate a **LetterContent** object by using the **Duplicate** property. The following example retrieves the letter elements in the active document and makes a duplicate copy. The example assigns the duplicate copy to `aLetter` and resets the recipient's name and address to empty strings. The **RunLetterWizard** method is used to run the Letter Wizard, using the values in the revised **LetterContent** object (`aLetter`) as the default settings.

```
Set aLetter = ActiveDocument.GetLetterContent.Duplicate
With aLetter
    .RecipientName = ""
    .RecipientAddress = ""
Documents.Add.RunLetterWizard LetterContent:=aLetter, WizardMode:=True
```

The **SetLetterContent** method inserts the contents of the specified **LetterContent** object in a document. The following example retrieves the letter elements from the active document, changes the attention line, and then uses the **SetLetterContent** method to update the active document to reflect the change.

```
Set myLetterContent = ActiveDocument.GetLetterContent
myLetterContent.AttentionLine = "Greetings"
ActiveDocument.SetLetterContent LetterContent:=myLetterContent
```

TextRetrievalMode Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTextRetrievalModeC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTextRetrievalModeP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTextRetrievalModeM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTextRetrievalModeE "}

TextRetrievalMode

Represents options that control how text is retrieved from a **Range** object.

Using the TextRetrievalMode Object

Use the **TextRetrievalMode** property to return a **TextRetrievalMode** object. The following example displays the text of the first sentence in the active document, excluding field codes and hidden text.

```
With ActiveDocument.Sentences(1).TextRetrievalMode
    .IncludeHiddenText = False
    .IncludeFieldCodes = False
    MsgBox .Parent.Text
End With
```

Remarks

Changing the **ViewType**, **IncludeHiddentText**, or **IncludeFieldCodes** property of the **TextRetrievalMode** object doesn't change the screen display. Instead, changing one of these properties determines what text is retrieved from a **Range** object when the **Text** property is used.

Hyperlink Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHyperlinkC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHyperlinkP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHyperlinkM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHyperlinkE "}

Hyperlinks (Hyperlink)

Represents a hyperlink. The **Hyperlink** object is a member of the **Hyperlinks** collection.

Using the Hyperlink Object

Use the **Hyperlink** property to return a **Hyperlink** object associated with a shape (a shape can have only one hyperlink). The following example activates the hyperlink associated with the first shape in the active document.

```
ActiveDocument.Shapes(1).Hyperlink.Follow
```

Use **Hyperlinks**(*index*), where *index* is the index number, to return a single **Hyperlink** object from a document, range, or selection. The following example activates the first hyperlink in the selection.

```
If Selection.HyperLinks.Count >= 1 Then
    Selection.HyperLinks(1).Follow
End If
```

Document, **InlineShape**, **Range**, **Selection**, **Shape**, **ShapeRange**

Hyperlinks Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjHyperlinksC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjHyperlinksP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjHyperlinksM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjHyperlinksE "}

Hyperlinks (Hyperlink)

Represents the collection of **Hyperlink** objects in a document, range, or selection.

Using the Hyperlinks Collection

Use the **Hyperlinks** property to return the **Hyperlinks** collection. The following example checks all the hyperlinks in document one for a link that contains the word "Microsoft" in the address. If a hyperlink is found, it's activated with the **Follow** method.

```
For Each hLink In Documents(1).Hyperlinks
    If InStr(hLink.Address, "Microsoft") <> 0 Then
        hLink.Follow
        Exit For
    End If
Next hLink
```

Use the **Add** method to create a hyperlink and add it to the **Hyperlinks** collection. The following example creates a new hyperlink to the MSN Web site.

```
ActiveDocument.Hyperlinks.Add Address:="http://www.msn.com/", _
    Anchor:=Selection.Range
```

Use **Hyperlinks**(*index*), where *index* is the index number, to return a single **Hyperlink** object in a document, range, or selection. The following example activates the first hyperlink in the selection.

```
If Selection.HyperLinks.Count >= 1 Then
    Selection.HyperLinks(1).Follow
End If
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

Range, **Shape**

KeyBinding Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjKeyBindingC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjKeyBindingP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjKeyBindingM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjKeyBindingE "}

KeyBindings (KeyBinding)

Represents a custom key assignment in the current context. The **KeyBinding** object is a member of the **KeyBindings** collection. Custom key assignments are made in the **Customize Keyboard** dialog box.

Using the KeyBinding Object

Use **KeyBindings**(*index*), where *index* is the index number, to return a single **KeyBinding** object. The following example displays the command associated with the first **KeyBinding** object in the **KeyBindings** collection.

```
MsgBox KeyBindings(1).Command
```

You can also use the **FindKey** property and the **Key** method to return a **KeyBinding** object.

Application, **KeysBoundTo**

KeyBindings Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjKeyBindingsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjKeyBindingsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjKeyBindingsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjKeyBindingsE "}

KeyBindings (KeyBinding)

A collection of **KeyBinding** objects that represent the custom key assignments in the current context. Custom key assignments are made in the **Customize Keyboard** dialog box.

Using the Keybindings Collection

Use the **KeyBindings** property to return the **KeyBindings** collection. The following example inserts after the selection the command name and key combination for each item in the **KeyBindings** collection.

```
CustomizationContext = NormalTemplate
For Each aKey In KeyBindings
    Selection.InsertAfter aKey.Command & vbTab & aKey.KeyString & vbCr
    Selection.Collapse Direction:=wdCollapseEnd
Next aKey
```

Use the **Add** method to add a **KeyBinding** object to the **KeyBindings** collection. The following example adds the CTRL+ALT+H key combination to the Heading 1 style in the active document.

```
CustomizationContext = ActiveDocument
KeyBindings.Add KeyCategory:=wdKeyCategoryStyle, Command:="Heading 1", _
    KeyCode:=BuildKeyCode(wdKeyControl, wdKeyAlt, wdKeyH)
```

Use **KeyBindings**(*index*), where *index* is the index number, to return a single **KeyBinding** object. The following example displays the command associated with the first **KeyBinding** object in the **KeyBindings** collection.

```
MsgBox KeyBindings(1).Command
```

KeysBoundTo Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjKeysBoundToC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjKeysBoundToP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjKeysBoundToM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjKeysBoundToE "}

KeysBoundTo (KeyBinding)

A collection of **KeyBinding** objects assigned to a command, style, macro, or other item in the current context.

Using the KeysBoundTo Collection

Use the **KeysBoundTo** property to return the **KeysBoundTo** collection. The following example displays the key combinations assigned to the **FileNew** command in the Normal template.

```
CustomizationContext = NormalTemplate
For Each myKey In KeysBoundTo(KeyCategory:=wdKeyCategoryCommand,
Command:="FileNew")
    myStr = myStr & myKey.KeyString & vbCr
Next myKey
MsgBox myStr
```

The following example displays the name of the document or template where the keys for the macro named "Macro1" are stored.

```
Set kb = KeysBoundTo(KeyCategory:=wdKeyCategoryMacro, Command:="Macro1")
MsgBox kb.Context.Name
```

MailMergeDataSource Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailMergeDataSourceC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailMergeDataSourceP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailMergeDataSourceM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjMailMergeDataSourceE "}

Represents the mail merge data source in a mail merge operation.

Using the MailMergeDataSource Object

Use the **DataSource** property to return the **MailMergeDataSource** object. The following example displays the name of the data source associated with to the active document.

```
If ActiveDocument.MailMerge.DataSource.Name <> "" Then _
    MsgBox ActiveDocument.MailMerge.DataSource.Name
```

The following example displays the field names in the data source associated with the active document.

```
For Each aField In ActiveDocument.MailMerge.DataSource.FieldNames
    MsgBox aField.Name
Next aField
```

The following example opens the data source associated with Form letter.doc and determines whether the FirstName field includes the name "Kate."

```
With Documents("Form letter.doc").MailMerge
    .EditDataSource
    If .DataSource.FindRecord(FindText:="Kate", Field:="FirstName") = True
Then
        MsgBox "Data was found"
    End If
End With
```

MailMergeDataFields, **MailMergeFieldNames**

Range Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjRangeC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjRangeP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjRangeM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjRangeE "}

Represents a contiguous area in a document. Each **Range** object is defined by a starting and ending character position. Similar to the way bookmarks are used in a document, **Range** objects are used in Visual Basic procedures to identify specific portions of a document. However, unlike a bookmark, a **Range** object only exists while the procedure that defined it is running.

Note **Range** objects are independent of the selection. That is, you can define and manipulate a range without changing the selection. You can also define multiple ranges in a document, while there can be only one selection per pane.

Using the Range Object

Use the **Range** method to return a **Range** object defined by the given starting and ending character positions. The following example returns a **Range** object that refers to the first 10 characters in the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=10)
```

Use the **Range** property to return a **Range** object defined by the beginning and end of another object. The **Range** property applies to many objects (for example, **Paragraph**, **Bookmark**, and **Cell**). The following example returns a **Range** object that refers to the first paragraph in the active document.

```
Set aRange = ActiveDocument.Paragraphs(1).Range
```

The following example returns a **Range** object that refers to the second through fourth paragraphs in the active document

```
Set aRange =
ActiveDocument.Range(Start:=ActiveDocument.Paragraphs(2).Range.Start, _
    End:=ActiveDocument.Paragraphs(4).Range.End)
```

For more information about working with **Range** objects, see Working with Range Objects.

Bookmark, **Cell**, **Characters**, **Comment**, **Document**, **Endnote**, **Footnote**, **FormField**, **Frame**, **HeaderFooter**, **HyperLink**, **Index**, **InlineShape**, **List**, **Paragraph**, **ProofreadingErrors**, **Revision**, **Row**, **Section**, **Selection**, **Sentences**, **Subdocument**, **Table**, **TableOfAuthorities**, , **TableOfContents**, **TableOfFigures**, **TextFrame**, **Words**

Bookmarks, **Borders**, **Cells**, **Characters**, **Columns**, **Comments**, **Endnotes**, **Fields**, **Find**, **Font**, **Footnotes**, **FormFields**, **Frames**, **HyperLinks**, **InlineShapes**, **ListFormat**, **ListParagraphs**, **PageSetup**, **ParagraphFormat**, **Paragraphs**, **ProofreadingErrors**, **ReadabilityStatistics**, **Revisions**, **Rows**, **Sections**, **Sentences**, **Shading**, **ShapeRange**, **Subdocuments**, **SynonymInfo**, **Tables**, **TextRetrievalMode**, **Words**

Dialog Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjDialogC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjDialogP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjDialogM"}

Represents a built-in dialog box. The **Dialog** object is a member of the **Dialogs** collection. The **Dialogs** collection contains all the built-in dialog boxes in Word. You cannot create a new built-in dialog box or add one to the **Dialogs** collection.

Using the Dialog Object

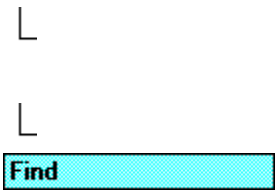
Use **Dialogs**(*index*), where *index* is a **WdWordDialog** constant that identifies the dialog box, to return a single **Dialog** object. The following example displays and carries out the actions taken in the built-in **Open** dialog box (**File** menu).

```
dlgAnswer = Dialogs(wdDialogFileOpen).Show
```

The **WdWordDialog** constants are formed from the prefix "wdDialog" followed by the name of the menu and the dialog box. For example, the constant for the **Page Setup** dialog box is **wdDialogFilePageSetup**, and the constant for the **New** dialog box is **wdDialogFileNew**. For more information about working with built-in Word dialog boxes, see Displaying built-in Word dialog boxes.

Find Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjFindC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjFindP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjFindM"}



Represents the criteria for a find operation. The properties and methods of the **Find** object correspond to the options in the **Find and Replace** dialog box.

Using the Find Object

Use the **Find** property to return a **Find** object. The following example finds and selects the next occurrence of the word "hi."

```
With Selection.Find
    .ClearFormatting
    .Text = "hi"
    .Execute Forward:=True
End With
```

The following example finds all occurrences of the word "hi" in the active document and replaces the word with "hello."

```
Set myRange = ActiveDocument.Content
myRange.Find.Execute FindText:="hi", ReplaceWith:="hello", _
    Replace:=wdReplaceAll
```

Remarks

If you've gotten to the **Find** object from the **Selection** object, the selection is changed when text matching the find criteria is found. The following example selects the next occurrence of the word "blue."

```
Selection.Find.Execute FindText:="blue", Forward:=True
```

If you've gotten to the **Find** object from the **Range** object, the selection isn't changed when text matching the find criteria is found, but the **Range** object is redefined. The following example locates the first occurrence of the word "blue" in the active document. If "blue" is found in the document, `myRange` is redefined and bold formatting is applied to "blue."

```
Set myRange = ActiveDocument.Content
myRange.Find.Execute FindText:="blue", Forward:=True
If myRange.Find.Found = True Then myRange.Bold = True
```

Range, **Selection**

Font, **Frame**, **ParagraphFormat**, **Replacement**

Mailer Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjMailerC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjMailerP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjMailerM"}

Mailer

Represents the PowerTalk mailer for a document. This object is available only on the Macintosh, with the PowerTalk system extension installed.

Using the Mailer Object

Use the **Mailer** property to return the **Mailer** object. The following example sets the **Subject** property for the mailer attached to the active document.

```
With ActiveDocument
    .HasMailer = True
    .Mailer.Subject = "Here's the document."
End With
```

Replacement Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjReplacementC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjReplacementP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjReplacementM"}

Find

Replacement

Represents the replace criteria for a find-and-replace operation. The properties and methods of the **Replacement** object correspond to the options in the **Find and Replace** dialog box.

Using the Replacement Object

Use the **Replacement** property to return a **Replacement** object. The following example replaces the next occurrence of the word "hi" with the word "hello."

```
With Selection.Find
    .Text = "hi"
    .ClearFormatting
    .Replacement.Text = "hello"
    .Replacement.ClearFormatting
    .Execute Replace:=wdReplaceOne, Forward:=True
End With
```

To find and replace formatting, set both the find text and the replace text to empty strings (""), and set the ***Format*** argument of the **Execute** method to **True**. The following example removes all the bold formatting in the active document. The **Bold** property is **True** for the **Find** object and **False** for the **Replacement** object.

```
With ActiveDocument.Content.Find
    .ClearFormatting
    .Font.Bold = True
    .Text = ""
    With .Replacement
        .ClearFormatting
        .Font.Bold = False
        .Text = ""
    End With
    .Execute Format:=True, Replace:=wdReplaceAll
End With
```

Font, **Frame**, **ParagraphFormat**

Find, **Range**, **Replacement**, **Selection**, **Style**

Borders, **ParagraphFormat**, **Shading**, **TabStops**

ParagraphFormat Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjParagraphFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjParagraphFormatP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjParagraphFormatM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjParagraphFormatE "}

Represents all the formatting for a paragraph.

