A

Excel 2003 Object Model

Most of the objects in the Excel Object Model have objects with associated collections. The collection object is usually the plural form of the associated object. For example, the Worksheets collection holds a collection of Worksheet objects. For simplicity, each object and associated collection will be grouped together under the same heading.

Common Properties with Collections and Associated Objects

In most cases the purpose of the collection object is only to hold a collection of the same objects. The common properties and methods of the collection objects are listed in the following section. Only unique properties, methods, or events will be mentioned in each object section.

Common Collection Properties

Name	Returns	Description
`Application`	`Application`	Read-only. Returns a reference to the owning Application of the current object. Excel, in this case
`Count`	`Long`	Read-only. Returns the number of objects in the collection
`Creator`	`Long`	Read-only. Returns a `Long` number that describes whether the object was created in Excel or not
`Parent`	`Object`	The `Parent` object is the owning object of the collection object. For example, Workbooks. `Parent` returns a reference to the `Application` object

Common Collection Methods

Name	Returns	Parameters	Description
Item	Single	Index as Variant	Returns the object from the collection with the Index value specified by the Index parameter. The Index value may also specify a unique string key describing one of the objects in the collection

Common Object Properties

Objects also have some common properties. To avoid redundancy the common properties and methods of all objects are listed next. They will be mentioned in each object description as existing but are only defined here.

Name	Returns	Description
Application	Application	Read-only. Returns a reference to the owning Application of the current object—Excel, in this case
Creator	Long	Read-only. Returns a Long number that describes whether the object was created in Excel or not
Parent	Object	Read-only. The owning object of the current object. For example, Characters. Parent may return a reference to a Range object, since a Range object is one of the possible owners of a Characters object

Excel Objects and Their Properties, Methods and Events

The objects are listed in alphabetical order. Each object has a general description of the object and possible parent objects. This is followed by a table format of each of the object's properties, methods, and events. The last section of each object describes some code examples of the object's use.

Addin Object and the Addins Collection

The Addins collection holds all of the Addin objects available to Excel. Each Addin object represents an Addin shown in Excel's Addins dialog box under the Tools ➪ Add-Ins . . . menu. The Addin must be installed (AddIn.Installed = True) to be able to use it in the current session. Examples of available Addin objects in Excel include the Analysis Toolpack, the MS Query Addin, and the Conditional Sum Wizard.

The Add method of the Addins collection can be used to add a new Addin to the collection. The Add method requires a FileName to be specified (usually with a XLL or XLA file extension). The Count property of the Addins collection returns the number of Addins that are available for use by the current Excel session.

Addin Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Addin Properties

Name	Returns	Description
CLSID	String	Read-only. Returns a unique identifier for the Addin
FullName	String	Read-only. Returns the full path and filename of the associated Addin
Installed	Boolean	Set/Get whether the Addin can be used in the current session
Name	String	Read-only. Returns the file name of the Addin
Path	String	Read-only. Returns the full file path of the associated Addin
Title	String	Read-only. This hidden property returns the string shown in the Addin Manager dialog box

Example: AddIn Object and the AddIns Collection

This example ensures that the Analysis Toolpack is installed:

```
Sub UseAnalysisToolpack()
   Dim oAddin As AddIn
   'Make sure the Analysis Toolpack is installed
   For Each oAddin In AddIns
      If oAddin.Name = "ANALYS32.XLL" Then
         oAddin.Installed = True
      End If
   Next
End Sub
```

Note that instead of looping through the Addins collection, you could follow the online Help and use:

```
AddIns("Analysis Toolpak").Installed = True
```

Unfortunately, this approach may not work with a non-English User-Interface language, if the Addin's title has been localised.

Adjustments Object

The Adjustments object holds a collection of numbers used to move the adjustment "handles" of the parent Shape object. Each Shape object can have up to eight different adjustments. Each specific adjustment handle can have one or two adjustments associated with it depending on if it can be moved both horizontally and vertically (two) or in just one dimension. Adjustment values are between 0 and 1 and hence are percentage adjustments—the absolute magnitude of a 100% change is defined by the shape being adjusted.

Adjustments Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Adjustments Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of adjustments values associated with the parent Shape object
Item	Single	Parameters: Index As Long. Set/Get the adjustment value or values indicated by the Index parameter

Example: Adjustments Object

This example draws a block arrow on the sheet, then modifies the dimensions of the arrow head:

```
Sub AddArrow()
    Dim oShp As Shape

    'Add an arrow head to the sheet
    Set oShp = ActiveSheet.Shapes.AddShape( _
                    msoShapeRightArrow, 10, 10, 100, 50)

    'Set the 'head' of the arrow to start 30% of the way across
    'and the 'shaft' to start 40% of the way down.
    oShp.Adjustments(1) = 0.3     'Left/right
    oShp.Adjustments(2) = 0.4     'Up/down
End Sub
```

AllowEditRange Object and the AllowEditRanges Collection

The AllowEditRange object represents a range of cells on a worksheet that can still be edited when protected. Each AllowEditRange object can have permissions set for any number of users on your network and can have a separate password.

Be aware of the Locked property of the Range object when using this feature. When you unlock cells, then protect the worksheet, you are allowing any user access to those cells, regardless of the AllowEditRange objects. When each AllowEditRange object's cells are locked, any user can still edit them unless you assign a password or add users and deny them permission without using a password.

The AllowEditRanges collection represents all AllowEditRange objects that can be edited on a protected worksheet. See the AllowEditRange object for more details.

AllowEditRanges Collection Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of AllowEditRange objects that are contained in the area
Item	AllowEdit Range	Parameter: Index As Variant. Returns a single AllowEditRange object in the AllowEditRanges collection

AllowEditRanges Collection Methods

Name	Returns	Parameters	Description
Add	AllowEdit Range	Title As String, Range As Range, [Password]	Adds an AllowEditRange object to the AllowEditRanges collection

AllowEditRange Properties

Name	Returns	Description
Range	Range	Returns a subset of the ranges that can be edited on a protected worksheet
Title	String	Returns or sets the title of the Web page when the document is saved as a Web page
Users	UserAccess List	Returns the list of users who are allowed access to the protected range on a worksheet

AllowEditRange Methods

Name	Returns	Parameters	Description
ChangePassword		Password As String	Sets the password for a range that can be edited on a protected worksheet
Delete			Deletes the object
Unprotect		[Password]	Removes any protection from a sheet or workbook

Example: AllowEditRange Object

The following routine loops through a list of range names in a worksheet and adds an `AllowEditRange` item for each one whose name begins with `"pc"`. It also denies access to the `pcNetSales` range to all but one user, who can only edit the range with a password.

```
Sub CreateAllowRanges()

    Dim lPos As Long
    Dim nm As Name
    Dim oAllowRange As AllowEditRange
    Dim sName As String

    With wksAllowEditRange
        'Loop through the worksheet level
        ' range names
        For Each nm In .Names
            'Store the name
            sName = nm.Name

            'Locate the position of the "!"
            lPos = InStr(1, sName, "!", vbTextCompare)

            'If there was an "!"...
            If lPos > 0 Then
                'Is there a "pc" just after the exclamation point
                'If so, it's a range we want to create an AllowEditRange
                ' object for
                If Mid(sName, lPos + 1, 2) = "pc" Then
                    'Make sure the cells are locked
                    'Unlocking them will allow any user
                    ' access to them.
                    nm.RefersToRange.Locked = True

                    'Pull out the worksheet reference (including the "!")
                    ' from the range name
                    sName = Right(sName, Len(sName) - lPos)

                    'Create the AllowEditRange
                    'Remove the old one if it exists
                    On Error Resume Next
                        Set oAllowRange = Nothing
                        Set oAllowRange = .Protection.AllowEditRanges(sName)
                    On Error GoTo 0
                    If Not oAllowRange Is Nothing Then oAllowRange.Delete
                    Set oAllowRange = .Protection.AllowEditRanges.Add(sName, _
                                                              nm.RefersToRange)
                    'If it's the sales range name...
                    If sName = "pcNetSales" Then
                        'Add a password, then
                        'Add a user and deny them from editing the range
                        ' without the password
                        oAllowRange.ChangePassword "pcnsw"
                        oAllowRange.Users.Add "RCR\AgamaOffice", False
                    End If
                End If
            End If
        Next nm
    End With

End Sub
```

Application Object

The Application object is the root object of the Excel Object Model. All the other objects in the Excel Object Model can only be accessed through the Application object. Many objects, however, are globally available. For example, the ActiveSheet property of the Application object is also available globally. That means that the active WorkSheet can be accessed by at least two ways: Application.ActiveSheet and ActiveSheet.

The Application object holds most of the application level attributes that can be set through the Tools ➪ Options menu in Excel. For example, the DefaultFilePath is equivalent to the Default File Location text box in the General tab of the Options dialog box.

Many of the Application object's properties and methods are equivalent to things that can be set with the Options dialog box.

The Application object is also used when automating Excel from another application, such as Word. The CreateObject function, GetObject function or the New keyword can be used to create a new instance of an Excel Application object from another application. Please refer to Chapter 15 for examples of automation from another application.

The Application object can also expose events. However, Application events are not automatically available for use. The following three steps must be completed before Application events can be used:

Create a new class module, say, called cAppObject, and declare a Public object variable in a class, say, called AppExcel, to respond to events. For example:

```
Public WithEvents AppExcel As Excel.Application
```

Now the Application object events will be available in the class for the AppExcel object variable. Write the appropriate event handling code in the class. For example, if you wanted a message to appear whenever a worksheet is activated then you could write the following:

```
Private Sub AppExcel_SheetActivate(ByVal Sh As Object)
    'display worksheet name
    MsgBox "The " & Sh.Name & " sheet has just been activated."
End Sub
```

Finally, in a procedure in a standard module instantiate the class created above with a current `Application` object:

```
Private App As New cAppObject 'class with the above code snippets
Sub AttachEvents()
    Set App.AppExcel = Application
End Sub
```

The `EnableEvents` property of the `Application` object must also be set to `True` for events to trigger at the appropriate time.

Application Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Application Properties

Name	Returns	Description
`ActiveCell`	`Range`	Read-only. Returns the cell in the active sheet where the cursor is located
`ActiveChart`	`Chart`	Read-only. Returns the currently selected chart in the active workbook. If no chart is currently selected, nothing is returned
`ActivePrinter`	`String`	Set/Get the name of the printer currently being used
`ActiveSheet`	`Object`	Read-only. Returns the currently active sheet in the active workbook
`ActiveWindow`	`Window`	Read-only. Returns the currently selected Excel window, if any
`ActiveWorkbook`	`Workbook`	Read-only. Returns the workbook that is currently active, if any
`AddIns`	`AddIns`	Read-only. Returns the collection of Addins currently available for use in Excel
`AlertBefore Overwriting`	`Boolean`	Set/Get whether a message pops up any time an attempt to overwrite non-blank cells by a drag-and-drop operation is made
`AltStartupPath`	`String`	Set/Get the alternative startup file location folder for Excel
`AnswerWizard`	`Answer Wizard`	Read-only. Returns an object allowing manipulation of the Answer Wizard
`ArbitraryXML Support Available`	`Boolean`	Returns a Boolean value indicating if the XML feature is available in Excel
`AskToUpdate Links`	`Boolean`	Set/Get whether the user is prompted to update links whenever a workbook with links is opened

Name	Returns	Description
Assistant	Assistant	Read-only. Returns an object allowing manipulation of the Office Assistant
AutoCorrect	AutoCorrect	Read-only. Returns an object allowing modification of Excel's AutoCorrect features
AutoFormatAs You TypeReplace Hyperlinks	Boolean	Set/Get whether Excel automatically formats/creates hyperlinks as you type
Automation Security	Mso Automation Security	Set/Get the level of macro security used when Excel opens a file programmatically. This setting is independent of the macro security setting found in Security dialog box in the Tools ➪ Macro command, though the msoAutomationSecurityByUI constant instructs the property to use the setting found there
AutoPercent Entry	Boolean	Set/Get whether Excel automatically adds a % sign when typing a number into a cell that has a Percentage format applied
AutoRecover	AutoRecover	Set/Get AutoRecover options such as Path and Time interval
Build	Long	Read-only. Returns the exact build number of Excel
Calculate BeforeSave	Boolean	Set/Get whether workbooks are calculated before they are saved to disk. This assumes that formula calculation is not set to automatic (Calculation property)
Calculation	Xl Calculation	Set/Get when calculations are made automatically, manually, or semi-automatically
Calculation InterruptKey	Xl Calculation Interrupt Key	Set/Get the key that can interrupt Excel when performing calculations
Calculation State	Xl Calculation State	Read-only. Indicates whether Excel calculations are in progress, pending, or done
Calculation Version	Long	Read-only. Returns the Excel version and calculation engine version used when the file was last saved
Caller	Variant	Read-only. Parameters: [Index]. Returns information describing what invoked the current Visual Basic code (for example, cell function, document event)
CanPlaySounds	Boolean	Read-only. Returns whether sound notes are heard in Excel. Property unused from Excel 2000 onwards

Continues

Name	Returns	Description
CanRecord Sounds	Boolean	Read-only. Returns whether sound notes can be recorded in Excel. Property unused from Excel 2000 onwards
Caption	String	Set/Get the caption that appears in the main Excel window
CellDragAnd Drop	Boolean	Set/Get whether dragging and dropping cells is possible
Cells	Range	Read-only. Returns all the cells in the active sheet
Charts	Sheets	Read-only. Returns all the charts in the active workbook
Clipboard Formats	Variant	Read-only. Parameters: [Index]. Returns an array of format values (XlClipboardFormat) that are currently in the clipboard
Columns	Range	Read-only. Returns all the columns in the currently active sheet
COMAddIns	COMAddIns	Read-only. Returns the collection of installed COM Addins
CommandBars	CommandBars	Read-only. Returns the collection of commandbars available to Excel
Command Underlines	XlCommand Underlines	Set/Get how commands are underlined in Excel. Used only on Macintosh systems
Constrain Numeric	Boolean	Set/Get whether only numbers and punctuation marks are recognized by handwriting recognition. Used only by Windows for Pen Computing
Control Characters	Boolean	Set/Get whether control characters are displayed for right-to-left languages. (Language support must be installed)
CopyObjects WithCells	Boolean	Set/Get whether objects (such as embedded objects) can be cut, copied, and sorted along with cell data
Cursor	XlMouse Pointer	Set/Get which mouse pointer is seen in Microsoft Excel
Cursor Movement	Long	Set/Get what type of cursor is used: visual or logical
CustomList Count	Long	Read-only. Returns the number of custom and built-in lists used in Excel (for example, Monday, Tuesday, Wednesday. . .)
CutCopyMode	XlCutCopy Mode	Set/Get whether a cut or copy operation is currently happening
DataEntry Mode	Long	Set/Get whether locked cells can be edited (xlOff for editing allowed, xlOn for editing of unlocked cells only, xlStrict for editing of unlocked cells only that can not be canceled by pressing Escape)
DDEAppReturn Code	Long	Read-only. Returns the result (confirmation/error) of the last DDE message sent by Excel

Name	Returns	Description
`DefaultFilePath`	`String`	Set/Get the default folder used when opening files
`DefaultSaveFormat`	`XlFileFormat`	Set/Get the default file format used when saving files
`DefaultSheetDirection`	`Long`	Set/Get which direction new sheets will appear in Excel
`DefaultWebOptions`	`DefaultWebOptions`	Read-only. Returns an object allowing manipulation of the items associated with the Web Options dialog box
`DecimalSeparator`	`String`	Set/Get the character used for the decimal separator. This is a global setting and will affect all workbooks when opened. Use `Application UseSystemSeparators = True` to globally reset custom separators
`Dialogs`	`Dialogs`	Read-only. Returns a collection of all the built-in dialog boxes
`DisplayAlerts`	`Boolean`	Set/Get whether the user is prompted by typical Excel messages (for example, “Save Changes to Workbook?”) or no prompts appear and the default answer is always chosen
`DisplayClipboardWindow`	`Boolean`	Set/Get whether the Clipboard window is displayed. Used in Microsoft Office Macintosh Edition
`DisplayCommentIndicator`	`XlCommentDisplayMode`	Set/Get how Excel displays cell comments and indicators
`DisplayDocumentActionTaskPane`	`Boolean`	Set to `True` to display the Document Actions task pane
`DisplayExcel4Menus`	`Boolean`	Set/Get whether Excel display Excel 4.0 menus
`Display Formula Bar`	`Boolean`	Set/Get whether the formula bar is displayed
`DisplayFullScreen`	`Boolean`	Set/Get whether the Excel is in full screen mode
`DisplayFunctionToolTips`	`Boolean`	Set/Get whether ToolTips for arguments appear in the cell when typing a function
`DisplayInsertOptions`	`Boolean`	Set/Get whether the Insert Options drop-down button appears next to a range after inserting cells, rows, or columns

Continues

Name	Returns	Description
`DisplayNoteIndicator`	`Boolean`	Set/Get whether comments inserted into cells have a little note indicator at the top right corner of the cell
`DisplayPasteOptions`	`Boolean`	Set/Get whether the Paste Options drop-down button appears next to a range after a paste operation. This is an Office XP setting and therefore affects all other Office applications that use this feature
`DisplayRecentFiles`	`Boolean`	Set/Get whether the most recently opened files are displayed under the File menu
`DisplayScrollBars`	`Boolean`	Set/Get whether scroll bars are displayed for all open workbooks in the current session
`DisplayStatusBar`	`Boolean`	Set/Get whether the status bar is displayed
`EditDirectlyInCell`	`Boolean`	Set/Get whether existing cell text can be modified directly in the cell. Note that cell text can still be overwritten directly
`EnableAnimations`	`Boolean`	Set/Get whether adding and deleting cells, rows, and columns are animated
`EnableAutoComplete`	`Boolean`	Set/Get whether the `AutoComplete` feature is enabled
`EnableCancelKey`	`XlEnableCancelKey`	Set/Get how an Excel macro reacts when the user tries to interrupt the macro (for example, *Ctrl-Break*). This can be used to disable any user interruption, send any interruption to the error handler, or to just stop the code (default). *Use with care*
`EnableEvents`	`Boolean`	Set/Get whether events are triggered for any object in the Excel Object Model that supports events
`EnableSound`	`Boolean`	Set/Get whether sounds are enabled for Excel
`ErrorCheckingOptions`	`ErrorCheckingOptions`	Set/Get error checking properties such as `BackgroundChecking`, `IndicatorColorIndex`, and `InconsistentFormula.` These options mirror rules found on the Error Checking tab of the Tools ➪ Options command
`Excel4IntlMacroSheets`	`Sheets`	Read-only. Returns the collection of sheets containing Excel 4 International macros
`Excel4MacroSheets`	`Sheets`	Read-only. Returns the collection of sheets containing Excel 4 macros
`ExtendList`	`Boolean`	Set/Get whether formatting and formulas are automatically added when adding new rows or columns to the existing lists of rows or columns

Name	Returns	Description
`FeatureInstall`	`MsoFeature Install`	Set/Get how Excel reacts when an Excel feature is accessed that is not installed (through the interface or programmatically)
`FileConverters`	`Variant`	Read-only. Parameters: `[Index1]`, `[Index2]`. Returns an array of all the file converters available in Excel
`FileDialog`	`FileDialog`	Parameters: `[fileDialogType]`. Returns an object that represents an instance of one of several types of file dialog boxes
`FileFind`	`IFind`	Returns an object that can be used to search for files. Used in Microsoft Office Macintosh Edition
`FileSearch`	`FileSearch`	Read-only. Returns an object that can be used to search for files
`FindFormat`	`FindFormat`	Set/Get search criteria for the types of cell formats to look for when using the `Find` and `Replace` methods
`FixedDecimal`	`Boolean`	Set/Get whether any numbers entered in the future will have the decimal points specified by `FixedDecimalPlaces`
`FixedDecimal Places`	`Long`	Set/Get the decimals places used for any future numbers
`GenerateGet PivotData`	`Boolean`	Set/Get whether Excel can get PivotTable report data
`Height`	`Double`	Set/Get the height of Excel's main application window. The value cannot be set if the main window is maximized or minimized
`Hinstance`	`Long`	Read-only. Returns the instance handle of the instance that is calling Excel. Used mainly by other custom applications like those written in Visual Basic
`Hwnd`	`Long`	Read-only. Returns the top-level window handle of the Excel window. Used mainly by other custom applications like those written in Visual Basic
`IgnoreRemote Requests`	`Boolean`	Set/Get whether remote requests through DDE are ignored
`Interactive`	`Boolean`	Set/Get whether Excel accepts keyboard and mouse input
`International`	`Variant`	Read-only. Parameters: `[Index]`. Returns international settings for Excel. Use the `XlApplicationInternational` constants as one of the values of `Index`

Continues

Name	Returns	Description
`Iteration`	`Boolean`	Set/Get whether Excel will iterate through and calculate all the cells in a circular reference trying to resolve the circular reference. Use with `MaxIterations` and `MaxChange`
`Language Settings`	`Language Settings`	Read-only. Returns an object describing the language settings in Excel
`Left`	`Double`	Set/Get the left edge of Excel's main application window. The value cannot be set if the main window is maximized or minimized
`LibraryPath`	`String`	Read-only. Returns the directory where Addins are stored
`MailSession`	`Variant`	Read-only. Returns the hexadecimal mail session number or `Null` if mail session is active
`MailSystem`	`XlMail System`	Read-only. Returns what type of mail system is being used by the computer (for example, `xlMapi`, `xlPowerTalk`)
`MapPaperSize`	`Boolean`	Set/Get whether documents formatted for another country's/region's standard paper size (for example, A4) are automatically adjusted so that they're printed correctly on your country's/region's standard paper size (for example, Letter)
`Math Coprocessor Available`	`Boolean`	Read-only. Returns whether a math coprocessor is available
`MaxChange`	`Double`	Set/Get the minimum change between iterations of a circular reference before iterations stop
`MaxIterations`	`Long`	Set/Get the maximum number of iterations allowed for circular references before iterations stop
`MemoryFree`	`Long`	Read-only. Returns how much free memory (in bytes) Excel can use
`MemoryTotal`	`Long`	Read-only. Returns how much total memory (in bytes) is available to Excel (including memory in use)
`MemoryUsed`	`Long`	Read-only. Returns how much memory (in bytes) Excel is using
`MouseAvailable`	`Boolean`	Read-only. Returns whether the mouse is available
`MoveAfter Return`	`Boolean`	Set/Get whether the current cell changes when the user hits Enter
`MoveAfter Return Direction`	`XlDirection`	Set/Get which direction the cursor will move when the user hits *Enter* changing the current cell
`Name`	`String`	Read-only. Returns "Microsoft Excel"

Name	Returns	Description
Names	Names	Read-only. Returns the collection of defined names in an active workbook
Network Templates Path	String	Read-only. Returns the location on the network where the Excel templates are kept, if any
NewWorkbook	Start Working	Read-only. Returns a StartWorking object
ODBCErrors	ODBCErrors	Read-only. Returns the collection of errors returned by the most recent query or PivotTable report that had an ODBC connection
ODBCTimeout	Long	Set/Get how long, in seconds, an ODBC connection will be kept before timing out
OLEDBErrors	OLEDBErrors	Read-only. Returns the collection of errors returned by the most recent query or PivotTable report that had an OLEDB connection
OnWindow	String	Set/Get the procedure that is executed every time a window is activated by the end user
Operating System	String	Read-only. Returns the name and version of the operating system
Organization Name	String	Read-only. Returns the organization name as seen in the About Microsoft Excel dialog box
Path	String	Read-only. Returns the path where Excel is installed
PathSeparator	String	Read-only. Returns a backslash ("\") on a PC or a colon ":" on a Macintosh
PivotTable Selection	Boolean	Set/Get whether PivotTables use structured selection. For example, when selecting a Row field title the associated data is selected with it
Previous Selections	Variant	Read-only. Parameters: [Index]. Returns an array of the last four ranges or named areas selected by using Name dialog box or Goto feature
ProductCode	String	Read-only. Returns the Guid for Excel
PromptFor SummaryInfo	Boolean	Set/Get whether the user is prompted to enter summary information when trying to save a file
Range	Range	Read-only. Parameters: Cell1, [Cell2]. Returns a Range object containing all the cells specified by the parameters
Ready	Boolean	Read-only. Determines whether the Excel application is ready

Continues

Name	Returns	Description
`RecentFiles`	`RecentFiles`	Read-only. Returns the collection of recently opened files
`RecordRelative`	`Boolean`	Read-only. Returns whether recorded macros use relative cell references (`True`) or absolute cell references (`False`)
`ReferenceStyle`	`XlReference Style`	Set/Get how cells are referenced: Letter-Number (for example, A1, A3) or RowNumber-ColumnNumber (for example, R1C1, R3C1)
`Registered Functions`	`Variant`	Read-only. Parameters: `[Index1]`, `[Index2]`. Returns the array of functions and function details relating to external DLLs or code resources. Using Addins will add external DLLs to your workbook
`ReplaceFormat`	`Replace Format`	Set/Get replacement criteria for the types of cell formats to replace when using the `Replace` method
`RollZoom`	`Boolean`	Set/Get whether scrolling with a scroll mouse will zoom instead of scroll
`Rows`	`Range`	Read-only. Returns all the rows in the active sheet
`RTD`	`RTD`	Read-only. Returns a reference to a real-time date (RTD) object connected to a RTD Server
`ScreenUpdating`	`Boolean`	Set/Get whether Excel updates its display while a procedure is running. This property can be used to speed up procedure code by turning off screen updates (setting the property to `False`) during processing. Use with the `ScreenRefresh` method to manually refresh the screen
`Selection`	`Object`	Read-only. Returns whatever object is currently selected (for example, sheet, chart)
`Sheets`	`Sheets`	Read-only. Returns the collection of sheets in the active workbook
`SheetsInNew Workbook`	`Long`	Set/Get how many blank sheets are put in a newly created workbook
`ShowChartTip Names`	`Boolean`	Set/Get whether charts show the tip names over data points
`ShowChartTip Values`	`Boolean`	Set/Get whether charts show the tip values over data points
`ShowStartup Dialog`	`Boolean`	Set/Get whether the New Workbook task pane appears when loading the Excel application
`ShowToolTips`	`Boolean`	Set/Get whether ToolTips are shown in Excel
`ShowWindowsIn Taskbar`	`Boolean`	Set/Get whether each workbook is visible on the taskbar (`True`) or only one Excel item is visible in the taskbar (`False`)

Name	Returns	Description
SmartTag Recognizers	SmartTag Recognizers	Read-only. Returns a collection of SmartTag recognition engines (recognizers) currently being used in the application
Speech	Speech	Read-only. Allows access to the properties and methods used to programmatically control the Office speech tools
Spelling Options	Spelling Options	Read-only. Allows access to the spelling options of the application
StandardFont	String	Set/Get what font is used as the standard Excel font
Standard FontSize	Double	Set/Get what font size is used as the standard Excel font size (in points)
StartupPath	String	Read-only. Returns the folder used as the Excel startup folder
StatusBar	Variant	Set/Get the status bar text. Returns False if Excel has control of the status bar. Set to False to give control of the status bar to Excel
TemplatesPath	String	Read-only. Returns the path to the Excel templates
ThisCell	Range	Set/Get the cell in which a user-defined function is being called
ThisWorkbook	Workbook	Read-only. Returns the workbook that contains the currently running VBA code
Thousands Separator	String	Set/Get the character used for the thousands separator. This is a global setting and will affect all workbooks when opened. Use Application UseSystemSeparators = True to globally reset custom separators
Top	Double	Set/Get the top of Excel's main application window. The value cannot be set if the main window is maximized or minimized
Transition MenuKey	String	Set/Get what key is used to bring up Excel's menu. The forward slash key ("/") is the default
Transition MenuKeyAction	Long	Set/Get what happens when the Transition Menu key is pressed. Either Excel menus appear (xlExcelMenu) or the Lotus Help dialog box (xlLotusHelp) appears
Transition NavigKeys	Boolean	Set/Get whether the Transition Navigation keys are active. These provide different key combinations for moving and selecting within a worksheet
UsableHeight	Double	Read-only. Returns the vertical space available in Excel's main window, in points, that is available to a sheet's Window. The value will be 1 if there is no space available

Continues

Name	Returns	Description
UsableWidth	Double	Read-only. Returns the horizontal space available in Excel's main window, in points, that is available to a sheet's Window. This property's value will be invalid if no space is available. Check the value of the UsableHeight property to check to see if there is any space available (>1)
UsedObjects	UsedObjects	Read-only. Represents objects allocated in a workbook
UserControl	Boolean	Read-only. True if the current Excel session was started by a user, and False if the Excel session was started programmatically
UserLibrary Path	String	Read-only. Returns the location of Excel's COM Addins
UserName	String	Set/Get the user name in Excel. Note that this is the name shown in the General tab of the Options dialog box and *not* the current user's network ID or the name shown in the Excel splash screen
UseSystem Separators	Boolean	Set/Get whether the system operators in Excel are enabled. When set to False, you can use Application .DecimalSeparator and Application .ThousandsSeparator to override the system separators, which are located in the Regional Settings/Options applet in the Windows Control Panel
Value	String	Read-only. Returns "Microsoft Excel"
VBE	VBE	Read-only. Returns an object allowing manipulation of the Visual Basic Editor
Version	String	Read-only. Returns the version of Excel
Visible	Boolean	Set/Get whether Excel is visible to the user
Watches	Watches	Read-only. Returns a Watches object that represents all of the ranges that are tracked when a worksheet is calculated
Width	Double	Set/Get the width of Excel's main application window. The value cannot be set if the main window is maximized or minimized
Windows	Windows	Read-only. Returns all the Windows open in the current Excel session
WindowsForPens	Boolean	Read-only. Returns whether Excel is running in a Windows for Pen Computing environment
WindowState	XlWindow State	Set/Get whether the window is maximized, minimized, or in a normal state
Workbooks	Workbooks	Read-only. Returns all the open workbooks (not including Addins) in the current Excel session

Name	Returns	Description
`WorksheetFunction`	`WorksheetFunction`	Read-only. Returns an object holding all the Excel's worksheet functions that can be used in VBA
`Worksheets`	`Sheets`	Read-only. Returns all the worksheets in the active workbook

Application Methods

Name	Returns	Parameters	Description
`ActivateMicrosoftApp`		`Index As XlMSApplication`	Activates an application specified by `XlMSApplication`. Opens the application if it is not open. Acts in a similar manner as the `GetObject` function in VBA
`AddChartAutoFormat`		`Chart, Name As String, [Description]`	Adds the formatting and legends of the `Chart` specified by the parameter to the custom chart types
`AddCustomList`		`ListArray, [ByRow]`	Adds the array of strings specified by `ListArray` to Excel's custom lists. The `ListArray` may also be a cell range
`Calculate`			Calculates all the formulas in all open workbooks that have changed since the last calculation. Only applicable if using manual calculation
`CalculateFull`			Calculates all the formulas in all open workbooks. Forces recalculation of every formula in every workbook, regardless of whether or not it has changed since the last calculation
`CalculateFullRebuild`			Completely calculates all open workbooks, including all formulas with dependencies
`CalculateFullRebuild`			Forces a full calculation of the data and rebuilds the dependencies for all open workbooks. Note that dependencies are the formulas that depend on other cells
`CentimetersToPoints`	`Double`	`Centimeters As Double`	Converts the `Centimeters` parameter to points where 1 cm = 28.35 points

Continues

Name	Returns	Parameters	Description
CheckAbort	—	[KeepAbort]	Stops any recalculations in an Excel application
CheckSpelling	Boolean	Word As String, [Custom Dictionary], [Ignore Uppercase]	Checks the spelling of the Word parameter and returns True if the spelling is correct or False if there are errors
Convert Formula	Variant	Formula, FromReference Style As XlReference Style, [ToReference Style], [ToAbsolute], [RelativeTo]	Converts the Formula parameter between R1C1 references and A1 references and returns the converted formula. Also can change the Formula parameter between relative references and absolute references using the ToReferenceStyle parameter and the XlReferenceStyle constants
DDEExecute		Channel As Long, String As String	Sends a Command to an application using DDE through the given Channel number. The properties starting with DDE are associated with the older technology, Dynamic Data Exchange, which was used to share data between applications
DDEInitiate	Long	App As String, Topic As String	Returns a channel number to use for DDE given an application name and the DDE topic
DDEPoke		Channel As Long, Item, Data	Sends Data to an item in an application using DDE through the given Channel number
DDERequest	Variant	Channel As Long, Item As String	Returns information given a specific DDE channel and a requested item
DDETerminate		Channel As Long	Closes the specified DDE channel
DeleteChart AutoFormat		Name As String	Deletes the custom chart type specified by the Name parameter
DeleteCustom List		ListNum As Long	Deletes the custom list specified by the list number. The first four lists are built-in to Excel and cannot be removed
DoubleClick			Triggered by a double-click to the active cell in the active sheet

Name	Returns	Parameters	Description
Evaluate	Variant	Name	Evaluates the Name string expression as if it were entered into a worksheet cell
ExecuteExcel4 Macro	Variant	String As String	Executes the Excel 4 macro specified by the String parameter and returns the results
FindFile	Boolean		Shows the Open dialog box allowing the user to choose a file to open. True is returned if the file opens successfully
GetCustomList Contents	Variant	ListNum As Long	Returns the custom list specified by the ListNum parameter as an array of strings
GetCustom ListNum	Long	ListArray	Returns the list number for the custom list that matches the given array of strings. A zero is returned if nothing matches
GetOpen Filename	Variant	FileFilter], [FilterIndex], [Title], [ButtonText], [MultiSelect]	The Open dialog box is displayed with the optional file filters, titles, and button texts specified by the parameters. The filename and path are returned from this method call. Optionally, can return an array of filenames if the MultiSelect parameter is True. Does not actually open the file
GetPhonetic	String	[Text]	Returns the phonetic text of the Japanese characters in the Text parameter. If no Text parameter is specified then an alternate phonetic text of the previous Text parameter is returned
GetSaveAs Filename	Variant	[Initial Filename], [FileFilter], [FilterIndex], [Title], [ButtonText]	The Save As dialog box is displayed with the optional default file name, file filters, titles, and button texts specified by the parameters. The filename and path are returned from this method call. Does not actually save the file
Goto		[Reference], [Scroll]	Selects the object specified by the Reference parameter and activates the sheet containing that object. The Reference parameter can be a cell, range, or the name of a VBA procedure. The Scroll parameter, if set to True, will scroll the selected object to the top left corner of the Excel window

Continues

Name	Returns	Parameters	Description
Help		[HelpFile], [HelpContext ID]	Displays the Help topic specified by the HelpContextID parameter in the Help file HelpFile
InchesTo Points	Double	Inches As Double	Converts the Inches parameter to points and returns the new value. (1 inch = 72 points)
InputBox	Variant	Prompt As String, [Title], [Default], [Left], [Top], [HelpFile], [HelpContext ID], [Type]	Displays a simple input box very similar to a standard VBA one. However, the [Type] parameter can be used to set the return type to a formula (0), number (1), text (2), Boolean (4), cell reference (8), an error value (16), or an array of values (64)
Intersect	Range	Arg1 As Range, Arg2 As Range, [Arg3], ... [Arg30]	Returns the intersection or overlap of the ranges specified by the parameters as a Range object
MacroOptions		[Macro], [Description], [HasMenu], [MenuText], [HasShortcut Key], [ShortcutKey], [Category], [StatusBar], [HelpContext ID], [HelpFile]	Allows modification of macro attributes such as the name, description, shortcut key, category and associated Help file. Equivalent to the Macro Options dialog box
MailLogoff			Logs off the current MAPI mail session (for example, Exchange, Outlook)
MailLogon		[Name], [Password], [DownloadNew Mail]	Logs on to the default MAPI mail client (for example, Exchange, Outlook). Credentials such as name and password can be specified
NextLetter	Workbook		Used in Macintosh systems with PowerTalk mail extensions to open the oldest unread workbook from the In Tray. Generates an error in Windows
OnKey		Key As String, [Procedure]	Executes the procedure specified by the Procedure parameter whenever the keystroke or key combination described in the Key parameter is pressed

Name	Returns	Parameters	Description
`OnRepeat`		`Text As String, Procedure As String`	Specifies the text to appear by the Edit ➪ Repeat menu item and the procedure to run when the user chooses Edit ➪ Repeat
`OnTime`		`EarliestTime, Procedure As String, [LatestTime], [Schedule]`	Chooses a procedure to run at the time specified by the `EarliestTime` parameter. Uses the `LatestTime` parameter to specify a time range
`OnUndo`		`Text As String, Procedure As String`	Specifies the text to appear by the Edit ➪ Undo menu item and the procedure to run when the user chooses Edit ➪ Undo
`Quit`			Shuts down Microsoft Excel
`RecordMacro`		`[BasicCode], [XlmCode]`	If the user is currently recording a macro, running this statement will put the code specified in the `BasicCode` parameter into the currently recording macro
`RegisterXLL`	`Boolean`	`Filename As String`	Loads the code resource specified by the `Filename` parameter and registers all the functions and procedures in that code resource
`Repeat`			Repeats the last user action made. Must be the first line of a procedure
`Run`	`Variant`	`[Macro], [Arg1], [Arg2], ... [Arg30]`	Runs the macro or procedure specified by the Macro parameter. Can also run Excel 4.0 macros with this method
`SaveWorkspace`		`[Filename]`	Saves the current workspace to the `Filename` parameter
`SendKeys`		`Keys, [Wait]`	Sends the keystrokes in the `Keys` parameter to Microsoft Excel user interface
`SetDefault Chart`		`[FormatName], [Gallery]`	Set the default chart type added when programmatically adding a chart. The `FormatName` parameter can be a built-in chart type or a custom chart type name
`Undo`			Undoes the last action done with the user interface

Continues

Name	Returns	Parameters	Description
Union	Range	Arg1 As Range, Arg2 As Range, [Arg3], ... [Arg30]	Returns the union of the ranges specified by the parameters
Volatile		[Volatile]	Sets the function that currently contains this statement to be either volatile (Volatile parameter to True) or not. A volatile function will be recalculated whenever the sheet containing it is calculated, even if its input values have not changed
Wait	Boolean	Time	Pauses the macro and Excel until the time in the Time parameter is reached

Application Events

Name	Parameters	Description
NewWorkbook	Wb As Workbook	Triggered when a new workbook is created. The new workbook is passed into the event
SheetActivate	Sh As Object	Triggered when a sheet is activated (brought up to front of the other sheets). The activated sheet is passed into the event
SheetBefore DoubleClick	Sh As Object, Target As Range, Cancel As Boolean	Triggered when a sheet is about to be double-clicked. The sheet and the potential double-click spot are passed into the event. The double-click action can be canceled by setting the Cancel parameter to True
SheetBefore RightClick	Sh As Object, Target As Range, Cancel As Boolean	Triggered when a sheet is about to be right-clicked. The sheet and the potential right-click spot are passed into the event. The right-click action can be canceled by setting the Cancel parameter to True
Sheet Calculate	Sh As Object	Triggered when a sheet is recalculated passing in the recalculated sheet
SheetChange	Sh As Object, Target As Range	Triggered when a range on a sheet is changed, for example, by clearing the range, entering data, deleting rows or columns, pasting data etc. *Not* triggered when inserting rows/columns
Sheet Deactivate	Sh As Object	Triggered when a sheet loses focus. Passes in the sheet

Name	Parameters	Description
SheetFollowHyperlink	Sh As Object, Target As Hyperlink	Triggered when the user clicks a hyperlink on a sheet. Passes in the sheet and the clicked hyperlink
SheetPivotTableUpdate	ByVal Sh As Object, Target As PivotTable	Triggered by an update of the PivotTable report. Passes in the sheet and the PivotTable report
SheetSelectionChange	Sh As Object, Target As Range	Triggered when the user selects a new cell in a worksheet. Passes in the new range and the sheet where the change occurred
WindowActivate	Wb As Workbook, Wn As Window	Triggered when a workbook window is activated (brought up to the front of other workbook windows). The workbook and the window are passed in
WindowDeactivate	Wb As Workbook, Wn As Window	Triggered when a workbook window loses focus. The related workbook and the window are passed in
WindowResize	Wb As Workbook, Wn As Window	Triggered when a workbook window is resized. The resized workbook and window are passed into the event. Not triggered when Excel is resized
WorkbookActivate	Wb As Workbook	Triggered when a workbook is activated (brought up to the front of other workbook windows). The workbook is passed in
WorkbookAddinInstall	Wb As Workbook	Triggered when an Addin is added to Excel that is also a workbook. The Addin workbook is passed into the event
WorkbookAddinUninstall	Wb As Workbook	Triggered when an Addin is removed to Excel that is also a workbook. The Addin workbook is passed into the event
WorkbookBeforeClose	Wb As Workbook, Cancel As Boolean	Triggered just before a workbook is closed. The workbook is passed into the event. The closure can be canceled by setting the Cancel parameter to True
WorkbookBeforePrint	Wb As Workbook, Cancel As Boolean	Triggered just before a workbook is printed. The workbook is passed into the event. The printing can be canceled by setting the Cancel parameter to True
WorkbookBeforeSave	Wb As Workbook, SaveAsUI As Boolean, Cancel As Boolean	Triggered just before a workbook is saved. The workbook is passed into the event. The saving can be canceled by setting the Cancel parameter to True. If the SaveAsUI is set to True then the Save As dialog box appears

Continues

Name	Parameters	Description
Workbook Deactivate	Wb As Workbook	Triggered when a workbook loses focus. The related workbook and the window are passed in
WorkbookNew Sheet	Wb As Workbook, Sh As Object	Triggered when a new sheet is added to a workbook. The workbook and new sheet are passed into the event
WorkbookOpen	Wb As Workbook	Triggered when a workbook is opened. The newly opened workbook is passed into the event
WorkbookPivot TableClose Connection	ByVal Wb As Workbook, Target As PivotTable	Triggered when a PivotTable report connection is closed. The selected workbook and PivotTable report are passed in to this event
WorkbookPivot TableOpen Connection	ByVal Wb As Workbook, Target As PivotTable	Triggered when a PivotTable report connection is opened. The selected workbook and PivotTable report are passed in to this event

Example: Application Object

This example demonstrates how to use Application.GetOpenFilename to get the name of a file to open. The key to using this function is to assign its return value to a Variant data type:

```
Sub UsingGetOpenFilename()
   Dim sFilter As String
   Dim vaFile As Variant
   'Build a filter list. If you omit the space before the first comma,
   'Excel will not display the pattern, (*.New)
   sFilter = "New Files (*.New) ,*.new," & _
             "Old Files (*.Old) ,*.old," & _
             "All Files (*.*) ,*.*"
   'Display the File Open dialog, putting the result in a Variant
   vaFile = Application.GetOpenFilename(FileFilter:=sFilter, FilterIndex:=1, _
                              Title:="Open a New or Old File",
MultiSelect:=False)
   'Did the user cancel?
   If vaFile <> False Then
      MsgBox "You want to open " & vaFile
   End If
End Sub
'The Application object is used to store and retrieve custom sort orders:
Sub UsingACustomSortOrder()
   Dim vaSortList As Variant
   Dim iListNum As Integer
   Dim bAdded As Boolean
   'Sort the products in this order
   vaSortList = Array("Oranges", "Mangoes", "Apples", "Pears")
   'Get the number of this custom sort, if it exists.
   iListNum = Application.GetCustomListNum(vaSortList)
   'If it doesn't exist, we get zero, NOT an error
   If iListNum = 0 Then
```

```
         'Create a custom list for this sort order
         Application.AddCustomList vaSortList

         'And retrieve its number (the last one!)
         iListNum = Application.CustomListCount

         'Remember that we added it - delete it after use
         bAdded = True
      End If
      'Sort the range using this custom list. Note that we have to
      'add 1 to the list number, as 'ordercustom:=1' means to use the
      'standard sort order (which is not a custom list)
      ActiveCell.CurrentRegion.Sort key1:=ActiveCell, _
                                 ordercustom:=iListNum + 1, header:=xlYes
      'If we added the list, remove it.
      If bAdded Then Application.DeleteCustomList iListNum
   End Sub
```

Chapter 3 in the first section of this book contains more examples of using the `Application` object.