Using the ParagraphFormat Object

Use the **Format** property to return the **ParagraphFormat** object for a paragraph or paragraphs. The **ParagraphFormat** property returns the **ParagraphFormat** object for a selection, range, style, **Find** object, or **Replacement** object. The following example centers the third paragraph in the active document.

```
ActiveDocument.Paragraphs(3).Format.Alignment = wdAlignParagraphCenter
```

The following example finds the next double-spaced paragraph after the selection.

```
With Selection.Find
    .ClearFormatting
    .ParagraphFormat.LineSpacingRule = wdLineSpaceDouble
    .Text = ""
    .Forward = True
    .Wrap = wdFindContinue
End With
Selection.Find.Execute
```

Remarks

You can use the **New** keyword to create a new, standalone **ParagraphFormat** object. The following example creates a **ParagraphFormat** object, sets some formatting properties for it, and then applies all of its properties to the first paragraph in the active document.

```
Dim myParaF As New ParagraphFormat
myParaF.Alignment = wdAlignParagraphCenter
myParaF.Borders.Enable = True
ActiveDocument.Paragraphs(1).Format = myParaF
```

You can also make a standalone copy of an existing **ParagraphFormat** object by using the **Duplicate** property. The following example duplicates the paragraph formatting of the first paragraph in the active document and stores the formatting in `myDup`. The example changes the left indent of `myDup` to 1 inch, creates a new document, inserts text into the document, and applies the paragraph formatting of `myDup` to the text.

```
Set myDup = ActiveDocument.Paragraphs(1).Format.Duplicate
myDup.LeftIndent = InchesToPoints(1)
Documents.Add
Selection.InsertAfter "This is a new paragraph."
```

```
Selection.Paragraphs.Format = myDup
```

Addins, **Application**, **Assistant**, **AutoCaptions**, **AutoCorrect**, **CaptionLabels**, **CommandBars**, **Dictionaries**, **Dialogs**, **Documents**, **FileConverters**, **FontNames**, **KeyBindings**, **KeysBoundTo**, **Languages**, **ListGalleries**, **Options**, **RecentFiles**, **Selection**, **SynonymInfo**, **System**, **Tasks**, **Templates**, **VBE**, **Windows**

Global Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjGlobalC"} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjGlobalP"} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjGlobalM"}

Global

Contains top-level properties and methods that don't need to be preceded by the **Application** property. For example, the following two statements have the same result.

```
Documents(1).Content.Bold = True
Application.Documents(1).Content.Bold = True
```

StoryRanges Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjStoryRangesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjStoryRangesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjStoryRangesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjStoryRangesE "}

StoryRanges (Range)

A collection of **Range** objects that represent stories in a document.

Using the StoryRanges Collection

Use the **StoryRanges** property to return the **StoryRanges** collection. The following example removes manual character formatting from the text in all stories other than the main text story in the active document.

```
For Each aStory In ActiveDocument.StoryRanges
    If aStory.StoryType <> wdMainTextStory Then aStory.Font.Reset
Next aStory
```

The **Add** method isn't available for the **StoryRanges** collection. The number of stories in the **StoryRanges** collection is finite.

Use **StoryRanges**(*index*), where *index* is a **wdStoryType** constant, to return a single story as a **Range** object. The following example adds text to the primary header story and then displays the text.

```
ActiveDocument.Sections(1).Headers(wdHeaderFooterPrimary).Range _
    .Text = "Header text"
MsgBox ActiveDocument.StoryRanges(wdPrimaryHeaderStory).Text
```

The following example copies the text of the footnotes from the active document into a new document.

```
If ActiveDocument.Footnotes.Count >= 1 Then
    ActiveDocument.StoryRanges(wdFootnotesStory).Copy
    Documents.Add.Content.Paste
End If
```

Remarks

If you attempt to return a story that isn't available in the specified document, an error occurs. The following example determines whether or not a footnote story is available in the active document.

```
On Error GoTo errhandler
Set MyRange = ActiveDocument.StoryRanges(wdFootnotesStory)
errhandler:
If Err = 5941 Then MsgBox "The footnotes story is not available."
```

Use the **NextStoryRange** property to loop through all stories in a document. The following example searches each story in the active document for the text "Microsoft Word." When the text is found, it's formatted as italic.

```
For Each myStoryRange In ActiveDocument.StoryRanges
    myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
    While myStoryRange.Find.Found
        myStoryRange.Italic = True
        myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
    Wend
    While Not (myStoryRange.NextStoryRange Is Nothing)
        Set myStoryRange = myStoryRange.NextStoryRange
        myStoryRange.Find.Execute FindText:="Microsoft Word", Forward:=True
        While myStoryRange.Find.Found
            myStoryRange.Italic = True
            myStoryRange.Find.Execute FindText:="Microsoft Word",
Forward:=True
        Wend
    Wend
Next myStoryRange
```

Style Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjStyleC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjStyleP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjStyleM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjStyleE "}

Styles (Style)

Represents a single built-in or user-defined style. The **Style** object includes style attributes (font, font style, paragraph spacing, and so on) as properties of the **Style** object. The **Style** object is a member of the **Styles** collection. The **Styles** collection includes all the styles in the specified document.

Using the Style Object

Use **Styles**(*index*), where *index* is the style name, a **WdBuiltinStyle** constant or index number, to return a single **Style** object. You must exactly match the spelling and spacing of the style name, but not necessarily its capitalization. The following example modifies the font name of the user-defined style named "Color" in the active document.

```
ActiveDocument.Styles("Color").Font.Name = "Arial"
```

The following example sets the built-in Heading 1 style to not be bold.

```
ActiveDocument.Styles(wdStyleHeading1).Font.Bold = False
```

The style index number represents the position of the style in the alphabetically sorted list of style names. Note that `Styles(1)` is the first style in the alphabetic list. The following example displays the base style and style name of the first style in the **Styles** collection.

```
MsgBox "Base style= " & ActiveDocument.Styles(1).BaseStyle & vbCr & _
    "Style name= " & ActiveDocument.Styles(1).NameLocal
```

To apply a style to a range, paragraph, or multiple paragraphs, set the **Style** property to a user-defined or built-in style name. The following example applies the Normal style to the first four paragraphs in the active document.

```
Set myRange =
ActiveDocument.Range(Start:=ActiveDocument.Paragraphs(1).Range.Start, _
    End:=ActiveDocument.Paragraphs(4).Range.End)
myRange.Style = wdStyleNormal
```

The following example applies the Heading 1 style to the first paragraph in the selection.

```
Selection.Paragraphs(1).Style = wdStyleHeading1
```

The following example creates a character style named "Bolded" and applies it to the selection.

```
Set myStyle = ActiveDocument.Styles.Add(Name:="Bolded", _
    Type:=wdStyleTypeCharacter)
myStyle.Font.Bold = True
Selection.Range.Style = "Bolded"
```

Remarks

Use the **OrganizerCopy** method to copy styles between documents and templates. Use the **UpdateStyles** method to update the styles in the active document to match the style definitions in the attached template. Use the **OpenAsDocument** method to open a template as a document so that you can modify the template styles.

Document, **Envelope**

Borders, **Font**, **Frame**, **ListTemplate** **ParagraphFormat**, **Shading**

Styles Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjStylesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjStylesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjStylesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjStylesE "}

Styles (Style)

A collection of **Style** objects that represent both the built-in and user-defined styles in a document.

Using the Styles Collection

Use the **Styles** property to return the **Styles** collection. The following example deletes all user-defined styles in the active document.

```
For Each sty In ActiveDocument.Styles
    If sty.BuiltIn = False Then sty.Delete
Next sty
```

Use the **Add** method to create a new user-defined style and add it to the **Styles** collection. The following example adds a new character style named "Introduction" and makes it 12-point Arial, with bold and italic formatting. The example then applies this new character style to the selection.

```
Set myStyle = ActiveDocument.Styles.Add(Name:="Introduction", _
    Type:=wdStyleTypeCharacter)
With myStyle.Font
    .Bold = True
    .Italic = True
    .Name = "Arial"
    .Size = 12
End With
Selection.Range.Style = "Introduction"
```

Use **Styles**(*index*), where *index* is the style name, a **WdBuiltinStyle** constant or index number, to return a single **Style** object. You must exactly match the spelling and spacing of the style name, but not necessarily its capitalization. The following example modifies the font of the user-defined style named "Color" in the active document.

```
ActiveDocument.Styles("Color").Font.Name = "Arial"
```

The following example sets the built-in Heading 1 style to not be bold.

```
ActiveDocument.Styles(wdStyleHeading1).Font.Bold = False
```

The style index number represents the position of the style in the alphabetically sorted list of style names. Note that `Styles(1)` is the first style in the alphabetic list. The following example displays the base style and style name of the first style in the **Styles** collection.

```
MsgBox "Base style= " & ActiveDocument.Styles(1).BaseStyle & vbCr & _
    "Style name= " & ActiveDocument.Styles(1).NameLocal
```

Remarks

The **Styles** object isn't available from the **Template** object. However, you can use the **OpenAsDocument** method to open a template as a document so that you can modify styles in the template. The following example changes the formatting of the Heading 1 style in the template attached to the active document.

```
Set aDoc = ActiveDocument.AttachedTemplate.OpenAsDocument
With aDoc
    .Styles(wdStyleHeading1).Font.Name = "Arial"
    .Close SaveChanges:=wdSaveChanges
End With
```

Use the **OrganizerCopy** method to copy styles between documents and templates. Use the **UpdateStyles** method to update the styles in the active document to match the style definitions in the attached template.

Subdocument Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSubdocumentC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSubdocumentP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSubdocumentM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSubdocumentE "}

Subdocuments (Subdocument)

Represents a subdocument within a document or range. The **Subdocument** object is a member of the **Subdocuments** collection. The **Subdocuments** collection includes all the subdocuments in the a range or document.

Using the Subdocument Object

Use **Subdocuments**(*index*), where *index* is the index number, to return a single **Subdocument** object. The following example displays the path and file name of the first subdocument in the active document.

```
If ActiveDocument.Subdocuments(1).HasFile = True Then
    MsgBox ActiveDocument.Subdocuments(1).Path & _
        Application.PathSeparator & ActiveDocument.Subdocuments(1).Name
End If
```

Use the **AddFromFile** or **AddFromRange** method to add a subdocument to a document. The following example adds a subdocument named "Setup.doc" at the end of the active document.

```
ActiveDocument.Subdocuments.Expanded = True
Selection.EndKey Unit:=wdStory
Selection.InsertParagraphBefore
ActiveDocument.Subdocuments.AddFromFile Name:="C:\Temp\Setup.doc"
```

The following example applies the Heading 1 style to the first paragraph in the selection and then creates a subdocument for the contents of the selection.

```
Selection.Paragraphs(1).Style = wdStyleHeading1
With ActiveDocument.Subdocuments
    .Expanded = True
    .AddFromRange Range:=Selection.Range
End With
```

Subdocuments Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSubdocumentsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSubdocumentsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSubdocumentsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSubdocumentsE "}

Subdocuments (Subdocument)

A collection of **Subdocument** objects that represent the subdocuments in a range or document.

Using the Subdocuments Collection

Use the **Subdocuments** property to return the **Subdocuments** collection. The following example expands all the subdocuments in the active document.

```
ActiveDocument.Subdocuments.Expanded = True
```

Use the **AddFromFile** or **AddFromRange** method to add a subdocument to a document. The following example adds a subdocument named "Setup.doc" at the end of the active document.

```
ActiveDocument.Subdocuments.Expanded = True
Selection.EndKey Unit:=wdStory
Selection.InsertParagraphBefore
ActiveDocument.Subdocuments.AddFromFile Name:="C:\Temp\Setup.doc"
```

The following example applies the Heading 1 style to the first paragraph in the selection and then creates a subdocument for the contents of the selection.

```
Selection.Paragraphs(1).Style = wdStyleHeading1
With ActiveDocument.Subdocuments
    .Expanded = True
    .AddFromRange Range:=Selection.Range
End With
```

Use **Subdocuments**(*index*), where *index* is the index number, to return a single **Subdocument** object. The following example displays the path and file name of the first subdocument in the active document.

```
If ActiveDocument.Subdocuments(1).HasFile = True Then
    MsgBox ActiveDocument.Subdocuments(1).Path & _
        Application.PathSeparator & ActiveDocument.Subdocuments(1).Name
End If
```

System Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjSystemC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjSystemP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjSystemM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjSystemE "}

System

Contains information about the computer system.

Using the System Object

Use the **System** property to return the **System** object. If the operating system is Windows, the following example makes a network connection to \\Project\Info.

```
If System.OperatingSystem = "Windows" Then
    System.Connect Path:="\\Project\Info"
End If
```

The following example displays the current screen resolution (for example, "1024 x 768").

```
horz = System.HorizontalResolution
vert = System.VerticalResolution
MsgBox "Resolution = " & horz & " x " & vert
```

Table Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTableC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTableP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTableM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTableE "}

Tables (Table)

Represents a single table. The **Table** object is a member of the **Tables** collection. The **Tables** collection includes all the tables in the specified selection, range, or document.

Using the Table Object

Use **Tables**(*index*), where *index* is the index number, to return a single **Table** object. The index number represents the position of the table in the selection, range, or document. The following example converts the first table in the active document to text.

```
ActiveDocument.Tables(1).ConvertToText Separator:=wdSeparateByTabs
```

Use the **Add** method to add a table at the specified range. The following example adds a 3x4 table at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.Tables.Add Range:=myRange, NumRows:=3, NumColumns:=4
```

Borders, **Cell**, **Columns**, **Range**, **Rows**, **Shading**

Tables Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTablesC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTablesP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTablesM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTablesE "}

Tables (Table)

A collection of **Table** objects that represent the tables in a selection, range, or document.

Using the Tables Collection

Use the **Tables** property to return the **Tables** collection. The following example applies a border around each of the tables in the active document.

```
For Each aTable In ActiveDocument.Tables
    aTable.Borders.OutsideLineStyle = wdLineStyleSingle
    aTable.Borders.OutsideLineWidth = wdLineWidth025pt
    aTable.Borders.InsideLineStyle = wdLineStyleNone
Next aTable
```

Use the **Add** method to add a table at the specified range. The following example adds a 3x4 table at the beginning of the active document.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.Tables.Add Range:=myRange, NumRows:=3, NumColumns:=4
```

Use **Tables**(*index*), where *index* is the index number, to return a single **Table** object. The index number represents the position of the table in the selection, range, or document. The following example converts the first table in the active document to text.

```
ActiveDocument.Tables(1).ConvertToText Separator:=wdSeparateByTabs
```

Remarks

The **Count** property for this collection in a document returns the number of items in the main story only. To count items in other stories use the collection with the **Range** object.

TablesOfAuthorities Collection Object

This item is not available for this version of Office.

TableOfAuthorities Object

this item is not available for this version of Office.

TablesOfAuthoritiesCategories Collection Object

This item is not available for this version of Office

TableOfAuthoritiesCategory Object

This item is not available for this version of Office.

TablesOfContents Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTablesOfContentsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTablesOfContentsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTablesOfContentsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTablesOfContentsE "}

TablesOfContents (TableOfContents)

A collection of **TableOfContents** objects that represent the tables of contents in a document.

Using the TablesOfContents Collection

Use the **TablesOfContents** property to return the **TablesOfContents** collection. The following example inserts a table of contents entry that references the selected text in the active document.

```
ActiveDocument.TablesOfContents.MarkEntry Range:=Selection.Range, _
    Level:=2, Entry:="Introduction"
```

Use the **Add** method to add a table of contents to a document. The following example adds a table of contents at the beginning of the active document. The example builds the table of contents from all paragraphs styled as either Heading 1, Heading 2, or Heading 3.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.TablesOfContents.Add Range:=myRange, _
    UseFields:=False, UseHeadingStyles:=True, LowerHeadingLevel:=3, _
    UpperHeadingLevel:=1
```

Use **TablesOfContents**(*index*), where *index* is the index number, to return a single **TableOfContents** object. The index number represents the position of the table of contents in the document. The following example updates the page numbers of the items in the first table of figures in the active document.

```
ActiveDocument.TablesOfContents(1).UpdatePageNumbers
```

HeadingStyles, **Range**

TableOfContents Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTableOfContentsC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTableOfContentsP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTableOfContentsM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTableOfContentsE "}

TablesOfContents (TableOfContents)

Represents a single table of contents in a document. The **TableOfContents** object is a member of the **TablesOfContents** collection. The **TablesOfContents** collection includes all the tables of contents in a document.

Using the TableOfCContents Object

Use **TablesOfContents**(*index*), where *index* is the index number, to return a single **TableOfContents** object. The index number represents the position of the table of contents in the document. The following example updates the page numbers of the items in the first table of figures in the active document.

```
ActiveDocument.TablesOfContents(1).UpdatePageNumbers
```

Use the **Add** method to add a table of contents to a document. The following example adds a table of contents at the beginning of the active document. The example builds the table of contents from all paragraphs styled as either Heading 1, Heading 2, or Heading 3.

```
Set myRange = ActiveDocument.Range(Start:=0, End:=0)
ActiveDocument.TablesOfContents.Add Range:=myRange, _
    UseFields:=False, UseHeadingStyles:=True, LowerHeadingLevel:=3, _
    UpperHeadingLevel:=1
```

TableOfFigures Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTableOfFiguresC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTableOfFiguresP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTableOfFiguresM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTableOfFiguresE "}

TablesOfFigures (TableOfFigures)

Represents a single table of figures in a document. The **TableOfFigures** object is a member of the **TablesOfFigures** collection. The **TablesOfFigures** collection includes all the tables of figures in a document.

Using the TableOfFigures Object

Use **TablesOfFigures**(*index*), where *index* is the index number, to return a single **TableOfFigures** object. The index number represents the position of the table of figures in the document. The following example updates the page numbers of the items in the first table of figures in the active document.

```
ActiveDocument.TablesOfFigures(1).UpdatePageNumbers
```

Use the **Add** method to add a table of figures to a document. A table of figures lists figure captions in the order in which they appear in the document. The following example replaces the selection in the active document with a table of figures that includes caption labels and page numbers.

```
ActiveDocument.TablesOfFigures.Add Range:=Selection.Range, _
    IncludeLabel:=True, IncludePageNumbers:=True
```

TablesOfFigures Collection Object

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woobjTablesOfFiguresC "} {ewc HLP95EN.DLL, DYNALINK, "Properties":"woobjTablesOfFiguresP "} {ewc HLP95EN.DLL, DYNALINK, "Methods":"woobjTablesOfFiguresM "} {ewc HLP95EN.DLL, DYNALINK, "Events":"woobjTablesOfFiguresE "}

TablesOfFigures (TableOfFigures)

A collection of **TableOfFigures** objects that represent the tables of figures in a document.

Using the TablesOfFigures Collection

Use the **TablesOfFigures** property to return the **TablesOfFigures** collection. The following example applies the Classic format to all tables of figures in the active document.

```
ActiveDocument.TablesOfFigures.Format = wdTOFClassic
```

Use the **Add** method to add a table of figures to a document. A table of figures lists figure captions in the order in which they appear in the document. The following example replaces the selection in the active document with a table of figures that includes caption labels and page numbers.

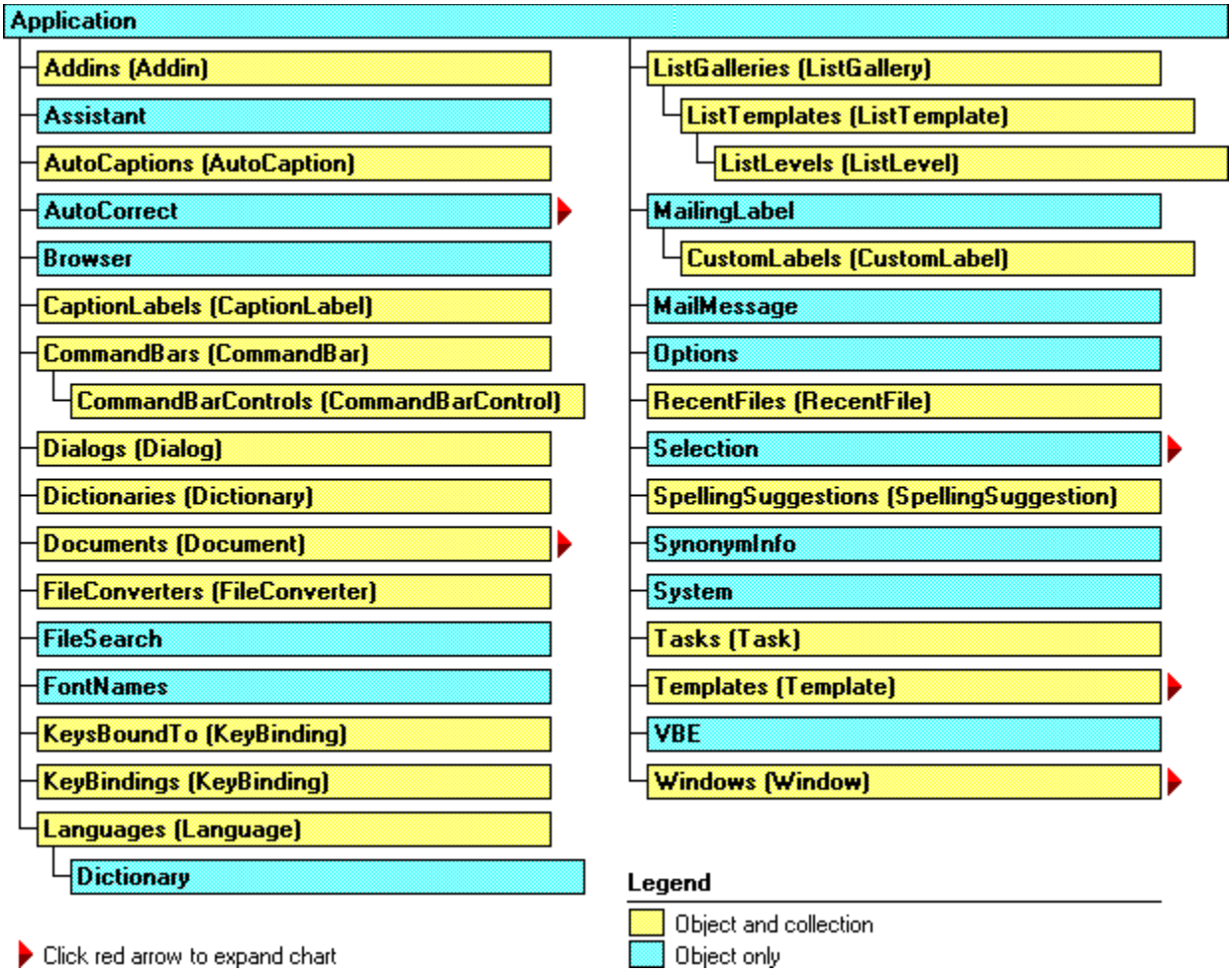
```
ActiveDocument.TablesOfFigures.Add Range:=Selection.Range, _
    IncludeLabel:=True, IncludePageNumbers:=True
```

Use **TablesOfFigures**(*index*), where *index* is the index number, to return a single **TableOfFigures** object. The index number represents the position of the table of figures in the document. The following example updates the page numbers of the items in the first table of figures in the active document.