Areas Collection

The `Areas` collection holds a collection of `Range` objects. Each `Range` object represents a block of cells (for example, A1:A10) or a single cell. The `Areas` collection can hold many ranges from different parts of a workbook. The parent of the `Areas` collection is the `Range` object.

Areas Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Areas Properties

Name	Returns	Description
`Count`	`Long`	Read-only. Returns the number of `Range` objects that are contained in the area
`Item`	`Range`	Parameter: `Index As Long`. Returns a single `Range` object in the `Areas` collection. The `Index` parameter corresponds to the order of the ranges selected

Example: Areas Collection

When using a `Range` containing a number of different areas, we cannot use code like `rgRange.Cells(20).Value` if the twentieth cell is not inside the first area in the range. This is because Excel only looks at the first area, implicitly doing `rgRange.Areas(1).Cells(20).Value`, as this example shows—with a function to provide a workaround:

```
   Sub TestMultiAreaCells()
      Dim oRNg As Range
      'Define a multi-area range
      Set oRNg = Range("D2:F5,H2:I5")
      'The 12th cell should be F5.
      MsgBox "Rng.Cells(12) is " & oRNg.Cells(12).Address & _
              vbCrLf & "Rng.Areas(1).Cells(12) is " & _
                           oRNg.Areas(1).Cells(12).Address & _
              vbCrLf & "MultiAreaCells(Rng, 12) is " & _
                           MultiAreaCells(Rng, 12).Address

      'The 13th cell of the multi-area range should be H2,
      'that is the first cell in the second area.
      MsgBox "Rng.Cells(13) is " & oRNg.Cells(13).Address & _
              vbCrLf & "Rng.Areas(1).Cells(13) is " & _
                           oRNg.Areas(1).Cells(13).Address & _
              vbCrLf & "MultiAreaCells(Rng, 13) is " & _
                           MultiAreaCells(Rng, 13).Address
   End Sub
   Function MultiAreaCells(oRange As Range, iCellNum As Long) As Range
      Dim iTotCells As Long, oArea As Range
      'Loop through all the areas in the range,
      'starting again from the first if we run out
      Do
         For Each oArea In oRange.Areas
            'Is the cell we want in this area?
            If iTotCells + oArea.Cells.Count >= iCellNum Then

               'Yes - return it and exit the function
               Set MultiAreaCells = oArea.Cells(iCellNum - iTotCells)
               Exit Function
            Else
               'No - count up the cells we've checked and carry on
               iTotCells = iTotCells + oArea.Cells.Count
            End If
         Next
      Loop
   End Function
```

AutoCorrect Object

The `AutoCorrect` object represents all of the functionality of the Excel's `AutoCorrect` features.

AutoCorrect Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

AutoCorrect Properties

Name	Returns	Description
`AutoExpand ListRange`	`Boolean`	Set to `True` to enable automatic expansion in lists
`Capitalize NamesOfDays`	`Boolean`	Set/Get whether the first letter of days of the weeks are capitalized

Name	Returns	Description
CorrectCaps Lock	Boolean	Set/Get whether typing mistakes made with leaving the Caps Lock on are automatically corrected
Correct SentenceCap	Boolean	Set/Get whether the first letter of a sentence is capitalized if accidentally left in small case
DisplayAuto Correct Options	Boolean	Displays / Hides the AutoCorrect Options button. The default value is True. This is an Office-wide setting. Changing it in Excel will also affect all the other Office applications
ReplaceText	Boolean	Set/Get whether Excel will automatically replace certain words with words from the AutoCorrect list
TwoInitial Capitals	Boolean	Set/Get whether Excel will automatically change the second letter of a word to lowercase if the first letter is uppercase

AutoCorrect Methods

Name	Returns	Parameters	Description
Add Replacement	Variant	What As String, Replacement As String	Adds a word (the What parameter) that will be automatically replaced with another word (the Replacement parameter) to the ReplacementList list array
Delete Replacement	Variant	What As String	Deletes a word from the ReplacementList list so that it does not get replaced with another word automatically
Replacement List	Variant	[Index]	Returns a multidimensional array of strings. The first column of the array holds the word that will be changed and the second column holds the replaced text. The Index parameter can be used to return an array containing a single word and its replacement

Example: AutoCorrect Object

This example uses the AutoCorrect object to find the replacement to use for a given word:

```
Sub TestAutoCorrect()
   MsgBox "'(c)' is replaced by " & UseAutoCorrect("(c)")
End Sub
```

```
Function UseAutoCorrect(ByVal sWord As String) As String
   Dim i As Integer
   Dim vaRepList As Variant
   Dim sReturn As String
   'Default to returning the word we were given
   sReturn = sWord
   'Get the replacement list into an array
   vaRepList = Application.AutoCorrect.ReplacementList
   'Go through the replacement list
   For i = LBound(vaRepList) To UBound(vaRepList)
      'Do we have a match?
      If vaRepList(i, 1) = sWord Then
         'Return the replacement text
         sReturn = vaRepList(i, 2)
         'Jump out of the loop
         Exit For
      End If
   Next
   'Return the word, or its replacement if it has one
   UseAutoCorrect = sReturn
End Function
```

AutoFilter Object

The `AutoFilter` object provides the functionality equivalent to the `AutoFilter` feature in Excel. This object can programmatically filter a range of text for specific types of rows, hiding the rows that do not meet the filter criteria. Examples of filters include top 10 rows in the column, rows matching specific values, and non-blank cells in the row. Using the Data ➪ Filter ➪ AutoFilter submenu in Excel can access this feature. The parent of the `AutoFilter` object is the `Worksheet` object (implying that a worksheet can have only one `AutoFilter`).

The `AutoFilter` object is used with the `AutoFilter` method of the `Range` object and the `AutoFilterType` property of the `Worksheet` object.

AutoFilter Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

AutoFilter Properties

Name	Returns	Description
`Filters`	`Filters`	Read-only. Returns the collection of filters associated with the range that was autofiltered (for example, non-blank rows)
`Range`	`Range`	Read-only. Returns the group of cells that have an `AutoFilter` applied to it

Example: AutoFilter Object

This example demonstrates how to use the `AutoFilter`, `Filters` and `Filter` objects, by displaying the complete set of auto-filters currently in use:

```
Sub ShowAutoFilterCriteria()
   Dim oAF As AutoFilter, oFlt As Filter
   Dim sField As String
   Dim sCrit1 As String, sCrit2 As String
   Dim sMsg As String, i As Integer
   'Check if the sheet is filtered at all
   If ActiveSheet.AutoFilterMode = False Then
      MsgBox "The sheet does not have an AutoFilter"
      Exit Sub
   End If
   'Get the sheet's AutoFilter object
   Set oAF = ActiveSheet.AutoFilter
   'Loop through the Filters of the AutoFilter
   For i = 1 To oAF.Filters.Count
      'Get the field name from the first row
      'of the AutoFilter range
      sField = oAF.Range.Cells(1, i).Value
      'Get the Filter object
      Set oFlt = oAF.Filters(i)
      'If it is on...
      If oFlt.On Then
         'Get the standard filter criteria
         sMsg = sMsg & vbCrLf & sField & oFlt.Criteria1
         'If it's a special filter, show it
         Select Case oFlt.Operator
            Case xlAnd
               sMsg = sMsg & " And " & sField & oFlt.Criteria2
            Case xlOr
               sMsg = sMsg & " Or " & sField & oFlt.Criteria2
            Case xlBottom10Items
               sMsg = sMsg & " (bottom 10 items)"
            Case xlBottom10Percent
               sMsg = sMsg & " (bottom 10%)"
            Case xlTop10Items
               sMsg = sMsg & " (top 10 items)"
            Case xlTop10Percent
               sMsg = sMsg & " (top 10%)"
         End Select
      End If
   Next
   If sMsg = "" Then
      'No filters are applied, so say so
      sMsg = "The range " & oAF.Range.Address & " is not filtered."
   Else
      'Filters are applied, so show them
      sMsg = "The range " & oAF.Range.Address & " is filtered by:" & sMsg
   End If
   'Display the message
   MsgBox sMsg
End Sub
```

AutoRecover Object

This object allows access to the AutoRecover settings for the Excel application. These settings can be found on the Save tab of the Tools ➪ Options command and apply to all workbooks. Note that each workbook can choose whether or not to have AutoRecover applied to it—also located on the Save tab.

AutoRecover Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

AutoRecover Properties

Name	Returns	Description
Enabled	Boolean	True if the AutoRecover object is enabled
Path	String	Set/Get the complete path to where the AutoRecover temporary files are saved
Time	Long	Set/Get the time interval for the AutoRecover object. Permissible values are integers from 1 to 120 minutes (default 10)

Example: AutoRecover Object

The following subroutine and function sets AutoRecover properties, then ensures that the workbook the code is in uses them:

```
Sub SetAutoRecoverOptions()
    'Set the AutoRecover options for the application
    ChangeAutoRecoverSettings True, "C:\Documents and Settings\AgamaOffice\My
Documents\Backup Files\AutoRecover\Excel", 2

    'Make sure this workbook uses them
    ThisWorkbook.EnableAutoRecover = True
End Sub
Function ChangeAutoRecoverSettings(Optional ByVal vEnable As Variant,
Optional ByVal vPath As Variant, Optional ByVal vTime As Variant)
    With Application.AutoRecover
         'Only set the property if a value was passed
         If Not IsMissing(vEnable) Then
             'Enable AutoRecover
             .Enabled = vEnable
         End If

         'Only set the property if a value was passed
         If Not IsMissing(vPath) Then
             'Change the path to a central backup files area
             .Path = vPath
         End If
         'Only set the property if a value was passed
         If Not IsMissing(vTime) Then
             'Save every AutoRecover file every 2 minutes
             .Time = vTime
         End If
     End With
End Function
```

Axis Object and the Axes Collection

The Axes collection represents all of the Axes in an Excel chart. Each Axis object is equivalent to an axis in an Excel chart (for example, X axis, Y axis, etc.). The parent of the Axes collection is the Chart object.

Besides the typical properties and methods associated with a collection object, the Axes collection also has a Count property that returns the number of Axis objects in the collection. Also, unlike most other collections, the Item method of the Axes collection has two parameters: Type and AxisGroup. Use one of the xlAxisType constants for the Type parameter (xlValue, xlCategory, or xlSeriesAxis). The optional second parameter, AxisGroup, can take one of the xlAxisGroup constants (xlPrimary or xlSecondary).

Axis Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Axis Properties

Name	Returns	Description
AxisBetween Categories	Boolean	Set/Get whether the value axis crosses the category axis between categories (as in Column charts) or aligned with the category label (as in Line charts)
AxisGroup	XlAxis Group	Read-only. Returns whether the current axis is of the primary group (xlPrimary) or the secondary group (xlSecondary)
AxisTitle	AxisTitle	Read-only. Returns an object manipulating the axis title properties
BaseUnit	XlTime Unit	Set/Get what type of base units to have for a category axis. Use with BaseUnitIsAuto property. Fails on a value axis
BaseUnitIsAuto	Boolean	Set/Get whether the Excel automatically chooses the base units for a category axis. Fails on a value axis
Border	Border	Read-only. Returns the border’s properties around the selected axis
CategoryNames	Variant	Set/Get the category names for the axis as a string array

Continues

Name	Returns	Description
CategoryType	XlCategory Type	Set/Get what type of axis to make the category axis. Fails on a value axis
Crosses	XlAxis Crosses	Set/Get where one axis crosses with the other axis: at the minimum value, maximum value, Excel automatic, or some custom value
CrossesAt	Double	Set/Get what value the other axis crosses the current one. Use when the Crosses property is xlAxisCrossesCustom
DisplayUnit	XlDisplay Unit	Set/Get what sort of unit to display for the axis (for example xlThousands)
DisplayUnit Custom	Double	Set/Get the value to display units if the DisplayUnit property is set to xlCustom
DisplayUnit Label	DisplayUnit Label	Read-only. Returns an object that manipulates a unit label for an axis
HasDisplay UnitLabel	Boolean	Set/Get whether a display unit label created using the DisplayUnit or DisplayUnitCustom property is visible on the axis
HasMajor Gridlines	Boolean	Set/Get whether major gridlines are displayed on the axis
HasMinor Gridlines	Boolean	Set/Get whether minor gridlines are displayed on the axis
HasTitle	Boolean	Set/Get whether the axis has a title
Height	Double	Read-only. Returns the height of the axis
Left	Double	Read-only. Returns the position of the axis from the left edge of the chart
Major Gridlines	Gridlines	Read-only. Returns an object to manipulate the major gridlines formatting associated
MajorTickMark	XlTickMark	Set/Get how the major ticks should look like (for example, inside the axis, outside the axis)
MajorUnit	Double	Set/Get what the value is between major blocks of a unit
MajorUnitIs Auto	Boolean	Set/Get whether the value of MajorUnit is set automatically
MajorUnit Scale	XlTimeUnit	Set/Get what type to set for the major units
MaximumScale	Double	Set/Get what the maximum value is for the axis
MaximumScale IsAuto	Boolean	Set/Get whether the maximum value for the axis is determined automatically

Name	Returns	Description
MinimumScale	Double	Set/Get what the minimum value is for the axis
MinimumScale IsAuto	Boolean	Set/Get whether the minimum value for the axis is determined automatically
Minor Gridlines	Gridlines	Read-only. Returns an object to manipulate major gridline formatting associated
MinorTickMark	XlTickMark	Set/Get what the minor ticks should look like (for example, inside the axis, outside the axis)
MinorUnit	Double	Set/Get what the value is between minor blocks of a unit
MinorUnitIs Auto	Boolean	Set/Get whether the value of MinorUnit is set automatically
MinorUnitScale	XlTimeUnit	Set/Get what scale to set for the minor units
ReversePlot Order	Boolean	Set/Get whether the unit values on the axis should be reversed
ScaleType	XlScale Type	Set/Get the type of scale to use for the units: Linear or Logarithmic
TickLabel Position	XlTickLabel Position	Set/Get the position that the tick marks will appear in relation to the axis (for example, low, high)
TickLabels	TickLabels	Read-only. Returns an object to manipulate properties of the tick labels of an axis
TickLabel Spacing	Long	Set/Get how often to display the tick labels
TickMark Spacing	Long	Set/Get how often to display tick marks on an axis. Fails on a value axis
Top	Double	Read-only. Returns the top of the axis in relation to the top edge of the chart
Type	XlAxisType	Set/Get the type of axis (xlCategory, xlSeriesAxis, or xlValue)
Width	Double	Read-only. Returns the width of the axis

Axis Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the axis from the axes collection
Select	Variant		Selects the axis on the chart

Example: Axis Object and the Axes Collection

This example sets the labels for the X-axis (independently of the data that's plotted) and applies some formatting:

```
Sub FormatXAxis()
   Dim oCht As Chart, oAxis As Axis
   'Get the first embedded chart on the sheet
   Set oCht = ActiveSheet.ChartObjects(1).Chart
   'Get it's X axis
   Set oAxis = oCht.Axes(xlCategory)
   'Format the X axis
   With oAxis
      .CategoryNames = Array("Item 1", "Item 2", "Item 3")
      .TickLabels.Orientation = 45
      .AxisBetweenCategories = True
      .ReversePlotOrder = False
      .MinorTickMark = xlTickMarkNone
 .MajorTickMark = xlTickMarkCross
   End With
End Sub
```

AxisTitle Object

The `AxisTitle` object contains the formatting and words associated with a chart axis title. The parent of the `AxisTitle` object is the `Axis` object. The `AxisTitle` object is used in coordination with the `HasTitle` property of the parent `Axis` object. The `HasTitle` property must be `True` for a child `AxisTitle` object to exist.

AxisTitle Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

AxisTitle Properties

Name	Returns	Description
`AutoScaleFont`	`Variant`	Set/Get whether the font size will change automatically if the parent chart changes sizes
`Border`	`Border`	Read-only. Returns the border's properties around the selected axis title
`Caption`	`String`	Set/Get the axis title's text
`Characters`	`Characters`	Read-only. Parameters: `[Start]`, `[Length]`. Returns an object containing all the characters in the axis title. Allows manipulation on a character-by-character basis
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the chart axis title

Name	Returns	Description
`Font`	`Font`	Read-only. Returns an object containing Font options for the chart axis title
`Horizontal Alignment`	`Variant`	Set/Get how you want the axis title horizontally aligned. Use the `xlAlign` constants
`Interior`	`Interior`	Read-only. Returns an object containing options to format the area in the chart title text area (for example, interior color)
`Left`	`Double`	Set/Get the distance from the left edge of the axis title text area to the chart's left edge
`Name`	`String`	Read-only. Returns the name of the axis title object
`Orientation`	`Variant`	Set/Get the angle of the text for the axis title. The value can be in degrees (from –90 to 90) or one of the `XlOrientation` constants
`ReadingOrder`	`Long`	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages
`Shadow`	`Boolean`	Set/Get whether the axis title has a shadow effect
`Text`	`String`	Set/Get the axis title's text
`Top`	`Double`	Set/Get the distance from the top edge of the axis title text area to the chart's top edge
`Vertical Alignment`	`Variant`	Set/Get how you want the axis title horizontally aligned. Use the `xlVAlign` constants

AxisTitle Methods

Name	Returns	Parameters	Description
`Delete`	`Variant`		Deletes the axis title from the axis
`Select`	`Variant`		Selects the axis title on the chart

Example: AxisTitle Object

This example ensures the X-axis has a title and sets the X-axis title's caption and formatting:

```
Sub FormatXAxisTitle()
   Dim oCht As Chart, oAT As AxisTitle
   'Get the first embedded chart on the sheet
   Set oCht = ActiveSheet.ChartObjects(1).Chart
   'Give the X axis a title
   oCht.Axes(xlCategory).HasTitle = True
   'Get the title
   Set oAT = oCht.Axes(xlCategory).AxisTitle
```

```
    'Format the title
    With oAT
        .AutoScaleFont = False
        .Caption = "X Axis Title"
        .Font.Bold = True
    End With
End Sub
```

Border Object and the Borders Collection

The Borders collection contains the properties associated with four borders around the parent object. Parent objects of the Borders collection are the Range and the Style object. A Borders collection always has four borders. Use the xlBordersIndex constants with the Item property of the Borders collection to access one of the Border objects in the collection.

Each Border object corresponds to a side or some sides of a border around a parent object. Some objects only allow access to all four sides of a border as a whole (for example, left side of border can not be accessed independently). The following objects are parents of the Border object (not the Borders collection): Axis, AxisTitle, ChartArea, ChartObject, ChartTitle, DataLabel, DataTable, DisplayUnitLabel, Downbars, DropLines, ErrorBars, Floor, GridLines, HiLoLines, LeaderLines, Legend, LegendKey, OleObject, PlotArea, Point, Series, SeriesLines, TrendLine, UpBars, and Walls. The following collections are also possible parents of the Border object: DataLabels, ChartObjects, and OleObjects.

The Borders collection has a few properties besides the typical collection attributes. They are listed in the following table.

Borders Collection Properties

Name	Returns	Description
Color	Variant	Set/Get the color for all four of the borders in the collection. Use the RGB function to set the color
ColorIndex	Variant	Set/Get the color for all four of the borders in the collection. Use the index number of a color in the current color palette to set the Color value
Count	Long	Read-only. Returns the number of Border objects in the collection. Always returns four
LineStyle	Variant	Set/Get the style of line to use for the borders (for example, xlDash). Use the xlLineStyle constants to set the value
Value	Variant	Set/Get the style of line to use for the borders (for example, xlDash). Use the xlLineStyle constants to set the value. Same as LineStyle
Weight	Variant	Set/Get how thick to make the borders in the collection (for example, xlThin, xlThick). Use the xlBorderWeight constants

Border Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Border Properties

Name	Returns	Description
`Color`	`Variant`	Set/Get the color for a border. Use the `RGB` function to set the color
`ColorIndex`	`Variant`	Set/Get the color for a border. Use the index number of a color in the current color palette to set the color value
`LineStyle`	`Variant`	Set/Get the style of line to use for a border (for example, `xlDash`). Use the `xlLineStyle` constants to set the value
`Weight`	`Variant`	Set/Get how thick to make the border (for example, `xlThin`, `xlThick`). Use the `xlBorderWeight` constants

Example: Border Object and the Borders Collection

Applies a 3D effect to a range:

```
Sub TestFormat3D()
   'Format the selected range as 3D sunken
   Format3D Selection
End Sub
Sub Format3D(oRange As Range, Optional bSunken As Boolean = True)
   'Using the range...
   With oRange
       'Surround it with a white border
       .BorderAround Weight:=xlMedium, Color:=RGB(255, 255, 255)

       If bSunken Then
          'Sunken, so make the left and top dark-grey
          .Borders(xlEdgeLeft).Color = RGB(96, 96, 96)
          .Borders(xlEdgeTop).Color = RGB(96, 96, 96)
       Else
          'Raised, so make the right and bottom dark-grey
          .Borders(xlEdgeRight).Color = RGB(96, 96, 96)
          .Borders(xlEdgeBottom).Color = RGB(96, 96, 96)
       End If
   End With
End Sub
```

CalculatedFields Collection

See the *PivotField Object, PivotFields Collection*, and the *CalculatedFields Collection* section.

CalculatedItems Collection

See the *PivotItem Object, PivotItems Collection,* and the *CalculatedItems Collection* section.

CalculatedMember Object and the CalculatedMembers Collection

The CalculatedMembers collection is a collection of all the CalculatedMember objects on the specified PivotTable. Each CalculatedMember object represents a calculated field, or calculated item.

CalculatedMembers Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CalculatedMembers Collection Properties

Name	Returns	Description
Count	Long	Returns the number of objects in the collection
Item	Calculated Member	Parameter: Index As Variant. Returns a single CalculatedMember object in the CalculatedMembers collection

CalculatedMembers Collection Methods

Name	Returns	Parameters	Description
Add	Calculated Member	Name As String, Formula As String, [SolveOrder], [Type]	Adds a CalculatedField or CalculatedItem to a PivotTable

CalculatedMember Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CalculatedMember Properties

Name	Returns	Description
Formula	String	Returns the CalculatedMember's formula in multidimensional expressions (MDX) syntax
IsValid	Boolean	Indicates whether the specified CalculatedMember object has been successfully instantiated with the OLAP provider during the current session. Will return True even if the PivotTable is not connected to its data source

Name	Returns	Description
Name	String	Set/Get the name of the object
SolveOrder	Long	Gets the value of the CalculatedMember's MDX (multidimensional expression) argument (default is zero)
SourceName	String	Gets the object's name as it appears in the original source data for the specified PivotTable report
Type	Xl Calculated MemberType	Gets the CalculatedMember object's type

CalculatedMember Methods

Name	Returns	Parameters	Description
Delete			Deletes the selected object

Example: CalculatedMembers Collection and CalculatedMember Object

The following routine returns information about each CalculatedMember from the data source used by the PivotTable on the wksPivotTable worksheet. It returns messages if either the data source is not an OLAP type or if there are no CalculatedMembers:

```
Sub ReturnCalculatedMembers()
    Dim lIcon As Long, lCount As Long
    Dim ptTable As PivotTable
    Dim oCalcMember As CalculatedMember
    Dim oCalcMembers As CalculatedMembers
    Dim sInfo As String
    'Set the reference to the PivotTable
    Set ptTable = wksPivotTable.PivotTables("WroxSales1")
    On Error Resume Next
        Set oCalcMembers = ptTable.CalculatedMembers
    On Error GoTo 0

    'Did we return a reference to Calculated Members?
    If Not oCalcMembers Is Nothing Then
        'If there's at least one Calculated Member...
        If oCalcMembers.Count > 0 Then
            'Initialize the Count
            ' and message variables
            lCount = 1
            lIcon = vbInformation

            'Loop through each Calculated Member
            ' And store its name and formula
            For Each oCalcMember In oCalcMembers
                With oCalcMember
```

```
                    sInfo = sInfo & lCount & ") " & .Name & ": " & .Formula
                    lCount = lCount + 1
                End With
            Next oCalcMember

        Else
            'It's a valid OLAP data source, but no
            ' Calculated Members are there
            lIcon = vbExclamation
            sInfo = "No Calculated Members found."
        End If
    Else
        'oCalcMembers returned nothing. Not an OLAP data source
        lIcon = vbCritical
        sInfo = "Could not retrieve Calculated Members. Data Source may not be
OLAP type."
    End If

    MsgBox sInfo, lIcon, "Calculated Members"

End Sub
```

CalloutFormat Object

The `CalloutFormat` object corresponds to the line callouts on shapes. The parent of the `CalloutFormat` object is the `Shape` object.

CalloutFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

CalloutFormat Properties

Name	Returns	Description
`Accent`	`MsoTriState`	Set/Get whether a vertical accent bar is used to separate the callout box from the line
`Angle`	`MsoCallout AngleType`	Set/Get the angle of the callout line in relation to the callout box
`AutoAttach`	`MsoTriState`	Set/Get whether a callout line automatically changes where it is attached to the callout box depending on the where the line is pointing (left or right of the callout box)
`AutoLength`	`MsoTriState`	Read-only. Return whether the callout line changes size automatically if the multisegment callout box is moved
`Border`	`MsoTriState`	Set/Get whether the callout box has a border around it
`Drop`	`Single`	Read-only. Returns the distance from the callout box to the spot where the callout line is pointing

Name	Returns	Description
`DropType`	`MsoCallout DropType`	Read-only. Returns the spot on the callout box that attaches to the callout line
`Gap`	`Single`	Set/Get the distance between the callout line end and the callout box
`Length`	`Single`	Read-only. Returns the length of the first part of a callout line. `AutoLength` must be `False`
`Type`	`MsoCallout Type`	Set/Get the type of line callout used

CalloutFormat Methods

Name	Returns	Parameters	Description
`Automatic Length`			Sets the `AutoLength` property to `True`
`CustomDrop`		`Drop As Single`	Uses the `Drop` parameter to set the distance from the callout box to the spot where the callout line is pointing
`CustomLength`		`Length As Single`	Sets the length of the first part of a callout line to the `Length` parameter and sets `AutoLength` to `False`
`PresetDrop`		`DropType As MsoCallout DropType`	Sets the spot on the callout box that attaches to the callout line using the `DropType` parameter

Example: CalloutFormat Object

This example applies the same formatting to all the callouts in a worksheet:

```
Sub FormatAllCallouts()
   Dim oShp As Shape
   Dim oCF As CalloutFormat
   'Loop through all the shapes in the sheet
   For Each oShp In ActiveSheet.Shapes
      'Is this a callout?
      If oShp.Type = msoCallout Then
         'Yes - set its text box to autosize
         oShp.TextFrame.AutoSize = True

         'Get the CalloutFormat object
         Set oCF = oShp.Callout

         'Format the callout
         With oCF
            .Gap = 0
            .Border = msoFalse
            .Accent = msoTrue
            .Angle = msoCalloutAngle30
            .PresetDrop msoCalloutDropCenter
         End With
      End If
   Next
End Sub
```

CellFormat Object

Represents both the FindFormat and ReplaceFormat property settings of the Application object, which are then used by the Find and Replace methods (respectively) of the Range object.

Set the FindFormat property settings before using the Find method to search for cell formats within a range. Set the ReplaceFormat property settings if you want the Replace method to replace formatting in cells. Any values specified in the What or Replacement arguments of either the Find or Replace methods will involve an And condition. For example, if you are searching for the word "wrox" and have set the FindFormat property to search for Bold, only those cells containing both will be found.

When searching for formats, make sure the SearchFormat argument of the Find method is set to True. When replacing formats, make sure the ReplaceFormat argument of the Replace method is set to True.

When you want to search for formats only, make sure the What argument of the Find method contains nothing. When you only want to replace formats, make sure the Replace argument of the Replace method contains nothing.

When replacing one format with another, make sure you explicitly specify formats you no longer want. For example, if you are searching for cells containing both bold and red and want to replace both formats with just blue, you'll need to make sure you set the bold property of the ReplaceFormat property to False. If you don't, you'll end up with blue and bold text.

When you need to search or replace using different format settings (or none at all), be sure to use the Clear method of either the CellFormat object—if you've declared a variable as such, or by directly accessing the Clear methods of the FindFormat and ReplaceFormat properties. Setting the SearchFormat and ReplaceFormat arguments to False for the Find and Replace methods will *not* prevent the FindFormat and/or ReplaceFormat settings from being used.

CellFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CellFormat Properties

Name	Returns	Description
AddIndent	Variant	Gets whether the text in a cell is automatically indented when the text alignment is set to equal distribution either horizontally or vertically
Borders	Borders	Set/Get the search criteria based on the cell's border format

Name	Returns	Description
Font	Font	Set/Get the search criteria based on the cell's font format
Formula Hidden	Variant	Gets whether the formula will be hidden when the worksheet is protected. Returns Null if the specified range contains some cells with hidden formulas and some cells without
Horizontal Alignment	Variant	Set/Get the horizontal alignment for the specified object
IndentLevel	Variant	Set/Get the indent level for the cell or range
Interior	Interior	Set/Get the search criteria based on the cell's interior format
Locked	Variant	Set/Get whether cells in the range can be modified if the sheet is protected. Returns Null if only some of the cells in the range are locked
MergeCells	Variant	Returns True if the range or style contains merged cells
NumberFormat	Variant	Set/Get the number format associated with the cells in the range. Null if all the cells don't have the same format
NumberFormat Local	Variant	Set/Get the number format associated with the cells in the range in the language of the end user. Null if all the cells don't have the same format
Orientation	Variant	Set/Get the text orientation for the cell text. A value from −90 to 90 degrees can be specified, or use an XlOrientation constant
ShrinkToFit	Variant	Set/Get whether the cell text will automatically shrink to fit the column width. Returns Null if the rows in the range have different ShrinkToFit properties
Vertical Alignment	Variant	Set/Get how the cells in the range are vertically aligned. Use the XLVAlign constants
WrapText	Variant	Set/Get whether cell text wraps in the cell. Returns Null if the cells in the range contain different text wrap properties

CellFormat Methods

Name	Returns	Parameters	Description
Clear			Removes the criteria set in the FindFormat and ReplaceFormat properties

Example: CellFormat Object

The following routine searches through the used range in a worksheet replacing any cells containing both a Tahoma font and a light blue background with Arial light green background:

```
Sub ReplaceFormats()

    Dim oCellFindFormat As CellFormat
    Dim oCellReplaceFormat As CellFormat
    Dim rngReplace As Boolean, sMessage As String

    'Define variables for Find and Replace formats
    Set oCellFindFormat = Application.FindFormat
    Set oCellReplaceFormat = Application.ReplaceFormat

    'Set the Search criteria for the Find Formats
    With oCellFindFormat
         .Clear
         .Font.Name = "Tahoma"
         .Interior.ColorIndex = 34
    End With

    'Set the Replace criteria for the Replace Formats
    With oCellReplaceFormat
         .Clear
         .Font.Name = "Arial"
         .Interior.ColorIndex = 35
    End With

    'Perform the replace
    wksAllowEditRange.UsedRange.Replace What:="", Replacement:="", _
                                        SearchFormat:=True, _
                                        ReplaceFormat:=True

    'Reset the Find and Replace formats
    oCellFindFormat.Clear
    oCellReplaceFormat.Clear

End Sub
```

Characters Object

The `Characters` object allows access to individual characters in a string of text. Characters can have some of the visual properties modified with this object. Possible parents of the `Characters` object are the `AxisTitle`, `ChartTitle`, `DataLabel`, and the `Range` object. Each of the parent objects can use the `Characters([Start],[Length])` property to access a part of their respective texts. The `Start` parameter can specify which character to start at and the `Length` parameter can specify how many to take from the `Start` position.

Characters Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Characters Properties

Name	Returns	Description
Caption	String	Set/Get the full string contained in the Characters object
Count	Long	Read-only. Returns the number of characters in the object
Font	Font	Read-only. Returns an object allowing manipulation of the character's font
Phonetic Characters	String	Set/Get the phonetic characters contained in the Characters object
Text	String	Set/Get the full string contained in the Characters object

Characters Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the characters in the collection
Insert	Variant	String As String	Replaces the characters in the collection with the specified string

Example: Characters Object

This example formats all the capital letters in the active cell in red with 16 point bold text:

```
Sub FormatCellCapitals()
   Dim sText As String
   Dim oChars As Characters
   Dim i As Integer
   'Get the text of the active cell
   sText = ActiveCell.Text
   'Loop through the text
   For i = 1 To Len(sText)
      'Is this character a capital letter?
      If Asc(Mid(sText, i, 1)) > 64 And Asc(Mid(sText, i, 1)) < 91 Then

         'Yes, so get the Characters object
         Set oChars = ActiveCell.Characters(i, 1)

         'Format the Characters object in Red, 16pt Bold.
         With oChars
            .Font.Color = RGB(255, 0, 0)
            .Font.Size = 16
            .Font.Bold = True
         End With
      End If
   Next
End Sub
```

Appendix A

Chart Object and the Charts Collection

The Charts collection holds the collection of chart sheets in a workbook. The Workbook object is always the parent of the Charts collection. The Charts collection only holds the chart sheets. Individual charts can also be embedded in worksheets and dialog sheets. The Chart objects in the Charts collection can be accessed using the Item property. Either the name of the chart can be specified as a parameter to the Item's parameter or an index number describing the position of the chart in the workbook (from left to right).

The Chart object allows access to all of the attributes of a specific chart in Excel. This includes chart formatting, chart types, and other charting properties. The Chart object also exposes events that can be used programmatically.

The Charts collection has a few properties and methods besides the typical collection attributes. These are listed in the following table.

Charts Collection Properties and Methods

Name	Returns	Description
Count	Long	Read-only. Returns the number of charts in the collection
HPageBreaks	HPageBreaks	Read-only. Returns a collection holding all the horizontal page breaks associated with the Charts collection
VPageBreaks	VPageBreaks	Read-only. Returns a collection holding all the vertical page breaks associated with the Charts collection
Visible	Variant	Set/Get whether the charts in the collection are visible. Also can set this to xlVeryHidden to not allow a user to make the charts in the collection visible
Add	Chart	Method. Parameters: [Before], [After], [Count] Adds a chart to the collection. You can specify where the chart goes by choosing which sheet object will be before the new chart object (Before parameter) or after the new chart (After parameter). The Count parameter decides how many charts are created
Copy		Method. Parameters: [Before], [After]. Adds a new copy of the currently active chart to the position specified at the Before or After parameters
Delete		Method. Deletes all the charts in the collection
Move		Method. Parameters: [Before], [After]. Moves the current chart to the position specified by the parameters

Name	Returns	Description
PrintOut		Method. Parameters: [From], [To], [Copies], [Preview], [ActivePrinter], [PrintToFile], [Collate], [PrToFileName]. Prints out the charts in the collection. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the PrintToFile and PrToFileName parameters. The From and To parameters can be used to specify the range of printed pages
PrintPreview		Method. Parameters: [EnableChanges]. Displays the current chart in the collection in a print preview mode. Set the EnableChanges parameter to False to disable the Margins and Setup buttons, hence not allowing the viewer to modify the chart's page setup
Select		Method. Parameters: [Replace]. Selects the current chart in the collection

Chart Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Chart Properties

Name	Returns	Description
Area3DGroup	ChartGroup	Read-only. Returns a ChartGroup object containing the area chart group for a 3D chart
AutoScaling	Boolean	Set/Get whether Excel will stretch a 3D chart to match its 2D chart equivalent. RightAngleAxes must be true
Bar3DGroup	ChartGroup	Read-only. Returns a ChartGroup object containing the bar chart group for a 3D chart
BarShape	XlBarShape	Set/Get the basic shape used in 3D bar or column charts (for example, box, cylinder, pyramid, etc.)
ChartArea	ChartArea	Read-only. Returns the part of a chart containing axes, titles, legends, and formatting properties
ChartTitle	ChartTitle	Read-only. Returns an object manipulating the chart title's properties. Use with the HasTitle property
ChartType	XlChart Type	Set/Get what the type of chart is. This property determines what other chart properties are valid. For example, if the ChartType is set to xl3DBarClustered then the Bar3DGroup property can be used to access the chart group properties

Continues

Name	Returns	Description
CodeName	String	Read-only. Returns the programmatic name of the chart set at design-time in the VBA editor
Column3DGroup	ChartGroup	Read-only. Returns a ChartGroup object containing the column chart group for a 3D chart
Corners	Corners	Read-only. Returns an object holding all the corners of a 3D chart
DataTable	DataTable	Read-only. Returns an object to manipulate a chart's data table
DepthPercent	Long	Set/Get the percentage that a 3D chart depth (y-axis) is in relation to its width (x-axis)
Display BlanksAs	XlDisplay BlanksAs	Set/Get how blank cells are treated when plotting data in a chart. (for example, xlNotPlotted, xlZero, or xlInterpolated)
Elevation	Long	Set/Get what angle of elevation, in degrees, the viewer sees a 3D chart. Valid degrees vary depending on the type of 3D chart
Floor	Floor	Read-only. Returns an object with the formatting properties of the floor (base) of a 3D chart
GapDepth	Long	Set/Get the percentage depth of a data series in relation to the marker width
HasAxis	Variant	Parameters: [Index1], [Index2]. Set/Get whether axes exist for the chart. The parameters can be used to specify the axis type (using the xlAxisType constants with the first parameter) and the axis group (using the xlAxisGroup constants with the second parameter)
HasDataTable	Boolean	Set/Get whether a data table is associated (and therefore displayed). Use with the DataTable property
HasLegend	Boolean	Set/Get whether the chart has a legend. Use with the Legend property
HasPivot Fields	Boolean	Set/Get whether PivotChart controls are displayed for the PivotChart. Can only set to True if using a PivotChart report
HasTitle	Boolean	Set/Get whether the chart has a title. Use with the ChartTitle property
Height Percent	Long	Set/Get the percentage that a 3D chart height (z-axis) is in relation to its width (x-axis)
Hyperlinks	Hyperlinks	Read-only. Returns the collection of hyperlinks associated with the chart
Index	Long	Read-only. Returns the spot in the parent collection where the current chart is located

Name	Returns	Description
`Legend`	`Legend`	Read-only. Returns the formatting properties for a `Legend`. Use with the `HasLegend` property
`Line3DGroup`	`ChartGroup`	Read-only. Returns a `ChartGroup` object containing the line chart group for a 3D chart
`MailEnvelope`	`MsoEnvelope`	Set/Get the e-mail header for a document
`Name`	`String`	Set/Get the name of the chart
`Next`	`Object`	Read-only. Returns the next sheet in the workbook (from left to right) as an object
`PageSetup`	`PageSetup`	Read-only. Returns an object to manipulate the page setup properties for the chart
`Perspective`	`Long`	Sets the perspective, in degrees, that a 3D chart will be viewed as if the `RightAngleAxes` property is set to `False`
`Pie3DGroup`	`ChartGroup`	Read-only. Returns a `ChartGroup` object containing the pie chart group for a 3D chart
`PivotLayout`	`Pivot Layout`	Read-only. Returns an object to manipulate the location of fields for a `PivotChart` report
`PlotArea`	`PlotArea`	Read-only. Returns an object to manipulate formatting, gridlines, data markers and other visual items for the area where the chart is actually plotted. Inside the chart area
`PlotBy`	`XlRowCol`	Set/Get whether columns in the original data are used as individual data series (`xlColumns`) or if the rows in the original data are used as data series (`xlRows`)
`PlotVisible Only`	`Boolean`	Set/Get whether only visible cells are plotted or if invisible cells are plotted too (`False`)
`Previous`		Read-only. Returns the previous sheet in the workbook (from right to left) as an object
`Protect Contents`	`Boolean`	Read-only. Returns whether the chart and everything in it is protected from changes
`ProtectData`	`Boolean`	Set/Get whether the source data can be redirected for a chart
`Protect Drawing Objects`	`Boolean`	Read-only. Returns whether the shapes in the chart can be modified (`ProtectDrawingObjects = False`)
`Protect Formatting`	`Boolean`	Set/Get whether formatting can be changed for a chart