```
ActiveDocument.TablesOfFigures(1).UpdatePageNumbers
```

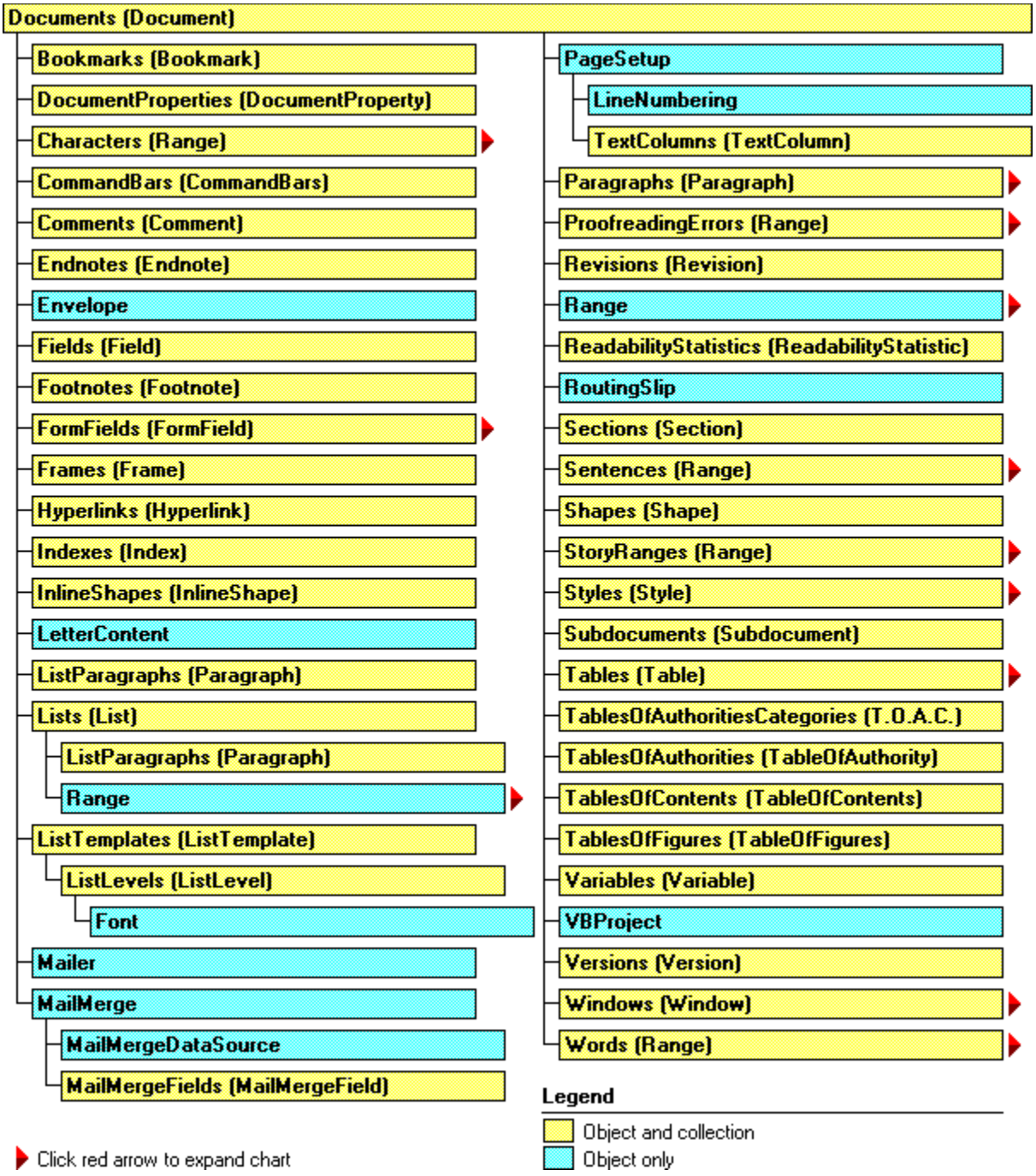
Microsoft Word Objects

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelApplicationC"}



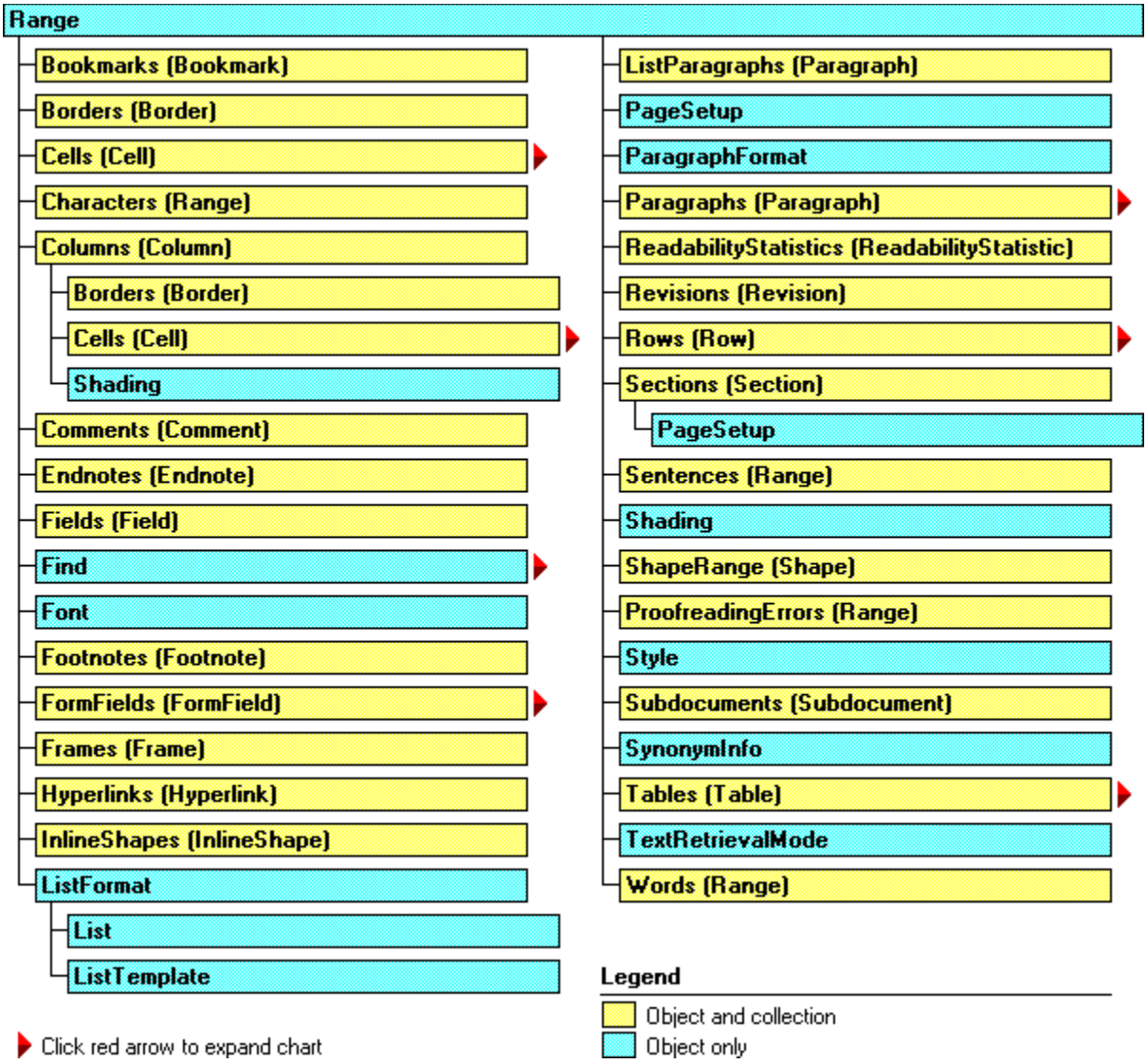
Microsoft Word Objects (Documents)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelDocumentsC"}



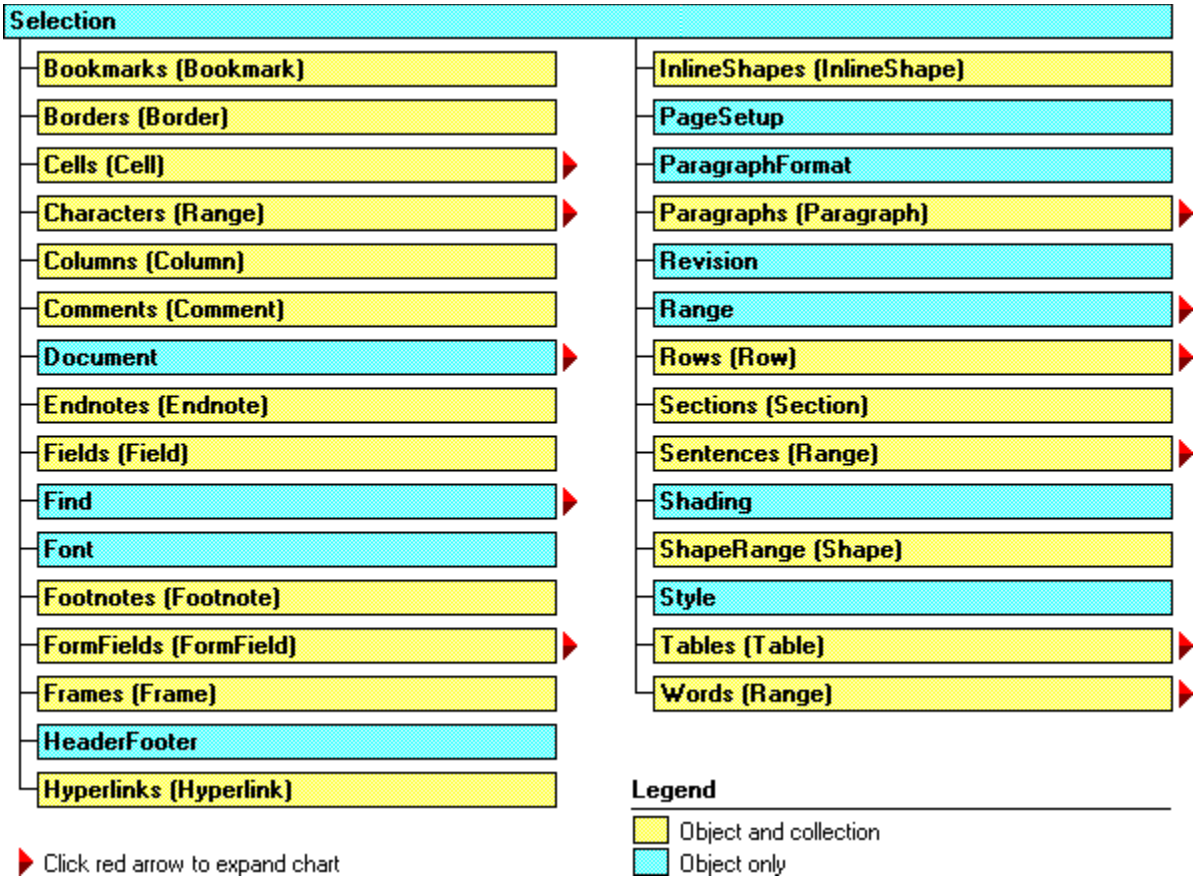
Microsoft Word Objects (Range)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelRangeC"}



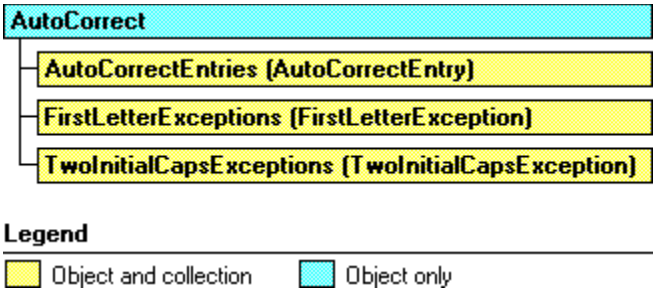
Microsoft Word Objects (Selection)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelSelectionC"}



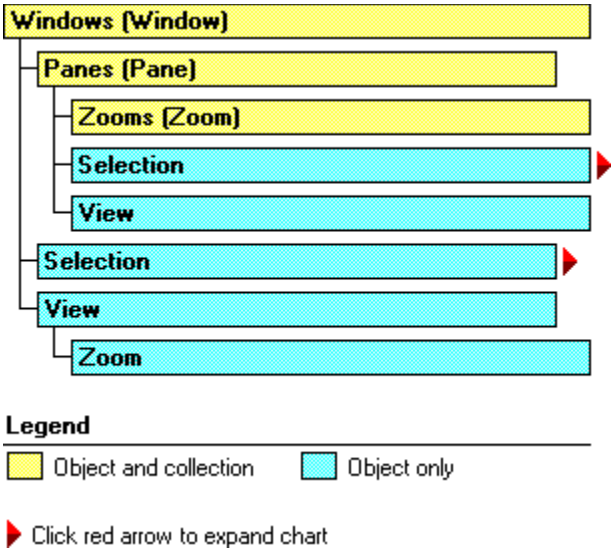
Microsoft Word Objects (AutoCorrect)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelAutoCorrectC"}



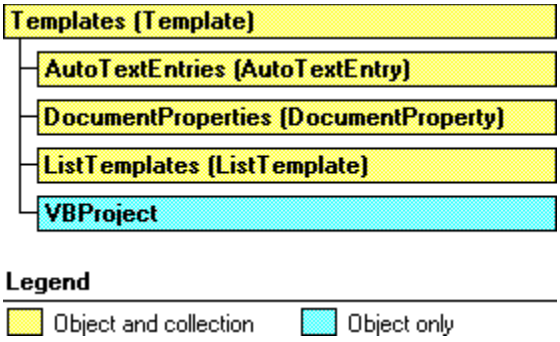
Microsoft Word Objects (Windows)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelWindowsC"}



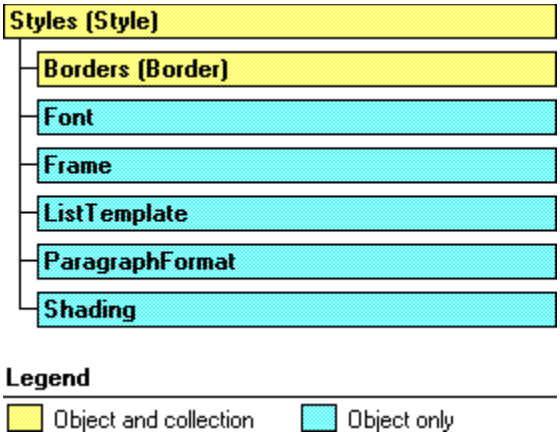
Microsoft Word Objects (Templates)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelTemplatesC"}



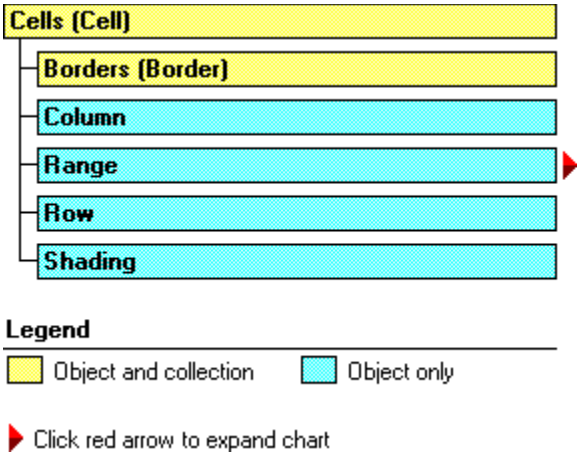
Microsoft Word Objects (Styles)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelStylesC"}



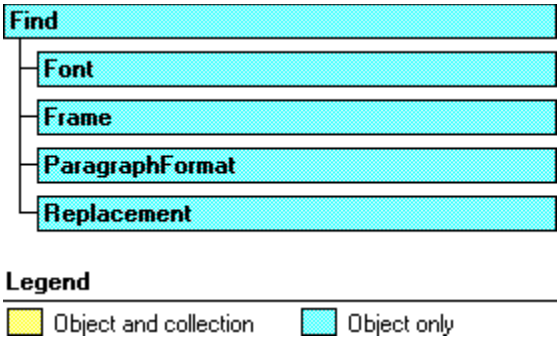
Microsoft Word Objects (Cells)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelCellsC"}



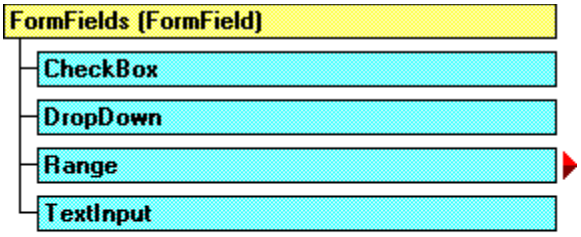
Microsoft Word Objects (Find)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelFindC"}



Microsoft Word Objects (FormFields)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelFormFieldsC"}



Legend

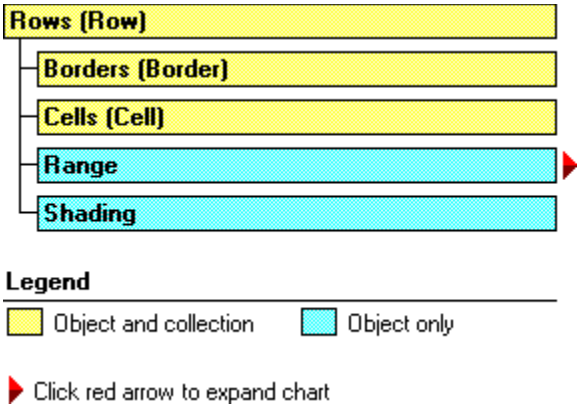
Object and collection

Object only

Click red arrow to expand chart

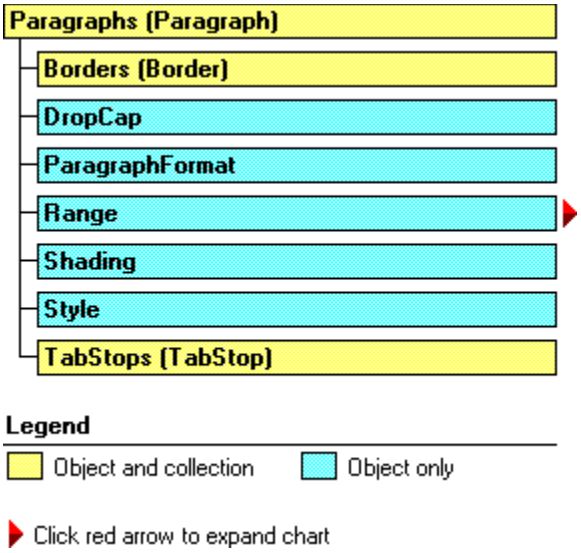
Microsoft Word Objects (Rows)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelRowsC"}



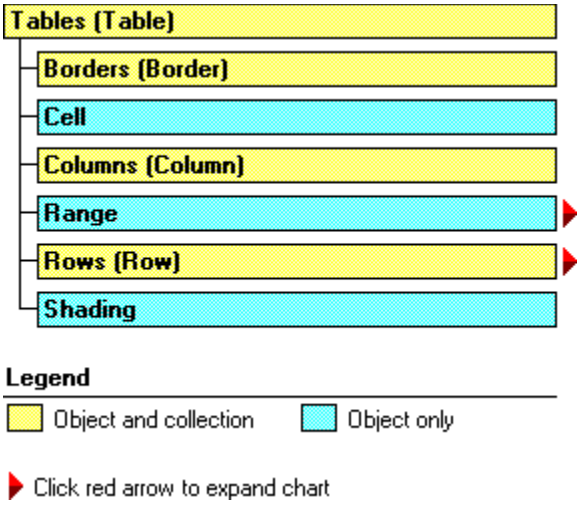
Microsoft Word Objects (Paragraphs)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelParagraphsC"}



Microsoft Word Objects (Tables)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"wotocObjectModelTablesC"}



OLE Programmatic Identifiers (Microsoft Word)

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womscProgrammaticIdentifiersC;vafctCreateObject;vafctGetObject;OLE Programmatic Identifiers"}

You use an OLE programmatic identifier (sometimes called a ProgID) to create an Automation object. To create a Microsoft Word **Application** object, use "Word.Application.8." To create a Microsoft Word **Document** object, use "Word.Document.8." Using "Word.Application" or "Word.Document" with no version number creates an object in the most recent version of Microsoft Word available on the machine where the macro is running.

Help Topic Not Available

The Help topic cannot be displayed because Visual Basic for Applications Help cannot be found or was not installed.

To install Visual Basic for Applications Help

1. Run Microsoft Office 97 Setup, and click **Add/Remove**.
2. Click **Microsoft Word**, and then click **Change Option**.
3. Click **Help**, and then click **Change Option**.
4. Make sure that the **Help for Visual Basic** check box is selected.
5. Continue with Setup.

Help Topic Not Available

The Help topic cannot be displayed because Microsoft Office Visual Basic Help cannot be found or was not installed.

To install Microsoft Office Visual Basic Help

1. Run Microsoft Office 97 Setup, and click **Add/Remove**.
2. Click **Microsoft Word**, and then click **Change Option**.
3. Click **Help**, and then click **Change Option**.
4. Make sure that the **Help for Visual Basic** check box is selected.
5. Continue with Setup.

Help Topic Not Available

The Help topic cannot be displayed because Microsoft Word Help cannot be found or was not installed.

To install Microsoft Word Help

1. Run Microsoft Office 97 Setup, and click **Add/Remove**.
2. Click **Microsoft Word**, and then click **Change Option**.
3. Click **Help**, and then click **Change Option**.
4. Make sure that the **Help for Microsoft Word** check box is selected.
5. Continue with Setup.

ConvertToFrame Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToFrameC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToFrameX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToFrameA"}

Converts the specified shape to a frame. Returns a **Frame** object that represents the new frame.

Syntax

expression.**ConvertToFrame**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Remarks

Shapes that don't support attached text cannot be converted to frames. For pictures, OLE objects, and ActiveX controls, use the **ConvertToInlineShape** method.

If you use this method on a **ShapeRange** object that contains more than one shape, an error occurs.

In Word 97, frames have been replaced by text boxes.

ConvertToFrame Method Example

This example creates a text box using the selected text, and then it converts the text box to a frame.

```
If Selection.Type = wdSelectionNormal Then
    Selection.CreateTextbox
    Selection.ShapeRange.ConvertToFrame
End If
```

ConvertToInlineShape Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthConvertToInlineShapeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthConvertToInlineShapeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthConvertToInlineShapeA"}

Converts the specified shape in the drawing layer of a document to an inline shape in the text layer. You can convert only shapes that represent pictures, OLE objects, or ActiveX controls. This method returns an **InlineShape** object that represents the picture or OLE object.

Syntax

expression.**ConvertToInlineShape**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Remarks

Shapes that support attached text cannot be converted to inline shapes. For these shapes, use the **ConvertToFrame** method.

If you use this method on a **ShapeRange** object that contains more than one shape, an error occurs.

ConvertToInlineShape Method Example

This example converts each picture in MyDoc.doc to an inline shape.

```
For Each s In Documents("MyDoc.doc").Shapes
    If s.Type = msoPicture Then
        s.ConvertToInlineShape
    End If
Next s
```

SetShapesDefaultProperties Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSetShapesDefaultPropertiesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSetShapesDefaultPropertiesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSetShapesDefaultPropertiesA"}

Applies the formatting of the specified shape to a default shape for that document. New shapes inherit many of their attributes from the default shape.

Syntax

expression.**SetShapesDefaultProperties**

expression Required. An expression that returns a **Shape** or **ShapeRange** object.

Remarks

Using this method is equivalent to using the **Set AutoShape Defaults** command on the **Draw** menu on the **Drawing** toolbar.

SetShapesDefaultProperties Method 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 = ActiveDocument
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
        .SetShapesDefaultProperties     ' Sets formatting for default
shapes
    End With
    .AddShape msoShapeRectangle, 90, 90, 40, 30  ' New shape has default
formatting
End With
```

WrapFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWrapFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWrapFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWrapFormatA"}

Returns a **WrapFormat** object that contains the properties for wrapping text around the specified shape or shape range. Read-only.

WrapFormat Property Example

This example adds an oval to the active document and specifies that the document text wrap around the left and right sides of the square that circumscribes the oval. The example sets a 0.1-inch margin between the document text and the top, bottom, left side, and right side of the square.

```
Set myOval = ActiveDocument.Shapes.AddShape(msoShapeOval, 36, 36, 90, 50)
With myOval.WrapFormat
    .Type = wdWrapSquare
    .Side = wdWrapBoth
    .DistanceTop = InchesToPoints(0.1)
    .DistanceBottom = InchesToPoints(0.1)
    .DistanceLeft = InchesToPoints(0.1)
    .DistanceRight = InchesToPoints(0.1)
End With
```

Anchor Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproAnchorC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproAnchorX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproAnchorA"}

Returns a **Range** object that represents the anchoring range for the specified shape or shape range. Read-only.

Remarks

All **Shape** objects are anchored to a range of text but can be positioned anywhere on the page that contains the anchor. If you specify the anchoring range when you create a shape, the anchor is positioned at the beginning of the first paragraph that contains the anchoring range. If you don't specify the anchoring range, the anchoring range is selected automatically and the shape is positioned relative to the top and left edges of the page.

The shape will always remain on the same page as its anchor. If the **LockAnchor** property for the shape is set to **True**, you cannot drag the anchor from its position on the page.

If you use this property on a **ShapeRange** object that contains more than one shape, an error occurs.

Anchor Property Example

This example selects the paragraph that the first shape in the active document is anchored to.

```
ActiveDocument.Shapes(1).Anchor.Paragraphs(1).Range.Select
```

BreakForwardLink Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthBreakForwardLinkC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthBreakForwardLinkX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthBreakForwardLinkA"}

Breaks the forward link for the specified text frame, if such a link exists.

Syntax

expression.**BreakForwardLink**

expression Required. An expression that returns a **TextFrame** object.

Remarks

Applying this method to a shape in the middle of a chain of shapes with linked text frames will break the chain, leaving two sets of linked shapes. All of the text, however, will remain in the first series of linked shapes.

BreakForwardLink Method Example

This example creates a new document adds a chain of three linked text boxes to it, and then breaks the link after the second text box.

```
Documents.Add
Set myTB = ActiveDocument.Shapes.AddTextbox _
    (Orientation:=msoTextOrientationHorizontal, _
    Left:=InchesToPoints(1.5), _
    Top:=InchesToPoints(0.5), _
    Width:=InchesToPoints(1), _
    Height:=InchesToPoints(0.5))
myTB.TextFrame.TextRange = "This is some text. This is some more text. This
is even more text."
Set myTB2 = ActiveDocument.Shapes.AddTextbox _
    (Orientation:=msoTextOrientationHorizontal, _
    Left:=InchesToPoints(1.5), _
    Top:=InchesToPoints(1.5), _
    Width:=InchesToPoints(1), _
    Height:=InchesToPoints(0.5))
Set myTB3 = ActiveDocument.Shapes.AddTextbox _
    (Orientation:=msoTextOrientationHorizontal, _
    Left:=InchesToPoints(1.5), _
    Top:=InchesToPoints(2.5), _
    Width:=InchesToPoints(1), _
    Height:=InchesToPoints(0.5))
myTB.TextFrame.Next = myTB2.TextFrame
myTB2.TextFrame.Next = myTB3.TextFrame
MsgBox "Textboxes 1, 2, and 3 are linked."
myTB2.TextFrame.BreakForwardLink
```

ContainingRange Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproContainingRangeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproContainingRangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproContainingRangeA"}

Returns a **Range** object that represents the entire story in a series of shapes with linked text frames that the specified text frame belongs to. Read-only.

ContainingRange Property Example

This example checks the spelling in TextBox 1 and any other text in text frames that are linked to TextBox 1.

```
Set myStory = ActiveDocument.Shapes("TextBox 1").TextFrame.ContainingRange
myStory.CheckSpelling
```

Overflowing Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOverflowingC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOverflowingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOverflowingA"}

True if the text inside the specified text frame doesn't all fit within the frame. Read-only **Boolean**.

Overflowing Property Example

This example checks to see whether the text in MyTextBox is overflowing its text frame. If so, the example adds another text box and links the two text boxes so that the text flows into the next one.

```
Set myTBox = ActiveDocument.Shapes("MyTextBox")
If myTBox.TextFrame.Overflowing = True Then
    Set nextTBox = ActiveDocument.Shapes. _
        AddTextbox(msoTextOrientationHorizontal, 72, 72, 100, 200)
    MyTBox.TextFrame.Next = nextTBox.TextFrame
End If
```

ValidLinkTarget Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthValidLinkTargetC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthValidLinkTargetX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthValidLinkTargetA"}

Determines whether the text frame of one shape can be linked to the text frame of another shape. Returns **True** if ***TargetTextFrame*** is a valid target. Returns **False** if ***TargetTextFrame*** already contains text or is already linked, or if the shape doesn't support attached text.

Syntax

expression.**ValidLinkTarget(*TargetTextFrame*)**

expression Required. An expression that returns a **TextFrame** object.

TargetTextFrame Required **TextFrame** object. The target text frame that you'd like to link the text frame returned by *expression* to.

ValidLinkTarget Method Example

This example checks to see whether the text frames for the first and second shapes in the active document can be linked to one another. If so, the example links the two text frames.

```
Set myFrame1 = ActiveDocument.Shapes(1).TextFrame
Set myFrame2 = ActiveDocument.Shapes(2).TextFrame
If myFrame1.ValidLinkTarget(myFrame2) = True Then
    myFrame1.Next = myFrame2
End If
```

ScaleWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproScaleWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproScaleWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproScaleWidthA"}

Scales the width of the specified inline shape, relative to its original size. Read/write **Single**.

ScaleWidth Property Example

This example sets the height and width of the first inline shape in the active document to 150 percent of the shape's original height and width.

```
With ActiveDocument.InlineShapes(1)
    .ScaleHeight = 150
    .ScaleWidth = 150
End With
```

ScaleHeight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproScaleHeightC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproScaleHeightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproScaleHeightA"}

Scales the height of the specified inline shape, relative to its original size. Read/write **Single**.

ScaleHeight Property Example

This example sets the height and width of the first inline shape in the active document to 150 percent of the shape's original height and width.

```
With ActiveDocument.InlineShapes(1)
    .ScaleHeight = 150
    .ScaleWidth = 150
End With
```

Arrange Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthArrangeC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthArrangeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthArrangeA "}

Arranges all open document windows in the application workspace.

Syntax

expression.**Arrange(*ArrangeStyle*)**

expression An expression that returns a **Windows** object.

ArrangeStyle Optional **Variant**. The window arrangement. Can be either of the following **WdArrangeStyle** constants: **wdIcons** or **wdTiled**.

Arrange Method Example

This example arranges all open windows so that they don't overlap.

```
Windows.Arrange ArrangeStyle:=wdTiled
```

This example minimizes all open windows and then arranges the minimized windows.

```
For Each aWindow In Windows
    aWindow.Activate
    aWindow.WindowState = wdWindowStateMinimize
Next aWindow
Windows.Arrange ArrangeStyle:=wdIcons
```

Close Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCloseC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCloseX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCloseA "}

Syntax 1: Closes the specified document or documents.

Syntax 2: Closes the specified window.

Syntax 3: Closes the specified pane or task.

Syntax 1

expression.**Close**(***SaveChanges, OriginalFormat, RouteDocument***)

Syntax 2

expression.**Close**(***SaveChanges, RouteDocument***)

Syntax 3

expression.**Close**

expression Syntax 1: Required. An expression that returns a **Document** or **Documents** object.

Syntax 2: Required. An expression that returns a **Window** object.

Syntax 3: Required. An expression that returns a **Pane** or **Task** object.

SaveChanges Optional **Variant**. Specifies the save action for the document. Can be one of the following **WdSaveOptions** constants: **wdDoNotSaveChanges**, **wdPromptToSaveChanges**, or **wdSaveChanges**.

OriginalFormat Optional **Variant**. Specifies the save format for the document. Can be one of the following **WdOriginalFormat** constants: **wdOriginalDocumentFormat**, **wdPromptUser**, or **wdWordDocument**.

RouteDocument Optional **Variant**. **True** to route the document to the next recipient. If the document doesn't have a routing slip attached, this argument is ignored.

Close Method Example

This example closes the active window and saves the document that's displayed in it.

```
ActiveWindow.Close SaveChanges:=wdSaveChanges
```

This example prompts the user to save the active document before closing it. If the user clicks **Cancel**, error 4198 (command failed) is trapped and a message is displayed.

```
On Error GoTo errorHandler
ActiveDocument.Close SaveChanges:=wdPromptToSaveChanges,
OriginalFormat:=wdPromptUser
errorHandler:
If Err = 4198 Then MsgBox "Document was not closed"
```

This example activates Microsoft Excel and then closes it.

```
For Each myTask In Tasks
    If InStr(myTask.Name, "Microsoft Excel") > 0 Then
        myTask.Activate
        myTask.Close
    End If
Next myTask
```

This example closes the active pane if the active window is split.

```
If ActiveWindow.Panes.Count >= 2 Then ActiveWindow.ActivePane.Close
```

DisplayHorizontalScrollBar Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayHorizontalScrollBarC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayHorizontalScrollBarX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayHorizontalScrollBarA "}

True if a horizontal scroll bar is displayed for the specified window. Read/write **Boolean**.