Continues

Name	Returns	Description
ProtectGoalSeek	Boolean	Set/Get whether the user can modify the points on a chart with a mouse action
ProtectionMode	Boolean	Read-only. Returns whether protection has been applied to the user interface. Even if a chart has user interface protection on, any VBA code associated with the chart can still be accessed
ProtectSelection	Boolean	Set/Get whether parts of a chart can be selected and if shapes can be put into a chart
RightAngleAxes	Variant	Set/Get whether axes are fixed at right angles for 3D charts even if the perspective of the chart changes
Rotation	Variant	Set/Get what angle of rotation around the z-axis, in degrees, the viewer sees on a 3D chart. Valid degrees vary depending on the type of 3D chart
Scripts	Scripts	Read-only. Returns the collection of VBScript code associated with a chart (typically to later use on Web pages)
Shapes	Shapes	Read-only. Returns all the shapes contained by the chart
ShowWindow	Boolean	Set/Get whether an embedded chart is shown in a separate window and not as an embedded object in the worksheet
SizeWithWindow	Boolean	Set/Get whether chart sheets automatically change sizes to match the window size
SurfaceGroup	ChartGroup	Read-only. Returns a ChartGroup object containing the surface chart group for a 3D chart
Tab	Tab	Read-only. Returns a Tab object for a chart or a worksheet
Visible	XlSheetVisibility	Set/Get whether the chart is visible or not. The Visible property can also be set to xlVeryHidden to make the chart inaccessible to the end user
Walls	Walls	Read-only. Returns an object to manipulate the formatting of the walls on a 3D chart
WallsAndGridlines2D	Boolean	Set/Get whether gridlines and walls are drawn in a 2D manner on a 3D bar charts, 3D stacked area charts, and 3D clustered column charts

Chart Methods

Name	Returns	Parameters	Description
Activate			Activates the chart making it the ActiveChart
ApplyCustom Type		ChartType As XlChartType, [TypeName]	Changes the chart type to the one specified in the ChartType parameter. If the ChartType is xlUserDefined then the second parameter can specify the custom chart type name
ApplyData Labels		[Type As Xl DataLabels Type], [Legend Key], [Auto Text], [Has LeaderLines], [ShowSeries Name], [Show CategoryName], [ShowValue], [Show Percentage], [ShowBubble Size], [Separator]	Sets the point labels for a chart. The Type parameter specifies whether no label, a value, a percentage of the whole, or a category label is shown. The legend key can appear by the point by setting the LegendKey parameter to True
AreaGroups	Object	[Index]	Returns either a single area chart group (ChartGroup) or a collection of area chart groups (ChartGroups) for a 2D chart
Axes	Object	Type, AxisGroup As XlAxisGroup	Returns the Axis object or the Axes collection for the associated chart. The type of axis and the axis group can be specified with the parameters
BarGroups	Object	[Index]	Returns either a single bar chart group (ChartGroup) or a collection of bar chart groups (ChartGroups) for a 2D chart
ChartGroups	Object	[Index]	Returns either a single chart group (ChartGroup) or a collection of chart groups (ChartGroups) for a chart
ChartObjects	Object	[Index]	Returns either a single embedded chart (ChartObject) or a collection of embedded charts (ChartObjecs) in a chart

Continues

Name	Returns	Parameters	Description
`ChartWizard`		`[Source], [Gallery], [Format], [PlotBy], [Category Labels], [Series Labels], [HasLegend], [Title], [Category Title], [ValueTitle], [ExtraTitle]`	A single method to modify the key properties associated with a chart. Specify the properties that you want to change. The `Source` specifies the data source. `Gallery` specifies the chart type. `Format` can specify one of the 10 built-in chart auto-formats. The rest of the parameters set up how the source will be read, the source of category labels, the source of the series labels, whether a legend appears, and the titles of the chart and the axis. If `Source` is not specified this method can only be used if the sheet containing the chart is active
`CheckSpelling`		`[Custom Dictionary], [Ignore Uppercase], [Always Suggest], [SpellLang]`	Checks the spelling of the text in the chart. A custom dictionary can be specified (`CustomDictionary`), all *uppercase* words can be ignored (`IgnoreUppercase`), and Excel can be set to display a list of suggestions (`AlwaysSuggest`)
`ColumnGroups`	`Object`	`[Index]`	Returns either a single column chart group (`ChartGroup`) or a collection of column chart groups (`ChartGroups`) for a 2D chart
`Copy`		`[Before], [After]`	Adds a new copy of the chart to the position specified at the `Before` or `After` parameters
`CopyPicture`		`[Appearance As XlPicture Appearance], [Format As Xl-CopyPicture Format], [Size As XlPicture Appearance]`	Copies the chart into the clipboard as a picture. The `Appearance` parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The `Format` parameter can specify the type of picture that will be put into the clipboard. The `Size` parameter is used when dealing with chart sheets to describe the size of the picture

Name	Returns	Parameters	Description
CreatePublisher		Edition, Appearance As XlPicture Appearance, Size As XlPictureAppearance, [Contains PICT], [Contains BIFF], [Contains RTF], [Contains VALU]	Used on the Macintosh to create an image of the chart in a standard format. Equivalent to CopyPicture on the PC
Delete			Deletes the chart
Deselect			Unselects the chart object
DoughnutGroups	Object	[Index]	Returns either a single doughnut chart group (ChartGroup) or a collection of doughnut chart groups (ChartGroups) for a 2D chart
Evaluate	Variant	Name	Evaluates the Name string expression as if it were entered into a worksheet cell
Export	Boolean	Filename As String, [FilterName], [Interactive]	Saves the chart as a picture (jpg or gif format) at the name specified by Filename
GetChartElement		x As Long, y As Long, ElementID As Long, Arg1 As Long, Arg2 As Long	Returns what is located at the coordinates x and y of the chart. Only the first two parameters are sent. Variables must be put in the last three parameters. After the method is run, the last three parameters can be checked for return values. The ElementID parameter will return one of the XlChartItem parameters. The Arg1 and Arg2 parameters may or may not hold data depending on the type of element
LineGroups	Object	[Index]	Returns either a single line chart group (ChartGroup) or a collection of line chart groups (ChartGroups) for a 2D chart

Continues

Name	Returns	Parameters	Description
`Location`	`Chart`	`Where As XlChart Location, [Name]`	Moves the chart to the location specified by the `Where` and `Name` parameters. The `Where` can specify if the chart is moving to become a chart sheet or an embedded object
`Move`		`[Before], [After]`	Moves the chart to the position specified by the parameters
`OLEObjects`	`Object`	`[Index]`	Returns either a single `OLE` Object (`OLEObject`) or a collection of `OLE` objects (`OLEObjects`) for a chart
`Paste`		`[Type]`	Pastes the data or pictures from the clipboard into the chart. The `Type` parameter can be used to specify if only formats, formulas or everything is pasted
`PieGroups`	`Object`	`[Index]`	Returns either a single pie chart group (`ChartGroup`) or a collection of pie chart groups (`ChartGroups`) for a 2D chart
`PrintOut`		`[From],[To], [Copies], [Preview], [Active Printer], [PrintToFile], [Collate], [PrToFile Name]`	Prints out the chart. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file by using the `PrintToFile` and `PrToFileName` parameters. The `From` and `To` parameters can be used to specify the range of printed pages
`PrintPreview`		`[Enable Changes]`	Displays the current chart in the collection in a print preview mode. Set the `EnableChanges` parameter to `False` to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup

Name	Returns	Parameters	Description
Protect		[Password], [Drawing Objects], [Contents], [Scenarios], [User Interface Only]	Protects the chart from changes. A case-sensitive Password can be specified. Also, whether shapes are protected (DrawingObjects), the entire contents are protected (Contents), and whether only the user interface is protected (UserInterfaceOnly)
RadarGroups	Object	[Index]	Returns either a single radar chart group (ChartGroup) or a collection of radar chart groups (ChartGroups) for a 2D chart
Refresh			Refreshes the chart with the data source
SaveAs		Filename As String, [FileFormat], [Password], [WriteRes Password], [ReadOnly Recommended], [Create Backup], [AddToMru], [Text Codepage], [TextVisual Layout], [Local]	Saves the current chart into a new workbook with the file name specified by the Filename parameter. A file format, password, write-only password, creation of backup files, and other properties of the saved file can be specified with the parameters
Select		[Replace]	Selects the chart
Series Collection	Object	[Index]	Returns either a single series (Series) or a collection of series (SeriesCollection) for a chart
SetBackground Picture		FileName As String	Sets the chart's background to the picture specified by the FileName parameter
SetSourceData		Source As Range, [PlotBy]	Sets the source of the chart's data to the range specified by the Source parameter. The PlotBy parameter uses the XlRowCol constants to choose whether rows or columns of data will be plotted

Continues

Name	Returns	Parameters	Description
Unprotect		[Password]	Deletes the protection set up for a chart. If the chart was protected with a password, the password must be specified now
XYGroups	Object	[Index]	Returns either a single scatter chart group (ChartGroup) or a collection of scatter chart groups (ChartGroups) for a 2D chart

Chart Events

Name	Parameters	Description
Activate		Triggered when a chart is made to have focus
BeforeDouble Click	ElementID As XlChartItem, Arg1 As Long, Arg2 As Long, Cancel As Boolean	Triggered just before a user double-clicks a chart. The element that was double-clicked in the chart is passed in to event procedure as ElementID. The Arg1 and Arg2 parameters may or may not hold values depending on the ElementID. The double-click action can be canceled by setting the Cancel parameter to True
BeforeRight Click	Cancel As Boolean	Triggered just before a user right-clicks a chart. The right-click action can be canceled by setting the Cancel parameter to True
Calculate		Triggered after new or changed data is plotted on the chart
Deactivate		Triggered when the chart loses focus
DragOver		Triggered when a cell range is dragged on top of a chart. Typically used to change the mouse pointer or give a status message
DragPlot		Triggered when a cell range is dropped onto a chart. Typically used to modify chart attributes
MouseDown	Button As XlMouse Button, Shift As Long, x As Long, y As Long	Triggered when the mouse button is pressed down on a chart. Which mouse button is pressed is passed in with the Button parameter. The Shift parameter holds information regarding the state of the Shift, Ctrl, and Alt keys. The x and y parameters hold the x and y coordinates of the mouse pointer

Name	Parameters	Description
`MouseMove`	`Button As XlMouse Button, Shift As Long, x As Long, y As Long`	Triggered when the mouse is moved on a chart. Which mouse button is pressed is passed in with the `Button` parameter. The `Shift` parameter holds information regarding the state of the Shift, Ctrl, and Alt keys. The `x` and `y` parameters hold the x and y coordinates of the mouse pointer
`MouseUp`	`Button As XlMouse Button, Shift As Long, x As Long, y As Long`	Triggered when the mouse button is released on a chart. Which mouse button is pressed is passed in with the `Button` parameter. The `Shift` parameter holds information regarding the state of the Shift, Ctrl, and Alt keys. The `x` and `y` parameters hold the x and y coordinates of the mouse pointer
`Resize`		Triggered when the chart is resized
`Select`	`ElementID As XlChartItem, Arg1 As Long, Arg2 As Long`	Triggered when one of the elements in a chart is selected. The element that was selected in the chart is passed in to event procedure as `ElementID`. The `Arg1` and `Arg2` parameters may or may not hold values depending on the `ElementID`
`SeriesChange`	`SeriesIndex As Long, PointIndex As Long`	Triggered when the value of a point on a chart is changed. `SeriesIndex` returns the location of the series in the chart series collection. `PointIndex` returns the point location in the series

Example: Chart Object and the Charts Collection

This example creates a 3D chart from the table containing the active cell, formats it and saves a picture of it as a `.jpg` image:

```
Sub CreateAndExportChart()
   Dim oCht As Chart
   'Create a new (blank) chart
   Set oCht = Charts.Add
   'Format the chart
   With oCht
      .ChartType = xl3DColumnStacked
      'Set the data source and plot by columns
      .SetSourceData Source:=Selection.CurrentRegion, PlotBy:=xlColumns

      'Create a new sheet for the chart
      .Location Where:=xlLocationAsNewSheet
      'Size and shape matches the window it's in
      .SizeWithWindow = True
      'Turn of stretching of chart
      .AutoScaling = False
      'Set up a title
      .HasTitle = True
      .ChartTitle.Caption = "Main Chart"
      'No titles for the axes
      .Axes(xlCategory).HasTitle = False
      .Axes(xlSeries).HasTitle = False
      .Axes(xlValue).HasTitle = False
      'Set the 3D view of the chart
      .RightAngleAxes = False
      .Elevation = 50   'degrees
      .Perspective = 30 'degrees
      .Rotation = 20    'degrees
      .HeightPercent = 100
      'No data labels should appear
      .ApplyDataLabels Type:=xlDataLabelsShowNone
      'Save a picture of the chart as a jpg image
      .Export "c:\" & .Name & ".jpg", "jpg", False
   End With
End Sub
```

ChartArea Object

The ChartArea object contains the formatting options associated with a chart area. For 2D charts ChartArea includes the axes, axes titles and chart titles. For 3D charts, ChartArea includes the chart title and its legend. The part of the chart where data is plotted (plot area) is not part of the ChartArea object. Please see the PlotArea object for formatting related to the plot area. The parent of the ChartArea is always the Chart object.

ChartArea Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ChartArea Properties

Name	Returns	Description
AutoScaleFont	Variant	Set/Get whether the font size changes in the ChartArea whenever the Chart changes sizes
Border	Border	Read-only. Returns the border’s attributes around the selected chart area
Fill	ChartFillFormat	Read-only. Returns an object to manipulate the fill attributes of the chart area
Font	Font	Read-only. Returns access to Font properties such as Type and Size
Height	Double	Set/Get the height of the chart area in points
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the chart area (for example, interior color)

Name	Returns	Description
`Left`	`Double`	Set/Get the left edge of the chart area in relation to the chart in points
`Name`	`String`	Read-only. Returns the name of the chart area
`Shadow`	`Boolean`	Set/Get whether a shadow effect appears around the chart area
`Top`	`Double`	Set/Get the top edge of the chart area in relation to the chart in points
`Width`	`Double`	Set/Get the width of the chart area in points

ChartArea Methods

Name	Returns	Parameters	Description
`Clear`	`Variant`		Clears the chart area
`ClearContents`	`Variant`		Clears the data from the chart area without affecting formatting
`ClearFormats`	`Variant`		Clears the formatting from the chart area without affecting the data
`Copy`	`Variant`		Copies the chart area into the clipboard
`Select`	`Variant`		Activates and selects the chart area

Example: ChartArea Object

Apply formatting to the chart area:

```
Sub FormatChartArea()
   Dim oCA As ChartArea
   Set oCA = Charts(1).ChartArea
   With oCA
      .Border.LineStyle = xlContinuous
      .Fill.PresetTextured msoTextureCanvas
      .Fill.Visible = msoTrue
   End With
End Sub
```

ChartColorFormat Object

The `ChartColorFormat` object describes a color of the parent `ChartFillFormat`. For example, the `ChartFillFormat` object contains a `BackColor` property that returns a `ChartColorFormat` object to set the color.

ChartColorFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

ChartColorFormat Properties

Name	Returns	Description
`RGB`	`Long`	Read-only. Returns the red-green-blue value associated with color
`SchemeColor`	`Long`	Set/Get the color of `ChartColorFormat` using an index value corresponding to the current color scheme
`Type`	`Long`	Read-only. Returns whether the color is an RGB, mixed, or scheme type

Example: ChartColorFormat Object

This example sets a chart's fill pattern to built-in colour number 6, then displays the RGB values for the color.

```
Sub SetChartColorFormat()
   Dim oCCF As ChartColorFormat
   With Charts(3).PlotArea.Fill
      'Make sure we're using a Fill pattern
      .Visible = True

      'Get the ChartColorFormat for the ForeColor
      Set oCCF = .ForeColor
      'Set it to built-in colour #6
      oCCF.SchemeColor = 6

      'Read off colour 6's RGB values
      MsgBox "ForeColor #6 RGB is:" & vbCrLf & _
      "Red = " & ((oCCF.RGB And &HFF0000) / &H10000) & vbCrLf & _
      "Green = " & ((oCCF.RGB And &HFF00) / &H100) & vbCrLf & _
      "Blue = " & ((oCCF.RGB And &HFF))
   End With
End Sub
```

ChartFillFormat Object

The `ChartFillFormat` object represents the fill formatting associated with its parent object. This object allows manipulation of foreground colors, background colors and patterns associated with the parent object.

ChartFillFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

ChartFillFormat Properties

Name	Returns	Description
BackColor	ChartColor Format	Read-only. Returns the background color through the ChartColorFormat object
ForeColor	ChartColor Format	Read-only. Returns the foreground color through the ChartColorFormat object
GradientColor Type	Mso Gradient ColorType	Read-only. Returns what type of gradient fill color concept is used
GradientDegree	Single	Read-only. Returns how dark or light the gradient fill is
GradientStyle	Mso Gradient Style	Read-only. Returns the orientation of the gradient that is used
Gradient Variant	Long	Read-only. Returns the variant used for the gradient from the center
Pattern	MsoPattern Type	Read-only. Returns the pattern used for the fill, if any
PresetGradient Type	MsoPreset Gradient Type	Read-only. Returns the type of gradient that is used
PresetTexture	MsoPreset Texture	Read-only. Returns the non-custom texture of the fill
TextureName	String	Read-only. Returns the custom texture name of the fill
TextureType	MsoTexture Type	Read-only. Returns whether the texture is custom, preset, or mixed
Type	MsoFillType	Set/Get how transparent the fill is. From 0 (opaque) to 1 (clear)
Visible	MsoTriState	Read-only. Returns if the fill is a texture, gradient, solid, background, picture or mixed

ChartFillFormat Methods

Name	Returns	Parameters	Description
OneColor Gradient		Style As MsoGradient Style, Variant As Long, Degree As Single	Sets the style, variant and degree for a one-color gradient fill

Continues

Name	Returns	Parameters	Description
Patterned		Pattern As MsoPattern Type	Set the pattern for a fill
Preset Gradient		Style As MsoGradient Style, Variant As Long, Preset GradientType As MsoPreset GradientType	Choose the style, variant, and preset gradient type for a gradient fill
Preset Textured		PresetTexture As MsoPreset Texture	Set the preset texture for a fill
Solid			Set the fill to a solid color
TwoColor Gradient		Style As MsoGradient Style, Variant As Long	Set the style for a two-color gradient fill
UserPicture		[PictureFile], [Picture Format], [Picture StackUnit], [Picture Placement]	Set the fill to the picture in the PictureFile format
UserTextured		TextureFile As String	Set the custom texture for a fill with the TextureFile format

Example: ChartFillFormat Object

```
Sub FormatPlotArea()
   Dim oCFF As ChartFillFormat
   'Get the ChartFillFormat for the plot area
   Set oCFF = ActiveSheet.ChartObjects(1).Chart.PlotArea.Fill
   'Format the fill area
   With oCFF
      .TwoColorGradient Style:=msoGradientDiagonalUp, Variant:=1
      .Visible = True
      .ForeColor.SchemeColor = 6
      .BackColor.SchemeColor = 7
   End With
End Sub
```

ChartGroup Object and the ChartGroups Collection

The ChartGroups collection holds all the plotting information associated with the parent chart. A chart can have more than one ChartGroup associated with it. For example, a single chart can contain both a line and a bar chart associated with it. The ChartGroups property of the Chart object can be used to access the ChartGroups collection. Also, the PieGroups and LineGroups properties of the Chart object will also return only chart groups of pie chart types and line chart types, respectively.

Besides the typical properties associated with a collection, the ChartGroups collection also has a Count property that returns the number of ChartGroup objects in the collection. The parent of the ChartGroups collection or the ChartGroup object is the Chart object.

The ChartGroup object includes all of the plotted points associated with a particular chart type. A ChartGroup can hold many series of points (each column or row of the original data). Each series can contain many points (each cell of the original data). A Chart can contain more than one ChartGroup associated with it. The Bar3DGroup, Column3DGroup, Line3DGroup, Pie3DGroup, and the SurfaceGroup properties of the Chart object can be used to access a particular chart group of the corresponding chart type. The AreaGroups, BarGroups, ColumnGroups, DoughnutGroups, LineGroups, PieGroups, RadarGroups, and XYGroups methods of the Chart object can be used to return either a ChartGroup object or a ChartGroups collection.

ChartGroup Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ChartGroup Properties

Name	Returns	Description
AxisGroup	XlAxis Group	Set/Get whether the chart group is primary or secondary
BubbleScale	Long	Set/Get the percentage increase in the size of bubbles from the default size. Valid values from 0 to 300 percent. Valid only for bubble chart group
DoughnutHole Size	Long	Set/Get how large the hole in a doughnut chart group is. The value is a percentage of the size of the chart. Valid values from 10 to 90 percent. Valid only on doughnut chart groups
DownBars	DownBars	Read-only. Returns an object to manipulate the formatting options of down bars on a line chart group. Valid only on line chart groups. Use with the HasUpDownBars property
DropLines	DropLines	Read-only. Returns an object to manipulate the formatting options of drop lines on a line or area chart group. Valid only on line or area chart groups. Use with the HasDropLines property

Continues

Name	Returns	Description
FirstSlice Angle	Long	Set/Get what angle to use for the first slice of a pie or doughnut chart groups (the first data point plotted on the chart)
GapWidth	Long	Set/Get how big to make the gap between the columns of different data series. Also, when dealing with Bar of Pie charts or Pie of Pie charts, the GapWidth describes the distance from the main chart to the secondary chart (when the ChartType is xlPieOfPie or xlBarOfPie for the parent chart)
Has3Dshading	Boolean	Set/Get whether 3D shading is applied to the chart group visuals
HasDropLines	Boolean	Set/Get whether the chart group has drop lines. Use with the DownLines property
HasHiLoLines	Boolean	Set/Get whether the chart group has high-low lines. Use with the HiLoLines property
HasRadarAxis Labels	Boolean	Set/Get whether axis labels are put on a radar chart. Valid only for radar chart groups
HasSeries Lines	Boolean	Set/Get whether the chart group has series lines. Use with the SeriesLines property
HasUpDownBars	Boolean	Set/Get whether the chart group has up and down bars. Use with the DownBars and UpBars property
HiLoLines	HiLoLines	Read-only. Returns an object to manipulate the formatting of high-low lines in a line chart. Valid only for line charts
Index	Long	Read-only. Returns the spot in the parent collection that the current ChartGroup object is located
Overlap	Long	Set/Get whether bars and columns in a series will overlap each other or have a gap between them. A value from −100 to 100 can be specified where −100 will put a gap between each bar / column equal to the bar / column width and 100 will stack the bars / columns on top of each other. Valid only for 2D bar and column chart groups
RadarAxis Labels	TickLabels	Read-only. Returns an object to manipulate the formatting and labels associated with radar axis labels. Valid only for radar chart groups
SecondPlot Size	Long	Set/Get the percentage of size of the secondary part of a Pie of Pie or Bar of Pie chart group as a percentage of the main Pie

Name	Returns	Description
`SeriesLines`	`SeriesLines`	Read-only. Returns an object to manipulate the formatting associated with the series lines in a chart group. A series line connects same series of data appearing in a stacked column chart groups, stacked bar chart groups, Pie of Pie chart groups, or Bar of Pie chart groups. Use with the `HasSeriesLines` property
`ShowNegative Bubbles`	`Boolean`	Set/Get whether bubbles with negative data values are shown. Valid only on bubble chart groups
`Size Represents`	`XlSizeRepresents`	Set/Get whether the value of the data points are represented by the size or the area of bubbles on a bubble chart group. Valid only on bubble chart groups
`SplitType`	`XlChartSplitType`	Set/Get how the two charts in Pie of Pie chart group and Bar of Pie chart group are split up. For example, the chart can be split by percentage of value (`xlSplitByPercentValue`) or be split by value (`xlSplitByValue`)
`SplitValue`	`Variant`	Set/Get the value that will be combined in the main pie chart but split up in the secondary chart in a Pie of Pie or Bar of Pie chart group
`UpBars`	`UpBars`	Returns an object to manipulate the formatting options of up bars on a line chart group. Valid only on line chart groups. Use with the `HasUpDownBars` property
`VaryBy Categories`	`Boolean`	Set/Get whether different colors are assigned to different categories in a single series of a chart group. The chart can only contain a single data series for this to work

ChartGroup Methods

Name	Returns	Parameters	Description
`Series Collection`	`Object`	`[Index]`	Returns either a single series (`Series`) or a collection of series (`SeriesCollection`) for a chart

Example: ChartGroup Object and the ChartGroups Collection

This sets the gap width of all column groups in the chart to 10% and set each column to have a different color:

```
Sub FormatColumns()
   Dim oCht As Chart
   Dim oCG As ChartGroup
   For Each oCG In Charts(1).ColumnGroups
      oCG.GapWidth = 10
      oCG.VaryByCategories = True
   Next
End Sub
```

ChartObject Object and the ChartObjects Collection

The ChartObjects collection holds all of the embedded Chart objects in a worksheet, chart sheet, or dialog sheet. This collection does not include the actual chart sheets themselves. Chart sheets can be accessed through the Charts collection. Each Chart in the ChartObjects collection is accessed through the ChartObject object. The ChartObject acts as a wrapper for the embedded chart itself. The Chart property of the ChartObject is used to access the actual chart. The ChartObject object also contains properties to modify the formatting of the embedded chart (for example, Height, Width).

The ChartObjects collection contains many properties besides the typical collection attributes. These properties are listed next.

ChartObjects Collection Properties and Methods

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the collection of chart objects
Count	Long	Read-only. Returns the number of ChartObject objects in the collection
Enabled	Boolean	Set/Get whether any macros associated with each ChartObject object in the collection can be triggered by the user
Height	Double	Set/Get the height of the ChartObject in the collection if there is only one object in the collection
Interior	Interior	Read-only. Returns an object containing options to format the inside area of all the Chart objects in the collection (for example, interior color)
Left	Double	Set/Get the distance from the left edge of the ChartObject to the left edge of the parent sheet. This property only works if there is only one ChartObject in the collection
Locked	Boolean	Set/Get whether the ChartObject is locked when the parent sheet is protected. This property only works if there is only one ChartObject in the collection

Name	Returns	Description
`Placement`	`Variant`	Set/Get how the `ChartObject` object is anchored to the sheet (for example, free floating, move with cells). Use the `XlPlacement` constants to set this property. This property only works if there is only one `ChartObject` in the collection
`PrintObject`	`Boolean`	Set/Get whether the embedded chart on the sheet will be printed when the sheet is printed. This property only works if there is only one `ChartObject` in the collection
`Rounded Corners`	`Boolean`	Set/Get whether the corners of the embedded chart are rounded (`True`) or right angles (`False`). This property only works if there is only one `ChartObject` in the collection
`Shadow`	`Boolean`	Set/Get whether a shadow appears around the embedded chart. This property only works if there is only one `ChartObject` in the collection
`ShapeRange`	`ShapeRange`	Read-only. Returns the `ChartObjects` in the collection as `Shape` objects
`Top`	`Double`	Set/Get the distance from top edge of the `ChartObject` to the top of the parent sheet. This property only works if there is only one `ChartObject` object in the collection
`Visible`	`Boolean`	Set/Get whether all the `ChartObject` objects in the collection are visible
`Width`	`Double`	Set/Get the width of the `ChartObject` in the collection if there is only one `ChartObject` object in the collection
`Add`	`ChartObject`	Method. Parameters: `Left As Double`, `Top As Double`, `Width As  Double`, `Height As Double`. Adds a `ChartObject` to the collection of `ChartObjects`. The position of the new `ChartObject` can be specified by using the `Left`, `Top`, `Width`, and `Height` parameters
`BringToFront`	`Variant`	Method. Brings all the `ChartObject` objects in the collection to the front of all the other objects
`Copy`	`Variant`	Method. Copies all the `ChartObject` objects in the collection into the clipboard
`CopyPicture`	`Variant`	Method. Parameters: `Appearance As XlPictureAppearance`, `Format As XlCopyPictureFormat`. Copies the `Chart` objects in the collection into the clipboard as a picture. The `Appearance` parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The `Format` parameter can specify the type of picture that will be put into the clipboard

Continues

Name	Returns	Description
Cut	Variant	Method. Cuts all the ChartObject objects in the collection into the clipboard
Delete	Variant	Method. Deletes all the ChartObject objects in the collection into the clipboard
Duplicate		Method. Duplicates all the ChartObject objects in the collection into the parent sheet. (for example, if you had two ChartObject objects in the parent sheet and used this method then you would have four ChartObject objects)
Select	Variant	Method. Parameters: [Replace]. Selects all the ChartObject objects in the collection
SendToBack	Variant	Method. Brings the ChartObject objects in the collection to the back of other objects

ChartObject Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ChartObject Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the embedded chart
BottomRight Cell	Range	Read-only. Returns the single cell range located under the lower-right corner of the ChartObject
Chart	Chart	Read-only. Returns the actual chart associated with the ChartObject
Enabled	Boolean	Set/Get whether a macro associated with the ChartObject is capable of being triggered
Height	Double	Set/Get the height of embedded chart
Index	Long	Read-only. Returns the position of the ChartObject among the parent collection
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the chart object (for example, interior color)
Left	Double	Set/Get the distance from the left edge of the ChartObject to the left edge of the parent sheet

Name	Returns	Description
`Locked`	`Boolean`	Set/Get whether the `ChartObject` is locked when the parent sheet is protected
`Name`	`String`	Set/Get the name of the `ChartObject`
`Placement`	`Variant`	Set/Get how the `ChartObject` object is anchored to the sheet (for example, free floating, move with cells). Use the `XlPlacement` constants to set this property
`PrintObject`	`Boolean`	Set/Get whether the embedded chart on the sheet will be printed when the sheet is printed
`ProtectChartObject`	`Boolean`	Set/Get whether the embedded chart can change sizes, be moved, or deleted from the parent sheet
`RoundedCorners`	`Boolean`	Set/Get whether the corners of the embedded chart are rounded (`True`) or right angles (`False`)
`Shadow`	`Boolean`	Set/Get whether a shadow appears around the embedded chart
`ShapeRange`	`ShapeRange`	Read-only. Returns the `ChartObject` as a `Shape` object
`Top`	`Double`	Set/Get the distance from top edge of the `ChartObject` to the top of the parent sheet
`TopLeftCell`	`Range`	Read-only. Returns the single cell range located above the top-left corner of the `ChartObject`
`Visible`	`Boolean`	Set/Get whether the `ChartObject` object is visible
`Width`	`Double`	Set/Get the width of embedded chart
`ZOrder`	`Long`	Read-only. Returns the position of the embedded chart among all the other objects on the sheet. The `ZOrder` also matches the location of the `ChartObject` in the parent collection

ChartObject Methods

Name	Returns	Parameters	Description
`Activate`	`Variant`		Makes the embedded chart the active chart
`BringToFront`	`Variant`		Brings the embedded chart to the front of all the other objects on the sheet. Changes the `ZOrder`
`Copy`	`Variant`		Copies the embedded chart into the clipboard

Continues

Name	Returns	Parameters	Description
CopyPicture	Variant	Appearance As XlPicture Appearance, Format As XlCopyPicture Format	Copies the Chart object into the clipboard as a picture. The Appearance parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The Format parameter can specify the type of picture that will be put into the clipboard. The Size parameter is used when dealing with chart sheets to describe the size of the picture
Cut	Variant		Cuts the embedded chart into the clipboard
Delete	Variant		Deletes the embedded chart from the sheet
Duplicate			Duplicates the embedded chart and places the duplicate in the same parent sheet
Select	Variant	[Replace]	Sets focus to the embedded chart
SendToBack	Variant		Sends the embedded object to the back of the other objects on the sheet

Example: ChartObject Object and the ChartObjects Collection

This example creates .jpg images from all the embedded charts in the active worksheet:

```
Sub ExportChartObjects()
   Dim oCO As ChartObject
   For Each oCO In ActiveSheet.ChartObjects
      'Export the chart as a jpg image, giving it the
      'name of the embedded object
      oCO.Chart.Export "c:\" & oCO.Name & ".jpg", "jpg"
   Next
End Sub
```

ChartTitle Object

The ChartTitle object contains all of the text and formatting associated with a chart's title. The parent of the ChartTitle object is the Chart object. This object is usually used along with the HasTitle property of the parent Chart object.

ChartTitle Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ChartTitle Properties

Name	Returns	Description
`AutoScaleFont`	`Variant`	Set/Get whether the font size will change automatically if the parent chart changes sizes
`Border`	`Border`	Read-only. Returns the border's properties around the selected chart title
`Caption`	`String`	Set/Get the chart title's text
`Characters`	`Characters`	Read-only. Parameters: `[Start],[Length]`. Returns an object containing all the characters in the chart title. Allows manipulation on a character-by-character basis
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the chart title
`Font`	`Font`	Read-only. Returns an object containing `Font` options for the chart title
`Horizontal Alignment`	`Variant`	Set/Get how the chart title is horizontally aligned. Use the `xlAlign` constants
`Interior`	`Interior`	Read-only. Returns an object containing options to format the area in the chart title text area (for example, interior color)
`Left`	`Double`	Set/Get the distance from the left edge of the chart title text area to the chart's left edge
`Name`	`String`	Read-only. Returns the name of the chart title object
`Orientation`	`Variant`	Set/Get the angle of the text for the chart title. The value can either be in degrees (from −90 to 90) or one of the `XlOrientation` constants
`ReadingOrder`	`Long`	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages
`Shadow`	`Boolean`	Set/Get whether the chart title has a shadow effect
`Text`	`String`	Set/Get the chart title's text
`Top`	`Double`	Set/Get the distance from the top edge of the chart title text area to the chart's top edge
`Vertical Alignment`	`Variant`	Set/Get how you want the chart title horizontally aligned. Use the `xlVAlign` constants

ChartTitle Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the chart title from the chart
Select	Variant		Selects the chart title on the chart

Example: ChartTitle Object

This example adds a chart title to a chart and formats it:

```
Sub AddAndFormatChartTitle()
   Dim oCT As ChartTitle
   'Make sure the chart has a title
   Charts(1).HasTitle = True
   'Get the ChartTitle object
   Set oCT = Charts(1).ChartTitle
   'Format the chart title
   With oCT
      .Caption = "Hello World"
      .Font.Name = "Times New Roman"
      .Font.Size = 16
      .Characters(1, 1).Font.Color = RGB(255, 0, 0)
      .Characters(7, 1).Font.Color = RGB(255, 0, 0)
      .Border.LineStyle = xlContinuous
      .Border.Weight = xlThin
      .Shadow = True
   End With
End Sub
```

ColorFormat Object

The ColorFormat object describes a single color used by the parent object. Possible parents of the ColorFormat object are the FillFormat, LineFormat, ShadowFormat, and ThreeDFormat objects.

ColorFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ColorFormat Properties

Name	Returns	Description
RGB	Long	Read-only. Returns the red-green-blue value associated with color
SchemeColor	Integer	Set/Get the color of the ColorFormat using an index value corresponding to the current color scheme

Name	Returns	Description
TintAndShade	Single	Set/Get a value that lightens or darkens the color of a specified shape. The values can be from −1 (darkest) to 1 (lightest). Zero is neutral
Type	MsoColor Type	Read-only. Returns whether the color is an RGB, mixed, or scheme type

Example: ColorFormat Object

Set the ForeColor of a shape's fill effect:

```
Sub FormatShapeColour()
   Dim oShp As Shape
   Dim oCF As ColorFormat
   Set oShp = ActiveSheet.Shapes(1)
   Set oCF = oShp.Fill.ForeColor
   oCF.SchemeColor = 53
End Sub
```

Comment Object and the Comments Collection

The Comments collection holds all of the cell comments in the parent Range object. Each Comment object represents a single cell comment.

Comment Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Comment Properties

Name	Returns	Description
Author	String	Read-only. Returns the name of the person who created the comment.
Shape	Shape	Read-only. Returns the comment box as a Shape object allowing manipulation of the comment box.
Visible	Boolean	Set/Get whether the comment is visible all the time (True) or only when the user hovers over the cell containing the comment.

Comment Methods

Name	Returns	Parameters	Description
Delete			Deletes the comment from the cell
Next	Comment		Returns the next cell comment in the parent collection

Continues

Name	Returns	Parameters	Description
`Previous`	`Comment`		Returns the previous cell comment in the parent collection

`Text`	`String`	`[Text],`
		`[Start],`
		`[Overwrite]`
			Sets the text associated with the comment. The `Text` parameter is used to set the comment text. Use the `Start` parameter to specify the starting point for `Text` in the existing comment. Set the `Overwrite` parameter to `True` to overwrite existing text

Example: Comment Object and the Comments Collection

This example removes the user name added by Excel at the start of the comment and formats the comment to make it more readable:

```
Sub FormatComments()
   Dim oComment As Comment, i As Integer
   'Loop through all the comments in the sheet
   For Each oComment In ActiveSheet.Comments
      'Using the text of the comment...
      With oComment.Shape.TextFrame.Characters

         'Find and remove the user name inserted by Excel
         i = InStr(1, .Text, ":" & vbLf)
         If i > 0 Then
            .Text = Mid(.Text, i + 2)
         End If

         'Increase the font size
         With .Font
            .Name = "Arial"
            .Size = 10
            .Bold = False
         End With
      End With

      'Make the text frame auto-fit
      oComment.Shape.TextFrame.AutoSize = True
   Next
End Sub
```

ConnectorFormat Object

The `ConnectorFormat` object represents the connector line used between shapes. This connector line connects two shapes together. If either of the shapes are moved, the connector automatically readjusts so the shapes still look visually connected. The parent of a `ConnectorFormat` object is the `Shape` object.

ConnectorFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

ConnectorFormat Properties

Name	Returns	Description
`Begin Connected`	`MsoTri State`	Read-only. Returns whether the beginning of the connector has a shape attached. Use with `BeginConnectedShape`
`Begin Connected Shape`	`Shape`	Read-only. Returns the shape that is connected to the beginning of the connector. Use with `BeginConnected`
`Begin Connection Site`	`Long`	Read-only. Returns which connection site (connection spot) on the shape that the beginning of the connector is connected to. Use with `BeginConnected`
`EndConnected`	`MsoTri State`	Read-only. Returns whether the end of the connector has a shape attached. Use with `BeginConnectedShape`
`EndConnected Shape`	`Shape`	Read-only. Returns the shape that is connected to the end of the connector. Use with `EndConnected`
`EndConnection Site`	`Long`	Read-only. Returns which connection site (connection spot) on the shape that the end of the connector is connected to. Use with `EndConnected`
`Type`	`Mso Connector Type`	Set/Get what type of connector is being used (for example, `msoConnectorStraight`, `msoConnectorCurve`)

ConnectorFormat Methods

Name	Returns	Parameters	Description
`BeginConnect`		`Connected Shape As Shape, Connection Site As Lon`	Sets the beginning of the connector to the shape specified by the `ConnectedShape` parameter at the connection site specified by the `ConnectionSite` parameter
`Begin Disconnect`			Disconnects the shape that was at the beginning of the connection. This method does not move the connection line
`EndConnect`		`Connected Shape As Shape, Connection Site As Lon`	Sets the end of the connector to the shape specified by the `ConnectedShape` parameter at the connection site specified by the `ConnectionSite` parameter
`End Disconnect`			Disconnects the shape that was at the end of the connection. This method does not move the connection line

Example: ConnectorFormat Object

This example formats all fully-connected connectors as curved lines:

```
Sub FormatConnectors()
   Dim oShp As Shape
   Dim oCF As ConnectorFormat
   'Loop through all the Shapes in the sheet
   For Each oShp In ActiveSheet.Shapes
      'Is it a Connector?
      If oShp.Connector Then

         'Yes, so get the ConnectorFormat object
         Set oCF = oShp.ConnectorFormat

         'If the connector is connected at both ends,
         'make it a curved line.
         With oCF
            If .BeginConnected And .EndConnected Then
               .Type = msoConnectorCurve
            End If
         End With
      End If
   Next
End Sub
```

ControlFormat Object

The `ControlFormat` object contains properties and methods used to manipulate Excel controls such as textboxes and listboxes. This object's parent is always the `Shape` object.

ControlFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

ControlFormat Properties

Name	Returns	Description
`DropDownLines`	`Long`	Set/Get how many lines are displayed in the drop-down part of a combo box. Valid only if the control is a combo box
`Enabled`	`Boolean`	Set/Get whether the control is enabled
`LargeChange`	`Long`	Set/Get the value that is added or subtracted every time the user clicks inside the scroll bar area for a scroll box. Valid only if the control is a scroll box
`LinkedCell`	`String`	Set/Get the range where the results of the control are placed
`ListCount`	`Long`	Read-only. Returns the number of items in the list box of combo box. Valid only for list box and combo box controls
`ListFillRange`	`String`	Set/Get the range that contains the items for a list box or combo box. Valid only for list box and combo box controls

Name	Returns	Description
`ListIndex`	`Long`	Set/Get the item that is currently selected in the list box or combo box. Valid only for list box and combo box controls
`LockedText`	`Boolean`	Set/Get whether the control text can be changed if the workbook is locked
`Max`	`Long`	Set/Get the maximum value allowed for a scroll bar or spinner. Valid only on a control that is a scroll bar or spinner
`Min`	`Long`	Set/Get the minimum value allowed for a scroll bar or spinner. Valid only on a control that is a scroll bar or spinner
`MultiSelect`	`Long`	Set/Get how a list box reacts to user selection. The property can be set to `xlNone` (only one item can be selected), `xlSimple` (each item the user clicks one is added to the selection), or `xlExtended` (the user has to hold down the *Ctrl* key to select multiple items). Valid only on list boxes
`PrintObject`	`Boolean`	Set/Get whether the control will be printed when the sheet is printed
`SmallChange`	`Long`	Set/Get the value that is added or subtracted every time the user clicks on the arrow button associated with the scroll bar. Valid only if the control is a scroll box
`Value`	`Long`	Set/Get the value of the control

ControlFormat Methods

Name	Returns	Parameters	Description
`AddItem`		`TextAs String, [Index]`	Adds the value of the `Text` parameter into a list box or combo box. Valid only for list box and combo box controls
`List`	`Variant`	`[Index]`	Set/Get the string list array associated with a combo box or list box. Can also Set/Get individual items in the list box or combo box if the `Index` parameter is specified. Valid only for list box and combo box controls
`RemoveAll Items`			Removes all the items from a list box or combo box. Valid only for list box and combo box controls
`RemoveItem`		`Index As Long, [Count]`	Removes the item specified by the Index parameter from a list box or combo box. Valid only for list box and combo box controls

Example: ControlFormat Object

This example resets all the list boxes, drop-downs, scrollbars, spinners, and check boxes on the sheet:

```
Sub ResetFormControls()
   Dim oShp As Shape
   Dim oCF As ControlFormat
   'Loop through all the shapes in the sheet
   For Each oShp In ActiveSheet.Shapes
      'Is this a Forms control?
      If oShp.Type = msoFormControl Then

         'Yes, so get the ControlFormat object
         Set oCF = oShp.ControlFormat

         'Reset the control as appropriate
         Select Case oShp.FormControlType
            Case xlListBox, xlDropDown
               oCF.RemoveAllItems

            Case xlSpinner, xlScrollBar
               oCF.Value = oCF.Min

            Case xlCheckBox
               oCF.Value = xlOff

         End Select
      End If
   Next
End Sub
```

Corners Object

The `Corners` object represents the corners of a 3D chart. The parent of the `Corners` object is the `Chart` object. The parent chart must be a 3D chart. Individual corners cannot be accessed.