DisplayHorizontalScrollBar Property Example

This example displays vertical and horizontal scroll bars for the active window.

```
With ActiveWindow
    .DisplayHorizontalScrollBar = True
    .DisplayVerticalScrollBar = True
End With
```

This example toggles the horizontal scroll bar of the window for Document1.

```
Set aWin = Windows("Document1")
aWin.DisplayHorizontalScrollBar = Not aWin.DisplayHorizontalScrollBar
```

DisplayRulers Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayRulersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayRulersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayRulersA "}

True if rulers are displayed for the specified window or pane. Equivalent to the **Ruler** command on the **View** menu. Read/write **Boolean**.

Note If **DisplayRulers** is **False**, the horizontal and vertical rulers won't be displayed, regardless of the state of the **DisplayVerticalRuler** property.

DisplayRulers Property Example

This example toggles the ruler display for the active window.

```
ActiveWindow.DisplayRulers = Not ActiveWindow.DisplayRulers
```

This example switches the window to page layout view and displays the horizontal and vertical rulers.

```
With ActiveWindow
    .View.Type = wdPageView
    .DisplayVerticalRuler = True
    .DisplayRulers = True
End With
```

DisplayVerticalRuler Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayVerticalRulerC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayVerticalRulerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayVerticalRulerA "}

True if a vertical ruler is displayed for the specified window or pane. Read/write **Boolean**.

Note A vertical ruler appears only in page layout view, and only if the **DisplayRulers** property is set to **True**.

DisplayVerticalRuler Property Example

This example switches each window in the **Windows** collection to page layout view and displays the horizontal and vertical rulers.

```
For Each myWindow In Windows
    With myWindow
        .View.Type = wdPageView
        .DisplayRulers = True
        .DisplayVerticalRuler = True
    End With
Next myWindow
```

This example hides the horizontal and vertical rulers for the active window.

```
With ActiveWindow
    .DisplayVerticalRuler = False
    .DisplayRulers = False
End With
```

DisplayVerticalScrollBar Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDisplayVerticalScrollBarC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDisplayVerticalScrollBarX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDisplayVerticalScrollBarA "}

True if a vertical scroll bar is displayed for the specified window. Read/write **Boolean**.

DisplayVerticalScrollBar Property Example

This example displays the vertical and horizontal scroll bars for each window in the **Windows** collection.

```
For Each myWindow In Windows
    myWindow.DisplayVerticalScrollBar = True
    myWindow.DisplayHorizontalScrollBar = True
Next myWindow
```

This example toggles the vertical scroll bar for the active window.

```
Set aWindow = ActiveWindow
aWindow.DisplayVerticalScrollBar = Not aWindow.DisplayVerticalScrollBar
```

LargeScroll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthLargeScrollC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthLargeScrollX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthLargeScrollA "}

Scrolls a window or pane by the specified number of screens. This method is equivalent to clicking just before or just after the scroll boxes on the horizontal and vertical scroll bars.

Syntax

expression.**LargeScroll(*Down, Up, ToRight, ToLeft*)**

expression Required. An expression that returns a **Pane** or **Window** object.

Down Optional **Variant**. The number of screens to scroll the window down.

Up Optional **Variant**. The number of screens to scroll the window up.

ToRight Optional **Variant**. The number of screens to scroll the window to the right.

ToLeft Optional **Variant**. The number of screens to scroll the window to the left.

Remarks

If ***Down*** and ***Up*** are both specified, the window is scrolled by the difference of the arguments. For example, if ***Down*** is 2 and ***Up*** is 4, the window is scrolled up two screens. Similarly, if ***ToLeft*** and ***ToRight*** are both specified, the window is scrolled by the difference of the arguments.

Any of these arguments can be a negative number. If no arguments are specified, the window is scrolled down one screen.

LargeScroll Method Example

This example scrolls the active window down one screen.

```
ActiveWindow.LargeScroll Down:=1
```

This example splits the active window and then scrolls up two screens and to the right one screen.

```
With ActiveWindow
    .Split = True
    .LargeScroll Up:=2, ToRight:=1
End With
```

NewWindow Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNewWindowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNewWindowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNewWindowA "}

Opens a new window with the same document as the specified window. Returns a **Window** object.

Note A colon (:) and a number appear in the window caption when more than one window is open for a document.

Syntax

expression.**NewWindow**

expression Required. An expression that returns an **Application** or **Window** object.

Remarks

If the **NewWindow** method is used with the **Application** object, a new window is opened for the active window. The following two instructions are functionally equivalent.

```
Set myWindow = ActiveWindow.NewWindow
Set myWindow = NewWindow
```

NewWindow Method Example

This example posts a message that indicates the number of windows that exist before and after you open a new window for Document1.

```
MsgBox Windows.Count & " windows open"
Windows("Document1").NewWindow
MsgBox Windows.Count & " windows open"
```

This example opens a new window, arranges all the open windows, closes the new window, and then rearranges the open windows.

```
Set myWindow = NewWindow
Windows.Arrange ArrangeStyle:=wdTiled
myWindow.Close
Windows.Arrange ArrangeStyle:=wdTiled
```

Panes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPanesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPanesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPanesA "}

Returns a **Panes** collection that represents all the window panes for the specified window. Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Panes Property Example

This example splits the active window in half.

```
If ActiveWindow.Panes.Count = 1 Then ActiveWindow.Panes.Add
```

This example activates the first pane in the window for Document2.

```
Windows("Document2").Panes(1).Activate
```

SmallScroll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthSmallScrollC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthSmallScrollX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthSmallScrollA "}

Scrolls a window or pane by the specified number of lines. This method is equivalent to clicking the scroll arrows on the horizontal and vertical scroll bars.

Syntax

expression.**SmallScroll(*Down, Up, ToRight, ToLeft*)**

expression Required. An expression that returns a **Pane** or **Window** object.

Down Optional **Variant**. The number of lines to scroll the window down. A "line" corresponds to the distance scrolled by clicking the down scroll arrow on the vertical scroll bar once.

Up Optional **Variant**. The number of lines to scroll the window up. A "line" corresponds to the distance scrolled by clicking the up scroll arrow on the vertical scroll bar once.

ToRight Optional **Variant**. The number of lines to scroll the window to the right. A "line" corresponds to the distance scrolled by clicking the right scroll arrow on the horizontal scroll bar once.

ToLeft Optional **Variant**. The number of lines to scroll the window to the left. A "line" corresponds to the distance scrolled by clicking the left scroll arrow on the horizontal scroll bar once.

Remarks

If ***Down*** and ***Up*** are both specified, the window is scrolled by the difference of the arguments. For example, if ***Down*** is 3 and ***Up*** is 6, the window is scrolled up three lines. Similarly, if ***ToLeft*** and ***ToRight*** are both specified, the window is scrolled by the difference of the arguments.

Any of these arguments can be a negative number. If no arguments are specified, the window is scrolled down by one line.

SmallScroll Method Example

This example scrolls the active window down one line.

```
ActiveWindow.SmallScroll Down:=1
```

This example splits the active window and then scrolls up and over to the left.

```
With ActiveWindow
    .Split = True
    .SmallScroll Up:=5, ToLeft:=5
End With
```

SplitVertical Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSplitVerticalC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSplitVerticalX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSplitVerticalA "}

Returns or sets the vertical split percentage for the specified window. Read/write **Long**.

Note To remove the split, set this property to zero (0) or set the **Split** property to **False**.

SplitVertical Property Example

This example splits the active window so that the top pane occupies 70 percent of the window.

```
ActiveWindow.SplitVertical = 70
```

This example splits the window for Document1 in half vertically.

```
Windows("Document1").SplitVertical = 50
```

View Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproViewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproViewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproViewA "}

Returns a **View** object that represents the view for the specified window or pane. Read-only.

View Property Example

This example switches the active window to full-screen view.

```
ActiveWindow.View.FullScreen = True
```

This example shows all nonprinting characters for panes associated with the first window in the **Windows** collection.

```
For Each myPane In Windows(1).Panes
    myPane.View.ShowAll = True
Next myPane
```

This example sets view options for each window in the **Windows** collection.

```
For Each myWindow In Windows
    With myWindow.View
        .ShowTabs = True
        .ShowParagraphs = True
        .Type = wdNormalView
    End With
Next myWindow
```

Windows Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWindowsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWindowsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWindowsA "}

Application object: Returns a **Windows** collection that represents all document windows. The collection corresponds to the window names that appear at the bottom of the **Window** menu. Read-only.

Document object: Returns a **Windows** collection that represents all windows for the specified document (for example, Sales.doc:1 and Sales.doc:2). Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Windows Property Example

This example displays the number of windows for the active document, both before and after the **NewWindow** method is run.

```
MsgBox Prompt:= ActiveDocument.Windows.Count & " window(s)", _
    Title:= ActiveDocument.Name
ActiveWindow.NewWindow
MsgBox Prompt:= ActiveDocument.Windows.Count & " windows", _
    Title:= ActiveDocument.Name
```

This example arranges all open windows so that they don't overlap.

```
Windows.Arrange ArrangeStyle:=wdTiled
```

WindowState Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWindowStateC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWindowStateX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWindowStateA "}

Returns or sets the state of the specified document window or task window. Can be one the following **WdWindowState** constants: **wdWindowStateMaximize**, **wdWindowStateMinimize**, or **wdWindowStateNormal**. Read/write **Long**.

Remarks

The **wdWindowStateNormal** constant indicates a window that's not maximized or minimized. The state of an inactive window cannot be set. Use the **Activate** method to activate a window prior to setting the window state.

WindowState Property Example

This example maximizes the active window if it's not maximized or minimized.

```
If ActiveWindow.WindowState = wdWindowStateNormal Then _
    ActiveWindow.WindowState = wdWindowStateMaximize
```

This example minimizes the Microsoft Excel application window.

```
For Each myTask In Tasks
    If InStr(myTask.Name, "Microsoft Excel") > 0 Then
        myTask.Activate
        myTask.WindowState = wdWindowStateMinimize
    End If
Next myTask
```

Zoom Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproZoomC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproZoomX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproZoomA "}

Returns a **Zoom** object that represents the magnification for the specified view. Read-only.

Zoom Property Example

This example changes the zoom percentage of each open window to 100 percent.

```
For Each myWindow In Windows
    myWindow.View.Zoom.Percentage = 100
Next myWindow
```

This example changes the zoom percentage of the active window so that the entire width of the text is visible.

```
ActiveWindow.View.Zoom.PageFit = wdPageFitBestFit
```

WrapToWindow Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWrapToWindowC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWrapToWindowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWrapToWindowA "}

True if lines wrap at the right edge of the document window rather than at the right margin or the right column boundary. Read/write **Boolean**.

Note This property has no effect in page layout or online layout view.

WrapToWindow Property Example

This example wraps the text to fit within the active window.

```
With ActiveWindow.View
    .Type = wdNormalView
    .WrapToWindow = True
End With
```

StyleAreaWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproStyleAreaWidthC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproStyleAreaWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproStyleAreaWidthA "}

Returns or sets the width of the style area, in points. Read/write **Single**.

Note When the **StyleAreaWidth** property is greater than 0 (zero), style names are displayed to the left of the text. The style area isn't visible in page layout view or online layout view.

StyleAreaWidth Property Example

This example switches the active window to normal view and sets the width of the style area to 1 inch.

```
With ActiveWindow
    .View.Type = wdNormalView
    .StyleAreaWidth = InchesToPoints(1)
End With
```

Draft Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDraftC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDraftX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDraftA "}

True if all the text in a window is displayed in the same sans-serif font with minimal formatting to speed up display. Read/write **Boolean**.

Draft Property Example

This example displays the contents of the window for Document1 in the draft font.

```
Windows("Document1").View.Draft = True
```

This example toggles the draft font option for the active window.

```
ActiveWindow.View.Draft = Not ActiveWindow.View.Draft
```

ShowFieldCodes Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowFieldCodesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowFieldCodesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowFieldCodesA "}

True if field codes are displayed. Read/write **Boolean**.

ShowFieldCodes Property Example

This example hides field codes in the window for Document1.

```
Windows("Document1").View.ShowFieldCodes = False
```

This example shows field codes in the first window.

```
Windows(1).View.ShowFieldCodes = True
```

This example toggles field codes in the active window.

```
ActiveWindow.View.ShowFieldCodes = Not ActiveWindow.View.ShowFieldCodes
```

FieldShading Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFieldShadingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFieldShadingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFieldShadingA "}

Returns or sets on-screen shading for form fields. Can be one of the following **WdFieldShading** constants: **wdFieldShadingAlways**, **wdFieldShadingNever**, or **wdFieldShadingWhenSelected**. Read/write **Long**.

FieldShading Property Example

This example enables field shading for all form fields in the active window.

```
ActiveWindow.View.FieldShading = wdFieldShadingAlways
```

FullScreen Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproFullScreenC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproFullScreenX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproFullScreenA "}

True if the window is in full-screen view. Read/write **Boolean**.

FullScreen Property Example

This example switches the active window to full-screen view.

```
ActiveWindow.View.FullScreen = True
```

This example activates the window for Sales.doc and switches out of full-screen view.

```
With Windows("Sales.doc")
    .Activate
    .View.FullScreen = False
End With
```

HorizontalPercentScrolled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproHorizontalPercentScrolledC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproHorizontalPercentScrolledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproHorizontalPercentScrolledA "}

Returns or sets the horizontal scroll position as a percentage of the document width. Read/write **Long**.

HorizontalPercentScrolled Property Example

This example displays the percentage that the active window is scrolled horizontally.

```
MsgBox ActiveWindow.HorizontalPercentScrolled & "%"
```

This example vertically scrolls the active pane of the window for Document1 all the way to the left.

```
With Windows("Document1")
    .Activate
    .ActivePane.HorizontalPercentScrolled = 0
End With
```

Magnifier Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMagnifierC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMagnifierX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMagnifierA "}

True if the pointer is displayed as a magnifying glass in print preview, indicating that the user can click to zoom in on a particular area of the page or zoom out to see an entire page or spread of pages. Read/write **Boolean**.

Note This property generates an error if the view is not print preview.