Corners Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Corners Properties

Name	Returns	Description
`Name`	`String`	Read-only. Returns the name of the `Corners` object—usually `"Corners"`

Corners Methods

Name	Returns	Parameters	Description
`Select`	`Variant`		Sets the corners on the chart

Example: Corners Object

No example—its only method is to select it, which is not particularly useful.

CubeField Object and the CubeFields Collection

The CubeFields collection holds all of the PivotTable report fields based on an OLAP cube. Each CubeField object represents a measure or hierarchy field from the OLAP cube. The parent of the CubeFields collection is the PivotTable object.

The CubeFields collection contains a Count property besides the typical collection attributes. The Count property returns the number of objects in the collection.

CubeFields Collection Methods

Name	Returns	Parameters	Description
AddSet	CubeField	Name As String, Caption As String	Adds a new CubeField object to the CubeFields collection

CubeField Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CubeField Properties

Name	Returns	Description
Caption	String	Read-only. Returns the text label to use for the cube field
CubeField Type	XlCube FieldType	Read-only. Returns whether the cube field is a hierarchy field (xlHierarchy) or a measure field (xlMeasure)
DragToColumn	Boolean	Set/Get whether the field can be dragged to a column position. False for measure fields
DragToData	Boolean	Set/Get whether the field can be dragged to the data position
DragToHide	Boolean	Set/Get whether the field can be dragged off the PivotTable report and therefore hidden
DragToPage	Boolean	Set/Get whether the field can be dragged to the page position. False for measure fields
DragToRow	Boolean	Set/Get whether the field can be dragged to a row position. False for measure fields
Enable Multiple PageItems	Boolean	Set/Get whether multiple items in the page field area for OLAP PivotTables can be selected

Continues

Name	Returns	Description
HasMemberProperties	Boolean	Read-only. Returns True when there are member properties specified to be displayed for the cube field
HiddenLevels	Long	Set/Get the top levels of the hierarchy cube field that are hidden. Set the value to 0 before setting it a value greater than 0 (displays all the levels then hide some)
LayoutForm	XlLayoutFormType	Set/Get the way the specified PivotTable items appear
LayoutSubtotalLocation	XlSubtotalLocationType	Set/Get the position of the PivotTable field subtotals in relation to the specified field
Name	String	Read-only. Returns the name of the field
Orientation	XlPivotFieldOrientation	Set/Get where the field is located in the PivotTable report
PivotFields	PivotFields	Read-only. Returns the PivotFields collection
Position	Long	Set/Get the position number of the hierarchy field among all the fields in the same orientation
ShowInFieldList	Boolean	Set/Get whether a CubeField object will be shown in the field list
TreeviewControl	TreeviewControl	Read-only. Returns an object allowing manipulation of the cube on an OLAP PivotTable report
Value	String	Read-only. Returns the name of the field

CubeField Methods

Name	Returns	Parameters	Description
AddMemberPropertyField		Property As String, [PropertyOrder]	Adds a member property field to the display for the cube field. Note that the property field specified will not be viewable if the PivotTable view has no fields
Delete			Deletes the object

CustomProperty Object and the CustomProperties Collection

This object allows you to store information within a worksheet or SmartTag. This information can then be used as metadata for XML, or can be accessed by any routine that needs information specific to the worksheet or SmartTag.

More important to a developer is the ability of this new object to store specifics regarding a worksheet or group of worksheets so that any routine can call up the CustomProperty, analyze the information contained within, then make decisions on how to handle that worksheet. In the past, many developers used worksheet level range names to store information about a worksheet. Worksheet level range names only reside in that worksheet, enabling each worksheet to have the same range name, but store different values.

For example, each worksheet in a workbook containing a dozen budget worksheets and three report worksheets could contain the same range name called IsBudget. All of the budget sheets would store the value of True in the range name while the report sheets would store False. Routines that need to loop through the worksheets applying different formats or calculations to budget sheets can call on the value of the range name to determine if it's a budget sheet before running code on it.

This new CustomProperty object makes storing such information (or any information for that matter) simpler than creating worksheet level range names, or storing such information in a hidden worksheet or in the Registry.

The CustomProperties collection represents CustomProperty objects for either worksheets or SmartTags. CustomProperties can store information within either a worksheet or SmartTag. They are similar to the DocumentProperties object in the Office XP model, except they are stored with a worksheet or SmartTag instead of the whole document.

CustomProperties Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CustomProperties Collection Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of objects in the collection
Item	Custom Property	Read-only. Index As Variant. Returns a single object from a collection

CustomProperties Collection Methods

Name	Returns	Parameters	Description
Add	Custom Property	Name As String, Value As Variant	Adds custom property information

CustomProperty Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CustomProperty Properties

Name	Returns	Description
`Name`	`String`	Set/Get the name of the object
`Value`	`Variant`	Set/Get the style of line to use for the borders (for example, `xlDash`). Use the `xlLineStyle` constants to set the value. Same as `LineStyle`

CustomProperty Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the object

Example: CustomProperty Object

This routine loops through the worksheets in a workbook and creates a `CustomProperty` called `IsBudget`. The value of `IsBudget` depends on whether or not the worksheet contains the phrase "Budget Analysis". It then lists the results:

```
Sub CreateCustomProperties()
    Dim bBudget As Boolean
    Dim lRow As Long
    Dim oCustomProp As CustomProperty
    Dim rng As Range, wks As Worksheet

    'Turn off the screen and clear the search formats
    With Application
        .FindFormat.Clear
        .ScreenUpdating = False
    End With

    'Clear the worksheet that will contain the
    ' Custom Property list
    wksCustomProperties.UsedRange.Offset(1, 0).ClearContents

    'Initialize the row counter
    lRow = 2    'Row 1 contains the Column Headings

    'Loop through the worksheet in this workbook
    For Each wks In ThisWorkbook.Worksheets

        'Supress errors resulting in no cells found and
        ' no Custom Property
        On Error Resume Next
            bBudget = False
            bBudget = _
                   (Len(wks.UsedRange.Find(What:="Budget
Analysis").Address) > 0)

            'Unfortunately, we cannot refer to a Custom Property by
            ' its name, only its numeric index
            Set oCustomProp = wks.CustomProperties(1)
        On Error GoTo 0

        'If the Custom Property exists, delete it and
        ' add it again
        If Not oCustomProp Is Nothing Then oCustomProp.Delete

        'Note the value of bBudget is encased in double quotes.
        'If we don't, True will be stored as -1 and False 0 (their
        'numeric values).
        Set oCustomProp = wks.CustomProperties.Add(Name:=
"IsBudget", Value:="" _
                                                   & bBudget & "")

        'List the Custom Property settings on the worksheet
        With wksCustomProperties
            'Parent.Name returns the name of the object
            ' holding the Custom Property - the worksheet name in this case
            .Cells(lRow, 1).Value = oCustomProp.Parent.Name
            .Cells(lRow, 2).Value = oCustomProp.Name
            .Cells(lRow, 3).Value = oCustomProp.Value
        End With

        'Move down one row
        lRow = lRow + 1

    Next wks

End Sub
```

CustomView Object and the CustomViews Collection

The CustomViews collection holds the list of custom views associated with a workbook. Each CustomView object holds the attributes associated with a workbook custom view. A custom view holds settings such as window size, window position, column widths, hidden columns, and print settings of a workbook. The parent object of the CustomViews collection is the Workbook object.

The CustomViews collection has two other properties besides the typical collection attributes. The Count property returns the number of CustomView objects in the collection. The Add method adds a custom view to the CustomViews collection. The Add method accepts a name for the view with the ViewName parameter. Optionally, the Add method accepts whether print settings are included (PrintSettings) and whether hidden rows and columns are included (RowColSettings).

CustomView Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

CustomView Properties

Name	Returns	Description
`Name`	`String`	Read-only. Returns the name of the custom view
`PrintSettings`	`Boolean`	Read-only. Returns whether print settings are included in the custom view
`RowCol Settings`	`Boolean`	Read-only. Returns whether hidden rows and columns are included in the custom view

CustomView Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the custom view
`Show`			Shows the custom view and the settings associated with it

Example: CustomView Object and the CustomViews Collection

Display all the custom views in the workbook as a screen-show, pausing for two seconds between each one:

```
Sub ShowCustomView()
   Dim oCV As CustomView
   'Cycle through all the custom views in the sheet
   'that contain row/column information
   For Each oCV In ActiveWorkbook.CustomViews
      If oCV.RowColSettings Then
         oCV.Show
      End If

      'Pause for 2 seconds between each view
      Application.Wait Now + TimeValue("00:00:02")
   Next
End Sub
```

DataLabel Object and the DataLabels Collection

The `DataLabels` collection holds all the labels for individual points or trendlines in a data series. Each series has only one `DataLabels` collection. The parent of the `DataLabels` collection is the `Series` object. Each `DataLabel` object represents a single data label for a trendline or a point. The `DataLabels` collection is used with the `HasDataLabels` property of the parent `Series` object.

The `DataLabels` collection has a few properties and methods besides the typical collection attributes. They are listed in the following table.

DataLabels Collection Properties and Methods

Name	Returns	Description
AutoScaleFont	Variant	Set/Get whether the font size will change automatically if the parent chart changes sizes
AutoText	Boolean	Set/Get whether Excel will generate the data label text automatically
Border	Border	Read-only. Returns the border's properties around the data label collection
Count	Long	Read-only. Returns the number of data labels in the collection
Fill	ChartFill Format	Read-only. Returns an object containing fill formatting options for the data labels in the collection
Font	Font	Read-only. Returns an object containing `Font` options for the data labels in the collection
Horizontal Alignment	Variant	Set/Get how the data labels are horizontally aligned. Use the `xlAlign` constants
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the data labels in the collection (for example, interior color)
Name	String	Read-only. Returns the name of the collection
NumberFormat	String	Set/Get the numeric formatting to use if the data labels are numeric values or dates
NumberFormat Linked	Boolean	Set/Get whether the same numerical format used for the cells containing the chart data is used by the data labels
NumberFormat Local	Variant	Set/Get the name of the numeric format being used by the data labels in the language being used by the user
Orientation	Variant	Set/Get the angle of the text for the data labels. The value can be in degrees (from –90 to 90) or one of the `XlOrientation` constants
Position	XlDataLabel Position	Set/Get where the data labels are going to be located in relation to points or trendlines
ReadingOrder	Long	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages
Separator	Variant	Set/Get the separator used for the data labels on a chart
Shadow	Boolean	Set/Get whether the data labels have a shadow effect
ShowBubble Size	Boolean	Set/Get whether to show the bubble size for the data labels on a chart

Continues

Name	Returns	Description
ShowCategory Name	Boolean	Set/Get whether to display the category name for the data labels on a chart
ShowLegendKey	Boolean	Set/Get whether the key being used in the legend, usually a specific color, will show along with the data label
Show Percentage	Boolean	Set/Get whether to display the percentage value for the data labels on a chart
ShowSeries Name	Boolean	Set/Get whether to show the series name
ShowValue	Boolean	Set/Get whether to display the specified chart's data label values
Type	Variant	Set/Get what sort of data label to show for the collection (for example, labels, percent, values)
Vertical Alignment	Variant	Set/Get how you want the data labels horizontally aligned. Use the `xlVAlign` constants
Delete	Variant	Method. Deletes the data labels
Select	Variant	Method. Selects the data labels on the chart

DataLabel Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

DataLabel Properties

Name	Returns	Description
AutoScaleFont	Variant	Set/Get whether the font size will change automatically if the parent chart changes sizes
AutoText	Boolean	Set/Get whether Excel will generate the data label text automatically
Border	Border	Read-only. Returns the border's properties around the data label
Caption	String	Set/Get the data label text
Characters	Characters	Read-only. Parameters: `[Start]`, `[Length]`. Returns an object that represents a range of characters within the text
Fill	ChartFill Format	Read-only. Returns an object containing fill formatting options for the data label
Font	Font	Read-only. Returns an object containing `Font` options for the data label

Name	Returns	Description
`HorizontalAlignment`	`Variant`	Set/Get how the data labels are horizontally aligned. Use the `xlAlign` constants
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of the data label (for example, interior color)
`Left`	`Double`	Set/Get the distance from the left edge of the data label to the parent chart's left edge
`Name`	`String`	Read-only. Returns the name of the data label
`NumberFormat`	`String`	Set/Get the numeric formatting to use if the data label is a numeric value or a date
`NumberFormatLinked`	`Boolean`	Set/Get whether the same numerical format used for the cells containing the chart data is used by the data label
`NumberFormatLocal`	`Variant`	Set/Get the name of the numeric format being used by the data label in the language being used by the user
`Orientation`	`Variant`	Set/Get the angle of the text for the data label. The value can be in degrees (from –90 to 90) or one of the `XlOrientation` constants
`Position`	`XlDataLabelPosition`	Set/Get where the data label is going to be located in relation to points or trendlines
`ReadingOrder`	`Long`	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages
`Separator`	`Variant`	Set/Get the separator used for the data labels on a chart
`Shadow`	`Boolean`	Set/Get whether the data label has a shadow effect
`ShowBubbleSize`	`Boolean`	Set/Get whether to show the bubble size for the data labels on a chart
`ShowCategoryName`	`Boolean`	Set/Get whether to display the category name for the data labels on a chart
`ShowLegendKey`	`Boolean`	Set/Get whether the key being used in the legend, usually a specific color, will show along with the data label
`ShowPercentage`	`Boolean`	Set/Get whether to display the percentage value for the data labels on a chart
`ShowSeriesName`	`Boolean`	Set/Get whether to show the series name
`ShowValue`	`Boolean`	Set/Get whether to display the specified chart's data label values
`Text`	`String`	Set/Get the data label text

Continues

Name	Returns	Description
`Top`	`Double`	Set/Get the distance from the top edge of the data label to the parent chart's top edge
`Type`	`Variant`	Set/Get what sort of data label to show (for example, labels, percent, values)
`Vertical Alignment`	`Variant`	Set/Get how you want the data label horizontally aligned. Use the `xlVAlign` constants

DataLabel Methods

Name	Returns	Parameters	Description
`Delete`	`Variant`		Deletes the data label
`Select`	`Variant`		Selects the data label on the chart

Example: DataLabel Object and the DataLabels Collection

This example adds data labels to all the points on the chart, using the column to the left of the X values range:

```
Sub AddDataLabels()
   Dim oSer As Series
   Dim vaSplits As Variant
   Dim oXRng As Range
   Dim oLblRng As Range
   Dim oLbl As DataLabel
   'Loop through all the series in the chart
   For Each oSer In Charts(1).SeriesCollection
      'Get the series formula and split it into its
      'constituent parts (Name, X range, Y range, order)
      vaSplits = Split(oSer.Formula, ",")

      'Get the X range
      Set oXRng = Range(vaSplits(LBound(vaSplits) + 1))

      'Get the column to the left of the X range
      Set oLblRng = oXRng.Offset(0, -1)

      'Show data labels for the series
      oSer.ApplyDataLabels

      'Loop through the points
      For i = 1 To oSer.Points.Count

         'Get the DataLabel object
         Set oLbl = oSer.Points(i).DataLabel
```

```
            'Set its text and alignment
            With oLbl
               .Caption = oLblRng.Cells(i)
               .Position = xlLabelPositionAbove
         End With
      Next
      Next
   End Sub
```

DataTable Object

A `DataTable` object contains the formatting options associated with a chart's data table. The parent of the `DataTable` object is the `Chart` object.

DataTable Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

DataTable Properties

Name	Returns	Description
`AutoScaleFont`	`Variant`	Set/Get whether the font size will change automatically if the parent chart changes sizes.
`Border`	`Border`	Read-only. Returns the border's properties around the data table
`Font`	`Font`	Read-only. Returns an object containing `Font` options for the data table
`HasBorderHorizontal`	`Boolean`	Set/Get whether the data table has horizontal cell borders
`HasBorderOutline`	`Boolean`	Set/Get whether the data table has a border around the outside
`HasBorderVertical`	`Boolean`	Set/Get whether the data table has vertical cell borders
`ShowLegendKey`	`Boolean`	Set/Get whether the legend key is shown along with the data table contents

DataTable Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the data table
`Select`			Selects the data table on the chart

Example: DataTable Object

Adds a data table to a chart and formats it to only have vertical lines between the values:

```
Sub FormatDataTable()
   Dim oDT As DataTable
   'Display the data table
   Charts(1).HasDataTable = True
   'Get the DataTable object
   Set oDT = Charts(1).DataTable
   'Format the data table to only have vertical lines
   With oDT
      .HasBorderOutline = False
      .HasBorderHorizontal = False
      .HasBorderVertical = True
   End With
End Sub
```

DefaultWebOptions Object

Allows programmatic changes to items associated with the default settings of the Web Options dialog box. These options include what Excel does when opening an HTML page and when saving a sheet as an HTML page.

DefaultWebOptions Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

DefaultWebOptions Properties

Name	Returns	Description
`AllowPNG`	`Boolean`	Set/Get whether Portable Network Graphics Format PNG is allowed as an output format. PNG is a file format for the lossless, portable, well-compressed storage of images
`AlwaysSaveInDefaultEncoding`	`Boolean`	Set/Get whether Web pages are always saved in the default encoding
`CheckIfOfficeIsHTMLEditor`	`Boolean`	Set/Get whether Office is the default Web editor for Office created pages
`DownloadComponents`	`Boolean`	Set/Get whether Office components are downloaded to the end user's machine when viewing Excel files in a Web browser
`Encoding`	`MsoEncoding`	Set/Get the type of encoding to save a document as
`FolderSuffix`	`String`	Read-only. Returns what the suffix name is for the support directory created when saving an Excel document as a Web page. Language dependent

Name	Returns	Description
`Fonts`	`WebPage Fonts`	Read-only. Returns a collection of possible Web type fonts
`LoadPictures`	`Boolean`	Set/Get whether images are loaded when opening up an Excel file
`LocationOf Components`	`String`	Set/Get the URL or path that contains the Office Web components needed to view documents in a Web browser
`Organize InFolder`	`Boolean`	Set/Get whether supporting files are organized in a folder
`PixelsPer Inch`	`Long`	Set/Get how dense graphics and table cells should be when viewed on a Web page
`RelyOnCSS`	`Boolean`	Set/Get whether Cascading Style Sheets (CSS) is used for font formatting
`RelyOnVML`	`Boolean`	Set/Get whether image files are not created when saving a document with drawn objects. Vector Markup Language is used to create the images on the fly. VML is an XML-based format for high-quality vector graphics on the Web
`SaveHidden Data`	`Boolean`	Set/Get whether all hidden data is saved in the Web page along with the regular data
`SaveNewWeb PagesAsWeb Archives`	`Boolean`	Set/Get whether a new Web page can be saved as a Web archive
`ScreenSize`	`MsoScreen Size`	Set/Get the target monitor's screen size
`Target Browser`	`MsoTarget Browser`	Set/Get the browser version
`UpdateLinks OnSave`	`Boolean`	Set/Get whether links are updated every time the document is saved
`UseLongFile Names`	`Boolean`	Set/Get whether long file names are used whenever possible

Example: DefaultWebOptions Object

This example shows how to open a Web page, without loading the pictures:

```
Sub OpenHTMLWithoutPictures()
   Dim bLoadImages As Boolean
   Dim oDWO As DefaultWebOptions
   'Get the Default Web options
   Set oDWO = Application.DefaultWebOptions
```

```
    'Remember whether to load pictures
    bLoadImages = oDWO.LoadPictures
    'Tell Excel not to load pictures, for faster opening
    oDWO.LoadPictures = False
    'Open a web page, without pictures
    Workbooks.Open "http://www.wrox.com"
    'Restore the setting
    oDWO.LoadPictures = bLoadImages
End Sub
```

Diagram Object

A `Diagram` represents a preset collection of shapes surrounded by an invisible border. It's a cross between adding shapes using the Drawing toolbar and an enhanced version of the Org Chart program used in previous versions of Microsoft Office. Within each `Diagram` are `Nodes`. Each Node represents an individual shape in the `Diagram`.

There are several different types of preset `Diagrams` you can choose from: `Cycle`, `Target`, `Radial`, `Venn`, `Pyramid`, and `OrgChart`.

It's important to note that the `Diagram` object belongs to the `Shape`(s) object, which, in turn, belongs to the `Worksheet` object. Consequently, to add a `Diagram` object to a worksheet, you go through the `Shapes` collection using the `AddDiagram` method:

```
ActiveSheet.Shapes.AddDiagram(msoDiagramOrgChart, 2, 2, 400, 300)
```

If you set the preceding code to an object variable, it returns a `Shape` object. To add shapes to the `Diagram`, use the `DiagramNode` object within the `Shape` object:

```
ActiveSheet.Shapes(1).DiagramNode.Children.AddNode
```

To reference the properties and methods of the `Diagram` object itself (listed next), you access the `Diagram` object through the `Shape` object, like so:

```
ActiveSheet.Shapes(1).Diagram.Nodes(1).TextShape.Fill.BackColor.
SchemeColor = 17
```

Diagram Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Diagram Properties

Name	Returns	Description
`AutoFormat`	`MsoTriState`	Get/Set the automatic formatting state for a diagram
`AutoLayout`	`MsoTriState`	Get/Set the constant which determines the automatic positioning of the nodes and connectors in a diagram
`Nodes`	`Diagram Nodes`	Read-only. Returns a `DiagramNodes` object that contains a flat list of all the nodes in the specified diagram

Name	Returns	Description
`Reverse`	`MsoTriState`	Set/Get whether to reverse the order of the nodes
`Type`	`MsoDiagram Type`	Read-only. Returns the diagram type

Diagram Methods

Name	Returns	Parameters	Description
`Convert`		`Type As Mso DiagramType`	Converts the current diagram to a different diagram

Example: Diagram Object

The following routine creates a diagram and adds and formats several shapes (called nodes) to the diagram. The shape color and font name come from a table on a worksheet, allowing you to easily experiment with different looks.

As of this writing, any attempt to programmatically add text to nodes in a `Diagram` results in an error:

```
Sub CreateDiagram()

    Const sRANGE_LEVELS As String = "Levels"

    Dim lCount As Long
    Dim oDiagramShape As Shape
    Dim oDiagramNode As DiagramNode
    Dim oDiagramNodeChild As DiagramNode

    'Clear the current shapes (except the Command Button)
    On Error Resume Next
        For Each oDiagramShape In wksDiagrams.Shapes
            If oDiagramShape.HasDiagram Then oDiagramShape.Delete
        Next oDiagramShape
    On Error GoTo 0

    'Turn off the screen
    Application.ScreenUpdating = False

    'Create the Diagram
    Set oDiagramShape = wksDiagrams.Shapes.AddDiagram(msoDiagramOrgChart, 2, 2,
300, 250)

    'Remove the transparent background
    oDiagramShape.Fill.Visible = msoTrue

    'Create the top level node
    Set oDiagramNode = oDiagramShape.DiagramNode.Children.AddNode

    With oDiagramNode
```

```
        'Format the top level node
    With .Shape
        .AutoShapeType = msoShapeBevel
        .TextFrame.Characters.Font.Name = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(1, 2).Text
        .Fill.ForeColor.SchemeColor = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(1, 3).Value
    End With

    'Create a child node under the top level node
    Set oDiagramNodeChild = .Children.AddNode

    'Format the child node
    With oDiagramNodeChild
        .Shape.TextFrame.Characters.Font.Name = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(2, 2).Text
        .Shape.Fill.ForeColor.SchemeColor = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(2, 3).Value
    End With

    'Place two child nodes under the top level's child
    For lCount = 1 To 2
        With oDiagramNodeChild.Children.AddNode
            .Shape.TextFrame.Characters.Font.Name = _
                wksDiagrams.Range(sRANGE_LEVELS).Cells(3, 2).Text
            .Shape.Fill.ForeColor.SchemeColor = _
                wksDiagrams.Range(sRANGE_LEVELS).Cells(3, 3).Value
        End With
    Next lCount

    'Create another child under the top level node
    Set oDiagramNodeChild = .Children.AddNode

    With oDiagramNodeChild
        .Shape.TextFrame.Characters.Font.Name = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(2, 2).Text
        .Shape.Fill.ForeColor.SchemeColor = _
            wksDiagrams.Range(sRANGE_LEVELS).Cells(2, 3).Value
    End With

    'Place two child nodes under this child
    '(which is under top level)
    For lCount = 1 To 2
        With oDiagramNodeChild.Children.AddNode
            .Shape.TextFrame.Characters.Font.Name = _
                wksDiagrams.Range(sRANGE_LEVELS).Cells(3, 2).Text
            .Shape.Fill.ForeColor.SchemeColor = _
                wksDiagrams.Range(sRANGE_LEVELS).Cells(3, 3).Value
        End With
    Next lCount

End With

End Sub
```

DiagramNode Object and the DiagramNodes Collection

The DiagramNode object represents one shape inside a Diagram. Shapes underneath a specific node are called children. Use the AddNode method of the Children property of this object to add nodes to the current node.

The DiagramNodes collection consists of all of the Nodes in a Diagram object. Each Node is a shape within the Diagram.

DiagramNodes Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

DiagramNodes Collection Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of objects in the collection

DiagramNodes Collection Methods

Name	Returns	Parameters	Description
Item	Diagram Node	Index As Variant	Returns a single object from a collection
SelectAll			Selects all the shapes in the collection

DiagramNode Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

DiagramNode Properties

Name	Returns	Description
Children	Diagram NodeChildren	Read-only. Returns the collection of child nodes of a particular node
Diagram	IMso Diagram	Read-only. Returns a representation of a diagram
Layout	MsoOrg Chart LayoutType	Set/Get the formatting style of the child nodes of an organization chart
Root	Diagram Node	Read-only. Returns the root of the root diagram node

Continues

Name	Returns	Description
Shape	Shape	Read-only. Returns the shape attached to the specified comment, diagram node, or hyperlink
TextShape	Shape	Read-only. Returns the shape of the text box associated with a diagram node

DiagramNode Methods

Name	Returns	Parameters	Description
AddNode	Diagram Node	[pos As MsoRelative NodePosition], [nodeType As MsoDiagram NodeType]	Creates a diagram node and returns a DiagramNode object that represents the new node
CloneNode	Diagram Node	copyChildren As Boolean, [pTarget Node As DiagramNode], [pos As MsoRelative NodePosition]	Clones a diagram node and returns a DiagramNode object representing the cloned node
Delete			Deletes the object
MoveNode		pTargetNode As DiagramNode, pos As MsoRelative NodePosition	Moves a diagram node and any of its child nodes, within a diagram
NextNode	Diagram Node		Selects the next diagram node in a series of nodes and returns a DiagramNode object representing the newly-selected node
PrevNode	Diagram Node		Returns the previous diagram node in a collection of diagram nodes
ReplaceNode		pTargetNode As DiagramNode	Replaces a target diagram node with the source diagram node
SwapNode		pTargetNode As DiagramNode, [swapChildren As Boolean]	Swaps the source diagram node with a target diagram node
Transfer Children		pReceiving Node As DiagramNode	The child nodes of a source diagram node are transferred to a receiving diagram node

DiagramNodeChildren Object

The DiagramNodeChildren object represents a Child shape one level below a DiagramNode object. Each DiagramNodeChildren object is a DiagramNode object itself. If a DiagramNodeChildren object contains Children below it (in the hiercharchy), then each of those would be considered DiagramNodeChildren objects.

DiagramNodeChildren Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

DiagramNodeChildren Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of objects in the collection
FirstChild	Diagram Node	Read-only. Returns the first child node of a parent node
LastChild	Diagram Node	Read-only. Returns the last child node of a parent node

DiagramNodeChildren Methods

Name	Returns	Parameters	Description
AddNode	Diagram Node	[Index], [nodeType As MsoDiagram NodeType]	Makes a new DiagramNode
SelectAll			Selects all the shapes in the collection

Dialog Object and the Dialogs Collection

The Dialogs collection represents the list of dialog boxes that are built-in to Excel. The XlBuiltinDialog constants are used to access an individual Dialog object in the Dialogs collection. A Dialog object represents a single built-in Excel dialog box. Each Dialog object will have additional custom properties depending on what type of Dialog object it is. Besides the typical collection attributes, the Dialogs collection also has a Count property that returns the number of Dialog objects in the collection.

Dialog Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Dialog Methods

Name	Returns	Parameters	Description

`Show`	`Boolean`	`[Arg1],`
		`[Arg2],`
		`...[Arg30]`
			Displays and executes the dialog box settings. `True` is returned if the user chose *OK* and `False` is returned if the user chose *Cancel*. The arguments to pass depend on the dialog box

Example: Dialog Object and the Dialogs Collection

```
Sub ShowPrinterSelection()
   'Show printer selection dialog
   Application.Dialogs(xlDialogPrinterSetup).Show
End Sub
```

DisplayUnitLabel Object

The `DisplayUnitLabel` object contains all of the text and formatting associated with the label used for units on axes. For example, if the values on an axis are in the millions it would be messy to display such large values on the axis. Using a unit label such as "Millions" would allow much smaller numbers to be used. The parent of the `DisplayUnitLabel` object is the `Axis` object. This object is usually used along with the `HasDisplayUnit` property of the parent `Axis` object.

DisplayUnitLabel Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

DisplayUnitLabel Properties

Name	Returns	Description
`AutoScaleFont`	`Variant`	Set/Get whether the font size will change automatically if the parent chart changes sizes
`Border`	`Border`	Read-only. Returns the border's properties around the unit label
`Caption`	`String`	Set/Get the unit label's text
`Characters`	`Characters`	Read-only. Parameters: `[Start]`, `[Length]`. Returns an object containing all the characters in the unit label. Allows manipulation on a character-by-character basis
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the unit label
`Font`	`Font`	Read-only. Returns an object containing `Font` options for the unit label

Name	Returns	Description
HorizontalAlignment	Variant	Set/Get how you want the unit label horizontally aligned. Use the xlAlign constants
Interior	Interior	Read-only. Returns an object containing options to format the area in the unit label text area (for example, interior color)
Left	Double	Set/Get the distance from the left edge of the unit label text area to the chart's left edge
Name	String	Read-only. Returns the name of the DisplayUnitLabel object
Orientation	Variant	Set/Get the angle of the text for the unit label. The value can be in degrees (from –90 to 90) or one of the XlOrientation constants
ReadingOrder	Long	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages
Shadow	Boolean	Set/Get whether the unit label has a shadow effect
Text	String	Set/Get the unit label's text
Top	Double	Set/Get the distance from the top edge of the unit label text area to the chart's top edge
VerticalAlignment	Variant	Set/Get how you want the unit label horizontally aligned. Use the xlVAlign constants

DisplayUnitLabel Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the unit label from the axis
Select	Variant		Selects the unit label on the chart

Example: DisplayUnitLabel Object

```
Sub AddUnitLabel()
   Dim oDUL As DisplayUnitLabel
   'Format the Y axis to have a unit label
   With Charts(1).Axes(xlValue)
      .DisplayUnit = xlThousands
      .HasDisplayUnitLabel = True
      'Get the unit label
      Set oDUL = .DisplayUnitLabel
   End With
   'Format the unit label
   With oDUL
      .Caption = "Thousands"
      .Font.Name = "Arial"
      .VerticalAlignment = xlCenter
   End With
End Sub
```

DownBars Object

The DownBars object contains formatting options for down bars on a chart. The parent of the DownBars object is the ChartGroup object. To see if this object exists, use the HasUpDownBars property of the ChartGroup object.

DownBars Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

DownBars Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the down bars
Fill	ChartFill Format	Read-only. Returns an object containing fill formatting options for the down bars
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the down bars (for example, interior color)
Name	String	Read-only. Returns the name of the down bars

DownBars Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the down bars
Select	Variant		Selects the down bars in the chart

DropLines Object

The DropLines object contains formatting options for drop lines in a chart. The parent of the DropLines object is the ChartGroup object. To see if this object exists, use the HasDropLines property of the ChartGroup object.

DropLines Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

DropLines Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the drop lines
`Name`	`String`	Read-only. Returns the name of the drop lines

DropLines Methods

Name	Returns	Parameters	Description
`Delete`	`Variant`		Deletes the drop lines
`Select`	`Variant`		Selects the drop lines in the chart

Example: DropLines Object

```
Sub AddAndFormatDropLines()
   Dim oDLine As DropLines
   'Show the drop lines
   Charts(1).ChartGroups(1).HasDropLines = True
   'Get the DropLines object
   Set oDLine = Charts(1).ChartGroups(1).DropLines
   'Format the drop lines
   With oDLine
      .Border.Weight = xlMedium
      .Border.LineStyle = xlDash
      .Border.ColorIndex = 3
   End With
End Sub
```

Error Object and the Errors Collection

The `Error` object contains one error in the `Errors` collection representing one error in a cell containing possible errors.

The `Errors` collection represents all the errors contained within a cell. Each cell can contain multiple errors.

These errors are analogous to the new Error Checking feature in Excel 2003. The different types of errors that Excel can check can be found on the Error Checking Tab of the Tools ➪ Options command. In the Excel application, cells containing errors appear with a small triangle in their upper left corner. The default color of the triangle on most systems is green, but can be changed using the Error Indicator Color option on the Error Checking Tab of the Options command.

When a user selects a range containing an error, a drop-down icon containing an exclamation point inside a yellow diamond appears. The user can then click the icon and choose how to handle the errors in the range. If action was taken, like ignoring the error or clicking one of the recommended choices, the green indicator disappears for all cells containing that error. Any cells still containing the green triangle indicate other error types are still present in those cells.

As of this writing, the Errors Collection object and Error object do not have the ability to handle multiple errors in a multicell range as described earlier. The Help file and object model indicate that the Parent object of the Errors Collection is a Range object. However, any attempt to reference the Errors in a multicell range results in an error. Since each cell can contain multiple errors, for all intent and purposes, the Error Collection object stores all the errors contained within one cell, not a range of cells. This requires that you loop through a range of cells if you need to programmatically handle errors in a multicell range.

Note that neither the Error nor Errors objects contains a count or Boolean property that would allow us to test whether an error even exists in a cell. For this reason, additional code would be needed to loop through each error type for every desired cell checking for the Error object's Value property, which returns True if that type of error occurs in the cell.

Use the Item property of the Errors Collection object to loop through the error types to determine which errors might have occurred.

Errors Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Errors Collection Properties

Name	Returns	Description
Item	Error	Returns an Error object that is contained in the Errors collection

Error Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Error Properties

Name	Returns	Description
Ignore	Boolean	Get/Set whether error checking is enabled for a range
Value	Boolean	Read-only. Returns whether all the validation criteria are met

ErrorBars Object

The ErrorBars object contains formatting options for error bars in a chart. The parent of the Errors object is the SeriesCollection object.

ErrorBars Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

ErrorBars Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the error bars
`EndStyle`	`XlEndStyle Cap`	Set/Get the style used for the ending of the error bars
`Name`	`String`	Read-only. Returns the name of the error bars

ErrorBars Methods

Name	Returns	Parameters	Description
`ClearFormats`	`Variant`		Clears the formatting set on the error bar
`Delete`	`Variant`		Deletes the error bars
`Select`	`Variant`		Selects the error bars in the chart

Example: ErrorBars Object

```
Sub AddAndFormatErrorBars()
   Dim oSer As Series
   Dim oErrBars As ErrorBars
   'Add error bars to the first series (at +/- 10% of the value)
   Set oSer = Charts(1).SeriesCollection(1)
   oSer.ErrorBar xlY, xlErrorBarIncludeBoth, xlErrorBarTypePercent, 10
   'Get the ErrorBars object
   Set oErrBars = oSer.ErrorBars
   'Format the error bars
   With oErrBars
      .Border.Weight = xlThick
      .Border.LineStyle = xlContinuous
      .Border.ColorIndex = 7
      .EndStyle = xlCap
   End With
End Sub
```

ErrorCheckingOptions Collection Object

Represents all of the Error Checking possibilities found on the Error Checking Tab of the Tools ➪ Options command. Using the `BackgroundChecking` property of this object hides all of the error indicators (small triangle in the upper right corner of cells).

Use the other properties in this object to specify which type of error checking you want Excel to perform.

The ErrorCheckingOptions object can be referenced through the Application object and therefore affect all open workbooks.

ErrorCheckingOptions Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ErrorCheckingOptions Collection Properties

Name	Returns	Description
BackgroundChecking	Boolean	Set/Get whether background error checking is set, that is whether the autocorrect button will apear in cells that contain errors
EmptyCellReferences	Boolean	Set/Get whether error checking is on for cells containing formulas that refer to empty cells
EvaluateToError	Boolean	Set/Get whether error checking is on for cells that evaluate to an error value
InconsistentFormula	Boolean	Set/Get whether error checking is on for cells containing an inconsistent formula in a region
IndicatorColorIndex	XlColorIndex	Set/Get the color of the indicator for error checking options
ListDataValidation	Boolean	The property will return True if data validation is enabled for a list.
NumberAsText	Boolean	Set/Get whether error checking is on for numbers written as text
OmittedCells	Boolean	Set/Get whether error checking is on for cells that contain formulas referring to a range that omits adjacent cells that could be included
TextDate	Boolean	Set/Get whether error checking is on for cells that contain a text date with a two-digit year
UnlockedFormulaCells	Boolean	Set/Get whether error checking is on for cells that are unlocked and contain a formula

Example: ErrorCheckingOptions Object

The following routine uses a table on a worksheet to set the Error Checking Options:

```
Sub SetErrorCheckingOptions()

    Dim rngSettings As Range
    Dim vSetting As Variant
```

```
     'Locate the start of the Settings table
     Set rngSettings = wksErrors.Range("ErrorSettings")

     'Go through each ErrorChecking Property and
     ' set it according to the values placed in teh table
     With Application.ErrorCheckingOptions

         vSetting = rngSettings.Cells(1, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .BackgroundChecking = vSetting
        End If

        vSetting = rngSettings.Cells(2, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .EvaluateToError = vSetting
        End If

        vSetting = rngSettings.Cells(3, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .TextDate = vSetting
        End If

        vSetting = rngSettings.Cells(4, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .NumberAsText = vSetting
        End If

        vSetting = rngSettings.Cells(5, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .InconsistentFormula = vSetting
        End If

        vSetting = rngSettings.Cells(6, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .OmittedCells = vSetting
        End If

        vSetting = rngSettings.Cells(7, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .UnlockedFormulaCells = vSetting
        End If

        vSetting = rngSettings.Cells(8, 2).Value
        If Len(vSetting) And (vSetting = True Or vSetting = False) Then
            .EmptyCellReferences = vSetting
        End If

        vSetting = rngSettings.Cells(9, 2).Value
        If LCase(vSetting) = "xlcolorindexautomatic" Then
            .IndicatorColorIndex = xlColorIndexAutomatic
        ElseIf Len(vSetting) And (vSetting > 1 And vSetting < 100) Then
            .IndicatorColorIndex = vSetting
        End If

    End With
```

```
        'Indicators sometimes don't appear
        ' after the routine finishes unless you
        ' update the screen
        Application.ScreenUpdating = True

    End Sub
```

FillFormat Object

The FillFormat object represents the fill effects available for shapes. For example, a FillFormat object defines solid, textured, and patterned fill of the parent shape. A FillFormat object can only be accessed through the parent Shape object.

FillFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

FillFormat Properties

Name	Returns	Description
BackColor	ColorFormat	Read-only. Returns the background color through the ColorFormat object
ForeColor	ColorFormat	Read-only. Returns the foreground color through the ColorFormat object
GradientColorType	MsoGradientColorType	Read-only. Returns what type of gradient fill color concept is used
GradientDegree	Single	Read-only. Returns how dark or light the gradient fill is
GradientStyle	MsoGradientStyle	Read-only. Returns the orientation of the gradient that is used
GradientVariant	Integer	Read-only. Returns the variant used for the gradient from the center
Pattern	MsoPatternType	Read-only. Returns the pattern used for the fill, if any
PresetGradientType	MsoPresetGradientType	Read-only. Returns the type of gradient that is used
PresetTexture	MsoPresetTexture	Read-only. Returns the non-custom texture of the fill
TextureName	String	Read-only. Returns the custom texture name of the fill
TextureType	MsoTextureType	Read-only. Returns whether the texture is custom, preset, or mixed
Transparency	Single	Set/Get how transparent the fill is. From 0 (opaque) to 1 (clear)

Name	Returns	Description
Type	MsoFillType	Read-only. Returns if the fill is a texture, gradient, solid, background, picture or mixed
Visible	MsoTriState	Set/Get whether the fill options are visible in the parent shape

FillFormat Methods

Name	Returns	Parameters	Description
OneColor Gradient		Style As MsoGradient Style, Variant As Integer, Degree As Single	Set the style, variant and degree for a one-color gradient fill
Patterned		Pattern As MsoPattern Type	Set the pattern for a fill
Preset Gradient		Style As MsoGradient Style, Variant As Integer, Preset GradientType As MsoPreset GradientType	Choose the style, variant, and preset gradient type for a gradient fill
Preset Textured		Preset Texture As MsoPreset Texture	Set the preset texture for a fill
Solid			Set the fill to a solid color
TwoColor Gradient		Style As MsoGradient Style, Variant As Integer	Set the style for a two-color gradient fill
UserPicture		PictureFile As String	Set the fill to the picture in the PictureFile format
UserTextured		TextureFile As String	Set the custom texture for a fill with the TextureFile format

Example: FillFormat Object

```
Sub FormatShape()
   Dim oFF As FillFormat
```

```
    'Get the Fill format of the first shape
    Set oFF = ActiveSheet.Shapes(1).Fill
    'Format the shape
    With oFF
       .TwoColorGradient msoGradientFromCorner, 1
       .ForeColor.SchemeColor = 3
       .BackColor.SchemeColor = 5
    End With
End Sub
```

Filter Object and the Filters Collection

The `Filters` collection holds all of the filters associated with the specific parent `AutoFilter`. Each Filter object defines a single filter for a single column in an autofiltered range. The parent of the `Filters` collection is the `AutoFilter` object.