Magnifier Property Example

This example switches to print preview and changes the pointer to an insertion point.

```
PrintPreview = True
ActiveWindow.View.Magnifier = False
```

NextHeaderFooter Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthNextHeaderFooterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthNextHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthNextHeaderFooterA "}

If the selection is in a header, this method moves to the next header within the current section (for example, from an odd header to an even header) or to the first header in the following section. If the selection is in a footer, this method moves to the next footer.

Note If the selection is in the last header or footer in the last section of the document, or if it's not in a header or footer at all, an error occurs.

Syntax

expression.**NextHeaderFooter**

expression Required. An expression that returns a **View** object.

NextHeaderFooter Method Example

This example displays the first page header in the active document and then switches to the next header. The document needs to be at least two pages long.

```
ActiveDocument.PageSetup.DifferentFirstPageHeaderFooter = True
With ActiveWindow.View
    .Type = wdPageView
    .SeekView =wdSeekFirstPageHeader
    .NextHeaderFooter
End With
```

PreviousHeaderFooter Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPreviousHeaderFooterC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPreviousHeaderFooterX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPreviousHeaderFooterA "}

If the selection is in a header, this method moves to the previous header within the current section (for example, from an even header to an odd header) or to the last header in the previous section. If the selection is in a footer, this method moves to the previous footer.

Note If the selection is in the first header or footer in the first section of the document, or if it's not in a header or footer at all, an error occurs.

Syntax

expression.**PreviousHeaderFooter**

expression Required. An expression that returns a **View** object.

PreviousHeaderFooter Method Example

This example inserts an even section break, switches the active window to page layout view, displays the current header, and then switches to the previous header.

```
Selection.Collapse Direction:=wdCollapseStart
Selection.InsertBreak Type:=wdSectionBreakEvenPage
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageHeader
    .PreviousHeaderFooter
End With
```

SeekView Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSeekViewC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSeekViewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSeekViewA "}

Returns or sets the document element displayed in page layout view. Can be one of the following **WdSeekView** constants: **wdSeekCurrentPageFooter**, **wdSeekCurrentPageHeader**, **wdSeekEndnotes**, **wdSeekEvenPagesFooter**, **wdSeekEvenPagesHeader**, **wdSeekFirstPageFooter**, **wdSeekFirstPageHeader**, **wdSeekFootnotes**, **wdSeekMainDocument**, **wdSeekPrimaryFooter**, or **wdSeekPrimaryHeader**. Read/write **Long**.

Note This property generates an error if the view is not page layout view.

SeekView Property Example

If the active document has footnotes, this example displays footnotes in page layout view.

```
If ActiveDocument.Footnotes.Count >= 1 Then
    With ActiveWindow.View
        .Type = wdPageView
        .SeekView = wdSeekFootnotes
    End With
End If
```

This example shows the first page footer for the current section.

```
ActiveDocument.PageSetup.DifferentFirstPageHeaderFooter = True
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekFirstPageFooter
End With
```

If the selection is in a footnote area or endnote area in page layout view, this example switches to the main document.

```
Set myView = ActiveWindow.View
If myView.SeekView = wdSeekFootnotes Or _
    myView.SeekView = wdSeekEndnotes Then
    myView.SeekView = wdSeekMainDocument
End If
```

ShowAll Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowAllC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowAllX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowAllA "}

True if all nonprinting characters – such as hidden text, tab marks, space marks, and paragraph marks – are displayed. Read/write **Boolean**.

ShowAll Property Example

This example displays all nonprinting characters in the active window.

```
ActiveWindow.View.ShowAll = True
```

This example toggles the display of nonprinting characters in the first window.

```
Windows(1).View.ShowAll = Not Windows(1).View.ShowAll
```

ShowAllHeadings Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShowAllHeadingsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShowAllHeadingsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShowAllHeadingsA "}

Toggles between showing all text (headings and body text) and showing only headings.

Note This method generates an error if the view isn't outline view or master document view.

Syntax

expression.**ShowAllHeadings**

expression Required. An expression that returns a **View** object.

ShowAllHeadings Method Example

This example uses the **ShowHeading** method to show all headings (without any body text) and then toggles the display to show all text (headings and body text) in outline view.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .ShowHeading 9
    .ShowAllHeadings
End With
```

ShowAnimation Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowAnimationC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowAnimationX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowAnimationA "}

True if text animation is displayed. Read/write **Boolean**.

ShowAnimation Property Example

This example turns on text animation in the active window and then applies sparkle-text animation to the selection.

```
ActiveWindow.View.ShowAnimation = True
Selection.Font.Animation = wdAnimationSparkleText
```

This example turns off font animation in all open windows.

```
For Each aWindow In Windows
    aWindow.View.ShowAnimation = False
Next aWindow
```

ShowBookmarks Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowBookmarksC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowBookmarksX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowBookmarksA "}

True if square brackets are displayed at the beginning and end of each bookmark. Read/write **Boolean**.

ShowBookmarks Property Example

This example displays square brackets around bookmarks in all windows.

```
For Each aWindow In Windows
    aWindow.View.ShowBookmarks = True
Next aWindow
```

This example marks the selection with a bookmark, displays square brackets around each bookmark in the active document, and then collapses the selection.

```
ActiveDocument.Bookmarks.Add Range:=Selection.Range, Name:="temp"
ActiveWindow.View.ShowBookmarks = True
Selection.Collapse Direction:=wdCollapseStart
```

ShowDrawings Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowDrawingsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowDrawingsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowDrawingsA "}

True if objects created with the drawing tools are displayed in page layout view. Read/write **Boolean**.

ShowDrawings Property Example

This example switches the active window to page layout view and displays objects created with the drawing tools.

```
With ActiveWindow.View
    .Type = wdPageView
    .ShowDrawings = True
End With
```

ShowFirstLineOnly Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowFirstLineOnlyC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowFirstLineOnlyX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowFirstLineOnlyA "}

True if only the first line of body text is shown in outline view. Read/write **Boolean**.

Note This property generates an error if the view isn't outline view or master document view.

ShowFirstLineOnly Property Example

This example switches the active window to outline view and hides all but the first line of body text.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .ShowFirstLineOnly = True
End With
```

ShowFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowFormatC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowFormatA "}

True if character formatting is visible in outline view. Read/write **Boolean**.

Note This property generates an error if the view isn't outline view or master document view.

ShowFormat Property Example

This example switches the active window to outline view and shows character formatting.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .ShowFormat = True
End With
```

ShowHeading Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthShowHeadingC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthShowHeadingX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthShowHeadingA "}

Shows all headings up to the specified heading level and hides subordinate headings and body text.

Note This method generates an error if the view isn't outline view or master document view.

Syntax

expression.**ShowHeading(*Level*)**

expression Required. An expression that returns a **View** object.

Level Required **Long**. The outline heading level (a number from 1 to 9).

ShowHeading Method Example

This example switches the active window to outline view and displays all text that's formatted with the Heading 1 style. Body text and all other types of headings are hidden.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .ShowHeading 1
End With
```

This example switches the window for Document1 to outline view and displays all text that's formatted with the Heading 1, Heading 2, or Heading 3 style.

```
With Windows("Document1").View
    .Type = wdOutlineView
    .ShowHeading 3
End With
```

ShowHiddenText Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowHiddenTextC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowHiddenTextX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowHiddenTextA "}

True if text formatted as hidden text is displayed. Read/write **Boolean**.

ShowHiddenText Property Example

This example hides text formatted as hidden text in each window.

```
For Each myWindow In Windows
    myWindow.View.ShowHiddenText = False
Next myWindow
```

This example toggles the display of hidden text.

```
ActiveWindow.View.ShowHiddenText = Not ActiveWindow.View.ShowHiddenText
```

ShowHighlight Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowHighlightC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowHighlightX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowHighlightA "}

True if highlight formatting is displayed and printed with a document. Read/write **Boolean**.

ShowHighlight Property Example

This example toggles the display of highlighting in the active document.

```
ActiveWindow.View.ShowHighlight = Not ActiveWindow.View.ShowHighlight
```

This example prints the active document without highlight formatting.

```
With ActiveDocument
    .ActiveWindow.View.ShowHighlight = False
    .PrintOut
End With
```

ShowHyphens Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowHyphensC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowHyphensX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowHyphensA "}

True if optional hyphens are displayed. An optional hyphen indicates where to break a word when it falls at the end of a line. Read/write **Boolean**.

ShowHyphens Property Example

This example inserts an optional hyphen before the selection and then displays optional hyphens in the active window.

```
Selection.InsertBefore Chr(31)
ActiveWindow.View.ShowHyphens = True
```

ShowObjectAnchors Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowObjectAnchorsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowObjectAnchorsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowObjectAnchorsA "}

True if object anchors are displayed next to items that can be positioned in page layout view. Read/write **Boolean**.

ShowObjectAnchors Property Example

This example adds a frame around the selection, switches the active window to page layout view, and shows object anchors for framed objects.

```
Selection.Frames.Add(Range:=Selection.Range).LockAnchor = True
With ActiveWindow.View
    .Type = wdPageView
    .ShowObjectAnchors = True
End With
```

ShowParagraphs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowParagraphsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowParagraphsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowParagraphsA "}

True if paragraph marks are displayed. Read/write **Boolean**.

ShowParagraphs Property Example

This example hides paragraph marks in the active window.

```
ActiveWindow.View.ShowParagraphs = False
```

ShowPicturePlaceHolders Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowPicturePlaceHoldersC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowPicturePlaceHoldersX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowPicturePlaceHoldersA "}

True if blank boxes are displayed as placeholders for pictures. Read/write **Boolean**.

ShowPicturePlaceHolders Property Example

This example inserts a picture in the active document and displays picture placeholders in the active window.

```
Selection.Collapse Direction:=wdCollapseStart
ActiveDocument.InlineShapes.AddPicture Range:=Selection.Range, _
    FileName:="C:\Windows\Bubbles.bmp"
ActiveWindow.View.ShowPicturePlaceHolders = True
```

ShowSpaces Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowSpacesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowSpacesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowSpacesA "}

True if space characters are displayed. Read/write **Boolean**.

ShowSpaces Property Example

This example inserts spaces before the selection and displays space characters in the active window.

```
Selection.InsertBefore "    "
ActiveWindow.View.ShowSpaces = True
```

ShowTabs Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowTabsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowTabsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowTabsA "}

True if tab characters are displayed. Read/write **Boolean**.

ShowTabs Property Example

This example inserts a tab before the selection and displays tab characters in the window for Document2.

```
With Windows("Document2")
    .Activate
    .View.ShowTabs = True
End With
Selection.InsertBefore vbTab
Selection.Collapse Direction:=wdCollapseEnd
```

This example splits the active window, shows tab characters in the first pane, and hides tab characters in the second pane.

```
With ActiveWindow
    .Split = True
    .Panes(1).View.ShowTabs = True
    .Panes(2).View.ShowTabs = False
End With
```

ShowTextBoundaries Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowTextBoundariesC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowTextBoundariesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowTextBoundariesA "}

True if dotted lines are displayed around page margins, text columns, objects, and frames in page layout view. Read/write **Boolean**.

ShowTextBoundaries Property Example

This example switches the active window to page view and displays text boundary lines.

```
With ActiveWindow.View
    .Type = wdPageView
    .ShowTextBoundaries = True
End With
```

SplitSpecial Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproSplitSpecialC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproSplitSpecialX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproSplitSpecialA "}

Returns or sets the active window pane. Read/write **Long**.

Can be one of the following **WdSpecialPane** constants:

wdPaneComments
wdCurrentPageFooter
wdPaneCurrentPageHeader
wdPaneEndnoteContinuationNotice
wdPaneEndnoteContinuationSeparator
wdPaneEndnotes
wdPaneEndnoteSeparator
wdPaneEvenPagesFooter
wdPaneEvenPagesHeader
wdPaneFirstPageFooter
wdPaneFirstPageHeader
wdPaneFootnoteContinuationNotice
wdPaneFootnoteContinuationSeparator
wdPaneFootnotes
wdPaneFootnoteSeparator
wdPaneNone
wdPanePrimaryFooter
wdPanePrimaryHeader

SplitSpecial Property Example

This example displays the primary footer in a separate pane in the active window.

```
ActiveWindow.View.SplitSpecial = wdPanePrimaryFooter
```

This example adds a footnote to the active document and displays all the footnotes in a separate pane in the active window.

```
ActiveDocument.Footnotes.Add Range:=Selection.Range, Text:="Footnote text"
With ActiveWindow.View
    .Type = wdNormalView
    .SplitSpecial = wdPaneFootnotes
End With
```

VerticalPercentScrolled Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproVerticalPercentScrolledC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproVerticalPercentScrolledX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproVerticalPercentScrolledA "}

Returns or sets the vertical scroll position as a percentage of the document length. Read/write **Long**.

VerticalPercentScrolled Property Example

This example displays the percentage that the active window is scrolled vertically.

```
MsgBox ActiveWindow.VerticalPercentScrolled & "%"
```

This example scrolls the active window vertically by 10 percent.

```
Set aWindow = ActiveWindow
aWindow.VerticalPercentScrolled = aWindow.VerticalPercentScrolled + 10
```

This example scrolls the active pane of the window for Document1 vertically to the end.

```
With Windows("Document1")
    .Activate
    .ActivePane.VerticalPercentScrolled = 100
End With
```

WindowNumber Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproWindowNumberC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproWindowNumberX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproWindowNumberA "}

Returns the window number of the document displayed in the specified window. For example, if the caption of the window is "Sales.doc:2," this property returns the number 2. Read-only **Long**.

Note Use the **Index** property to return the number of the specified window in the **Windows** collection.

WindowNumber Property Example

This example retrieves the window number of the active window, opens a new window, and then activates the original window.

```
windowNum = ActiveDocument.ActiveWindow.WindowNumber
NewWindow
ActiveDocument.Windows(windowNum).Activate
```

DocumentMap Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDocumentMapC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDocumentMapX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDocumentMapA"}

True if the document map is visible. Read/write **Boolean**.

DocumentMap Property Example

This example toggles the document map for the active window.

```
ActiveWindow.DocumentMap = Not ActiveWindow.DocumentMap
```

This example displays the document map in the window for Sales.doc.

```
Set myDoc = Documents.Open(FileName:="C:\Documents\Sales.doc")
myDoc.ActiveWindow.DocumentMap = True
```

TableGridlines Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproTableGridlinesC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproTableGridlinesX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproTableGridlinesA"}

True if table gridlines are displayed. Read/write **Boolean**.