The `Filters` collection has one other property besides the typical collection attributes. The `Count` property returns the number of `Filter` objects in the collection.

Filter Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Filter Properties

Name	Returns	Description
`Criteria1`	`Variant`	Read-only. Returns the first criteria defined for the filter (for example, “>=5”)
`Criteria2`	`Variant`	Read-only. Returns the second criteria for the filter, if defined
`On`	`Boolean`	Read-only. Returns whether the filter is in use
`Operator`	`XlAuto Filter Operator`	Read-only. Returns what sort of operator has been defined for the filter (for example, `xlTop10Items`)

Example: Filter Object and the Filters Collection

See the `AutoFormat` object for an example of using the `Filter` object and the `Filters` collection.

Floor Object

The `Floor` object contains formatting options for the floor area of a 3D chart. The parent of the `Floor` object is the `Chart` object.

Floor Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Floor Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the floor of the 3D chart
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the floor of a 3D chart
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of the chart floor (for example, interior color)
`Name`	`String`	Read-only. Returns the name of the `Floor` object
`PictureType`	`Variant`	Set/Get how an associated picture is displayed on the floor of the 3D chart (for example, stretched, tiled). Use the `XlPictureType` constants

Floor Methods

Name	Returns	Parameters	Description
`ClearFormats`	`Variant`		Clears the formatting made on the `Floor` object
`Paste`			Pastes the picture in the clipboard into the `Floor` object
`Select`	`Variant`		Selects the floor on the parent chart

Example: Floor Object

```
Sub FormatFloor()
   Dim oFlr As Floor
   'Get the chart's Floor
   Set oFlr = Charts(1).Floor
   'Format the floor in white marble
   With oFlr
      .Fill.PresetTextured msoTextureWhiteMarble
      .Fill.Visible = True
   End With
End Sub
```

Font Object

The `Font` object contains all of the formatting attributes related to fonts of the parent including font type, size and color. Possible parents of the `Font` object are the `AxisTitle`, `Characters`, `ChartArea`, `ChartTitle`, `DataLabel`, `Legend`, `LegendEntry`, `Range`, `Style`, and `TickLabels` objects. Also, the `DataLabels` collection is another possible parent of the `Font` object.

Font Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Font Properties

Name	Returns	Description
`Background`	`Variant`	Set/Get the type of background used behind the font text (`xlBackgroundAutomatic`, `xlBackgroundOpaque`, and `xlBackgroundTransparent`). Use the `XlBackground` constants. Valid only for text on charts
`Bold`	`Variant`	Set/Get whether the font is bold
`Color`	`Variant`	Set/Get the color of the font. Use the `RGB` function to create the color value
`ColorIndex`	`Variant`	Set/Get the color of the font. Use the `XlColorIndex` constants or an index value in the current color palette
`FontStyle`	`Variant`	Set/Get what style to apply to the font (for example, "`Bold`")
`Italic`	`Variant`	Set/Get whether the font is italic
`Name`	`Variant`	Set/Get the name of the font
`OutlineFont`	`Variant`	Set/Get whether the font is an outline font. Not used in Windows
`Shadow`	`Variant`	Set/Get whether the font is a shadow font. Not used in Windows
`Size`	`Variant`	Set/Get the font size of the font
`Strikethrough`	`Variant`	Set/Get whether the font has a strikethrough effect
`Subscript`	`Variant`	Set/Get whether the font characters look like a subscript
`Superscript`	`Variant`	Set/Get whether the font characters look like a superscript
`Underline`	`Variant`	Set/Get whether the font is underlined

Example: Font Object

```
Sub FormatCellFont()
   Dim oFont As Font
   'Get the font of the currently selected range
   Set oFont = Selection.Font
   'Format the font
   With oFont
      .Name = "Times New Roman"
      .Size = 16          'Points
      .ColorIndex = 5     'Blue
      .Bold = True
      .Underline = xlSingle
   End With
End Sub
```

FormatCondition Object and the FormatConditions Collection

The FormatConditions collection contains the conditional formatting associated with the particular range of cells. The Parent of the FormatConditions collection is the Range object. Up to three FormatCondition objects can be contained in the FormatConditions collection. Each FormatCondition object represents some formatting that will be applied if the condition is met.

The FormatConditions collection has one property and two methods besides the typical collection attributes. The Count property returns how many FormatCondition objects are in the collection. The Add method can be used to add a formatting condition to the collection. The Type parameter must be specified (XlFormatConditionType constants) and the condition may be specified with the Operator, Formula1, and Formula2 parameters.

Name	Returns	Description
Count	Long	Read-only. Returns the number of objects in the collection
Add	Format Condition	Method. Parameters: Type As XlFormatConditionType, [Operator],[Formula1],[Formula2]
Delete		Method

FormatCondition Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

FormatCondition Properties

Name	Returns	Description
Borders	Borders	Read-only. Returns a collection holding all the individual border attributes for the formatting condition
Font	Font	Read-only. Returns an object containing Font options for the formatting condition
Formula1	String	Read-only. Returns the value that the cells must contain or an expression or formula evaluating to True/False. If the formula or expression evaluates to True then the formatting is applied
Formula2	String	Read-only. Returns the value that the cells must contain or an expression evaluating to True/False. Valid only if the Operator property is xlBetween or xlNotBetween
Interior	Interior	Read-only. Returns an object containing options to format the inside area for the formatting condition (for example, interior color)
Operator	Long	Read-only. Returns the operator to apply to the Formula1 and Formula2 property. Use the XlFormatConditionOperator constants

Continues

Name	Returns	Description
`Type`	`Long`	Read-only. Returns whether the `FormatCondition` is applying formatting based on cell values or a formula. Use the `XlFormatConditionType` constants

FormatCondition Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the formatting condition
`Modify`		`Type As XlFormat Condition Type, [Operator], [Formula1], [Formula2]`	Modifies the formatting condition. Since all the properties are read-only, this is the only way to modify the format condition

Example: FormatCondition Object and the FormatConditions Collection

```
Sub AddConditionalFormat()
   Dim oFC As FormatCondition
   'Remove any existing conditions
   For Each oFC In Selection.FormatConditions
      Selection.FormatConditions(1).Delete
   Next
   'Add first condition
   Set oFC = Selection.FormatConditions.Add(Type:=xlCellValue,
Operator:=xlLess, _
                                                               Formula1:="10")
   With oFC
      .Font.ColorIndex = 2                'white
      .Font.Bold = True
      .Interior.Pattern = xlSolid
      .Interior.Color = RGB(255, 0, 0)    'red
   End With
   'Add second condition
   Set oFC = Selection.FormatConditions.Add(Type:=xlCellValue,
Operator:=xlBetween, _
                                               Formula1:="10", Formula2:="40")
   With oFC
      .Font.Color = RGB(0, 255, 0)
      .Font.Bold = False
      .Interior.Pattern = xlNone
   End With
   'Add third condition
   Set oFC = Selection.FormatConditions.Add(Type:=xlCellValue,
Operator:=xlGreater, _
                                                               Formula1:="40")
   With oFC
      .Font.Color = RGB(0, 0, 255)
      .Font.Bold = True
      .Interior.Pattern = xlNone
   End With
End Sub
```

FreeformBuilder Object

The `FreeformBuilder` object is used by the parent `Shape` object to create new “free hand” shapes. The `BuildFreeform` method of the `Shape` object is used to return a `FreeformBuilder` object.

FreeformBuilder Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

FreeformBuilder Methods

Name	Returns	Parameters	Description
`AddNodes`		`SegmentType As MsoSegment Type, EditingType As MsoEditing Type, X1 As Single, Y1 As Single, [X2], [Y2], [X3], [Y3]`	This method adds a point in the current shape being drawn. A line is drawn from the current node being added to the last node added. `SegmentType` describes the type of line to add between the nodes. `X1, Y1, X2, Y2, X3, Y3` is used to define the position of the current node being added. The coordinates are taken from the upper left corner of the document
`ConvertTo Shape`	`Shape`		Converts the nodes added above into a `Shape` object

Example: FreeformBuilder Object

```
   Sub MakeArch()
      Dim oFFB As FreeformBuilder
      'Create a new freeform builder
      Set oFFB = ActiveSheet.Shapes.BuildFreeform(msoEditingCorner, 100, 300)
      'Add the lines to the builder
      With oFFB
         .AddNodes msoSegmentLine, msoEditingAuto, 100, 200
         .AddNodes msoSegmentCurve, msoEditingCorner, 150, 150, 0, 0, 200, 200
         .AddNodes msoSegmentLine, msoEditingAuto, 200, 300
         .AddNodes msoSegmentLine, msoEditingAuto, 100, 300

         'Convert it to a shape
         .ConvertToShape
      End With
   End Sub
```

Graphic Object

Represents a picture that can be placed in any one of the six locations of the Header and Footer in the Page Setup of a sheet. It's analogous to using both the Insert Picture and Format Picture buttons in the Header or Footer dialogs inside the Page Setup command.

It's important to note that none of the Property settings of this object will result in anything appearing in the Header or Footer unless you insert "`&G`" (via VBA code) in any of the six different areas of the Header or Footer.

Graphic Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Graphic Properties

Name	Returns	Description
`Brightness`	`Single`	Set/Get the brightness of the specified picture. This property's value must be from 0.0 (dimmest) to 1.0 (brightest)
`ColorType`	`MsoPicture ColorType`	Set/Get the color transformation applied to the specified picture or OLE object
`Contrast`	`Single`	Set/Get the contrast of the specified picture. This property's value must be from 0.0 (least) to 1.0 (greatest)
`CropBottom`	`Single`	Set/Get the number of points that are cropped off the bottom of the specified picture or OLE object
`CropLeft`	`Single`	Set/Get the number of points that are cropped off the left-hand side of the specified picture or OLE object
`CropRight`	`Single`	Set/Get the number of points that are cropped off the right-hand side of the specified picture or OLE object
`CropTop`	`Single`	Set/Get the number of points that are cropped off the top of the specified picture or OLE object
`Filename`	`String`	Set/Get the URL or path to where the specified object was saved
`Height`	`Single`	Set/Get the height of the object
`LockAspectRatio`	`MsoTriState`	Set/Get whether the specified shape retains its original proportions when you resize it
`Width`	`Single`	Set/Get the width of the object

Example: Graphic Object

The following routine prompts the user for a graphic file. If chosen, it places the graphic in the header of the active sheet as a Watermark and sizes it to fit the page:

```
Sub AddWatermark()
    Dim oSheet As Object
    Dim sFile As String

    On Error Resume Next
        Set oSheet = ActiveSheet
    On Error GoTo 0

    'Make sure there is an active sheet
    If Not oSheet Is Nothing Then

        'Set the properties of the File Open dialog
        With Application.FileDialog(msoFileDialogFilePicker)

            'Change the default dialog title
            .Title = "Insert Graphic In Center Header"

            'Allow only one file
            .AllowMultiSelect = False

            'Clear the filters and create your own
            'Switch to the custom filter before showing the dialog
            .Filters.Add "All Pictures", "*.gif; *.jpg; *.jpeg; *.bmp; *.wmf; _
                                           *.gif; *.emf; *.dib; *.jfif; *.jpe", 1

            'Show thumbnails to display small representations
            ' of the images
            .InitialView = msoFileDialogViewThumbnail

            'Show the dialog
            '-1 means they didn't cancel
            If .Show = -1 Then
                'Store the chosen file
                sFile = .SelectedItems(1)

                'Set up the graphic in the Header
                With oSheet.PageSetup
                    With .CenterHeaderPicture
                        .Filename = sFile
                        .ColorType = msoPictureWatermark
                        .LockAspectRatio = True

                        'Make it fill the page
                        'c Assumes a letter size portrait)
                        .Width = Application.InchesToPoints(17)
                    End With

                    'Make the graphic appear
                    'Without this, nothing happens
                    .CenterHeader = "&G"
                End With
            End If

            'Remove the filter when done
            .Filters.Clear

        End With

    End If

End Sub
```

Gridlines Object

The `Gridlines` object contains formatting properties associated with the major and minor gridlines on a chart's axes. The gridlines are an extension of the tick marks seen in the background of a chart allowing the end user to more easily see what a chart object's value is. The parent of the `Gridlines` object is the `Axis` object. To make sure the object is valid and to create the `Gridlines` object use the `HasMajorGridlines` and `HasMinorGridlines` properties of the `Axis` object first.

Gridlines Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Gridlines Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the gridlines
`Name`	`String`	Read-only. Returns the name of the `Gridlines` object

Gridlines Methods

Name	Returns	Parameters	Description
`Delete`	`Variant`		Deletes the `Gridline` object
`Select`	`Variant`		Selects the gridlines on the chart

Example: Gridlines Object

```
Sub FormatGridlines()
   Dim oGL As Gridlines
   'Make sure the Y axis has gridlines
   With Charts(1).Axes(xlValue)
      .HasMajorGridlines = True

      'Get the Gridlines object for the major gridlines
      Set oGL = .MajorGridlines
   End With
```

```
    'Format the gridlines
    With oGL
       .Border.ColorIndex = 5
       .Border.LineStyle = xlDash
       .Border.Weight = xlThin
    End With
 End Sub
```

GroupShapes Collection

The GroupShapes collection holds all of the shapes that make up a grouped shape. The GroupShapes collection holds a collection of Shape objects. The parent of the GroupShapes object is the Shape object.

The GroupShapes collection only has two properties besides the typical collection attributes. The Count property returns the number of Shape objects in the GroupShapes collection, and the Range property returns a subset of the shapes in the Shapes collection.

HiLoLines Object

The HiLoLines object contains formatting attributes for a chart's high-low lines. The parent of the HiLoLines object is the ChartGroup object. High-low lines connect the largest and smallest points on a 2D line chart group.

HiLoLines Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

HiLoLines Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the high-low lines
Name	String	Read-only. Returns the name of the HiLoLines object

HiLoLines Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the high-low lines
Select	Variant		Selects the high-low lines on the chart

Example: HiLoLines Object

```
 Sub AddAndFormatHiLoLines()
    Dim oHLL As HiLoLines
    'Add high-low lines to the first group
    Charts(1).ChartGroups(1).HasHiLoLines = True
    'Get the HiLoLines object
    Set oHLL = Charts(1).ChartGroups(1).HiLoLines
    'Format the lines
    With oHLL
       .Border.Weight = xlMedium
       .Border.LineStyle = xlContinuous
       .Border.ColorIndex = 3
    End With
 End Sub
```

HPageBreak Object and the HPageBreaks Collection

The HPageBreaks collection contains all of the horizontal page breaks in the printable area of the parent object. Each HPageBreak object represents a single horizontal page break for the printable area of the parent object. Possible parents of the HPageBreaks collection are the WorkSheet and the Chart objects.

The HPageBreaks collection contains one property and one method besides the typical collection attributes. The Count property returns the number of HPageBreak objects in the collection. The Add method is used to add a HPageBreak object to the collection (and horizontal page break to the sheet). The Add method has a Before parameter to specify the range above where the horizontal page break will be added.

HPageBreak Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

HPageBreak Properties

Name	Returns	Description
Extent	XlPageBreak Extent	Read-only. Returns whether the horizontal page break is full screen or only for the print area
Location	Range	Set/Get the cell that the horizontal page break is located. The top edge of the cell is the location of the page break
Type	XlPageBreak	Set/Get whether the page break is automatic or manually set

HPageBreak Methods

Name	Returns	Parameters	Description
Delete			Deletes the page break
DragOff		Direction As XlDirection, RegionIndex As Long	Drags the page break out of the printable area. The Direction parameter specifies the direction the page break is dragged. The RegionIndex parameter specifies which print region the page break is being dragged out of

Example: HPageBreak Object and the HPageBreaks Collection

```
Sub AddHPageBreaks()
   Dim oCell As Range
   'Loop through all the cells in the first column of the sheet
   For Each oCell In ActiveSheet.UsedRange.Columns(1).Cells
      'If the font size is 16, add a page break above the cell
      If oCell.Font.Size = 16 Then
         ActiveSheet.HPageBreaks.Add oCell
      End If
   Next
End Sub
```

Hyperlink Object and the Hyperlinks Collection

The `Hyperlinks` collection represents the list of hyperlinks in a worksheet or range. Each `Hyperlink` object represents a single hyperlink in a worksheet or range. The `Hyperlinks` collection has an `Add` and `Delete` method besides the typical collection of properties and methods. The `Add` method takes the text or graphic that is to be converted into a hyperlink (`Anchor`) and the URL address or filename (`Address`) and creates a `Hyperlink` object. The `Delete` method deletes the `Hyperlinks` in the collection. The `Hyperlinks` collection also has a `Count` property that returns the number of `Hyperlink` objects in the collection.

Hyperlink Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Hyperlink Properties

Name	Returns	Description
`Address`	`String`	Set/Get the file name or URL address of the hyperlink
`EmailSubject`	`String`	Set/Get the e-mail subject line if the address is an e-mail address
`Name`	`String`	Read-only. Returns whether the `ExtraInfo` property needs to be filled
`Range`	`Range`	Read-only. Returns the name of the hyperlink
`ScreenTip`	`String`	Read-only. Returns the spot in the document where the hyperlink is
`Shape`	`Shape`	Set/Get the text that appears when the mouse hovers over the hyperlink
`SubAddress`	`String`	Read-only. Returns the shape associated with the hyperlink, if any
`TextToDisplay`	`String`	Set/Get the spot in the target location that the hyperlink points to
`Type`	`Long`	Set/Get the target location of the HTML frame of the Address

Hyperlink Methods

Name	Returns	Parameters	Description
AddTo Favorites			Adds the Address property to the Favorites folder
CreateNew Document		Filename As String, EditNow As Boolean, Overwrite As Boolean	Creates a new document with the FileName name from the results of the hyperlink's address. Set the EditNow property to True to open up the document in the appropriate editor. Set Overwrite to True to overwrite any existing document with the same name
Delete			Deletes the Hyperlink object
Follow		[NewWindow], [AddHistory], [ExtraInfo], [Method], [HeaderInfo]	Opens up the target document specified by the Address property. Setting NewWindow to True opens up a new window with the target document. Set AddHistory to True to display the item in history folder. Use the Method parameter to choose if the ExtraInfo property is sent as a Get or a Post

Example: Hyperlink Object and the Hyperlinks Collection

This example creates a hyperlink-based "Table of Contents" worksheet:

```
Sub CreateHyperlinkTOC()
   Dim oBk As Workbook
   Dim oShtTOC As Worksheet, oSht As Worksheet
   Dim iRow As Integer
   Set oBk = ActiveWorkbook
   'Add a new sheet to the workbook
   Set oShtTOC = oBk.Worksheets.Add
   With oShtTOC
      'Add the title to the sheet
      .Range("A1").Value = "Table of Contents"

      'Add Mail and web hyperlinks
      .Hyperlinks.Add .Range("A3"), "mailto:Me@MyISP.com", _
                 TextToDisplay:="Email your comments"
      .Hyperlinks.Add .Range("A4"), "http://www.wrox.com", _
                 TextToDisplay:="Visit Wrox Press"
   End With
   'Loop through the sheets in the workbook
   'adding location hyperlinks
   iRow = 6
   For Each oSht In oBk.Worksheets
```

```
        If oSht.Name <> oShtTOC.Name Then
           oShtTOC.Hyperlinks.Add oShtTOC.Cells(iRow, 1), "", _
                     SubAddress:="'" & oSht.Name & "'!A1", _
                     TextToDisplay:=oSht.Name
           iRow = iRow + 1
        End If
     Next
  End Sub
```

Interior Object

The `Interior` object contains the formatting options associated with the inside area of the parent object. Possible parents of the `Interior` object are the `AxisTitle`, `ChartArea`, `ChartObject`, `ChartTitle`, `DataLabel`, `DownBars`, `Floor`, `FormatCondition`, `Legend`, `LegendKey`, `OLEObject`, `PlotArea`, `Point`, `Range`, `Series`, `Style`, `Upbars`, and `Walls` objects. The `ChartObjects`, `DataLabels`, and `OLEObjects` collections also are possible parents of the `Interior` object.

Interior Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Interior Properties

Name	Returns	Description
`Color`	`Variant`	Set/Get the color of the interior. Use the `RGB` function to create the color value
`ColorIndex`	`Variant`	Set/Get the color of the interior. Use the `XlColorIndex` constants or an index value in the current color palette
`InvertIf Negative`	`Variant`	Set/Get whether the color in the interior of the parent object is inverted if the values are negative
`Pattern`	`Variant`	Set/Get the pattern to use for the interior of the parent object. Use one of the `XlPattern` constants
`PatternColor`	`Variant`	Set/Get the color of the interior pattern. Use the `RGB` function to create the color value
`PatternColor Index`	`Variant`	Set/Get the color of the interior pattern. Use the `XlColorIndex` constants or an index value in the current color palette

Example: Interior Object

```
  Sub FormatRange()
     Dim oInt As Interior
     'Get the interior of the current selection
     Set oInt = Selection.Interior
```

```
    'Format the interior in solid yellow
    '(colour depends on the workbook palette)
    With oInt
       .Pattern = xlSolid
       .ColorIndex = 6
    End With
End Sub
```

IRtdServer Object

This object allows the ability to connect to a Real-Time Data Server (RTD). This type of server allows Excel to receive timed interval data updates without the need for extra coding. In prior versions of Excel, when regular updates were needed, you could use the `OnTime` method to set up regular data update intervals. RTDs send updates automatically based on an interval set within the server or by using the `HeartbeatInterval` method of the `IRTDUpdateEvent` object.

This object is similar in nature to using the RTD worksheet function, which displays data at regular intervals in a worksheet cell.

Note that to use this object you must instantiate it using the `Implements` keyword.

IRtdServer Methods

Name	Returns	Parameters	Description
`ConnectData`		`TopicID`, `Strings`, `GetNewValues`	Called when a file is opened that contains real-time data (RTD) functions or when a new formula which contains a RTD function is entered
`Disconnect Data`		`TopicID`	Used to notify the RTD server that a topic is no longer in use
`Heartbeat`	`Long`		Checks to see if the RTD server is still active. Negative numbers or zero are a failure, while positive numbers indicate success
`RefreshData`		`ByRef TopicCount As Long`	This method is called to get new data, but only after being notified by the RTD server that there is new data
`ServerStart`	`Long`	`Callback Object`	Called immediately after a RTD server is instantiated. Negative numbers or zero are a failure, while positive numbers indicate success
`Server Terminate`			Used to terminate the connection to the server

IRTDUpdateEvent Object

Represents Real-Time update events. This object is used to set the interval between updates for an `IrtdServer` object using the `HeartbeatInterval` property. This object is returned when you use the `ServerStart` method of the `IrtdServer` object to connect to a Real-Time Data server.

IRTDUpdateEvent Properties

Name	Returns	Description
`Heartbeat Interval`	`Long`	Set/Get the interval between updates for RTD

IRTDUpdateEvent Methods

Name	Returns	Parameters	Description
`Disconnect`			Instructs the RTD server to disconnect from the specified object
`UpdateNotify`			Excel is informed by the RTD server that new data has been received

LeaderLines Object

The `LeaderLines` object contains the formatting attributes associated with leader lines on charts connecting data labels to the actual points. The parent of the `LeaderLines` object is the `Series` object. Use the `HasLeaderLines` property of the `Series` object to create a `LeaderLines` object and make sure one exists.

LeaderLines Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

LeaderLines Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the leader lines

LeaderLines Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the `LeaderLines` object
`Select`			Selects the leader lines on the chart

Example: LeaderLines Object

```
Sub AddAndFormatLeaderLines()
   Dim oLL As LeaderLines
   'Using the first series of the PIE chart
   With Charts(1).SeriesCollection(1)
      'Add labels with leader lines (if required)
      .ApplyDataLabels HasLeaderLines:=True

      'Position the labels
      .DataLabels.Position = xlLabelPositionBestFit

      'Get the LeaderLines Object. If all labels are
      'in their default position, this will give an error
      Set oLL = .LeaderLines
   End With
   'Format the leader lines
   With oLL
      .Border.LineStyle = xlContinuous
      .Border.ColorIndex = 5
   End With
End Sub
```

Legend Object

The `Legend` object contains the formatting options and legend entries for a particular chart. The parent of the `Legend` object is the `Chart` object. Use the `HasLegend` property of the `Chart` object to create a `Legend` object and to make sure one exists.

Legend Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Legend Properties

Name	Returns	Description
`AutoScaleFont`	`Variant`	Set/Get whether the font size will change automatically if the parent chart changes sizes
`Border`	`Border`	Read-only. Returns the border's properties around the legend
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the legend of a chart
`Font`	`Font`	Read-only. Returns an object containing `Font` options for the legend text
`Height`	`Double`	Set/Get the height of the legend box
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of a legend (for example, interior color)
`Left`	`Double`	Set/Get the distance from the left edge of the legend box to the left edge of the chart containing the legend

Name	Returns	Description
Name	String	Read-only. Returns the name of the Legend object
Position	XlLegend Position	Set/Get the position of the legend on the chart (for example, xlLegendPositionCorner, xlLegendPositionLeft)
Shadow	Boolean	Set/Get whether the legend has a shadow effect
Top	Double	Set/Get the distance from the top edge of the legend box to the top edge of the chart containing the legend
Width	Double	Set/Get the width of the legend box

Legend Methods

Name	Returns	Parameters	Description
Clear	Variant		Clears the legend
Delete	Variant		Deletes the legend
LegendEntries	Object	[Index]	Returns either one LegendEntry object or a LegendEntries collection depending if an Index parameter is specified. Contains all the legend text and markers
Select	Variant		Selects the legend on the chart

Example: Legend Object

```
Sub PlaceLegend()
   Dim oLgnd As Legend
   'Make sure the chart has a legend
   Charts(1).HasLegend = True
   'Get the Legend
   Set oLgnd = Charts(1).Legend
   'Position and format the legend
   With oLgnd
      .Position = xlLegendPositionRight
      .Border.LineStyle = xlNone
      .AutoScaleFont = False
   End With
End Sub
```

LegendEntry Object and the LegendEntries Collection

The LegendEntries collection contains the collection of entries in a legend. Each LegendEntry object represents a single entry in a legend. This consists of the legend entry text and the legend entry marker. The legend entry text is always the associated series name or trendline name. The parent of the LegendEntries collection is the Legend object.

The LegendEntries collection contains one property besides the typical collection attributes. The Count property returns the number of LegendEntry objects in the collection.

LegendEntry Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

LegendEntry Properties

Name	Returns	Description
AutoScaleFont	Variant	Set/Get whether the font size will change automatically if the parent chart changes sizes
Font	Font	Read-only. Returns an object containing Font options for the legend entry text
Height	Double	Read-only. Returns the height of the legend entry
Index	Long	Read-only. Returns the position of the LegendEntry in the LegendEntries collection
Left	Double	Read-only. Returns the distance from the left edge of the legend entry box to the left edge of the chart
LegendKey	LegendKey	Read-only. Returns an object containing formatting associated with the legend entry marker
Top	Double	Read-only. Returns the distance from the top edge of the legend entry box to the top edge of the chart
Width	Double	Read-only. Returns the width of the legend entry

LegendEntry Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the LegendEntry object
Select	Variant		Selects the legend entry on the chart

Example: LegendEntry Object and the LegendEntries Collection

```
Sub FormatLegendEntries()
   Dim oLE As LegendEntry
   'Make sure the chart has a legend
   Charts(1).HasLegend = True
   'Loop through all the legend entries
   For Each oLE In Charts(1).Legend.LegendEntries
      'Format each entry with a different font style
      With oLE
         .Font.Size = 10 + .Index * 4
```

```
            .Font.Bold = (.Index Mod 2) = 0
            .Font.ColorIndex = .Index
         End With
      Next
   End Sub
```

LegendKey Object

The LegendKey object contains properties and methods to manipulate the formatting associated with a legend key entry marker. A legend key is a visual representation, such as a color, that identifies a specific series or trendline.

LegendKey Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

LegendKey Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the legend key
Fill	ChartFill Format	Read-only. Returns an object containing fill formatting options for the legend key of a series or trendline in a chart
Height	Double	Read-only. Returns the height of the legend entry key
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the legend key (for example, interior color)
InvertIf Negative	Boolean	Set/Get whether the color in the legend key is inverted if the values are negative
Left	Double	Read-only. Returns the distance from the left edge of the legend key entry box to the left edge of the chart
Marker Background Color	Long	Set/Get the color of the legend key background. Use the RGB function to create the color value
Marker Background ColorIndex	XlColor Index	Set/Get the color of the legend key background. Use the XlColorIndex constants or an index value in the current color palette
Marker Foreground Color	Long	Set/Get the color of the legend key foreground. Use the RGB function to create the color value
Marker Foreground ColorIndex	XlColor Index	Set/Get the color of the legend key foreground. Use the XlColorIndex constants or an index value in the current color palette

Continues

Name	Returns	Description
MarkerSize	Long	Set/Get the size of the legend key marker
MarkerStyle	XlMarker Style	Set/Get the type of marker to use as the legend key (for example, square, diamond, triangle, picture, etc.)
PictureType	Long	Set/Get how an associated picture is displayed on the legend (for example, stretched, tiled). Use the XlPictureType constants
PictureUnit	Long	Set/Get how many units a picture represents if the PictureType property is set to xlScale
Shadow	Boolean	Set/Get whether a shadow effect appears around the legend entry key
Smooth	Boolean	Set/Get whether the legend key has smooth curving enabled
Top	Double	Read-only. Returns the distance from the top edge of the legend entry key box to the top edge of the chart
Width	Double	Read-only. Returns the width of the legend entry key box.

LegendKey Methods

Name	Returns	Parameters	Description
ClearFormats	Variant		Clears the formatting made on the LegendKey object
Delete	Variant		Deletes the LegendKey object
Select	Variant		Selects the legend key on the parent chart

Example: LegendKey Object

```
Sub FormatLegendKeys()
   Dim oLE As LegendEntry
   Dim oLK As LegendKey
   'Make sure the chart has a legend
   Charts(1).HasLegend = True
   'Loop through all the legend entries
   For Each oLE In Charts(1).Legend.LegendEntries
      'Get the legend key for the entry
      Set oLK = oLE.LegendKey

      'Format each legend key with a different colour and size
      With oLK
         .MarkerForegroundColor = oLE.Index
         .MarkerSize = oLE.Index * 2 + 1
      End With
   Next
End Sub
```

LineFormat Object

The LineFormat object represents the formatting associated with the line of the parent Shape object. The Line property of the Shape object is used to access the LineFormat object. The LineFormat object is commonly used to change line properties such as arrowhead styles and directions.

LineFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

LineFormat Properties

Name	Returns	Description
BackColor	Color Format	Read-only. Returns an object allowing manipulation of the background color of the line
Begin Arrowhead Length	Mso Arrowhead Length	Set/Get the arrowhead length on the start of the line
Begin Arrowhead Style	Mso Arrowhead Style	Set/Get how the arrowhead looks on the start of the line
Begin Arrowhead Width	Mso Arrowhead Width	Set/Get the arrowhead width on the start of the line
DashStyle	MsoLine Dash Style	Set/Get the style of the line
EndArrowhead Length	Mso Arrowhead Length	Set/Get the arrowhead length on the end of the line
EndArrowhead Style	Mso Arrowhead Style	Set/Get how the arrowhead looks on the end of the line
EndArrowhead Width	Mso Arrowhead Width	Set/Get the arrowhead width on the end of the line
ForeColor	Color Format	Read-only. Returns an object allowing manipulation of the background color of the line
Pattern	MsoPattern Type	Set/Get the pattern used on the line
Style	MsoLine Style	Set/Get the line style
Transparency	Single	Set/Get how transparent (1) or opaque (0) the line is
Visible	MsoTri State	Set/Get whether the line is visible
Weight	Single	Set/Get how thick the line is

Example: LineFormat Object

```
Sub AddAndFormatLine()
   Dim oShp As Shape
   Dim oLF As LineFormat
   'Add a line shape
   Set oShp = ActiveSheet.Shapes.AddLine(100, 100, 200, 250)
   'Get the line format object
   Set oLF = oShp.Line
   'Set the line format
   With oLF
      .BeginArrowheadStyle = msoArrowheadOval
      .EndArrowheadStyle = msoArrowheadTriangle
      .EndArrowheadLength = msoArrowheadLong
      .EndArrowheadWidth = msoArrowheadWide
      .Style = msoLineSingle
   End With
End Sub
```

LinkFormat Object

The `LinkFormat` object represents the linking attributes associated with an `OLE` object or picture. The `LinkFormat` object is associated with a `Shape` object. Only `Shape` objects that are valid `OLE` objects can access the `LinkFormat` object.

LinkFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

LinkFormat Properties

Name	Returns	Description
`AutoUpdate`	`Boolean`	Set/Get whether the parent `Shape` object is updated whenever the source file changes or when the parent object is opened
`Locked`	`Boolean`	Set/Get whether the parent `Shape` object does not update itself against the source file

LinkFormat Methods

Name	Returns	Parameters	Description
`Update`			Updates the parent `Shape` object with the source file data

Example: LinkFormat Object

```
Sub UpdateShapeLinks()
   Dim oShp As Shape
```

```
    Dim oLnkForm As LinkFormat
    'Loop through all the shapes in the sheet
    For Each oShp In ActiveSheet.Shapes
       'Is it a linked shape?
       If oShp.Type = msoLinkedOLEObject Or oShp.Type = msoLinkedPicture Then

          'Yes, so get the link format
          Set oLnkForm = oShp.LinkFormat

          'and update the link
          oLnkForm.Update
       End If
    Next
 End Sub
```

ListColumn Object

The ListColumn object represents a column in a List. The ListColumns Collection contains all columns within in list, represented by many ListColumn objects.

ListColumn Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ListColumn Properties

Name	Returns	Description
Index	Integer	Index number of ListColumn object in the List
ListDataFormat	ListDataFormat Object	Returns a ListDataFormat for the ListColumn object
Name	String	Name of this object
Range		Returns a range representing the range for which the current ListColumn applies to the current list
SharePointFormula	String	Returns a string which is the formula for a calculated field
TotalsCalulation		Determines the type of calculation in the totals row
XPath		Returns an XPath object for the element mapped to the specific Range

ListColumn Methods

Name	Returns	Parameters	Description
Delete			Deletes a column of data in the list

ListDataFormat Object

The ListDataFormat holds all of the data type properties for a ListColumn object.

ListDataFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ListDataFormat Properties

Name	Returns	Description
AllowFillIn	Boolean	Determines if the user can provide values or if they are restricted to use information from a list
Choices	String Array	Contains a string array of choices
DecimalPlaces	Long	Determines the number of decimal places to show
DefaultValue	Variant	Default value to use for this item
IsPercent	Boolean	Determines if this item should be displayed as a percentage
LCID	Long	Determines the currency symbol that is used
MaxCharacters	Long	The maximum number of characters that can be entered
MaxNumber	Variant	The largest number used
MinNumber	Variant	The minimum number that can be used
ReadOnly	Boolean	Determines if the item should be read-only and therefore disallow changes
Required	Boolean	Determines if the item is required
Type	XlListDataType	Used when a item is links to a SharePoint site

ListObject Object

The ListObject represents a list object within a workbook.

ListObject Properties

Name	Returns	Description
Application	Object	Returns an Application object which represents the Excel application
Count	Integer	Return the number of items within the collection
Creator	Long	Returns a Long value that represents the application that created the object. This value can then be converted to a hexadecimal value and then to ASCII

Name	Returns	Description
`Item`	`Object`	Returns a single object from a collection of objects
`Parent`	`Object`	Returns a reference to the parent of the object

ListObject Methods

Name	Returns	Parameters	Description
`Delete`			Deletes an item from the collection

`Publish`	`String`	`[Target – String Array]`
		`[LinkSource – Boolean]`
			Publishes the object to a Windows Sharepoint Server. The Target string array must contain the following elements:
			0 – URL of SharePoint Server
			1 – ListName
			2 – Description of list

`Refresh`			Refreshes the current data from a Windows SharePoint Server
`Resize`		`Range`	Allows a ListObject to be resized of a certain range
`Unlink`			Removes the current link to a Windows SharePoint Server
`Unlist`			Removes the list functionality from a `ListObject` thus turning all data into a regular range of data
`Update Change`			Updates all information on a Windows SharePoint Server by synchronizing all changes

ListRow Object

The ListRow, as the name implies, represents a Row within a List object.

ListRow Properties

Name	Returns	Description
`Application`	`Object`	Returns an Application object which represents the Excel application
`Creator`	`Long`	Returns a Long value that represents the application that created the object. This value can then be converted to a hexadecimal value and then to ASCII

Continues

Name	Returns	Description
`Index`	`Long`	Returns an index number for an item within a collection
`InvalidData`	`Boolean`	Returns a Boolean value for a row in a list that determines if the row has or has not passed validation
`Parent`	`Object`	Returns a reference to the parent of the object
`Range`	`Range`	Returns a range to which the current object applies

ListRow Methods

Name	Returns	Parameters	Description
`Delete`			Deletes an item from the collection

Mailer Object

The `Mailer` object is used on the Macintosh to mail Excel files using the PowerTalk Mailer.

Mailer Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Mailer Properties

Name	Returns	Description
`BCCRecipients`	`Variant`	Set/Get the list of blind copies
`CCRecipients`	`Variant`	Set/Get the list of carbon copies
`Enclosures`	`Variant`	Set/Get the list of enclosures
`Received`	`Boolean`	Read-only. Returns whether the mail message was received
`SendDateTime`	`Date`	Read-only. Returns the date and time the message was sent
`Sender`	`String`	Read-only. Returns the name of the mail message sender
`Subject`	`String`	Set/Get the subject line of the mail message
`ToRecipients`	`Variant`	Set/Get the array of recipient names
`WhichAddress`	`Variant`	Set/Get the address that the mail message originates from

Name Object and the Names Collection

The `Names` collection holds the list of named ranges in a workbook. Each `Name` object describes a range of cells in a workbook that can be accessed by the name. Some `Name` objects are built-in (for example, Print_Area) and others are user defined. The parent of the Names collection can be the WorkBook, Application, and Worksheet object. The Name object can also be accessed through the Range object.

The Names collection has an Add method besides the typical collection attributes. The Add method adds a Name object to the collection. The parameters of the Add method correspond to the properties of the Name object.

Name Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Name Properties

Name	Returns	Description
Category	String	Set/Get the category of the Name in the language used to create the macro. Valid only if the Name is a custom function or command
CategoryLocal	String	Set/Get the category of the Name in the language of the end user. Valid only if the Name is a custom function or command
Index	Long	Read-only. Returns the spot that Name is located in the Names collection
MacroType	XlXLMMacro Type	Set/Get if the Name refers to command, function, or just a range
Name	String	Set/Get the name of the Name object in the language of the macro
NameLocal	String	Set/Get the name of the Name object in the language of the end user
RefersTo	Variant	Set/Get the range text that the Name refers to in the language of the macro and in A1 notation style
RefersToLocal	Variant	Set/Get the range text that the Name refers to in the language of the user and in A1 notation style
RefersToR1C1	Variant	Set/Get the range text that the Name refers to in the language of the macro and in R1C1 notation style
RefersToR1C1 Local	Variant	Set/Get the range text that the Name refers to in the language of the user and in R1C1 notation style
RefersToRange	Range	Read-only. Returns the range that the Name refers to
ShortcutKey	String	Set/Get the shortcut key to trigger a Microsoft Excel 4.0 macro associated with a Name
Value	String	Set/Get the range text that the Name refers to in the language of the macro and in A1 notation style
Visible	Boolean	Set/Get whether the name of the Name object appears in the Names dialog box in Excel

Name Methods

Name	Returns	Parameters	Description
Delete			Deletes the Name object from the collection

Example: Name Object and the Names Collection

```
Sub DeleteInvalidNames()
   Dim oName As Name
   'Loop through all the names in the active workbook
   For Each oName In ActiveWorkbook.Names
      'Is it an invalid name?
      If InStr(1, oName.RefersTo, "#REF") > 0 Then

         'Yes, so log it
         Debug.Print "Deleted name " & oName.Name & " - " & oName.RefersToLocal

         'and delete it from the collection
         oName.Delete
      End If
   Next
End Sub
```

ODBCError Object and the ODBCErrors Collection

The ODBCErrors collection contains a list of errors that occurred by the most recent query using an ODBC connection. Each ODBCError object contains information describing an error that occurred on the most recent query using an ODBC connection. If the most recent query against an ODBC source did not generate any errors then the collection is empty.

The ODBCErrors collection has a Count property besides the typical collection attributes. The Count property returns the number of ODBCError objects in the collection.

ODBCError Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ODBCError Properties

Name	Returns	Description
ErrorString	String	Read-only. Returns the error string generated from the ODBC connection
SqlState	String	Read-only. Returns the SQL state error generated from the ODBC connection

Example: ODBCError Object and the ODBCErrors Collection

```
Sub CheckODBCErrors()
   Dim oErr As ODBCError
   Dim sMsg As String
   'Continue after errors
   On Error Resume Next
   'Don't show logon prompts etc
   Application.DisplayAlerts = False
   'Update an ODBC query table
   ActiveSheet.QueryTables(1).Refresh
   'Any errors?
   If Application.ODBCErrors.Count = 0 Then
      'No, so all OK
      MsgBox "Updated OK"
   Else
      'Yes, so list them all
      sMsg = "The following error(s) occured during the update"
      For Each oErr In Application.ODBCErrors
         sMsg = sMsg & vbCrLf & oErr.ErrorString & " (" & oErr.SqlState & ")"
      Next
      MsgBox sMsg
   End If
End Sub
```

OLEDBError Object and the OLEDBErrors Collection

The `OLEDBErrors` collection contains a list of errors that occurred by the most recent query using an OLE DB provider. Each `OLEDBError` object contains information describing an error that occurred on the most recent query using an OLE DB provider. If the most recent query against an OLE DB provider did not generate any errors then the collection is empty.

The `OLEDBErrors` collection has a `Count` property besides the typical collection attributes. The `Count` property returns the number of `OLEDBError` objects in the collection.

OLEDBError Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

OLEDBError Properties

Name	Returns	Description
`ErrorString`	`String`	Read-only. Returns the error string generated from the OLE DB provider
`Native`	`Long`	Read-only. Returns a provider-specific error number describing the error
`Number`	`Long`	Read-only. Returns the error number describing the error

Continues

Name	Returns	Description
`SqlState`	`String`	Read-only. Returns the SQL state error generated from the OLE DB provider
`Stage`	`Long`	Read-only. Returns the stage of an error generated from the OLE DB provider

Example: OLEDBError Object and the OLEDBErrors Collection

```
Sub CheckOLEDbErrors()
   Dim oErr As OLEDBError
   Dim sMsg As String
   'Continue after errors
   On Error Resume Next
   'Don't show logon prompts etc
   Application.DisplayAlerts = False
   'Update an OLE DB pivot table
   ActiveSheet.PivotTables(1).Refresh
   'Any errors?
   If Application.OLEDBErrors.Count = 0 Then
      'No, so all OK
      MsgBox "Updated OK"
   Else
      'Yes, so list them all
      sMsg = "The following error(s) occured during the update"
      For Each oErr In Application.OLEDBErrors
         sMsg = sMsg & vbCrLf & oErr.ErrorString & " (" & oErr.SqlState & ")"
      Next
      MsgBox sMsg
   End If
End Sub
```

OLEFormat Object

The `OLEFormat` object represents all attributes associated with an `OLE` object or `ActiveX` object for linking. Linking characteristics are taken care of by the `LinkFormat` object. The `Shape` object is the parent of the `OLEFormat` object. The parent `Shape` object must be a linked or embedded object to be able to use this object.

OLEFormat Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

OLEFormat Properties

Name	Returns	Description
`Object`	`Object`	Read-only. Returns a reference to the parent `OLE` object
`ProgId`	`String`	Read-only. Returns the `ProgId` of the parent `OLE` object

OLEFormat Methods

Name	Returns	Parameters	Description
Activate			Activates and opens the parent OLE object
Verb		[Verb]	Performs an action on the parent OLE object that triggers a reaction in the OLE object (for example, xlOpen)

Example: OLEFormat Object

```
Sub PrintEmbeddedWordDocuments1()
   Dim oShp As Shape
   Dim oOF As OLEFormat
   'Loop through all the shapes in the sheet
   For Each oShp In ActiveSheet.Shapes
      'Is it an embedded object
      If oShp.Type = msoEmbeddedOLEObject Then
         'Get the embedded object's format
         Set oOF = oShp.OLEFormat

         'Is it a Word document?
         If oOF.ProgId Like "Word.Document*" Then

            'Yes, so print the Word document.
            'The first .Object gives us the generic
            'OLEObject contained in the Shape.
            'The second .Object gives us the Word object
            'contained within the OLEObject
            oOF.Object.Object.PrintOut
         End If
      End If
   Next
End Sub
```

OLEObject Object and the OLEObjects Collection

The OLEObjects collection holds all the ActiveX controls, linked OLE objects and embedded OLE objects on a worksheet or chart. An OLE object represents an ActiveX control, a linked OLE object, or an embedded OLE object on a worksheet or chart.

The OLEObjects collection has many properties and methods besides the typical collection attributes. These are listed in the following table.

OLEObjects Collection Properties and Methods

Name	Returns	Description
AutoLoad	Boolean	Set/Get whether the OLE object is automatically loaded when the workbook is opened. Not valid for ActiveX controls. Usually set to False. This property only works if there is one OLEObject in the collection

Continues

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the OLE object. This property only works if there is one OLEObject in the collection
Count	Long	Read-only. Returns the number of OLEObject objects in the collection
Enabled	Boolean	Set/Get whether the OLEObject is enabled. This property only works if there is one OLEObject in the collection
Height	Double	Set/Get the height of OLEObject frame. This property only works if there is one OLEObject in the collection
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the OLE object (for example, interior color). This property only works if there is one OLEObject in the collection
Left	Double	Set/Get the distance from the left edge of the OLEObject frame to the left edge of the sheet. This property only works if there is one OLEObject in the collection
Locked	Boolean	Set/Get whether editing will be possible when the parent sheet is protected. This property only works if there is one OLEObject in the collection
Placement	Variant	Set/Get how the OLEObject object is anchored to the sheet (for example, free floating, move with cells). Use the XlPlacement constants to set this property. This property only works if there is one OLEObject in the collection
PrintObject	Boolean	Set/Get whether the OLEObject on the sheet will be printed when the sheet is printed. This property only works if there is one OLEObject in the collection
Shadow	Boolean	Set/Get whether a shadow appears around the OLE object. This property only works if there is one OLEObject in the collection
ShapeRange	ShapeRange	Read-only. Returns the OLE object as a Shape object. This property only works if there is one OLEObject in the collection
SourceName	String	Set/Get the link source name of the OLE object. This property only works if there is one OLEObject in the collection
Top	Double	Set/Get the distance from top edge of the OLE object to the top of the parent sheet. This property only works if there is one OLEObject in the collection
Visible	Boolean	Set/Get whether all the OLEObjects in the collection are visible

Name	Returns	Description
Width	Double	Set/Get the width of the OLE object frame. This property only works if there is one OLEObject in the collection
ZOrder	Long	Read-only. Returns the position of the OLE object among all the other objects on the sheet. This property only works if there is one OLEObject in the collection
Add	OLEObject	Method. Parameters: [ClassType],[Filename], [Link],[DisplayAsIcon],[IconFileName], [IconIndex],[IconLabel],[Left],[Top], [Width], [Height]. Adds an OLE object to the collection of OLEObjects. The position of the new OLE object can be specified by using the Left, Top, Width, and Height parameters. The type of OLEObject (ClassType) or its location (FileName) can be specified as well. The other parameters have equivalent OLEObject properties
BringToFront	Variant	Method. Brings all the OLE objects in the collection to the front of all the other objects
Copy	Variant	Method. Copies all the OLE objects in the collection into the clipboard
CopyPicture	Variant	Method. Parameters: Appearance As XlPictureAppearance, Format As XlCopyPictureFormat. Copies the OLE objects in the collection into the clipboard as a picture. The Appearance parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The Format parameter can specify the type of picture that will be put into the clipboard
Cut	Variant	Method. Cuts all the OLE objects in the collection into the clipboard
Delete	Variant	Method. Deletes all the OLEObject objects in the collection into the clipboard
Duplicate		Method. Duplicates all the OLEObject objects in the collection into the parent sheet
Select	Variant	Method. Parameters: [Replace]. Selects all the OLEObject objects in the collection
SendToBack	Variant	Method. Brings the OLEObject objects in the collection to the back of other objects

OLEObject Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

OLEObject Properties

Name	Returns	Description
AltHTML	String	Set/Get what HTML to use when the document is saved as a Web page instead of embedding the OLE control
AutoLoad	Boolean	Set/Get whether the OLE object is automatically loaded when the workbook is opened. Not valid for ActiveX controls. Usually set to False
AutoUpdate	Boolean	Set/Get whether the OLE object is automatically updated when the source changes. Valid only for linked objects (OLEType=xlOLELink)
Border	Border	Read-only. Returns the border's properties around the OLE object
BottomRight Cell	Range	Read-only. Returns the single cell range located under the lower-right corner of the OLE object
Enabled	Boolean	Set/Get whether the OLEObject is enabled
Height	Double	Set/Get the height of OLEObject frame
Index	Long	Read-only. Returns the spot in the collection where the current OLEObject is located
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the OLE object (for example, interior color)
Left	Double	Set/Get the distance from the left edge of the OLEObject frame to the left edge of the sheet
LinkedCell	String	Set/Get the range that receives the value from the results of the OLE object
ListFill Range	String	Set/Get the range that holds the values used by an ActiveX list box
Locked	Boolean	Set/Get whether editing will be possible when the parent sheet is protected
Name	String	Set/Get the name of the OLE object
Object		Read-only. Returns access to some of the properties and methods of the underlying object in the OLE object
OLEType	Variant	Read-only. Returns the type OLE object: xlOLELink or xlOLEEmbed. Use the XlOLEType constants
Placement	Variant	Set/Get how the OLEObject object is anchored to the sheet (for example, free floating, move with cells). Use the XlPlacement constants to set this property
PrintObject	Boolean	Set/Get whether the OLEObject on the sheet will be printed when the sheet is printed

Name	Returns	Description
ProgId	String	Read-only. Returns the programmatic identifier associated with the OLE object (for example, "Excel.Application")
Shadow	Boolean	Set/Get whether a shadow appears around the OLE object
ShapeRange	ShapeRange	Read-only. Returns the OLE object as a Shape object
SourceName	String	Set/Get the link source name of the OLE object
Top	Double	Set/Get the distance from top edge of the OLE object to the top of the parent sheet
TopLeftCell	Range	Read-only. Returns the single cell range located above the top-left corner of the OLE object
Visible	Boolean	Set/Get whether the OLEObject is visible
Width	Double	Set/Get the width of the OLE object frame
ZOrder	Long	Read-only. The position of the OLE object among all the other objects on the sheet

OLEObject Methods

Name	Returns	Parameters	Description
Activate	Variant		Sets the focus and activates the OLE object
BringToFront	Variant		Brings the OLE object to the front of all the other objects
Copy	Variant		Copies the OLE object into the clipboard
CopyPicture	Variant	Appearance As XlPicture Appearance, Format As XlCopyPicture Format	Copies the OLE object into the clipboard as a picture. The Appearance parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The Format parameter can specify the type of picture that will be put into the clipboard
Cut	Variant		Cuts the OLE object into the clipboard
Delete	Variant		Deletes the OLEObject object into the clipboard
Duplicate			Duplicates the OLEObject object into the parent sheet
Select	Variant	[Replace]	Selects the OLEObject object
SendToBack	Variant		Brings the OLEObject object to the back of other objects

Continues

Name	Returns	Parameters	Description
Update	Variant		Updates the OLE object link, if applicable
Verb	Variant	Verb As XlOLEVerb	Performs an action on the parent OLE object that triggers a reaction in the OLE object (for example, xlOpen)

OLEObject Events

Name	Parameters	Description
GotFocus		Triggered when the OLE object gets focus
LostFocus		Triggered when the OLE object loses focus

Example: OLEObject Object and the OLEObjects Collection

```
Sub PrintEmbeddedWordDocuments2()
   Dim oOLE As OLEObject
   'Loop through all the shapes in the sheet
   For Each oOLE In ActiveSheet.OLEObjects
      'Is it a Word document?
      If oOLE.ProgId Like "Word.Document*" Then
         'Yes, so print the Word document.
         oOLE.Object.PrintOut
      End If
   Next
End Sub
```

Outline Object

The Outline object represents the outline feature in Excel. The parent of the Outline object is the WorkSheet object.

Outline Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Outline Properties

Name	Returns	Description
Automatic Styles	Boolean	Set/Get whether the outline has styles automatically assigned by Excel
Summary Column	XlSummary Column	Set/Get whether the summary columns are to the left (xlLeft) or the right (xlRight) of the detail columns
SummaryRow	XlSummary Row	Set/Get whether the summary rows are above (xlAbove) or below (xlBelow) the detail rows

Outline Methods

Name	Returns	Parameters	Description
ShowLevels	Variant	[RowLevels], [Column Levels]	Show the detail of rows and columns at a higher level as specified by the RowLevels and ColumnLevels parameters, respectively. The rest of the detail for the other levels is hidden

Example: Outline Object

```
Sub ShowOutlines()
   Dim oOutl As Outline
   'Group some rows
   ActiveSheet.Range("4:5").Group
   'Get the Outline object
   Set oOutl = ActiveSheet.Outline
   'Format the outline display
   With oOutl
      .ShowLevels 1
      .SummaryRow = xlSummaryAbove
   End With
End Sub
```

PageSetup Object

The PageSetup object contains the functionality of the Page Setup dialog box. Possible parents of the PageSetup object are the Chart and Worksheet object.

PageSetup Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PageSetup Properties

Name	Returns	Description
BlackAndWhite	Boolean	Set/Get whether worksheet items will be printed in black and white only. Not valid when parents are Chart objects
BottomMargin	Double	Set/Get the bottom margin of the page in points
CenterFooter	String	Set/Get the text for the center part of the footer
CenterFooter Picture	Graphic	Read-only. Returns the picture for the center section of the footer

Continues

Name	Returns	Description
CenterHeader	String	Set/Get the text for the center part of the header
CenterHeader Picture	Graphic	Read-only. Returns the picture for the center section of the footer
Center Horizontally	Boolean	Set/Get whether the worksheet or chart will be horizontally centered on the page
Center Vertically	Boolean	Set/Get whether the worksheet or chart will be vertically centered on the page
ChartSize	XlObjectSize	Set/Get how the chart is scaled to fit one page (xlFullPage or xlFitToPage) or the same way it appears on the screen (xlScreenSize). Not valid when parents are Worksheet objects
Draft	Boolean	Set/Get whether graphics will be printed. True means graphics will not be printed
FirstPage Number	Long	Set/Get which number will be used as the first page number. Use xlAutomatic to have Excel choose this (default)
FitToPages Tall	Variant	Set/Get how many pages tall the sheet will be scaled to. Setting this property to False will mean the FitToPagesWide property will be used
FitToPages Wide	Variant	Set/Get how many pages wide the sheet will be scaled to. Setting this property to False will mean the FitToPagesTall property will be used
FooterMargin	Double	Set/Get the distance from the page bottom to the footer of the page in points
HeaderMargin	Double	Set/Get the distance from the page top to the header of the page in points
LeftFooter	String	Set/Get the text for the left part of the footer
LeftFooter Picture	Graphic	Read-only. Returns the picture for the left section of the footer
LeftHeader	String	Set/Get the text for the center part of the header
LeftHeader Picture	Graphic	Read-only. Returns the picture for the left section of the header
LeftMargin	Double	Set/Get the left margin of the page in points
Order	XlOrder	Set/Get the manner that Excel numbers pages for large worksheets (for example, xlDownTheOver, xlOverThenDown). Not valid for parents that are Chart objects

Name	Returns	Description
`Orientation`	`XlPage Orientation`	Set/Get the page orientation: `xlLandscape` or `xlPortrait`
`PaperSize`	`XlPaper Size`	Set/Get the paper size (for example, `xlPaperLetter`, `xlPaperLegal`, etc.)
`PrintArea`	`String`	Set/Get the range on a worksheet that will be printed. If this property is set to `False` then the entire sheet is printed. Not valid for parents that are `Chart` objects
`PrintComments`	`XlPrint Location`	Set/Get how comments are printed or if they are at all (for example, `xlPrintInPlace`, `xlPrintNoComments`)
`PrintErrors`	`XlPrint Errors`	Set/Get the type of print error displayed. This allows the suppression of error values when printing a worksheet
`Print Gridlines`	`Boolean`	Set/Get whether cell gridlines are printed for a worksheet. Not valid for parents that are `Chart` objects
`PrintHeadings`	`Boolean`	Set/Get whether row and column headings are printed
`PrintNotes`	`Boolean`	Set/Get whether notes attached to the cells are printed at the end as endnotes. Not valid if parents are `Chart` objects
`PrintTitle Columns`	`String`	Set/Get which columns to repeat on the left side of every printed page
`PrintTitleRows`	`String`	Set/Get which rows to repeat on the top of every page
`RightFooter`	`String`	Set/Get the text for the right part of the footer
`RightFooter Picture`	`Graphic`	Read-only. Returns the picture for the right section of the footer
`RightHeader`	`String`	Set/Get the text for the center part of the header
`RightHeader Picture`	`Graphic`	Read-only. Returns the picture for the right section of the header
`RightMargin`	`Double`	Set/Get the right margin of the page in points
`TopMargin`	`Double`	Set/Get the top margin of the page in points
`Zoom`	`Variant`	Set/Get the percentage scaling that will occur for the worksheet. Not valid for parents that are `Chart` objects. (10 to 400 percent)

PageSetup Methods

Name	Returns	Parameters	Description
`PrintQuality`	`Variant`	`[Index]`	Set/Get the print quality. The `Index` parameter can be used to specify horizontal (`1`) or vertical (`2`) print quality

Example: PageSetup Object

```
Sub SetUpPage()
   Dim oPS As PageSetup
   'Get the sheet's PageSetup object
   Set oPS = ActiveSheet.PageSetup
   'Set up the page
   With oPS
      'Set the paper size to the local default
      .PaperSize = fnLocalPaperSize
      .Orientation = xlPortrait
      'etc.
   End With
End Sub
Function fnLocalPaperSize() As XlPaperSize
   'Remember the paper size when we've read it
   Static iPaperSize As XlPaperSize
   'Is it set?
   If iPaperSize = 0 Then
      'No, so create a new workbook and read off the paper size
      With Workbooks.Add
         iPaperSize = .Worksheets(1).PageSetup.PaperSize
         .Close False
      End With
   End If
   'Return the paper size
   fnLocalPaperSize = iPaperSize
End Function
```

Pane Object and the Panes Collection

The `Panes` collection allows manipulation of the different panes of a window. A `Pane` object is equivalent to the single pane of a window. The parent object of the `Panes` collection is the `Window` object.

Besides the typical collection properties and methods, the `Panes` collection has a `Count` property. The `Count` property returns the number of `Pane` objects in the collection.

Pane Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Pane Properties

Name	Returns	Description
`Index`	`Long`	Read-only. Returns the spot in the collection where `Pane` object is located
`ScrollColumn`	`Long`	Set/Get which column number is the leftmost column in the pane window
`ScrollRow`	`Long`	Set/Get which row number is the top row in the pane window
`VisibleRange`	`Range`	Read-only. Returns the cell range that is visible in the pane

Pane Methods

Name	Returns	Parameters	Description
`Activate`	`Boolean`		Activates the pane
`LargeScroll`	`Variant`	`[Down], [Up], [ToRight], [ToLeft]`	Causes the document to scroll in a certain direction a screen-full at a time, as specified by the parameters
`ScrollIntoView`		`Left As Long, Top As Long, Width As Long, Height As Long, [Start]`	Scrolls the spot specified by the `Left`, `Top`, `Width`, and `Height` parameters to either the upper-left corner of the pane (`Start = True`) or the lower-right corner of the pane (`Start = False`). The `Left`, `Top`, `Width`, and `Height` parameters are specified in points
`SmallScroll`	`Variant`	`[Down], [Up], [ToRight], [ToLeft]`	Causes the document to scroll in a certain direction a document line at a time, as specified by the parameters

Example: Pane Object and the Panes Collection

```
Sub ScrollActivePane()
   Dim oPane As Pane
   Dim oRNg As Range
   'The range to show in the pane
   Set oRNg = Range("G3:J10")
   'Get the active pane
   Set oPane = Application.ActiveWindow.ActivePane
   'Scroll the pane to show the range in the top-left corner
   oPane.ScrollColumn = oRNg.Column
   oPane.ScrollRow = oRNg.Row
End Sub
```

Parameter Object and the Parameters Collection

The Parameters collection holds the list of parameters associated with a query table. If no parameters exist then the collection has no Parameter objects inside of it. Each Parameter object represents a single parameter for a query table. The parent of the Parameters collection is the QueryTable object.

The Parameters collection has a few extra properties and methods besides the typical collection attributes. They are listed in the following table.

Parameters Collection Properties and Methods

Name	Returns	Description
Count	Long	Read-only. Returns the number of Parameter objects in the collection
Add	Parameter	Method. Parameters: Name As String, [iDataType]. Adds a parameter to the collection creating a new query parameter for the parent query table. The type of parameter can be specified by iDataType. Use the XlParamaterDataType constants for iDataType
Delete		Method. Deletes the parameters in the collection

Parameter Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Parameter Properties

Name	Returns	Description
DataType	XlParameter DataType	Set/Get the data type of the parameter
Name	String	Set/Get the name of the parameter
Prompt String	String	Read-only. Returns the prompt that is displayed to the user when prompted for a parameter value
RefreshOn Change	Boolean	Set/Get whether the query table results are refreshed when the parameter value changes
Source Range	Range	Read-only. Returns the range of text that contains the parameter value
Type	XlParameter Type	Read-only. Returns the type of parameter (for example, xlConstant, xlPrompt, or xlRange). XlConstant means that the Value parameter has the value of the parameter. XlPrompt means that the user is prompted for the value. XlRange means that the value defines the cell range that contains the value
Value	Variant	Read-only. Returns the parameter value

Parameter Methods

Name	Returns	Parameters	Description
SetParam		Type As XlParameter Type, Value	Set/Get the type of the parameter and the value of the parameter

Example: Parameter Object and the Parameters Collection

```
Sub UpdateQuery()
   Dim oParam As Parameter
   'Using the Query Table...
   With ActiveSheet.QueryTables(1)
      'Get the first parameter
      Set oParam = .Parameters(1)

      'Set its value
      oParam.SetParam xlConstant, "Company"

      'Refresh the query
      .Refresh
   End With
End Sub
```

Phonetic Object and the Phonetics Collection

The Phonetics collection holds all of the phonetic text in a range. The Phonetic object represents a single phonetic text string. The parent of the Phonetics object is the Range object.

The Phonetics collection has a few properties and methods besides the typical collection attributes. They are listed in the following table.

Phonetics Collection Properties and Methods

Name	Returns	Description
Alignment	Long	Set/Get the alignment for the phonetic text. Use the XlPhoneticAlignment constants
Character Type	Long	Set/Get the type of phonetic text to use. Use the XLPhoneticCharacterType constants
Count	Long	Read-only. Returns the number of Phonetic objects in the collection
Font	Font	Read-only. Returns an object containing Font options for the text in the Phonetics collection

Continues

Name	Returns	Description
Length	Long	Read-only. Returns the number of phonetic text characters starting from the Start parameter
Start	Long	Read-only. Returns what the position is that represents the first character of the phonetic text strings. Valid only if there is only one Phonetic object in the collection
Text	String	Set/Get the phonetic text. Valid only if there is only one Phonetic object in the collection
Visible	Boolean	Set/Get whether the phonetic text is visible to the end user. Valid only if there is only one Phonetic object in the collection
Add		Method. Parameters: Start As Long, Length As Long, Text As String. Adds a Phonetic object to the collection at the cell specified by the parent Range object
Delete		Method. Deletes all the Phonetic objects in the collection

Phonetic Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Phonetic Properties

Name	Returns	Description
Alignment	Long	Set/Get the alignment for the phonetic text. Use the XlPhoneticAlignment constants
CharacterType	Long	Set/Get the type of phonetic text to use. Use the XLPhoneticCharacterType constants
Font	Font	Read-only. Returns an object containing Font options for the phonetic text
Text	String	Set/Get the phonetic text
Visible	Boolean	Set/Get whether the phonetic text is visible to the end user

PictureFormat Object

The PictureFormat object allows manipulation of the picture properties of the parent Shape object.

PictureFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PictureFormat Properties

Name	Returns	Description
Brightness	Single	Set/Get the brightness of the parent shape (0 to 1 where 1 is the brightest)
ColorType	MsoPicture ColorType	Set/Get the type of color setting of the parent shape
Contrast	Single	Set/Get the contrast of the parent shape (0 to 1 where 1 is the greatest contrast)
CropBottom	Single	Set/Get how much is cropped off the bottom
CropLeft	Single	Set/Get how much is cropped off the left
CropRight	Single	Set/Get how much is cropped off the right
CropTop	Single	Set/Get how much is cropped off the top
Transparency Color	Long	Set/Get the color used for transparency
Transparent Background	MsoTriState	Set/Get whether transparent colors appear transparent

PictureFormat Methods

Name	Returns	Parameters	Description
Increment Brightness		Increment As Single	Increases the brightness by the Increment value
Increment Contrast		Increment As Single	Increases the contrast by the Increment value

Example: PictureFormat Object

```
Sub SetPictureFormat()
   Dim oShp As Shape
   Dim oPF As PictureFormat
   For Each oShp In ActiveSheet.Shapes
      If oShp.Type = msoPicture Then

         'Get the PictureFormat
         Set oPF = oShp.PictureFormat

         'Format the picture
         With oPF
            .TransparentBackground = msoTrue
            .TransparencyColor = RGB(255, 0, 0)
```

```
                .ColorType = msoPictureWatermark
            End With
        End If
    Next
End Sub
```

PivotCache Object and the PivotCaches Collection

The PivotCaches collection holds the collection of memory "caches" holding the data associated with a PivotTable report. Each PivotCache object represents a single memory cache for a PivotTable report. The parent of the PivotCaches collection is the Workbook object. Also a possible parent of the PivotCache object is the PivotTable object.

The PivotCaches has a Count property and Add method besides the typical collection attributes. The Count property returns the number of items in the collection. The Add method takes a SourceType constant (from the XlPivotTableSourceType constants) and SourceData to add a PivotCache to the collection.

PivotCache Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotCache Properties

Name	Returns	Description
ADOConnection	Object	Read-only. Returns an ADO connection object if the PivotCache is connected to an OLE DB data source
Background Query	Boolean	Set/Get if the processing of queries in a PivotTable report is done asynchronously. False for OLAP data sources
CommandText	Variant	Set/Get the SQL command used to retrieve data
CommandType	XlCmdType	Set/Get the type of ComandText (for example, xlCmdSQL, xlCmdTable)
Connection	Variant	Set/Get the OLE DB connection string, the ODBC string, Web data source, path to a text file, or path to a database
EnableRefresh	Boolean	Set/Get whether the PivotTable cache data can be refreshed. Always False for OLAP data sources
Index	Long	Read-only. Returns the spot in the collection for the specific cache
IsConnected	Boolean	Read-only. Returns whether the PivotCache is still connected to a data source
Local Connection	String	Set/Get the connection string to an offline cube file. Blank for non-OLAP data sources. Use with UseLocalConnection

Name	Returns	Description
`MaintainConnection`	`Boolean`	Set/Get whether the connection to the data source does not close until the workbook is closed. Valid only against an OLE DB source
`MemoryUsed`	`Long`	Read-only. Returns the amount of bytes used by the `PivotTable` cache
`MissingItemsLimit`	`XlPivotTableMissingItems`	Set/Get the maximum number of unique items that are retained per `PivotTable` field, even when they have no supporting data in the cache records
`OLAP`	`Boolean`	Read-only. Returns whether the `PivotCache` is connected to an OLAP server
`OptimizeCache`	`Boolean`	Set/Get whether the `PivotTable` cache is optimized when it is built. Always `False` for OLE DB data sources
`QueryType`	`xlQueryType`	Read-only. Returns the type of connection associated with the query table. (For example, `xlOLEDBQuery`, `xlDAOQuery`, `xlTextImport`)
`RecordCount`	`Long`	Read-only. Returns the number of records in the `PivotTable` cache
`Recordset`		Set/Get the recordset used as the data source for the `PivotTable` cache
`RefreshDate`	`Date`	Read-only. Returns the date the cache was last refreshed
`RefreshName`	`String`	Read-only. Returns the name of the person who last refreshed the cache
`RefreshOnFileOpen`	`Boolean`	Set/Get whether the `PivotTable` cache is refreshed when the workbook is opened
`RefreshPeriod`	`Long`	Set/Get how long (minutes) between automatic refreshes from the data source. Set to 0 to disable
`RobustConnect`	`XlRobustConnect`	Set/Get the method by which the `PivotCache` connects to its data source
`SavePassword`	`Boolean`	Set/Get whether an ODBC connection password is saved with the query table
`SourceConnectionFile`	`String`	Set/Get the name of the file that was used to create the `PivotTable`
`SourceData`	`Variant`	Set/Get the data source for the `PivotTable` report
`SourceDataFile`	`String`	Read-only. Returns the name of the source data file for the `PivotCache`

Continues

Name	Returns	Description
SourceType	XlPivot TableSource Type	Read-only. Returns a value that identifies the type of item being published
UseLocal Connection	Boolean	Set/Get if the LocalConnection property is used to set the data source. False means the Connection property is used. Allows you to store some data sources offline

PivotCache Methods

Name	Returns	Parameters	Description
CreatePivot Table	PivotTable	Table Destination As Variant, [TableName], [ReadData], [Default Version]	Creates a PivotTable report that is based on the current PivotCache object. The TableDestination parameter specifies where the new PivotTable report will be located. A TableName can also be specified. Set ReadData to True to fill the cache with all the records from the external database. Set ReadData to True to only retrieve some of the data. DefaultVersion is the default version of the PivotTable report
Make Connection			Makes a connection for the specified PivotCache
Refresh			Refreshes the data in the PivotTable cache with the latest copy of the external data. Set the BackgroundQuery parameter to True to get the data to refresh asynchronously
ResetTimer			Resets the time for the automatic refresh set by RefreshPeriod property
SaveAsODC		ODCFileName As String, [Description], [Keywords]	Saves the PivotCache source as an Office Data Connection (ODC) file. ODCFileName is the location where the file is to be saved. Description is the description that will be saved in the file. Keywords is a list of space-separated keywords that can be used to search for this file

Example: PivotCache Object and the PivotCaches Collection

```
Sub RefreshPivotCache()
   Dim oPC As PivotCache
   Set oPC = ActiveWorkbook.PivotCaches(1)
   With oPC
      'Refresh in the foreground
      .BackgroundQuery = False

      'Only refresh if the data is over 1 hour old
      If .RefreshDate < Now - TimeValue("01:00:00") Then
         .Refresh
      End If
   End With
End Sub
```

PivotCell Object

Represents a cell somewhere inside a PivotTable. Use access the PivotCell object through the range object. Once obtained, you can use the various properties of the PivotCell object to retrieve data from a PivotTable. For example, you can use the PivotCellType, ColumnItems, and RowItems properties to locate a particular sales person's total sales for a specific region.

This object mirrors the functionality of the GETPIVOTDATA worksheet function and the GetPivotData method of the PivotTable object. The difference is the PivotCell object can render information about where the cell is in the report. The GETPIVOTDATA worksheet function and the GetPivotData method do just the opposite. They yield the value associated with row and column heading you provide.

PivotCell Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotCell Properties

Name	Returns	Description
ColumnItems	PivotItemList	Read-only. Returns the items which represent the selected range on the column axis
Custom Subtotal Function	Xl Consolidation Function	Read-only. Returns the custom subtotal function field setting of the PivotCell
DataField	PivotField	Read-only. Returns the selected data field
PivotCellType	XlPivotCell Type	Read-only. Returns the PivotTable entity that the selected cell corresponds to
PivotField	PivotField	Read-only. Returns the PivotTable field containing the upper-left corner of the specified range

Continues

Name	Returns	Description
PivotItem	PivotItem	Read-only. Returns the PivotTable item containing the upper-left corner of the specified range
PivotTable	PivotTable	Read-only. Returns the PivotTable report containing the upper-left corner of the specified range, or the PivotTable report associated with the PivotChart report
Range	Range	Read-only. Returns the range to which the specified PivotCell applies
RowItems	PivotItemList	Read-only. Returns the items which represent the selected range on the row axis

PivotField Object, PivotFields Collection and the CalculatedFields Collection

The PivotFields collection holds the collection of fields associated with the parent PivotTable report. The CalculatedFields collection holds the collection of calculated fields associated with the parent PivotTable report. Each PivotField object represents single field in a PivotTable report. The parent of the PivotFields and CalculatedFields collection is the PivotTable object.

The PivotFields and CalculatedFields collections have one extra property besides the typical collection attributes. The Count property returns the number of fields in the parent collection. The CalculatedFields collection also has an Add method that adds a new calculated field to the collection given a Name and a Formula.

PivotField Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotField Properties

Name	Returns	Description
AutoShowCount	Long	Read-only. Returns the number of top or bottom items that are automatically displayed in the PivotTable field
AutoShowField	String	Read-only. Returns the name of the data field used to figure out what top or bottom items to show automatically for a PivotTable field
AutoShowRange	Long	Read-only. Returns either xlTop if the top items are shown automatically or xlBottom if the bottom items are shown

Name	Returns	Description
AutoShowType	Long	Read-only. Returns either xlAutomatic if AutoShow is True or xlManual if AutoShow is disabled
AutoSortField	String	Read-only. Returns the data field name that will be used to sort the PivotTable field automatically
AutoSortOrder	Long	Read-only. Returns one of the XLSortOrder constants specifying the automatic sort order type used for the field
BaseField	Variant	Set/Get the base field used for a calculation. Data fields only
BaseItem	Variant	Set/Get the base item in the base field used for a calculation. Data fields only
Calculation	XlPivotField Calculation	Set/Get the type of calculation to do on the data field
Caption	String	Set/Get the text label to use for the field
ChildField	PivotField	Read-only. Returns the child field of the current field, if any
ChildItems	Variant	Read-only. Parameters: [Index]. Returns an object or collection containing a single PivotTable item (PivotItem) or group of PivotTable items (PivotItems) associated with the field
CubeField	CubeField	Read-only. Returns the cube field that the current PivotTable field comes from.
CurrentPage	Variant	Set/Get the current page showing for the page field. Page fields only
Current PageList	Variant	Set or Get an array of strings corresponding to the list of items included in a multiple-item page field of a PivotTable report
CurrentPageName	String	Set/Get the displayed page of the PivotTable report
DataRange	Range	Read-only. Returns a range containing the data or items in the field
DataType	XlPivotField DataType	Read-only. Returns the data type of the PivotTable field
DatabaseSort	Boolean	Set/Get whether manual repositioning of items in a PivotField is allowed. Returns True, if the field has no manually positioned items

Continues

Name	Returns	Description
DragToColumn	Boolean	Set/Get whether the field can be dragged to a column position
DragToData	Boolean	Set/Get whether the field can be dragged to the data position
DragToHide	Boolean	Set/Get whether the field can be dragged off the PivotTable report and therefore hidden
DragToPage	Boolean	Set/Get whether the field can be dragged to the page position
DragToRow	Boolean	Set/Get whether the field can be dragged to a row position
DrilledDown	Boolean	Set/Get whether the PivotTable field can be drilled down
EnableItem Selection	Boolean	Set/Get whether the ability to use the field dropdown in the user interface is enabled
Formula	String	Set/Get the formula associated with the field, if any
Function	Xl Consolidation Function	Set/Get the type of function used to summarize the PivotTable field
GroupLevel	Variant	Read-only. Returns how the field is placed within a group of fields
HiddenItems	Variant	Read-only. Parameters: [Index]. Returns an object or collection containing a single hidden PivotTable item (PivotItem) or group of hidden PivotTable items (PivotItems) associated with the field
HiddenItems List	Variant	Set/Get an array of strings that are hidden items for the PivotField
IsCalculated	Boolean	Read-only. Returns whether the PivotTable field is calculated
IsMember Property	Boolean	Read-only. Returns whether the PivotField contains member properties
LabelRange	Range	Read-only. Returns the cell range containing the field's label
LayoutBlank Line	Boolean	Set/Get whether a blank row is added just after the current row field
LayoutForm	XlLayoutForm Type	Set/Get how the items will appear in the field

Name	Returns	Description
`LayoutPageBreak`	`Boolean`	Set/Get whether a page break is inserted after each field
`LayoutSubtotalLocation`	`XlSubtotalLocationType`	Set/Get the location for the field subtotals as compared to the current field
`MemoryUsed`	`Long`	Read-only. Returns the number of bytes of computer memory being used for the field
`Name`	`String`	Set/Get the name of the field
`NumberFormat`	`String`	Set/Get the format used for numbers in the field
`Orientation`	`XlPivotFieldOrientation`	Set/Get where the field is located in the `PivotTable` report
`ParentField`	`PivotField`	Read-only. Returns the parent field of the current field, if any
`ParentItems`	`Variant`	Read-only. Parameters: `[Index]`. Returns an object or collection containing a single parent `PivotTable` item (`PivotItem`) or group of parent `PivotTable` items (`PivotItems`) associated with the field
`Position`	`Variant`	Set/Get the position number of the field among all the fields in the same orientation
`PropertyOrder`	`Long`	Set/Get the display position of the member property within the cube field to which it belongs (setting will rearrange the order). Valid only for `PivotField` objects that are member property fields
`PropertyParentField`	`PivotField`	Read-only. Returns the field to which the properties in this field are linked
`ServerBased`	`Boolean`	Set/Get whether only items that match the page field selection are retrieved from the external data source
`ShowAllItems`	`Boolean`	Set/Get whether all items in the field are displayed
`SourceName`	`String`	Read-only. Returns the name of the source data for the field
`StandardFormula`	`String`	Set/Get the formulas with standard US formatting
`SubtotalName`	`String`	Set/Get the label used for the subtotal column or row for this field

Continues

Name	Returns	Description
Subtotals	Variant	Parameters: [Index]. Set/Get the subtotals displayed for the field
TotalLevels	Variant	Read-only. Returns the total number of fields in the current field group
Value	String	Set/Get the name of the field
VisibleItems	Variant	Read-only. Parameters: [Index]. Returns an object or collection containing a single visible PivotTable item (PivotItem) or group of visible PivotTable items (PivotItems) associated with the field

PivotField Methods

Name	Returns	Parameters	Description
AddPageItem		Item As String, [ClearList]	Adds an additional item to a multiple item page field. Item is the source name of a PivotItem object, corresponding to the specific OLAP member unique name. ClearList indicates whether to delete all existing items before adding the new item
AutoShow		Type As Long, Range As Long, Count As Long, Field As String	Set the number of top or bottom items to display for a row, page, or column field. Type describes whether the items are shown as xlAutomatic or xlManual. Range is the location to start showing items. Count is the number of items to show for the field. Field is the base data field name
AutoSort		Order As Long, Field As String	Sets the field to automatically sort based on the Order specified (using XlSortOrder constants) and the base data Field
Calculated Items	Calculated Items		Returns the group of calculated PivotTable items associated with the field
Delete			Deletes the PivotField object

Name	Returns	Parameters	Description
`PivotItems`	`Variant`	`[Index]`	Returns an object or collection containing a single `PivotTable` item (`PivotItem`) or group of `PivotTable` items (`PivotItems`) associated with the field

Example: PivotField Object, PivotFields Collection and the CalculatedFields

```
Sub AddField()
   Dim oPT As PivotTable
   Dim oPF As PivotField
   Set oPT = ActiveSheet.PivotTables(1)
  'Set the UseStandardFormula argument to true
  'This will format the field names in the formula for
  ' Standard U.S.English instead of using Local Settings
  'Note that running/debugging this workbook in versions of Excel
  ' prior to Excel 2003 will result in a "Wrong number of arguments" error.
   Set oPF = oPT.CalculatedFields.Add("Total", "=Price * Volume",  True)
   oPF.Orientation = xlDataField
End Sub
```

PivotFormula Object and the PivotFormulas Collection

The `PivotFormulas` collection holds the formulas associated with the `PivotTable`. Each `PivotFormula` object represents a formula being used in a `PivotTable` report. The parent of the `PivotFormulas` collection is the `PivotTable` object.

The `PivotFormulas` collection has a `Count` property and an `Add` method besides the typical collection attributes. The `Count` property returns the number of items in the collection. The `Add` method takes a `Formula` string and adds a `PivotFormula` to the collection.

PivotFormula Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

PivotFormula Properties

Name	Returns	Description
`Formula`	`String`	Set/Get the formula associated with the table. Use the A1-style reference notation
`Index`	`Long`	Read-only. Returns the order that the formulas in the parent collection will be processed
`Standard Formula`	`String`	Set/Get the formulas with standard US formatting
`Value`	`String`	Set/Get the formula associated with the table

PivotFormula Methods

Name	Returns	Parameters	Description
Delete			Deletes the formula from the parent collection

PivotItem Object, PivotItems Collection, and the CalculatedItems Collection

The PivotItems collection holds the collection of individual data entries in a field. The CalculatedItems collection holds the collection of individual calculated entries in a field. Each PivotItem object represents a single entry in a data field. The parent of the PivotItems and CalculatedItems collections is the PivotField object.

The PivotItems and CalculatedItems have one extra property besides the typical collection attributes. The Count property returns the number of objects in the collection. Also, the Add method of the PivotItems collection adds another item to the collection (only a Name is required). The Add method of the CalculatedItems collection adds another item to the collection but requires a Name and a Formula to be specified.

PivotItem Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotItem Properties

Name	Returns	Description
Caption	String	Set/Get the label text associated with the item
ChildItems	Variant	Read-only. Parameters: [Index]. Returns an object or collection containing a single PivotTable item (PivotItem) or group of PivotTable items (PivotItems) associated with the item
DataRange	Range	Read-only. Returns a range containing the data or items in the item
DrilledDown	Boolean	Set/Get whether the PivotTable item can be drilled down
Formula	String	Set/Get the formula associated with item, if any
IsCalculated	Boolean	Read-only. Returns whether the item that was calculated is a data item
LabelRange	Range	Read-only. Returns the cell range containing the field's item

Name	Returns	Description
Name	String	Set/Get the name of the item
ParentItem	PivotItem	Read-only. Returns the parent item of the current item, if any
ParentShowDetail	Boolean	Read-only. Returns whether the current item is being shown because the one of the item's parents is set to show detail
Position	Long	Set/Get the position number of the item among all the items in the same orientation
RecordCount	Long	Read-only. Returns the number of records in the PivotTable cache that contain the item
ShowDetail	Boolean	Set/Get whether the detail items are being displayed
SourceName	Variant	Read-only. Returns the name of the source data for the item
SourceName Standard	String	Read-only. Returns the PivotItem's source name in standard US format settings
StandardFormula	String	Set/Get the formulas with standard US formatting
Value	String	Set/Get the name of the specified item
Visible	Boolean	Set/Get whether the item is visible

PivotItem Methods

Name	Returns	Parameters	Description
Delete			Deletes the item from the collection

Example: PivotItem Object, PivotItems Collection, and the CalculatedItems Collection

```
Sub ShowPivotItemData()
   Dim oPT As PivotTable
   Dim oPI As PivotItem
   'Get the pivot table
   Set oPT = ActiveSheet.PivotTables(1)
   'Get the pivot item
   Set oPI = oPT.PivotFields("Product").PivotItems("Oranges")
   'Show all the source data rows for that pivot item
   oPI.ShowDetail = True
End Sub
```

PivotItemList Object

Represents a list of PivotItems associated with a particular cell in a PivotTable. You access the list through the PivotCell object. PivotItemLists are accessed either through the ColumnItems or RowItems properties of the PivotCell object. How many row and column items in the PivotItemList depends on the structure of the PivotTable.