TableGridlines Property Example

This example displays table gridlines in the active window.

```
ActiveWindow.View.TableGridlines = True
```

This example shows table gridlines for the panes associated with window one in the **Windows** collection.

```
For Each myPane In Windows(1).Panes
    myPane.View.TableGridlines = True
Next myPane
```

CollapseOutline Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthCollapseOutlineC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthCollapseOutlineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthCollapseOutlineA "}

Collapses the text under the selection or the specified range by one heading level.

Note If the document isn't in outline or master document view, an error occurs.

Syntax

expression.**CollapseOutline(*Range*)**

expression Required. An expression that returns a **View** object.

Range Optional **Range** object. The range of paragraphs to be collapsed. If this argument is omitted, the entire selection is collapsed.

CollapseOutline Method Example

This example applies the Heading 2 style to the second paragrah in the active document, switches the active window to outline view, and collapses the text under the second paragraph in the document.

```
ActiveDocument.Paragraphs(2).Style = wdStyleHeading2
With ActiveWindow.View
    .Type = wdOutlineView
    .CollapseOutline Range:=ActiveDocument.Paragraphs(2).Range
End With
```

This example collapses every heading in the document by one level.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .CollapseOutline Range:=ActiveDocument.Content
End With
```

ExpandOutline Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthExpandOutlineC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthExpandOutlineX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthExpandOutlineA "}

Expands the text under the selection or the specified range by one heading level.

Note If the document isn't in outline or master document view, an error occurs.

Syntax

expression.**ExpandOutline(*Range*)**

expression Required. An expression that returns a **View** object.

Range Optional **Range** object. The range of paragraphs to be expanded. If this argument is omitted, the entire selection is expanded.

ExpandOutline Method Example

This example expands every heading in the document by one level.

```
With ActiveWindow.View
    .Type = wdOutlineView
    .ExpandOutline Range:=ActiveDocument.Content
End With
```

This example expands the active paragraph in the Document2 window.

```
With Windows("Document2")
    .Activate
    .View.Type = wdOutlineView
    .View.ExpandOutline
End With
```

AutoScroll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoScrollC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoScrollX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoScrollA"}

Scrolls automatically through the specified pane.

Note This method continues to run until you stop it manually by pressing a key or clicking the mouse.

Syntax

expression.**AutoScroll(*Velocity*)**

expression Required. An expression that returns a **Pane** object.

Velocity Required **Long**. The speed for scrolling. Can be a number from – 100 through 100. Use – 100 for full-speed backward scrolling, and use 100 for full-speed forward scrolling.

AutoScroll Method Example

This example scrolls backward through the active window pane slowly.

```
ActiveWindow.ActivePane.AutoScroll Velocity:=-20
```

This example scrolls forward through the active window pane at full speed.

```
ActiveWindow.ActivePane.AutoScroll Velocity:=100
```

BrowseToWindow Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBrowseToWindowC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBrowseToWindowX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBrowseToWindowA"}

Pane object: **True** if lines wrap at the right edge of the pane rather than at the right margin of the page. Read/write **Boolean**.

View object: **True** if lines wrap at the right edge of the window rather than at the right margin of the page. Can be **True**, **False**, or **wdUndefined**. Read/write **Long**.

Note This property works only when you're in online layout view.

BrowseToWindow Property Example

This example wraps the text to fit within the active window.

```
With ActiveWindow.View
    .Type = wdOnlineView
    .BrowseToWindow = True
End With
```

This example wraps the text to fit within the active window pane.

```
With ActiveWindow
    .View.Type = wdOnlineView
    .ActivePane.BrowseToWindow = True
End With
```

BrowseWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproBrowseWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproBrowseWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproBrowseWidthA"}

Returns the width (in points) of the area in which text wraps in the specified pane. Read-only **Long**.

Note This property works only when you're in online layout view.

BrowseWidth Property Example

This example wraps the text to fit within the active pane, and then it displays a message box that indicates the width of the wrapping text.

```
With ActiveWindow
    .View.Type = wdOnlineView
    .ActivePane.BrowseToWindow = True
End With
MsgBox PointsToInches _
    (ActiveWindow.ActivePane.BrowseWidth) & "inches"
```

DocumentMapPercentWidth Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproDocumentMapPercentWidthC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproDocumentMapPercentWidthX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproDocumentMapPercentWidthA"}

Returns or sets the width of the document map as a percentage of the width of the specified window. Read/write **Long**.

DocumentMapPercentWidth Property Example

This example displays the document map for the active window and sets the map's width to 25 percent of the window's width.

```
With ActiveWindow
    .DocumentMap = True
    .DocumentMapPercentWidth = 25
End With
```

MinimumFontSize Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproMinimumFontSizeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproMinimumFontSizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproMinimumFontSizeA"}

Returns or sets the minimum font size (in points) displayed for the specified pane. Read/write **Long**.

Note This property only affects the text as shown in online layout view. The point sizes that are displayed on the **Formatting** command bar and used for printing aren't changed.

MinimumFontSize Property Example

This example sets the active window to online view and then sets the minimum font size for the active pane to 12 points.

```
With ActiveWindow
    .View.Type = wdOnlineView
    .ActivePane.MinimumFontSize = 12
End With
```

This example returns the minimum font size for the active pane.

```
Msgbox ActiveWindow.ActivePane.MinimumFontSize
```

PageScroll Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthPageScrollC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthPageScrollX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthPageScrollA"}

Scrolls through the specified pane or window page by page.

Syntax

expression.**PageScroll(*Down, Up*)**

expression Required. An expression that returns a **Window** or **Pane** object.

Down Optional **Variant**. The number of pages to be scrolled down. If this argument is omitted, this value is assumed to be 1.

Up Optional **Variant**. The number of pages to be scrolled up.

Remarks

The **PageScoll** method is available only if you're in page layout view or online layout view. This method doesn't affect the position of the insertion point.

If ***Down*** and ***Up*** are both specified, the window is scrolled by the difference of the arguments. For example, if ***Down*** is 2 and ***Up*** is 4, the window is scrolled up two pages.

PageScroll Method Example

This example scrolls down three pages in the active window.

```
ActiveWindow.View.Type = wdPageView
ActiveWindow.PageScroll Down:=3
```

This example scrolls up one page in the active pane.

```
ActiveWindow.View.Type = wdPageView
ActiveWindow.ActivePane.PageScroll Up:=1
```

This example scrolls down one page in the active window.

```
ActiveWindow.View.Type = wdPageView
ActiveWindow.PageScroll
```

ShowMainTextLayer Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowMainTextLayerC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowMainTextLayerX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowMainTextLayerA"}

True if the text in the specified document is visible when the header and footer areas are displayed. This property is equivalent to the **Show/Hide Document Text** button on the **Header and Footer** toolbar. Read/write **Boolean**.

ShowMainTextLayer Property Example

This example displays the document header in the active window and hides the document text.

```
With ActiveWindow.View
    .Type = wdPageView
    .SeekView = wdSeekCurrentPageHeader
    .ShowMainTextLayer = False
End With
```

PageColumns Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageColumnsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageColumnsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageColumnsA "}

Returns or sets the number of pages to be displayed side by side on-screen at the same time in page layout view or print preview. Read/write **Long**.

PageColumns Property Example

This example switches the active window to page layout view and displays two pages side by side.

```
With ActiveWindow.View
    .Type = wdPageView
    .Zoom.PageColumns = 2
    .Zoom.PageRows = 1
End With
```

This example switches the document window for Hello.doc to page layout view and displays one full page.

```
With Windows("Hello.doc").View
    .Type = wdPageView
    With .Zoom
        .PageColumns = 1
        .PageRows = 1
        .PageFit = wdPageFitFullPage
    End With
End With
```

PageFit Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageFitC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageFitX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageFitA "}

Returns or sets the view magnification of a window so that either the entire page is visible or the entire width of the page is visible. Can be one of the following **WdPageFit** constants: **wdPageFitBestFit**, **wdPageFitFullPage**, or **wdPageFitNone**. Read/write **Long**.

Remarks

The **wdPageFitFullPage** constant has no effect if the document isn't in page layout view.

When the **PageFit** property is set to **wdPageFitBestFit**, the zoom percentage is automatically recalculated every time the document window size is changed. Setting this property to **wdPageFitNone** keeps the zoom percentage from being recalculated whenever this happens.

PageFit Property Example

This example changes the magnification percentage of the window for Letter.doc so that the entire width of the text is visible.

```
With Windows("Letter.doc").View
    .Type = wdNormalView
    .Zoom.PageFit = wdPageFitBestFit
End With
```

This example switches the active window to page layout view and changes the magnification so that the entire page is visible.

```
With ActiveWindow.View
    .Type = wdPageView
    .Zoom.PageFit = wdPageFitFullPage
End With
```

PageRows Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPageRowsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPageRowsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPageRowsA "}

Returns or sets the number of pages to be displayed one above the other on-screen at the same time in page layout view or print preview. Read/write **Long**.

PageRows Property Example

This example switches the active window to print preview and displays two pages one above the other.

```
PrintPreview = True
With ActiveWindow.View.Zoom
    .PageColumns = 1
    .PageRows = 2
End With
```

Percentage Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproPercentageC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproPercentageX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproPercentageA "}

Returns or sets the magnification for a window as a percentage. Read/write **Long**.

Percentage Property Example

This example switches the active window to normal view and sets the magnification to 80 percent.

```
With ActiveWindow.View
    .Type = wdNormalView
    .Zoom.Percentage = 80
End With
```

This example increases the magnification of the active window by 10 percent.

```
Set myZoom = ActiveWindow.View.Zoom
myZoom.Percentage = myZoom.Percentage + 10
```

Zooms Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproZoomsC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproZoomsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproZoomsA "}

Returns a **Zooms** collection that represents the magnification options for each view (normal view, outline view, page layout view, and so on). Read-only.

For information about returning a single member of a collection, see Returning an Object from a Collection.

Zooms Property Example

This example sets the magnification in normal view to 100 percent for each open window.

```
For Each myWindow In Windows
    myWindow.ActivePane.Zooms(wdNormalView).Percentage = 100
Next myWindow
```

This example sets the magnification in page layout view so that an entire page is visible.

```
ActiveWindow.Panes(1).Zooms(wdPageView).PageFit = wdPageFitFullPage
```

ShowSummary Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproShowSummaryC "} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproShowSummaryX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproShowSummaryA "}

True if an automatic summary is displayed for the specified document. Read/write **Boolean**.

ShowSummary Property Example

This example hides everything in the active document except the summary text.

```
With ActiveDocument
    .SummaryViewMode = wdSummaryModeHideAllButSummary
    .SummaryLength = 30
    .ShowSummary = True
End With
```

AutoSummarize Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthAutoSummarizeC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthAutoSummarizeX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthAutoSummarizeA"}

Creates an automatic summary of the specified document, and returns a **Range** object. Corresponds to the options in **AutoSummarize** command (**Tools** menu).

Syntax

expression.**AutoSummarize(*Length, Mode, UpdateProperties*)**

expression Required. An expression that returns a **Document** object.

Length Optional **Variant**. The length of the summary as a percentage of the total document length (the larger the number, the more detail that's included in the summary).

Mode Optional **Variant**. Specifies the way the summary is displayed. Can be one of the following **WdSummaryMode** constants.

Constant	Description
wdSummaryModeHighlight	Highlights the key points in the specified document and displays the **AutoSummarize** toolbar.
wdSummaryModeInsert	Inserts a summary at the beginning of the specified document.
wdSummaryModeCreateNew	Creates a new document and inserts the summary.
wdSummaryModeHideAllButSummary	Hides everything except the summary and displays the **AutoSummarize** toolbar.

UpdateProperties Optional **Variant**. **True** to update the keyword and comment text in the **Properties** dialog box (**File** menu) to reflect the content of the summary for the specified document.

AutoSummarize Method Example

This example creates an automatic summary of the active document by highlighting its key points.

```
ActiveDocument.AutoSummarize Length:=30, Mode:=wdSummaryModeHighlight, _
    UpdateProperties:=True
```

ClosePrintPreview Method

{ewc HLP95EN.DLL, DYNALINK, "See Also":"womthClosePrintPreviewC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"womthClosePrintPreviewX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"womthClosePrintPreviewA"}

Switches the specified document from print preview to the previous view. If the specified document isn't in print preview, an error occurs.

Syntax

expression.**ClosePrintPreview**

expression Required. An expression that returns a **Document** object.

ClosePrintPreview Method Example

This example switches the active window from print preview to normal view.

```
If PrintPreview = True Then ActiveDocument.ClosePrintPreview
ActiveWindow.View.Type = wdNormalView
```

Exists Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproExistsC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproExistsX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproExistsA"}

True if the specified **HeaderFooter** object exists. Read/write **Boolean**.

Note The primary header and footer exist in all new documents by default. Use this method to determine whether a first-page or odd-page header or footer exists. You can also use the **DifferentFirstPageHeaderFooter** or **OddAndEvenPagesHeaderFooter** property to return or set the number of headers and footers in the specified document or section.

Exists Property Example

If a first-page header exists in section one, this example sets the text for the header.

```
Set aSection = ActiveDocument.Sections(1)
If aSection.Headers(wdHeaderFooterFirstPage).Exists = True Then
    aSection.Headers(wdHeaderFooterFirstPage).Range.Text = "First Page"
End If
```

OpenFormat Property

{ewc HLP95EN.DLL, DYNALINK, "See Also":"woproOpenFormatC"} {ewc HLP95EN.DLL, DYNALINK, "Example":"woproOpenFormatX":1} {ewc HLP95EN.DLL, DYNALINK, "Applies To":"woproOpenFormatA"}

Returns the file format of the specified file converter. Can be one of the following **WdOpenFormat** constants: **wdOpenFormatAuto**, **wdOpenFormatDocument**, **wdOpenFormatRTF**, **wdOpenFormatTemplate**, **wdOpenFormatText**, or **wdOpenFormatUnicodeText**; or it can be a unique number that represents an external file converter. Read-only **Long**.

OpenFormat Property Example

This example displays the unique format value and the format name for the converters you can use to open documents.

```
For Each fc In FileConverters
    If fc.CanOpen = True Then MsgBox fc.OpenFormat & vbCr & fc.FormatName
Next fc
```

This example opens the file named "Data.wp" by using the WordPerfect 6x file converter.

```
Documents.Open FileName:="C:\Data.wp",
Format:=FileConverters("WordPerfect6x").OpenFormat
```