For example, cell D5 is in a PivotTable called WroxSales1. In the row area to the left of cell D5 is the row heading OR (Oregon). To the left of OR is another row label called Region1. Based on this information the following will yield 2:

```
MsgBox wksPivotTable.Range("D5").PivotCell.RowItems.Count
```

The following will yield Region1, the farthest label to the left of cell D5:

```
MsgBox wksPivotTable.Range("D5").PivotCell.RowItems(1)
```

Finally, the following will yield OR, the second farthest label to the left of cell D5:

```
MsgBox wksPivotTable.Range("D5").PivotCell.RowItems(2)
```

We have yet to find a use for both the PivotItemList and PivotCell objects. Normally, we are looking for the opposite. We want to retrieve information based on row or column items (headings) we provide, something the GetPivotData method and the GETPIVOTDATA worksheet function can obtain.

PivotItemList Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this appendix.

PivotItemList Properties

Name	Returns	Description
Count	Long	Returns the number of objects in the collection

PivotItemList Methods

Name	Returns	Parameters	Description
Item	PivotItem	Index As Variant	Returns a single PivotItem from the PivotItemList

PivotLayout Object

The PivotLayout object describes how the fields of a PivotChart are placed in the parent chart. Either the Chart object or the ChartGroup object is the parent of the PivotChart object.

PivotLayout Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotLayout Properties

Name	Returns	Description
ColumnFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the columns of the PivotChart
CubeFields	CubeFields	Read-only. Returns the collection of cube fields associated with the PivotChart
DataFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the data fields of the PivotChart
HiddenFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the hidden fields of the PivotChart
InnerDetail	String	Set/Get the name of the field that will show the detail when the ShowDetail property is True
PageFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the page fields of the PivotChart
PivotCache	PivotCache	Read-only. Returns the PivotChart's data cache
PivotFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the fields of the PivotChart
PivotTable	PivotTable	Read-only. Returns the PivotTable associated with the PivotChart
RowFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the rows of the PivotChart
Visible Fields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the visible fields of the PivotChart

PivotLayout Methods

Name	Returns	Parameters	Description
AddFields		[RowFields], [Column Fields], [PageFields], [AppendField]	Adds row, column, and page fields to a PivotChart report RowFields, ColumnFields, and PageFields can hold a single string field name or an array of string field names. Set AppendField to True to add the fields to the chart. Set AppendField to False to replace the fields in the chart

Example: PivotLayout Object

```
Sub SetPivotLayout()
   Dim oPL As PivotLayout
   'Get the pivot layout
   Set oPL = Charts(1).PivotLayout
   'Show sales of Oranges by region
   With oPL
      .AddFields RowFields:="Region", PageFields:="Product"
      .PageFields("Product").CurrentPage = "Oranges"
   End With
End Sub
```

PivotTable Object and the PivotTables Collection

The PivotTables collection contains the collection of PivotTables in the parent worksheet. Each PivotTable object in the collection allows manipulation and creation of Excel PivotTables. The parent of the PivotTables collection is the Worksheet object.

The PivotTables collection has a Count property and an Add method besides the typical collection attributes. The Count property returns the number of PivotTable objects in the collection. The Add method takes a new PivotTable cache (containing the data) and the destination single cell range determining the upper-left corner of the PivotTable report to create a new PivotTable report. The name of the new PivotTable report can also be specified in the Add method.

PivotTable Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PivotTable Properties

Name	Returns	Description
CacheIndex	Long	Set/Get the index number pointing to the PivotTable cache of the current PivotTable
Calculated Members	Calculated Members	Read-only. Returns all the calculated fields and calculated items for the PivotTable

Name	Returns	Description
ColumnFields	Object	Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the columns of the PivotTable
ColumnGrand	Boolean	Set/Get whether grand totals are shown for columns in the PivotTable
ColumnRange	Range	Read-only. Returns the range of cells containing the column area in the PivotTable report
CubeFields	CubeFields	Read-only. Returns the collection of cube fields associated with the PivotTable report
DataBody Range	Range	Read-only. Returns the range of cells containing the data area of the PivotTable report
DataFields	Object	Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the data fields of the PivotTable
DataLabel Range	Range	Read-only. Returns the range of cells that contain the labels for the data fields in the PivotTable report
DataPivot Field	PivotField	Read-only. Returns all the data fields in a PivotTable
DisplayEmpty Column	Boolean	Read-only. Returns whether the non-empty MDX keyword is included in the query to the OLAP provider for the value axis
DisplayEmptyRow	Boolean	Read-only. Returns whether the non-empty MDX keyword is included in the query to the OLAP provider for the category axis
DisplayError String	Boolean	Set/Get whether the string in the ErrorString property is displayed in cells that contain errors
Display Immediate Items	Boolean	Set/Get whether items in the row and column areas are visible when the data area of the PivotTable is empty
DisplayNull String	Boolean	Set/Get whether the string in the NullString property is displayed in cells that contain null values
EnableData ValueEditing	Boolean	Set/Get whether to show an alert when the user overwrites values in the data area of the PivotTable

Continues

Name	Returns	Description
`Enable Drilldown`	`Boolean`	Set/Get whether drilldown in the `PivotTable` report is enabled
`EnableField Dialog`	`Boolean`	Set/Get whether the `PivotTable` Field dialog box is displayed when the user double-clicks a `PivotTable` field
`EnableField List`	`Boolean`	Set/Get whether to disable the ability to display the field well for the `PivotTable`. If the list was already visible, it disappears
`EnableWizard`	`Boolean`	Set/Get whether the `PivotTable` Wizard is available
`ErrorString`	`String`	Set/Get the string that is displayed in cells that contain errors. Use with the `DisplayErrorString` property
`GrandTotal Name`	`String`	Set/Get the string label that will be displayed on the grand total column or row heading of a `PivotTable` report. Default is "`Grand Total`"
`HasAuto Format`	`Boolean`	Set/Get whether the `PivotTable` report is automatically reformatted when the data is refreshed or the fields are moved around
`HiddenFields`	`Object`	Read-only. Parameters: `[Index]`. Returns an object or collection containing the `PivotTable` field (`PivotField`) or `PivotTable` fields (`PivotFields`) associated with the hidden fields of the `PivotTable`
`InnerDetail`	`String`	Set/Get the name of the field that will show the detail when the `ShowDetail` property is `True`
`MDX`	`String`	Read-only. Returns the MDX (Multidimensional Expression) that would be sent to the provider to populate the current `PivotTable` view
`ManualUpdate`	`Boolean`	Set/Get whether the `PivotTable` report is only recalculated manually
`MergeLabels`	`Boolean`	Set/Get whether the outer-row item, column item, subtotal, and grand total labels of a `PivotTable` report have merged cells
`Name`	`String`	Set/Get the name of the `PivotTable` report
`NullString`	`String`	Set/Get the string that is displayed in cells that contain `null` strings. Use with the `DisplayNullString` property
`PageField Order`	`Long`	Set/Get how new page fields are added to a `PivotTable` report's layout. Use the `XLOrder` constants

Name	Returns	Description
PageFields	Object	Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the page fields of the PivotTable
PageField Style	String	Set/Get the style used for a page field area in a PivotTable
PageField WrapCount	Long	Set/Get how many page fields are in each column or row of the PivotTable report
PageRange	Range	Read-only. Returns the range containing the page area in the PivotTable report
PageRange Cells	Range	Read-only. Returns the range containing the page fields and item drop-down lists in the PivotTable report
Pivot Formulas	Pivot Formulas	Read-only. Returns the collection of formulas used in the PivotTable report
Pivot Selection	String	Set/Get the data and label selection in the PivotTable using the standard PivotTable report selection format. For example, to select the data and label for the Country equal to "Canada" then the string would be "Country[Canada]"
Pivot Selection Standard	String	Set/Get the PivotTable selection in standard PivotTable report format using US settings
Preserve Formatting	Boolean	Set/Get whether formatting of the PivotTable report is preserved when the report is changed, sorted, pivoted, refreshed or recalculated
PrintTitles	Boolean	Set/Get whether the print title set on the PivotTable report is printed whenever the parent worksheet is printed
RefreshDate	Date	Read-only. Returns the date that the PivotTable report data was refreshed last
RefreshName	String	Read-only. Returns the name of the user who last refreshed the PivotTable report data
RepeatItems OnEachPrinted Page	Boolean	Set/Get whether row, column, and item labels appear on the first row of each page when the PivotTable report is printed
RowFields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the rows of the PivotTable

Continues

Name	Returns	Description
`RowGrand`	`Boolean`	Set/Get whether grand totals are shown for rows in the `PivotTable`
`RowRange`	`Range`	Read-only. Returns the range of cells containing the row area in the `PivotTable` report
`SaveData`	`Boolean`	Set/Get whether the `PivotTable` report data is saved with the workbook
`Selection Mode`	`XlPT Selection Mode`	Set/Get how the `PivotTable` report selection mode is set (for example, `xlLabelOnly`)
`ShowCell Background FromOLAP`	`Boolean`	Set/Get whether the MDX that Excel asks for includes the `BackColor` property for each cell in the data area that corresponds to a cell in the OLAP data set
`ShowPage MultipleItem Label`	`Boolean`	Set/Get whether "(Multiple Items)" will appear in the `PivotTable` cell whenever items are hidden and an aggregate of non-hidden items is shown in the `PivotTable` view
`SmallGrid`	`Boolean`	Set/Get whether a two-by-two grid is used for a newly created `PivotTable` report (`True`) or a blank stencil outline (`False`)
`SourceData`	`Variant`	Set/Get the source of the `PivotTable` report data. Can be a cell reference, an array, multiple ranges, and another `PivotTable` report. Not valid to use with OLE DB data sources
`Subtotal HiddenPage Items`	`Boolean`	Set/Get whether hidden page fields are included in row and column subtotals, block totals, and grand totals
`TableRange1`	`Range`	Read-only. Returns the range containing the whole `PivotTable` report, not including page fields
`TableRange2`	`Range`	Read-only. Returns the range containing the whole `PivotTable` report, with page fields
`TableStyle`	`String`	Set/Get the `PivotTable` report body style
`Tag`	`String`	Set/Get a string to be saved with the `PivotTable` report (for example, a description of the `PivotTable` report)
`VacatedStyle`	`String`	Set/Get the style to use for vacated cells when a `PivotTable` report is refreshed
`Value`	`String`	Set/Get the name of the `PivotTable` report

Name	Returns	Description
Version	XlPivot Table VersionList	Read-only. Returns the version number of Excel
View Calculated Members	Boolean	Set/Get whether calculated members for OLAP PivotTables can be viewed
Visible Fields		Read-only. Parameters: [Index]. Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the visible fields of the PivotTable
VisualTotals	Boolean	Set/Get whether PivotTables should retotal after an item has been hidden from view

PivotTable Methods

Name	Returns	Parameters	Description
AddDataField	PivotField	Field As Object, [Caption], [Function]	Adds a data field to a PivotTable report. Field is the unique field on the server, Caption is the label used to identify this data field, and Function is the function performed in the added data field
AddFields	Variant	[RowFields], [Column Fields], [PageFields], [AddToTable]	Adds row, column, and page fields to a PivotTable report. The RowFields, ColumnFields, and PageFields can hold a single string field name or an array of string field names. Set AddToTable to True to add the fields to the report. Set AddToTable to False to replace the fields in the report
Calculated Fields	Calculated Fields		Returns the collection of calculated fields in the PivotTable Report

Continues

Name	Returns	Parameters	Description
CreateCube File	String	File As String, [Measures], [Levels], [Members], [Properties]	Creates a cube file from a PivotTable report connected to an OLAP data source. File is the name of the cube file to be created, Measures is an array of unique names of measures that are to be part of the slice, and Levels an array of strings, where each array item is a unique level name. Members is an array of string arrays, where the elements correspond, in order, to the hierarchies represented in the Levels array. Properties should be set to False if you don't want member properties being included in the slice
Format		Format As xlPivotFormat Type	Set the PivotTable report format to the predefined style specified in the Format parameter
GetData	Double	Name As String	Get the value of a specific cell in the PivotTable report. The Name parameter must be in the standard PivotTable report selection format
GetPivotData	Range	[DataField], [Field1],[Item1], [Field2],[Item2], [Field3],[Item3], [Field4],[Item4], [Field5],[Item5], [Field6],[Item6], [Field7],[Item7], [Field8],[Item8], [Field9],[Item9], [Field10], [Item10], [Field11], [Item11], [Field12], [Item12], [Field13], [Item13], [Field14], [Item14]	Returns information about a data item in a PivotTable report. FieldN is the name of a column or row field in the PivotTable report, and ItemN is the name of an item in FieldN
ListFormulas			Creates a separate worksheet with the list of all the calculated PivotTable items and fields

Name	Returns	Parameters	Description
PivotCache	PivotCache		Returns a data cache associated with the current PivotTable
PivotFields	Object	[Index]	Returns an object or collection containing the PivotTable field (PivotField) or PivotTable fields (PivotFields) associated with the fields of the PivotTable
PivotSelect		Name As String, [Mode As XlPTSelectionMode], [UseStandard Name]	Selects the part of the PivotTable specified by Name parameter in the standard PivotTable report selection format. Mode decides which part of the PivotTable to select (for example, xlBlanks). Set UseStandardName to True for recorded macros that will play back in other locales
PivotTableWizard		[SourceType], [SourceData], [Table Destination], [TableName], [RowGrand], [ColumnGrand], [SaveData], [HasAuto Format], [AutoPage], [Reserved], [Background Query], [Optimize Cache], [PageField Order], [PageField WrapCount], [ReadData], [Connection]	Creates a PivotTable report. The SourceType uses the XLPivotTableSourceType constants to specify the type of SourceData being used for the PivotTable. The TableDestination holds the range in the parent worksheet that the report will be placed. TableName holds the name of the new report. Set RowGrand or ColumnGrand to True to show grand totals for rows and columns, respectively. Set HasAutoFormat to True for Excel to format the report automatically when it is refreshed or changed. Use the AutoPage parameter to set if a page field is created for consolidation automatically. Set BackgroundQuery to True for Excel to query the data source asynchronously.

Continues

Name	Returns	Parameters	Description
			Set `OptimizeCache` to `True` for Excel to optimize the cache when it is built. Use the `PageFieldOrder` with the `XLOrder` constants to set how new page fields are added to the report. Use the `PageFieldWrapCount` to set the number of page fields in each column or row. Set `ReadData` to `True` to copy the data from the external database into a cache. Finally, use the `Connection` parameter to specify an ODBC connection string for the `PivotTable`'s cache
`RefreshTable`	`Boolean`		Refreshes the `PivotTable` report from the source data and returns `True`, if successful
`ShowPages`	`Variant`	`[PageField]`	Creates a new `PivotTable` report for each item in the page field (`PageField`) in a new worksheet
`Update`			Updates the `PivotTable` report

Example: PivotTable Object and the PivotTables Collection

```
Sub PreviewPivotTable()
   Dim oPT As PivotTable
   'Get the pivot layout
   Set oPT = ActiveSheet.PivotTables(1)
   'Add column and row titles, then printpreview the table
   With oPT
      .ColumnGrand = False
      .RowGrand = True
      .TableRange2.PrintPreview
   End With
End Sub
```

PlotArea Object

The `PlotArea` object contains the formatting options associated with the plot area of the parent chart. For 2D charts the `PlotArea` includes trendlines, data markers, gridlines, data labels, and the axis labels—but not titles. For 3D charts the `PlotArea` includes the walls, floor, axes, axis titles, tick-marks, and all of the items mentioned for the 2D charts. The area surrounding the plot area is the chart area. Please see the `ChartArea` object for formatting related to the chart area. The parent of the `PlotArea` is always the `Chart` object.

PlotArea Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

PlotArea Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the plot area
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for a chart's plot area
`Height`	`Double`	Set/Get the height of the chart plot area
`InsideHeight`	`Double`	Read-only. Returns the height inside the plot area that does not include the axis labels
`InsideLeft`	`Double`	Read-only. Returns the distance from the left edge of the plot area, not including axis labels, to the chart's left edge
`InsideTop`	`Double`	Read-only. Returns the distance from the left edge of the plot area, not including axis labels, to the chart's left edge
`InsideWidth`	`Double`	Read-only. Returns the width inside the plot area that does not include the axis labels
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of the plot area (for example, interior color)
`Left`	`Double`	Set/Get the distance from the left edge of the plot area to the chart's left edge
`Name`	`String`	Read-only. Returns the name of the plot area
`Top`	`Double`	Set/Get the distance from the top edge of the plot area to the chart's top edge
`Width`	`Double`	Set/Get the width of the chart plot area

PlotArea Methods

Name	Returns	Parameters	Description
`ClearFormats`	`Variant`		Clears any formatting made to the plot area
`Select`	`Variant`		Selects the plot area on the chart

Example: PlotArea Object

This example uses the `PlotArea` object to make all the charts in the workbook have the same size and position plot area, regardless of the formatting of the axes (for example, different fonts and number scales):

```
Sub MakeChartAreasSameSizeAsFirst()
   Dim oCht As Chart, oPA As PlotArea
   Dim dWidth As Double, dHeight As Double
   Dim dTop As Double, dLeft As Double
   'Get the dimensions of the inside of the
   'plot area of the first chart
   With Charts(1).PlotArea
      dWidth = .InsideWidth
      dHeight = .InsideHeight
      dLeft = .InsideLeft
      dTop = .InsideTop
   End With
   'Loop through the charts in the workbook
   For Each oCht In Charts
      'Get the PlotArea
      Set oPA = oCht.PlotArea

      'Size and move the plot area
      With oPA
         If .InsideWidth > dWidth Then
            'Too big, make it smaller
            .Width = .Width - (.InsideWidth - dWidth)
         Else
            'Too small, move it left and make bigger
            .Left = .Left - (dWidth - .InsideWidth)
            .Width = .Width + (dWidth - .InsideWidth)
         End If

         If .InsideHeight > dHeight Then
            'Too big, make it smaller
            .Height = .Height - (.InsideHeight - dHeight)
         Else
            'Too small, move it left and make bigger
            .Top = .Top - (dHeight - .InsideHeight)
            .Height = .Height + (dHeight - .InsideHeight)
         End If
         'Set the position of the inside of the plot area
         .Left = .Left + (dLeft - .InsideLeft)
         .Top = .Top + (dTop - .InsideTop)
      End With
   Next
End Sub
```

Point Object and the Points Collection

The `Points` collection holds all of the data points of a particular series of a chart. In fact, a chart (`Chart` object) can have many chart groups (`ChartGroups / ChartGroup`) that can contain many series (`SeriesCollection / Series`), which, in turn, can contain many points (`Points / Point`). A `Point` object describes the particular point of a series on a chart. The parent of the Points collection is the Series object.

The Points collection contains a Count property besides the typical collection attributes. The Count property returns the number of Point objects in the collection.

Point Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Point Properties

Name	Returns	Description
ApplyPictToEnd	Boolean	Set/Get whether pictures are added to the end of the point
ApplyPictToFront	Boolean	Set/Get whether pictures are added to the front of the point
ApplyPictToSides	Boolean	Set/Get whether pictures are added to the sides of the point
Border	Border	Read-only. Returns the border's properties around the point
DataLabel	DataLabel	Read-only. Returns an object allowing you to manipulate the data label attributes (for example, formatting, text). Use with HasDataLabel
Explosion	Long	Set/Get how far out a slice (point) of a pie or doughnut chart will explode out. 0 for no explosion
Fill	ChartFillFormat	Read-only. Returns an object containing fill formatting options for a point
HasDataLabel	Boolean	Set/Get whether the point has a data label. Use with DataLabel
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the point (for example, interior color)
InvertIfNegative	Boolean	Set/Get whether the point's color will be inverted if the point value is negative
MarkerBackgroundColor	Long	Set/Get the color of the point marker background Use the RGB function to create the color value
MarkerBackgroundColorIndex	XlColorIndex	Set/Get the color of the point marker background. Use the XlColorIndex constants or an index value in the current color palette

Continues

Name	Returns	Description
`MarkerForegroundColor`	`Long`	Set/Get the color of the point marker foreground. Use the `RGB` function to create the color value
`MarkerForegroundColorIndex`	`XlColorIndex`	Set/Get the color of the point marker foreground. Use the `XlColorIndex` constants or an index value in the current color palette
`MarkerSize`	`Long`	Set/Get the size of the point key marker
`MarkerStyle`	`XlMarkerStyle`	Set/Get the type of marker to use as the point key (for example, square, diamond, triangle, picture, etc.)
`PictureType`	`XlChartPictureType`	Set/Get how an associated picture is displayed on the point (for example, stretched, tiled). Use the `XlPictureType` constants
`PictureUnit`	`Long`	Set/Get how many units a picture represents if the `PictureType` property is set to `xlScale`
`SecondaryPlot`	`Boolean`	Set/Get if the point is on the secondary part of a Pie of Pie chart of a Bar of Pie chart
`Shadow`	`Boolean`	Set/Get whether the point has a shadow effect

Point Methods

Name	Returns	Parameters	Description
`ApplyDataLabels`	`Variant`	`[Type As XlDataLabelsType], [LegendKey], [AutoText], [HasLeaderLines], [ShowSeriesName], [ShowCategoryName], [ShowValue], [ShowPercentage], [ShowBubbleSize], [Separator]`	Applies the data label properties specified by the parameters to the point. The `Type` parameter specifies whether no label, a value, a percentage of the whole, or a category label is shown. The legend key can appear by the point by setting the `LegendKey` parameter to `True`. Set `AutoText` to `True` if the object automatically generates appropriate text based on content. `HasLeaderLines` should be set to `True` if the series has leader lines. All the other parameters are simply the property of the data label that they describe
`ClearFormats`	`Variant`		Clears the formatting made to a point

Name	Returns	Parameters	Description
`Copy`	`Variant`		Cuts the point and places it in the clipboard
`Delete`	`Variant`		Deletes the point
`Paste`	`Variant`		Pastes the picture in the clipboard into the current point so it becomes the marker
`Select`	`Variant`		Selects the point on the chart

Example: Point Object and the Points Collection

```
Sub ExplodePie()
   Dim oPt As Point
   'Get the first data point in the pie chart
   Set oPt = Charts(1).SeriesCollection(1).Points(1)
   'Add a label to the first point only and
   'set it away from the pie
   With oPt
      .ApplyDataLabels xlDataLabelsShowLabelAndPercent
      .Explosion = 20
   End With
End Sub
```

Protection Object

Represents the group of the much-needed sheet protection options new to Excel 2003. When you protect a sheet, you now have the option to only allow unlocked cells selected, allow cell, column, and row formatting, allow insertion and deletion of rows and columns, allow sorting, and more.

Setting Protection options is done via the `Protect` method of the `Worksheet` object. Use the `Protection` property of the `Worksheet` object to check the current protection settings:

```
MsgBox ActiveSheet.Protection.AllowFormattingCells
```

Protection Properties

Name	Returns	Description
`AllowDeleting Columns`	`Boolean`	Read-only. Returns whether the deletion of columns is allowed on a protected worksheet
`AllowDeleting Rows`	`Boolean`	Read-only. Returns whether the deletion of rows is allowed on a protected worksheet
`AllowEdit Ranges`	`AllowEdit Ranges`	Read-only. Returns an `AllowEditRanges` object

Continues

Name	Returns	Description
`AllowFiltering`	`Boolean`	Read-only. Returns whether the user is allowed to make use of an `AutoFilter` that was created before the sheet was protected
`AllowFormattingCells`	`Boolean`	Read-only. Returns whether the formatting of cells is allowed on a protected worksheet
`AllowFormattingColumns`	`Boolean`	Read-only. Returns whether the formatting of columns is allowed on a protected worksheet
`AllowFormattingRows`	`Boolean`	Read-only. Returns whether the formatting of rows is allowed on a protected worksheet
`AllowInsertingColumns`	`Boolean`	Read-only. Returns whether the inserting of columns is allowed on a protected worksheet
`AllowInsertingHyperlinks`	`Boolean`	Read-only. Returns whether the inserting of hyperlinks is allowed on a protected worksheet
`AllowInsertingRows`	`Boolean`	Read-only. Returns whether the inserting of rows is allowed on a protected worksheet
`AllowSorting`	`Boolean`	Read-only. Returns whether the sorting option is allowed on a protected worksheet
`AllowUsingPivotTables`	`Boolean`	Read-only. Returns whether the manipulation of `PivotTables` is allowed on a protected worksheet

Example: Protection Object

The following routine sets `Protection` options based on the user name found on the General tab of the Tools ➪ Options command and that user's settings on a table on the worksheet. If the user isn't found, a message appears and the default settings are used:

```
Sub ProtectionSettings()
    Dim rngUsers As Range, rngUser As Range
    Dim sCurrentUser As String

    'Grab the current username
    sCurrentUser = Application.UserName

    'Define the list of users in the table
    With wksAllowEditRange
        Set rngUsers = .Range(.Range("Users"), .Range("Users").End(xlToRight))
    End With

    'Locate the current user on the table
    Application.FindFormat.Clear
    Set rngUser = rngUsers.Find(What:=sCurrentUser, SearchOrder:=xlByRows,
MatchCase:=False, SearchFormat:=False)

    'If current user is found on the table...
    If Not rngUser Is Nothing Then
        'Set the Protection properties based
        ' on a table
        wksAllowEditRange.Protect Password:="wrox1", _
        DrawingObjects:=True, _
        Contents:=True, _
        AllowFormattingCells:=rngUser.Offset(1, 0).Value, _
        AllowFormattingColumns:=rngUser.Offset(2, 0).Value, _
        AllowFormattingRows:=rngUser.Offset(3, 0).Value, _
        AllowSorting:=rngUser.Offset(4, 0).Value, _
        UserInterfaceOnly:=True

        'Select Unlocked cells, Locked and Unlocked cells, or neither
        ' is NOT part of the Protection object
        If rngUser.Offset(5, 0).Value = True Then
            wksAllowEditRange.EnableSelection = xlUnlockedCells
        Else
            wksAllowEditRange.EnableSelection = xlNoRestrictions
        End If
    Else
        'Current user is not on the table
        MsgBox "User not found on User Table. Default Options will be used.",
vbExclamation, "Protection Settings"
        wksAllowEditRange.Protect , True, True, False, False, False, _
                                    False, False, False, False, False, _
                                    False, False, False, False, False

        wksAllowEditRange.EnableSelection = xlNoRestrictions

    End If

End Sub
```

PublishObject Object and the PublishObjects Collection

The PublishObjects collection holds all of the things in a workbook that have been saved to a Web page. Each PublishObject object contains items from a workbook that have been saved to a Web page and may need some occasional refreshing of values on the Web page side. The parent of the PublishObjects collection is the Workbook object.

The PublishObjects collection has a few properties and methods besides the typical collection attributes. The unique attributes are listed in the following table.

PublishObjects Properties and Methods

Name	Returns	Description
AutoRepublish	Boolean	Set/Get whether an item in the PublishObjects collection should be republished when a workbook is saved
Count	Long	Read-only. Returns the number of PublishObject objects in the collection

Continues

Name	Returns	Description
Add	Publish Object	Method. Parameters: SourceType As XlSourceType, Filename As String, [Sheet], [Source], [HtmlType], [DivID], [Title]. Adds a PublishObject to the collection
Delete		Method. Deletes the PublishObject objects from the collection
Publish		Method. Publishes all the items associated with the PublishObject objects to a Web page

PublishObject Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

PublishObject Properties

Name	Returns	Description
DivID	String	Read-only. Returns the id used for the <DIV> tag on a Web page
Filename	String	Set/Get the URL or path that the object will be saved to as a Web page
HtmlType	XlHtmlType	Set/Get what type of Web page to save (for example, xlHtmlStatic, xlHtmlChart). Pages saved as other than xlHtmlStatic need special ActiveX components
Sheet	String	Read-only. Returns the Excel sheet that will be saved as a Web page
Source	String	Read-only. Returns the specific item, like range name, chart name, or report name from the base type specified by the SourceType property
SourceType	XlSource Type	Read-only. Returns the type of source being published (for example, xlSourceChart, xlSourcePrintArea, etc.)
Title	String	Set/Get the Web page title for the published Web page

PublishObject Methods

Name	Returns	Parameters	Description
Delete			Deletes the PublishObject object
Publish		[Create]	Publishes the source items specified by the PublishObject as a Web file. Set the Create parameter to True to overwrite existing files. False will append to the existing Web page with the same name, if any

Example: PublishObject Object and the PublishObjects Collection

```
Sub UpdatePublishedCharts()
   Dim oPO As PublishObject
   For Each oPO In ActiveWorkbook.PublishObjects
      If oPO.SourceType = xlSourceChart Then
         oPO.Publish
      End If
   Next
End Sub
```

QueryTable Object and the QueryTables Collection

The `QueryTables` collection holds the collection of data tables created from an external data source. Each `QueryTable` object represents a single table in a worksheet filled with data from an external data source. The external data source can be an ODBC source, an OLE DB source, a text file, a Data Finder, a Web-based query, or a DAO/ADO recordset. The parent of the `QueryTables` collection is the `Worksheet` object.

The `QueryTables` collection has a few properties and methods not typical of a collection. These atypical attributes are listed next.

QueryTables Properties and Methods

Name	Returns	Description
`Count`	`Long`	Read-only. Returns the number of items in the collection
`Add`	`QueryTable`	Method. Parameters: `Connection`, `Destination As Range`, `[Sql]`. Adds a `QueryTable` to the collection. The `Connection` parameter can specify the ODBC or OLE DB connection string, another `QueryTable` object, a DAO or ADO recordset object, a Web-based query, a Data Finder string, or a text file name. The `Destination` parameter specifies the upper-left corner that the query table results will be placed. The `SQL` parameter can specify the SQL for the connection, if applicable

QueryTable Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

QueryTable Properties

Name	Returns	Description
`AdjustColumn Width`	`Boolean`	Set/Get whether the column widths automatically adjust to best fit the data every time the query table is refreshed

Continues

Name	Returns	Description
`BackgroundQuery`	`Boolean`	Set/Get if the query table processing is done asynchronously
`CommandText`	`Variant`	Set/Get the SQL command used to retrieve data
`CommandType`	`XlCmdType`	Set/Get the type of `ComandText` (for example, `xlCmdSQL`, `xlCmdTable`)
`Connection`	`Variant`	Set/Get the OLE DB connection string, the ODBC string, Web data source, path to a text file, or path to a database
`Destination`	`Range`	Read-only. Returns the upper-left corner cell that the query table results will be placed
`EditWebPage`	`Variant`	Set/Get the Webpage URL for a Web query
`EnableEditing`	`Boolean`	Set/Get whether the query table data can be edited or only refreshed (`False`)
`EnableRefresh`	`Boolean`	Set/Get whether the query table data can be refreshed
`FetchedRowOverflow`	`Boolean`	Read-only. Returns whether the last query table refresh retrieved more rows than available on the worksheet
`FieldNames`	`Boolean`	Set/Get whether the field names from the data source become column headings in the query table
`FillAdjacentFormulas`	`Boolean`	Set/Get whether formulas located to the right of the query table will update automatically when the query table data is refreshed
`ListObject`	`Range`	Returns a Range object
`MaintainConnection`	`Boolean`	Set/Get whether the connection to the data source does not close until the workbook is closed. Valid only against an OLE DB source
`Name`	`String`	Set/Get the name of the query table
`Parameters`	`Parameters`	Read-only. Returns the parameters associated with the query table
`PostText`	`String`	Set/Get the post message sent to the Web server to return data from a Web query
`PreserveColumnInfo`	`Boolean`	Set/Get whether column location, sorting and filtering does not disappear when the data query is refreshed
`PreserveFormatting`	`Boolean`	Set/Get whether common formatting associated with the first five rows of data are applied to new rows in the query table

Name	Returns	Description
QueryType	xlQuery Type	Read-only. Returns the type of connection associated with the query table. (For example, xlOLEDBQuery, xlDAOQuery, xlTextImport)
Recordset		Read-only. Returns a recordset associated with the data source query
Refreshing	Boolean	Read-only. Returns whether an asynchronous query is currently in progress
RefreshOn FileOpen	Boolean	Set/Get whether the query table is refreshed when the workbook is opened
Refresh Period	Long	Set/Get how long (minutes) between automatic refreshes from the data source. Set to 0 to disable
RefreshStyle	XlCell Insertion Mode	Set/Get how worksheet rows react when data rows are retrieved from the data source. Worksheet cells can be overwritten (xlOverwriteCells), cell rows can be partial inserted / deleted as necessary (xlInsertDeleteCells), or only cell rows that need to be added are added (xlInsertEntireRows)
ResultRange	Range	Read-only. Returns the cell range containing the results of the query table
Robust Connect	XlRobust Connect	Set/Get how the PivotCache connects to its data source
RowNumbers	Boolean	Set/Get whether a worksheet column is added to the left of the query table containing row numbers
SaveData	Boolean	Set/Get whether query table data is saved with the workbook
SavePassword	Boolean	Set/Get whether an ODBC connection password is saved with the query table
SourceConnection File	String	Set/Get the name of the file that was used to create the PivotTable
SourceData File	String	Read-only. Returns the name of the source data file for the PivotCache
TextFile Column DataTypes	Variant	Set/Get the array of column constants representing the data types for each column. Use the XlColumnDataType constants. Used only when QueryType is xlTextImport
TextFile Comma Delimiter	Boolean	Set/Get whether a comma is the delimiter for text file imports into a query table. Used only when QueryType is xlTextImport and for a delimited text file

Continues

Name	Returns	Description
`TextFileConsecutiveDelimiter`	`Boolean`	Set/Get whether consecutive delimiters (for example, “,,,”) are treated as a single delimeter. Used only when `QueryType` is `xlTextImport`
`TextFileDecimalSeparator`	`String`	Set/Get the type of delimeter to use to define a decimal point. Used only when `QueryType` is `xlTextImport`
`TextFileFixedColumnWidths`	`Variant`	Set/Get the array of widths that correspond to the columns. Used only when `QueryType` is `xlTextImport` and for a fixed width text file
`TextFileOtherDelimiter`	`String`	Set/Get the character that will be used to delimit columns from a text file. Used only when `QueryType` is `xlTextImport` and for a delimited text file
`TextFileParseType`	`XlTextParsingType`	Set/Get the type of text file that is being imported: `xlDelimited` or `xlFixedWidth`. Used only when `QueryType` is `xlTextImport`
`TextFilePlatform`	`XlPlatform`	Set/Get which code pages to use when importing a text file (for example, `xlMSDOS`, `xlWindows`). Used only when `QueryType` is `xlTextImport`
`TextFilePromptOnRefresh`	`Boolean`	Set/Get whether the user is prompted for the text file to use to import into a query table every time the data is refreshed. Used only when `QueryType` is `xlTextImport`. The prompt does not appear on the initial refresh of data
`TextFileSemicolonDelimiter`	`Boolean`	Set/Get whether the semicolon is the text file delimiter for importing text files. Used only when `QueryType` is `xlTextImport` and the file is a delimited text file
`TextFileSpaceDelimiter`	`Boolean`	Set/Get whether the space character is the text file delimiter for importing text files. Used only when `QueryType` is `xlTextImport` and the file is a delimited text file
`TextFileStartRow`	`Long`	Set/Get which row number to start importing from a text file. Used only when `QueryType` is `xlTextImport`
`TextFileTabDelimiter`	`Boolean`	Set/Get whether the tab character is the text file delimiter for importing text files. Used only when `QueryType` is `xlTextImport` and the file is a delimited text file
`TextFileTextQualifier`	`XlTextQualifier`	Set/Get which character will be used to define string data when importing data from a text file. Used only when `QueryType` is `xlTextImport`

Name	Returns	Description
TextFile Thousands Separator	String	Set/Get which character is used as the thousands separator in numbers when importing from a text file (for example, “,”)
TextFile TrailingMinus Numbers	Boolean	Set/Get whether to treat numbers imported as text that begin with a “-” symbol as negative numbers
TextFile VisualLayout	Long	Returns 1 or 2 depending on the visual layout of the file. A value of 1 represents left-to-right while 2 represents right-to-left
Web Consecutive DelimitersAsOne	Boolean	Set/Get whether consecutive delimiters are treated as a single delimeter when importing data from a Web page. Used only when QueryType is xlWebQuery
WebDisable Date Recognition	Boolean	Set/Get whether data that looks like dates are parsed as text when importing Web page data. Used only when QueryType is xlWebQuery
WebDisable Redirections	Boolean	Set/Get whether Web query redirections are disabled for the QueryTable object
WebFormatting	xlWeb Formatting	Set/Get whether to keep any of the formatting when importing a Web page (for example, xlAll, xlNone). Used only when QueryType is xlWebQuery
WebPre Formatted TextToColumns	Boolean	Set/Get whether HTML data with the <PRE> tag is parsed into columns when importing Web pages. Used only when QueryType is xlWebQuery
WebSelection Type	xlWeb Selection Type	Set/Get what data from a Web page is imported. Either all tables (xlAllTables), the entire page (xlEntirePage), or specified tables (xlSpecifiedTables). Used only when QueryType is xlWebQuery
WebSingleBlock TextImport	Boolean	Set/Get whether all the Web page data with the <PRE> tags are imported all at once. Used only when QueryType is xlWebQuery
WebTables	String	Set/Get a comma-delimited list of all the table names that will be imported from a Web page. Used only when QueryType is xlWebQuery and WebSelectionType is xlSpecifiedTables

QueryTable Methods

Name	Returns	Parameters	Description
CancelRefresh			Cancels an asynchronously running query table refresh
Delete			Deletes the query table
Refresh	Boolean	[Background Query]	Refreshes the data in the query table with the latest copy of the external data. Sets the BackgroundQuery parameter to True to get the data to refresh asynchronously
ResetTimer			Resets the time for the automatic refresh set by RefreshPeriod property
SaveAsODC		ODCFileName As String, [Description], [Keywords]	Saves the PivotCache source as an Office Data Connection file. ODCFileName is the location of the source file. Description is the description that will be saved in the file. Keywords is a list of space-separated keywords that can be used to search for this file

Example: QueryTable Object and the QueryTables Collection

```
Sub UpdateAllWebQueries()
   Dim oQT As QueryTable
   For Each oQT In ActiveSheet.QueryTables
      If oQT.QueryType = xlWebQuery Then
         oQT.BackgroundQuery = False
         oQT.Refresh
      End If
   Next
End Sub
```

Range Object

The Range object is one of the more versatile objects in Excel. A range can be a single cell, a column, a row, a contiguous block of cells, or a non-contiguous range of cells. The main parent of a Range object is the Worksheet object. However, most of the objects in the Excel Object Model use the Range object. The Range property of the Worksheet object can be used to choose a certain range of cells using the Cell1 and Cell2 parameters.

Range Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Range Properties

Name	Returns	Description
AddIndent	Variant	Set/Get whether text in a cell is automatically indented if the text alignment in a cell is set to equally distribute
Address	String	Read-only. Parameters: RowAbsolute, ColumnAbsolute, ReferenceStyle As XlReferenceStyle, [External], [RelativeTo]. Returns the address of the current range as a string in the macro's language. The type of address (reference, absolute, A1 reference style, R1C1 reference style) is specified by the parameters
AddressLocal	String	Read-only. Parameters: RowAbsolute, ColumnAbsolute, ReferenceStyle As XlReferenceStyle, [External], [RelativeTo]. Returns the address of the current range as a string in the user's language. The type of address (reference, absolute, A1 reference style, R1C1 reference style) is specified by the parameters
AllowEdit	Boolean	Read-only. Returns True if the range can be edited on a protected worksheet
Areas	Areas	Read-only. Returns an object containing the different non-contiguous ranges in the current range
Borders	Borders	Read-only. Returns all the individual borders around the range. Each border side can be accessed individually in the collection
Cells	Range	Read-only. Returns the cells in the current range. The Cells property will return the same range as the current range
Characters	Characters	Read-only. Parameters: [Start], [Length]. Returns all the characters in the current range, if applicable
Column	Long	Read-only. Returns the column number of the first column in the range
Columns	Range	Read-only. Returns a range of the columns in the current range
ColumnWidth	Variant	Set/Get the column width of all the columns in the range. Returns Null if the columns in the range have different widths
Comment	Comment	Read-only. Returns an object representing the range comment, if any

Continues

Name	Returns	Description
Count	Long	Read-only. Returns the number of cells in the current range
CurrentArray	Range	Read-only. Returns a Range object that represents the array associated with the particular cell range, if the cell is part of an array
CurrentRegion	Range	Read-only. Returns the current region that contains the Range object. A region is defined as an area that is surrounded by blank cells
Dependents	Range	Read-only. Returns the dependents of a cell on the same sheet as the range
Direct Dependents	Range	Read-only. Returns the direct dependents of a cell on the same sheet as the range
Direct Precedents	Range	Read-only. Returns the direct precedents of a cell on the same sheet as the range
End	Range	Read-only. Parameters: Direction As XlDirection. Returns the cell at end of the region containing the Range object. Which end of the region is specified by the Direction parameter
EntireColumn	Range	Read-only. Returns the full worksheet column(s) occupied by the current range
EntireRow	Range	Read-only. Returns the full worksheet row(s) occupied by the current range
Errors	Errors	Read-only. Returns the Errors collection associated with the Range object
Font	Font	Read-only. Returns an object containing Font options for the text in the range
Format Conditions	Format Conditions	Read-only. Returns an object holding conditional formatting options for the current range
Formula	Variant	Set/Get the formula in the cells of the range
FormulaArray	Variant	Set/Get the array formula of the cells in the range
Formula Hidden	Variant	Set/Get whether the formula will be hidden if the workbook/worksheet is protected
FormulaLabel	XlFormula Label	Set/Get the type of formula label to use for the specified range
FormulaLocal	Variant	Set/Get the formula of the range in the language of the user using the A1 style references
FormulaR1C1	Variant	Set/Get the formula of the range in the language of the macro using the R1C1 style references

Name	Returns	Description
FormulaR1C1 Local	Variant	Set/Get the formula of the range in the language of the user using the R1C1 style references
HasArray	Variant	Read-only. Returns whether a single cell range is part of an array formula
HasFormula	Variant	Read-only. Returns whether all the cells in the range contain formulas (True). If only some of the cells contain formulas then Null is returned
Height	Variant	Read-only. Returns the height of the range
Hidden	Variant	Set/Get whether the cells in the range are hidden. Only works if the range contains whole columns or rows
Horizontal Alignment	Variant	Set/Get how the cells in the range are horizontally aligned. Use the XLHAlign constants
Hyperlinks	Hyperlinks	Read-only. Returns the collection of hyperlinks in the range
ID	String	Set/Get the ID used for the range if the worksheet is saved as a Web page
IndentLevel	Variant	Set/Get the indent level for the range
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the range if applicable (for example, interior color)
Left	Variant	Read-only. Returns the distance from the left edge of the left-most column in the range to the left edge of ColumnA
ListHeader Rows	Long	Read-only. Returns the number of header rows in the range
ListOject	Range	Returns a Range object
LocationIn Table	XlLocation InTable	Read-only. Returns the location of the upper-left corner of the range
Locked	Variant	Set/Get whether cells in the range can be modified if the sheet is protected. Returns Null if only some of the cells in the range are locked
MergeArea	Range	Read-only. Returns a range containing the merged range of the current cell range
MergeCells	Variant	Set/Get whether the current range contains merged cells
Name	Variant	Set/Get the Name object that contains the name for the range

Continues

Name	Returns	Description
`Next`	`Range`	Read-only. Returns the next range in the sheet
`NumberFormat`	`Variant`	Set/Get the number format associated with the cells in the range. `Null` if all the cells don't have the same format
`NumberFormat Local`	`Variant`	Set/Get the number format associated with the cells in the range in the language of the end user. `Null` if all the cells don't have the same format
`Offset`	`Range`	Read-only. Parameters: `[RowOffset]`, `[ColumnOffset]`. Returns the cell as a `Range` object that is the offset from the current cell as specified by the parameters. A positive `RowOffset` offsets the row downward. A negative `RowOffset` offsets the row upward. A positive `ColumnOffset` offsets the column to the right and a negative `ColumnOffset` offsets the column to the left
`Orientation`	`Variant`	Set/Get the text orientation for the cell text. A value from −90 to 90 degrees can be specified, or use an `XlOrientation` constant
`OutlineLevel`	`Variant`	Set/Get the outline level for the row or column range
`PageBreak`	`Long`	Set/Get how page breaks are set in the range. Use the `XLPageBreak` constants
`Phonetic`	`Phonetic`	Read-only. Returns the `Phonetic` object associated with the cell range
`Phonetics`	`Phonetics`	Read-only. Returns the `Phonetic` objects in the range
`PivotCell`	`PivotCell`	Read-only. Returns a `PivotCell` object that represents a cell in a `PivotTable` report
`PivotField`	`PivotField`	Read-only. Returns the `PivotTable` field associated with the upper-left corner of the current range
`PivotItem`	`PivotItem`	Read-only. Returns the `PivotTable` item associated with the upper-left corner of the current range
`PivotTable`	`PivotTable`	Read-only. Returns the `PivotTable` report associated with the upper-left corner of the current range
`Precedents`	`Range`	Read-only. Returns the range of precedents of the current cell range on the same sheet as the range
`Prefix Character`	`Variant`	Read-only. Returns the character used to define the type of data in the cell range. For example, "'" for a text label

Name	Returns	Description
`Previous`	`Range`	Read-only. Returns the previous range in the sheet
`QueryTable`	`QueryTable`	Read-only. Returns the query table associated with the upper-left corner of the current range
`Range`	`Range`	Read-only. Parameters: `Cell1`, `[Cell2]`. Returns a `Range` object as defined by the `Cell1` and optionally `Cell2` parameters. The cell references used in the parameters are relative to the range. For example, `Range.Range ("A1")` would return the first column in the parent range but not necessarily the first column in the worksheet
`ReadingOrder`	`Long`	Set/Get whether the text is from right-to-left (`xlRTL`), left-to-right (`xlLTR`), or context-sensitive (`xlContext`)
`Resize`	`Range`	Read-only. Parameters: `[RowSize]`, `[ColumnSize]`. Returns a new resized range as specified by the `RowSize` and `ColumnSize` parameters
`Row`	`Long`	Read-only. Returns the row number of the first row in the range
`RowHeight`	`Variant`	Set/Get the height of the rows in the range. Returns `Null` if the rows in the range have different row heights
`Rows`	`Range`	Read-only. Returns a `Range` object containing the rows of the current range
`ShowDetail`	`Variant`	Set/Get if all the outline levels in the range are expanded. Applicable only if a summary column or row is the range
`ShrinkToFit`	`Variant`	Set/Get whether the cell text will automatically shrink to fit the column width. Returns `Null` if the rows in the range have different `ShrinkToFit` properties
`SmartTags`	`SmartTags`	Read-only. Returns a `SmartTags` object representing the identifier for the specified cell
`SoundNote`	`SoundNote`	Property is kept for backwards compatibility only
`Style`	`Variant`	Read-only. Returns the `Style` object associated with the range
`Summary`	`Variant`	Read-only. Returns whether the range is an outline summary row or column

Continues

Name	Returns	Description
Text	Variant	Read-only. Returns the text associated with a range cell
Top	Variant	Read-only. Returns the distance from the top edge of the topmost row in the range to the top edge of RowA
UseStandard Height	Variant	Set/Get whether the row height is the standard height of the sheet. Returns Null if the rows in the range contain different heights
UseStandard Width	Variant	Set/Get whether the column width is the standard width of the sheet. Returns Null if the columns in the range contain different widths
Validation	Validation	Read-only. Returns the data validation for the current range
Value	Variant	Parameters: [RangeValueDataType]. Set/Get the value of a cell or an array of cells depending on the contents of the Range object
Value2	Variant	Set/Get the value of a cell or an array of cells depending on the contents of the Range object. No Currency or Date types are returned by Value2
Vertical Alignment	Variant	Set/Get how the cells in the range are vertically aligned. Use the XLVAlign constants
Width	Variant	Read-only. Returns the height of the range
Worksheet	Worksheet	Read-only. Returns the worksheet that has the Range object
WrapText	Variant	Set/Get whether cell text wraps in the cell. Returns Null if the cells in the range contain different text wrap properties
XPath	Object	Represents the XPath element of the object at the current range

Range Methods

Name	Returns	Parameters	Description
Activate	Variant		Selects the range cells
AddComment	Comment	[Text]	Adds the text specified by the parameter to the cell specified in the range. Must be a single cell range

Name	Returns	Parameters	Description
AdvancedFilter	Variant	Action As XlFilterAction, [CriteriaRange], [CopyToRange], [Unique]	Copies or filters the data in the current range. The Action parameter specifies whether a copy or filter is to take place. CriteriaRange optionally specifies the range containing the criteria. CopyToRange specifies the range that the filtered data will be copied to (if Action is xlFilterCopy)
ApplyNames	Variant	Names, IgnoreRelativeAbsolute, UseRowColumnNames, OmitColumn, OmitRow, Order As XlApplyNamesOrder, [AppendLast]	Applies defined names to the formulas in a range. For example, if a cell contained =A1*100 and A1 was given the name "TopLeft", you could apply the "TopLeft" name to the range, resulting in the formula changing to =TopLeft*100. Note that there is no UnApplyNames method
ApplyOutlineStyles	Variant		Applies the outline styles to the range
AutoComplete	String	String As String	Returns and tries to AutoComplete the word specified in the String parameter. Returns the complete word, if found. Returns an empty string if no word or more than one word is found
AutoFill	Variant	Destination As Range, Type As XlAutoFillType	Uses the current range as the source to figure out how to AutoFill the range specified by the Destination parameter. The Type parameter can also be used to specify the type of fill to use (for example, xlFillCopy, xlFillDays)

Continues

Name	Returns	Parameters	Description
AutoFilter	Variant	Field, Criteria1, Operator As XlAutoFilter Operator, [Criteria2], [Visible Drop Down]	Creates an auto-filter on the data in the range. See the AutoFilter object for details on the parameters
AutoFit	Variant		Changes the column widths in the range to best fit the data in the cells. The range must contain full rows or columns
AutoFormat	Variant	Format As XlRangeAuto Format, [Number], [Font], [Alignment], [Border], [Pattern], [Width]	Formats the range using the format specified by the Format parameter. The other parameters are Boolean indicators to specify: if numbers are formatted appropriately (Number), fonts applied (Font), alignments applied (Alignment), border formats applied (Border), pattern formats applied (Pattern), and if row/ column widths are applied from the autoformat
AutoOutline	Variant		Creates an outline for the range
BorderAround	Variant	LineStyle, Weight As XlBorder Weight, ColorIndex As XlColorIndex, [Color]	Creates a border around the range with the associated line style (LineStyle), thickness (Weight), and color (ColorIndex)
Calculate	Variant		Calculates all the formulas in the range

Name	Returns	Parameters	Description
`CheckSpelling`	`Variant`	`[Custom Dictionary], [Ignore Uppercase], [Always Suggest], [SpellLang]`	Checks the spelling of the text in the range. A custom dictionary can be specified (`CustomDictionary`), all UPPERCASE words can be ignored (`IgnoreUppercase`), and Excel can be set to display a list of suggestions (`AlwaysSuggest`)
`Clear`	`Variant`		Clears the text in the cells of the range
`ClearComments`			Clears all the comments in the range cells
`ClearContents`	`Variant`		Clears the formulas and values in a range
`ClearFormats`	`Variant`		Clears the formatting in a range
`ClearNotes`	`Variant`		Clears comments from the cells in the range
`ClearOutline`	`Variant`		Clears the outline used in the current range
`Column Differences`	`Range`	`Comparison`	Returns the range of cells that are different to the cell specified by the `Comparison` parameter
`Consolidate`	`Variant`	`[Sources], [Function], [TopRow], [Left Column], [Create Links]`	Consolidates the source array of range reference strings in the `Sources` parameter and returns the results to the current range. The `Function` parameter can be used to set the consolidation function. Use the `XLConsolidation Function` constants
`Copy`	`Variant`	`[Destination]`	Copies the current range to the range specified by the parameter or to the clipboard if no destination is specified

Continues

Name	Returns	Parameters	Description
`CopyFromRecordset`	`Long`	`Data As Recordset, [MaxRows], [MaxColumns]`	Copies the records from the ADO or DAO recordset specified by the `Data` parameter into the current range. The recordset can't contain `OLE` objects
`CopyPicture`	`Variant`	`Appearance As XlPicture Appearance, Format As XlCopyPicture Format`	Copies the range into the clipboard as a picture. The `Appearance` parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The `Format` parameter can specify the type of picture that will be put into the clipboard
`CreateNames`	`Variant`	`[Top], [Left], [Bottom], [Right]`	Creates a named range for the items in the current range. Set `Top` to `True` to make the first row hold the names for the ranges below. Set `Bottom` to `True` to use the bottom row as the names. Set `Left` or `Right` to `True` to make the left or right column contain the `Names`, respectively
`Create Publisher`	`Variant`	`Edition, Appearance As XlPicture Appearance, [Contains PICT], [Contains BIFF], [ContainsRTF], [Contains VALU]`	Creates a publisher based on the range. Available only on the Macintosh with System 7 or later
`Cut`	`Variant`	`[Destination]`	Cuts the current range to the range specified by the parameter, or to the clipboard if no destination is specified

Name	Returns	Parameters	Description
`DataSeries`	`Variant`	`Rowcol, Type As XlDataSeries Type, Date As XlDataSeries Date, [Step], [Stop], [Trend]`	Creates a data series at the current range location
`Delete`	`Variant`	`[Shift]`	Deletes the cells in the current range and optionally shifts the cells in the direction specified by the `Shift` parameter. Use the `XlDeleteShift Direction` constants for the `Shift` parameter
`DialogBox`	`Variant`		Displays a dialog box defined by an Excel 4.0 macro sheet
`Dirty`			Selects a range to be recalculated when the next recalculation occurs
`Edition Options`	`Variant`	`Type As XlEdition Type, Option As XlEdition Options Option, Name, Reference, Appearance As XlPicture Appearance, ChartSize As XlPicture Appearance, [Format]`	Used on the Macintosh. `EditionOptions` set how the range should act when being used as the source (publisher) or target (subscriber) of the link. `Editions` are basically the same as Windows' DDE links
`FillDown`	`Variant`		Copies the contents and formatting from the top row into the rest of the rows in the range
`FillLeft`	`Variant`		Copies the contents and formatting from the rightmost column into the rest of the columns in the range

Continues

Name	Returns	Parameters	Description
FillRight	Variant		Copies the contents and formatting from the leftmost column into the rest of the columns in the range
FillUp	Variant		Copies the contents and formatting from the bottom row into the rest of the rows in the range
Find	Range	What As Variant, [After], [LookIn], [LookAt], [SearchOrder], [Search Direction As XlSearch Direction], [MatchCase], [MatchByte], [Search Format]	Looks through the current range for the text of data type specified by the What parameter. Use a single cell range in the After parameter to choose the starting position of the search. Use the LookIn parameter to decide where the search is going to take place
FindNext	Range	[After]	Finds the next instance of the search criteria defined with the Find method
FindPrevious	Range	[After]	Finds the previous instance of the search criteria defined with the Find method
Function Wizard	Variant		Displays the Function Wizard for the upper-left cell of the current range
GoalSeek	Boolean	Goal, ChangingCell As Range	Returns True if the value specified by the Goal parameter is returned when changing the ChangingCell cell range
Group	Variant	[Start], [End], [By], [Periods]	Either demotes the outline in the range or groups the discontinuous ranges in the current Range object

Name	Returns	Parameters	Description

`Insert`	`Variant`	`[Shift],`
		`[CopyOrigin]`
			Inserts the equivalent rows or columns in the range into the range's worksheet

`InsertIndent`		`InsertAmount`
		`As Long`
			Indents the range by the amount specified by the `InsertAmount` parameter

`Justify`	`Variant`		Evenly distributes the text in the cells from the current range
`ListNames`	`Variant`		Pastes the names of all the named ranges in the current range starting at the top-left cell in the range
`Merge`		`[Across]`	Merges the cells in the range. Set the `Across` parameter to `True` to merge each row as a separate cell

`Navigate`
`Arrow`
	`Variant`	`[Toward`
		`Precedent],`
		`[ArrowNumber],`
		`[LinkNumber]`
			Moves through the tracer arrows in a workbook from the current range, returning the range of cells that make up the tracer arrow destination. Tracer arrows must be turned on. Use the `ShowDependents` and `ShowPrecendents` methods

`NoteText`	`String`	`[Text],`
		`[Start],`
		`[Length]`
			Set/Get the cell notes associated with the cell in the current range

`Parse`	`Variant`	`[ParseLine],`
		`[Destination]`
			Parses the string specified by the `ParseLine` parameter and returns it to the current range parsed out by column. Optionally, can specify the destination range with the `Destination` parameter. The `ParseLine` string should be in the "`[ColumnA] [ColumnB]`" format

Continues

Name	Returns	Parameters	Description
`PasteSpecial`	`Variant`	`Paste As XlPasteType, Operation As XlPaste Special Operation, [SkipBlanks], [Transpose]`	Pastes the range from the clipboard into the current range. Use the `Paste` parameter to choose what to paste (for example, formulas, values). Use the `Operation` parameter to specifiy what to do with the paste. Set `SkipBlanks` to `True` to not have blank cells in the clipboard's range pasted. Set `Transpose` to `True` to transpose columns with rows
`PrintOut`	`Variant`	`[From], [To], [Copies], [Preview], [Active Printer], [PrintToFile], [Collate], [PrToFile Name]`	Prints out the charts in the collection. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file with using the `PrintToFile` and `PrToFileName` parameters. The `From` and `To` parameters can be used to specify the range of printed pages
`PrintPreview`	`Variant`	`[Enable Changes]`	Displays the current range in a print preview. Set the `EnableChanges` parameter to `False` to disable the `Margins` and `Setup` buttons, hence not allowing the viewer to modify the page setup
`Remove Subtotal`	`Variant`		Removes subtotals from the list in the current range
`Replace`	`Boolean`	`What As Variant, Replacement As Variant, [LookAt], [SearchOrder], [MatchCase], [MatchByte], [Search Format], [Replace Format]`	Finds the text specified by the What parameter in the range. Replaces the found text with the `Replacement` parameter. Use the `SearchOrder` parameters with the `XLSearchOrder` constants to choose whether the search occurs by rows or by columns

Name	Returns	Parameters	Description
`Row Differences`	`Range`	`Comparison.`	Returns the range of cells that are different to the cell specified by the `Comparison` parameter

`Run`	`Variant`	`[Arg1],`
		`[Arg2],`
		`...`
		`[Arg30]`
			Runs the Excel 4.0 macro specified by the current range. The potential arguments to the macro can be specified with the `Argx` parameters

`Select`	`Variant`		Selects the cells in the range
`SetPhonetic`			Creates a `Phonetic` object for each cell in the range
`Show`	`Variant`		Scrolls the Excel window to display the current range. This only works if the range is a single cell
`ShowDependents`	`Variant`	`[Remove]`	Displays the dependents for the current single cell range using tracer arrows
`ShowErrors`	`Variant`		Displays the source of the errors for the current range using tracer arrows
`ShowPrecedents`	`Variant`	`[Remove]`	Displays the precedents for the current single cell range using tracer arrows
`Sort`	`Variant`	`[Key1], [Order1 As XlSortOrder], [Key2], [Type], [Order2 As XlSortOrder], [Key3], [Order3 As XlSortOrder], [Header As XlYesNoGuess], [OrderCustom], [MatchCase],`	Sorts the cells in the range. If the range contains only one cell then the active region is searched. Use the `Key1`, `Key2`, and `Key3` parameters to set which columns will be the sort columns. Use the `Order1`, `Order2`, and `Order3` parameters to set the sort order. Use the `Header` parameter to set whether the first row contains headers. Set the

Continues

Name	Returns	Parameters	Description
		[Orientation As XlSort Orientation], [SortMethod As XlSortMethod], [DataOption1 As XlSortData Option], [DataOption2 As XlSortData Option], [DataOption3 As XlSortData Option]	MatchCase parameter to True to sort data and to treat uppercase and lowercase characters differently. Use the Orientation parameter to choose whether rows are sorted or columns are sorted. Finally, the SortMethod parameter is used to set the sort method for other languages (for example, xlStroke or xlPinYin). Use the SortSpecial method for sorting in East Asian languages
SortSpecial	Variant	[SortMethod As XlSortMethod], [Key1], [Order1 As XlSortOrder] [Type], [Key2], [Order2 As XlSortOrder], [Key3], [Order3 As XlSortOrder] [Header As XlYesNoGuess], [OrderCustom], [MatchCase], [Orientation As XlSort Orientation], [DataOption1 As XlSortData Option], [DataOption2 As XlSortData Option], [DataOption3 As XlSortData Option]	Sorts the data in the range using East Asian sorting methods. The parameters are the same as the Sort method
Speak		[Speak Direction], [Speak Formulas]	Causes the cells of the range to be spoken in row order or column order

Name	Returns	Parameters	Description
`SpecialCells`	`Range`	`Type As XlCellType, [Value]`	Returns the cells in the current range that contain some special attribute as defined by the `Type` parameter. For example, if `Type` is `xlCellTypeBlanks` then a `Range` object containing all of the empty cells are returned
`SubscribeTo`	`Variant`	`Edition As String, Format As XlSubscribeTo Format`	Only valid on the Macintosh. Defines the source of a link that the current range will contain
`Subtotal`	`Variant`	`GroupBy As Long, Function As Xl Consolidation Function, TotalList, Replace, PageBreaks, SummaryBelow Data As XlSummaryRow`	Creates a subtotal for the range. If the range is a single cell then a subtotal is created for the current region. The `GroupBy` parameter specifies the field to group (for subtotaling). The `Function` parameter describes how the fields will be grouped. The `TotalList` parameter uses an array of field offsets that describe the fields that will be subtotaled. Set the `Replace` parameter to `True` to replace existing subtotals. Set the `PageBreaks` to `True` for page breaks to be added after each group. Use the `SummaryBelowData` parameter to choose where the summary row will be added
`Table`	`Variant`	`[RowInput], [Column Input]`	Creates a new data table at the current range

Continues

Name	Returns	Parameters	Description
TextToColumns	Variant	[Destination], [DataType As XlTextParsing Type], [Text Qualifier As XlText Qualifier], [Consecutive Delimiter], [Tab], [Semicolon], [Comma], [Space], [Other], [OtherChar], [FieldInfo], [Decimal Separator], [Thousands Separator], [Trailing MinusNumbers]	Parses text in cells into several columns. The Destination specifies the range that the parsed text will go into. The DataType parameter can be used to choose whether the text is delimited or fixed width. The TextQualifier parameter can specify which character denotes string data when parsing. Set the Consecutive Delimiter to True for Excel to treat consecutive delimiters as one. Set the Tab, Semicolon, Comma, or Space parameter to True to use the associated character as the delimiter. Set the Other parameter to True and specify an OtherChar to use another character as the delimiter. FieldInfo takes a two-dimensional array containing more parsing information. The DecimalSeparator and ThousandsSeparator can specify how numbers are treated when parsing
Ungroup	Variant		Either promotes the outline in the range or ungroups the range in a PivotTable report
UnMerge			Splits up a merged cell into single cells

Example: Range Object

See Chapter 5 for examples of working with the Range object.

RecentFile Object and the RecentFiles Collection

The RecentFiles collection holds the list of recently modified files. Equivalent to the files listed under the File menu in Excel. Each RecentFile object represents one of the recently modified files.

RecentFiles has a few attributes besides the typical collection ones. The Maximum property can be used to set or return the maximum number of files that Excel will "remember" modifying. The value can range from 0 to 9. The Count property returns the number of RecentFile objects in the collection. The Add method is used to add a file (with the Name parameter) to the collection.

RecentFile Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

RecentFile Properties

Name	Returns	Description
Index	Long	Read-only. Returns the spot in the collection that the current object is located
Name	String	Read-only. Returns the name of the recently modified file
Path	String	Read-only. Returns the file path of the recently modified file

RecentFile Methods

Name	Returns	Parameters	Description
Delete			Deletes the object from the collection
Open	Workbook		Opens up the recent file and returns the opened workbook

Example: RecentFile Object and the RecentFiles Collection

```
Sub CheckRecentFiles()
   Dim oRF As RecentFile
   'Remove any recent files that refer to the floppy drive
   For Each oRF In Application.RecentFiles
      If Left(oRF.Path, 2) = "A:" Then
         oRF.Delete
      End If
   Next
End Sub
```

RoutingSlip Object

The RoutingSlip object represents the properties and methods of the routing slip of an Excel document. The parent object of the RoutingSlip object is the Workbook object. The HasRoutingSlip property of the Workbook object has to set to True before the RoutingSlip object can be manipulated.

RoutingSlip Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

RoutingSlip Properties

Name	Returns	Description
Delivery	XlRoutingSlipDelivery	Set/Get how the delivery process will proceed
Message	Variant	Set/Get the body text of the routing slip message
Recipients	Variant	Parameters: [Index]. Returns the list of recipient names to send the parent workbook to
ReturnWhenDone	Boolean	Set/Get whether the message is returned to the original sender
Status	XlRoutingSlipStatus	Read-only. Returns the current status of the routing slip
Subject	Variant	Set/Get the subject text for the routing slip message
TrackStatus	Boolean	Set/Get whether the message is sent to the original sender each time the message is forwarded

RoutingSlip Methods

Name	Returns	Parameters	Description
Reset	Variant		Reset the routing slip

RTD Object

Represents a Real-Time Data object, like one referenced using the IrtdServer object. As of this writing, there was very little documentation.

RTD Properties

Name	Returns	Description
ThrottleInterval	Long	Set/Get the time interval between updates

RTD Methods

Name	Returns	Parameters	Description
RefreshData	RTD		Requests an update of RTD from the RTD server
Restart Servers	RTD		Reconnects to servers for RTD

Scenario Object and the Scenarios Collection

The Scenarios collection contains the list of all the scenarios associated with a worksheet. Each Scenario object represents a single scenario in a worksheet. A scenario holds the list of saved cell values that can later be substituted into the worksheet. The parent of the Scenarios collection is the Worksheet object.

The Scenarios collection has a few extra properties and methods besides the typical collection attributes. These are listed in the following table.

Scenarios Properties and Methods

Name	Returns	Description
Count	Long	Read-only. Returns the number of Scenario objects in the collection
Add	Scenario	Method. Parameters: Name As String, ChangingCells, [Values], [Comment], [Locked], [Hidden]. Adds a scenario to the collection. The Name parameter specifies the name of the scenario. See the Scenario object for a description of the parameters
CreateSummary	Variant	Method. Parameters: ReportType As XlSummaryReportType, [ResultCells]. Creates a worksheet containing a summary of all the scenarios of the parent worksheet. The ReportType parameter can specify the report type. The ResultCells parameter can be a range of cells containing the formulas related to the changing cells
Merge	Variant	Method. Parameters: Source. Merges the scenarios in the Source parameter into the current worksheet

Scenario Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Scenario Properties

Name	Returns	Description
`ChangingCells`	`Range`	Read-only. Returns the range of cells in the worksheet that will have values plugged in for the specific scenario
`Comment`	`String`	Set/Get the scenario comment
`Hidden`	`Boolean`	Set/Get whether the scenario is hidden
`Index`	`Long`	Read-only. Returns the spot in the collection that the current `Scenario` object is located
`Locked`	`Boolean`	Set/Get whether the scenario cannot be modified when the worksheet is protected
`Name`	`String`	Set/Get the name of the scenario
`Values`	`Variant`	Read-only. Parameters: `[Index]`. Returns an array of the values to plug in to the changing cells for this particular scenario

Scenario Methods

Name	Returns	Parameters	Description
`ChangeScenario`	`Variant`	`ChangingCells, [Values]`	Changes which set of cells in the worksheet are able to change for the scenario. Optionally, can choose new values for the scenario
`Delete`	`Variant`		Deletes the `Scenario` object from the collection
`Show`	`Variant`		Shows the scenario results by putting the scenario values into the worksheet

Example: Scenario Object and the Scenarios Collection

```
Sub GetBestScenario()
   Dim oScen As Scenario
   Dim oBestScen As Scenario
   Dim dBestSoFar As Double
   'Loop through the scenarios in the sheet
```

```
    For Each oScen In ActiveSheet.Scenarios
       'Show the secnario
       oScen.Show

       'Is it better?
       If Range("Result").Value > dBestSoFar Then
          dBestSoFar = Range("Result").Value

          'Yes - remember it
          Set oBestScen = oScen
       End If
    Next
    'Show the best scenario
    oBestScen.Show
    MsgBox "The best scenario is " & oBestScen.Name
 End Sub
```

Series Object and the SeriesCollection Collection

The SeriesCollection collection holds the collection of series associated with a chart group. Each Series object contains a collection of points associated with a chart group in a chart. For example, a simple line chart contains a series (Series) of points brought in from the originating data. Since some charts can have many series plotted on the same chart, the SeriesCollection is used to hold that information. The parent of the SeriesCollection is the ChartGroup.

The SeriesCollection has a few attributes that are not typical of a collection. These are listed in the following table.

SeriesCollection Properties and Methods

Name	Returns	Description
Add	Series	Method. Parameters: Source, Rowcol As XlRowCol, [SeriesLabels], [CategoryLabels], [Replace]. Adds a Series to the collection. The Source parameter specifies either a range or an array of data points describing the new series (and all the points in it). The Rowcol parameter sets whether the row or the column of the Source contains a series of points. Set SeriesLabels or CategoryLabels to True to make the first row or column of the Source contain the labels for the series and category, respectively
Count	Long	Read-only. Returns the number of Series objects in the collection

Continues

Name	Returns	Description
Extend	Variant	Method. Parameters: Source, [Rowcol], [CategoryLabels]. Adds the points specified by the range or array of data points in the Source parameter to the SeriesCollection. See the Add method for details on the other parameters
Paste	Variant	Method. Parameters: Rowcol As XlRowCol, [SeriesLabels], [CategoryLabels], [Replace], [NewSeries]. Pastes the data from the Clipboard into the SeriesCollection as a new Series. See the Add method for details on the other parameters
NewSeries	Series	Method. Creates a new series and returns the newly created series

Series Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Series Properties

Name	Returns	Description
ApplyPictToEnd	Boolean	Set/Get whether pictures are added to the end of the points in the series
ApplyPictTo Front	Boolean	Set/Get whether pictures are added to the front of the points in the series
ApplyPictTo Sides	Boolean	Set/Get whether pictures are added to the sides of the points in the series
AxisGroup	XlAxis Group	Set/Get the type of axis type being used by the series (primary or secondary)
BarShape	XlBarShape	Set/Get the type of shape to use in a 3D bar or column chart (for example, xlBox)
Border	Border	Read-only. Returns the collection of borders (sides) around the series. Each border’s attributes can be accessed individually
BubbleSizes	Variant	Set/Get the cell references (A1 reference style) that contain data relating to how big the bubble should be for bubble charts
ChartType	XlChartType	Set/Get the type of chart to use for the series
ErrorBars	ErrorBars	Read-only. Returns the error bars in a series. Use with HasErrorBars

Name	Returns	Description
`Explosion`	`Long`	Set/Get how far out the slices (points) of a pie or doughnut chart will explode out. `0` for no explosion
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the series of points on a chart
`Formula`	`String`	Set/Get the type of formula label to use for the series
`FormulaLocal`	`String`	Set/Get the formula of the series in the language of the user using the A1 style references
`FormulaR1C1`	`String`	Set/Get the formula of the series in the language of the macro using the R1C1 style references
`FormulaR1C1 Local`	`String`	Set/Get the formula of the series in the language of the user using the R1C1 style references
`Has3DEffect`	`Boolean`	Set/Get if bubble charts have a 3D appearance
`HasDataLabels`	`Boolean`	Set/Get if the series contains data labels
`HasErrorBars`	`Boolean`	Set/Get if the series contains error bars. Use with the `ErrorBars` property
`HasLeader Lines`	`Boolean`	Set/Get if the series contains leader lines. Use with the `LeaderLines` property
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of the series (for example, interior color)
`InvertIf Negative`	`Boolean`	Set/Get whether the color of the series' points should be the inverse if the value is negative
`LeaderLines`	`Leader Lines`	Read-only. Returns the leader lines associated with the series
`Marker Background Color`	`Long`	Set/Get the color of the series points marker background. Use the `RGB` function to create the color value
`Marker Background ColorIndex`	`XlColor Index`	Set/Get the color of the series points marker background. Use the `XlColorIndex` constants or an index value in the current color palette
`Marker Foreground Color`	`Long`	Set/Get the color of the series points marker foreground. Use the `RGB` function to create the color value
`Marker Foreground ColorIndex`	`XlColor Index`	Set/Get the color of the series points marker foreground. Use the `XlColorIndex` constants or an index value in the current color palette
`MarkerSize`	`Long`	Set/Get the size of the point key marker
`MarkerStyle`	`XlMarker Style`	Set/Get the type of marker to use as the point key (for example, square, diamond, triangle, picture, etc.)

Continues

Name	Returns	Description
Name	String	Set/Get the name of the series
PictureType	XlChart Picture Type	Set/Get how an associated picture is displayed on the series (for example, stretched, tiled). Use the XlPictureType constants
PictureUnit	Long	Set/Get how many units a picture represents if the PictureType property is set to xlScale
PlotOrder	Long	Set/Get the plotting order for this particular series in the SeriesCollection
Shadow	Boolean	Set/Get whether the points in the series will have a shadow effect
Smooth	Boolean	Set/Get whether scatter or line charts will have curves smoothed
Type	Long	Set/Get the type of series
Values	Variant	Set/Get the range containing the series values or an array of fixed values containing the series values
XValues	Variant	Set/Get the array of x values coming from a range or an array of fixed values

Series Methods

Name	Returns	Parameters	Description
ApplyCustom Type		ChartType As XlChartType	Changes the chart type to the one specified in the ChartType parameter
ApplyData Labels	Variant	[Type As XlDataLabels Type], [LegendKey], [AutoText], [HasLeader Lines], [ShowSeries Name], [ShowCategory Name], [ShowValue], [Show Percentage], [ShowBubble Size], [Separator]	Applies the data label properties specified by the parameters to the series. The Type parameter specifies whether no label, a value, a percentage of the whole, or a category label is shown. The legend key can appear by the point by setting the LegendKey parameter to True. Set the HasLeaderLines to True to add leader lines to the series

Name	Returns	Parameters	Description
`ClearFormats`	`Variant`		Clears the formatting made on the series
`Copy`	`Variant`		Copies the series into the clipboard
`DataLabels`	`Object`	`[Index]`	Returns the collection of data labels in a series. If the `Index` parameter is specified then only a single data label is returned
`Delete`	`Variant`		Deletes the series from the series collection
`ErrorBar`	`Variant`	`[Direction As XlErrorBar Direction], [Include As XlErrorBar Include], [Type As XlErrorBar Type], [Amount], [MinusValues]`	Adds error bars to the series. The `Direction` parameter chooses whether the bar appears on the x or y axis. The `Include` parameter specifies which error parts to include. The `Type` parameter decides the type of error bar to use. The `Amount` parameter is used to choose an error amount. The `MinusValues` parameter takes the negative error amount to use when the `Type` parameter is `xlErrorBarTypeCustom`
`Paste`	`Variant`		Uses the picture in the `Clipboard` as the marker on the points in the series
`Points`		`[Index]`	Returns either the collection of points associated with the series or a single point if the `Index` parameter is specified
`Select`	`Variant`		Selects the series' points on the chart
`Trendlines`		`[Index]`	Returns either the collection of trendlines associated with the series or a single trendline if the `Index` parameter is specified

Example: Series Object and the SeriesCollection Collection

See the DataLabel object for an example of using the Series object.

SeriesLines Object

The SeriesLines object accesses the series lines connecting data values from each series. This object only applies to 2D stacked bar or column chart groups. The parent of the SeriesLines object is the ChartGroup object.

SeriesLines Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

SeriesLines Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the series lines
Name	String	Read-only. Returns the name of the SeriesLines object

SeriesLines Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the SeriesLines object
Select	Variant		Selects the series lines in the chart

Example: SeriesLines Object

```
Sub FormatSeriesLines()
    Dim oCG As ChartGroup
    Dim oSL As SeriesLines
    'Loop through the column groups on the chart
    For Each oCG In Charts(1).ColumnGroups
       'Make sure we have some series lines
       oCG.HasSeriesLines = True
       'Get the series lines
       Set oSL = oCG.SeriesLines
       'Format the lines
       With oSL
          .Border.Weight = xlThin
          .Border.ColorIndex = 5
       End With
   Next
End Sub
```

ShadowFormat Object

The ShadowFormat object allows manipulation of the shadow formatting properties of a parent Shape object. Use the Shadow property of the Shape object to access the ShadowFormat object.

ShadowFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ShadowFormat Properties

Name	Returns	Description
ForeColor	ColorFormat	Read-only. Allows manipulation of the shadow forecolor
Obscured	MsoTriState	Set/Get whether the shape obscures the shadow or not
OffsetX	Single	Set/Get the horizontal shadow offset
OffsetY	Single	Set/Get the vertical shadow offset
Transparency	Single	Set/Get the transparency of the shadow (0 to 1 where 1 is clear)
Type	MsoShadow Type	Set/Get the shadow type
Visible	MsoTriState	Set/Get whether the shadow is visible

ShadowFormat Methods

Name	Returns	Parameters	Description
Increment OffsetX		Increment As Single	Changes the horizontal shadow offset
Increment OffsetY		Increment As Single	Changes the vertical shadow offset

Example: ShadowFormat Object

```
Sub AddShadow()
   Dim oSF As ShadowFormat
   Set oSF = ActiveSheet.Shapes.Range(1).Shadow
   With oSF
      .Type = msoShadow6
      .OffsetX = 5
      .OffsetY = 5
      .ForeColor.SchemeColor = 2
      .Visible = True
   End With
End Sub
```

Shape Object and the Shapes Collection

The Shapes collection holds the list of shapes for a sheet. The Shape object represents a single shape such as an AutoShape, a free-form shape, an OLE object (like an image), an ActiveX control or a picture. Possible parent objects of the Shapes collection are the Worksheet and Chart object.

The Shapes collection has a few methods and properties besides the typical collection attributes. They are listed in the following table.

Shapes Collection Properties and Methods

Name	Returns	Description
Count	Long	Read-only. Returns the number of shapes in the collection
Range	ShapeRange	Read-only. Parameters: Index. Returns a ShapeRange object containing only some of the shapes in the Shapes collection
AddCallout	Shape	Method. Parameters: Type As MsoCalloutType, Left As Single, Top As Single, Width As Single, Height As Single. Adds a callout line shape to the collection
AddConnector	Shape	Method. Parameters: Type As MsoConnectorType, BeginX As Single, BeginY As Single, EndX As Single, EndY As Single. Adds a connector shape to the collection
AddCurve	Shape	Method. Parameters: SafeArrayOfPoints. Adds a Bezier curve to the collection
AddDiagram	Shape	Method. Parameters: Type As MsoDiagramType, Left As Single, Top As Single, Width As Single, Height As Single. Adds a new AutoShape to the worksheet
AddFormControl	Shape	Method. Parameters: Type As XlFormControl, Left As Long, Top As Long, Width As Long, Height As Long. Adds an Excel control to the collection
AddLabel	Shape	Method. Parameters: Orientation As MsoTextOrientation, Left As Single, Top As Single, Width As Single, Height As Single. Adds a label to the collection
AddLine	Shape	Method. Parameters: BeginX As Single, BeginY As Single, EndX As Single, EndY As Single. Adds a line shape to the collection
AddOLEObject	Shape	Method. Parameters: [ClassType], [Filename], [Link], [DisplayAsIcon], [IconFileName], [IconIndex], [IconLabel], [Left], [Top], [Width], [Height]. Adds an OLE control to the collection

Name	Returns	Description
AddPicture	Shape	Method. Parameters: Filename As String, LinkToFile As MsoTriState, SaveWithDocument As MsoTriState, Left As Single, Top As Single, Width As Single, Height As Single. Adds a picture object to the collection
AddPolyline	Shape	Method. Parameters: SafeArrayOfPoints. Adds an open polyline or a closed polygon to the collection
AddShape	Shape	Method. Parameters: Type As MsoAutoShapeType, Left As Single, Top As Single, Width As Single, Height As Single. Adds a shape using the Type parameter to the collection
AddTextbox	Shape	Method. Parameters: Orientation As MsoTextOrientation, Left As Single, Top As Single, Width As Single, Height As Single. Adds a textbox to the collection
AddText Effect	Shape	Method. Parameters: PresetTextEffect As MsoPresetTextEffect, Text As String, FontName As String, FontSize As Single, FontBold As MsoTriState, FontItalic As MsoTriState, Left As Single, Top As Single. Adds a WordArt object to the collection
Build Freeform	Freeform Builder	Method. Parameters: EditingType As MsoEditingType, X1 As Single, Y1 As Single. Accesses an object that allows creation of a new shape based on ShapeNode objects
SelectAll		Method. Selects all the shapes in the collection

Shape Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Shape Properties

Name	Returns	Description
Adjustments	Adjustments	Read-only. An object accessing the adjustments for a shape
Alternative Text	String	Set/Get the alternate text to appear if the image is not loaded. Used with a Web page
AutoShape Type	MsoAuto ShapeType	Set/Get the type of AutoShape used

Continues

Name	Returns	Description
BlackWhite Mode	MsoBlack WhiteMode	Property used for compatibility to other drawing packages only. Does not do anything
BottomRight Cell	Range	Read-only. Returns the single cell range that describes the cell under the lower-right corner of the shape
Callout	Callout Format	Read-only. An object accessing the callout properties of the shape
Child	MsoTriState	Read-only. Returns whether the specified shape is a child shape, or if all shapes in a shape range are child shapes of the same parent
Connection SiteCount	Long	Read-only. Returns the number of potential connection points (sites) on the shape for a connector
Connector	MsoTriState	Read-only. Returns whether the shape is a connector
Connector Format	Connector Format	Read-only. Returns an object containing formatting options for a connector shape. Shape must be a connector shape
Control Format	Control Format	Read-only. Returns an object containing formatting options for an Excel control. Shape must be an Excel control
Diagram	Diagram	Read-only. Returns a Diagram object
DiagramNode	DiagramNode	Read-only. Returns a node in the diagram
Fill	FillFormat	Read-only. Returns an object containing fill formatting options for the Shape object
FormControl Type	XlForm Control	Read-only. Returns the type of Excel control the current shape is (for example, xlCheck Box). Shape must be an Excel control
GroupItems	GroupShapes	Read-only. Returns the shapes that make up the current shape
HasDiagram	MsoTriState	Read-only. Returns whether a shape or shape range contains a diagram
HasDiagram Node	MsoTriState	Read-only. Returns whether a diagram node exists in a given shape or shape range
Height	Single	Set/Get the height of the shape
Horizontal Flip	MsoTriState	Read-only. Returns whether the shape has been flipped

Name	Returns	Description
`Hyperlink`	`Hyperlink`	Read-only. Returns the hyperlink of the shape, if any
`ID`	`Long`	Read-only. Returns the type for the specified object
`Left`	`Single`	Set/Get the horizontal position of the shape
`Line`	`LineFormat`	Read-only. An object accessing the line formatting of the shape
`LinkFormat`	`LinkFormat`	Read-only. An object accessing the OLE linking properties
`LockAspect Ratio`	`MsoTriState`	Set/Get whether the dimensional proportions of the shape is kept when the shape is resized
`Locked`	`Boolean`	Set/Get whether the shape can be modified if the sheet is locked (`True` = cannot modify)
`Name`	`String`	Set/Get the name of the `Shape` object
`Nodes`	`ShapeNodes`	Read-only. An object accessing the nodes of the free-form shape
`OLEFormat`	`OLEFormat`	Read-only. An object accessing OLE object properties, if applicable
`OnAction`	`String`	Set/Get the macro to run when the shape is clicked
`ParentGroup`	`Shape`	Read-only. Returns the common parent shape of a child shape or a range of child shapes
`Picture Format`	`Picture Format`	Read-only. An object accessing the picture format options
`Placement`	`XlPlacement`	Set/Get how the object will react with the cells around the shape
`Rotation`	`Single`	Set/Get the degrees rotation of the shape
`Script`	`Script`	Read-only. Returns the VBScript associated with the shape
`Shadow`	`Shadow Format`	Read-only. An object accessing the shadow properties
`TextEffect`	`TextEffect Format`	Read-only. An object accessing the text effect properties
`TextFrame`	`TextFrame`	Read-only. An object accessing the text frame properties
`ThreeD`	`ThreeD Format`	Read-only. An object accessing the 3D effect formatting properties

Continues

Name	Returns	Description
Top	Single	Set/Get the vertical position of the shape
TopLeftCell	Range	Read-only. Returns the single cell range that describes the cell over the upper-left corner of the shape
Type	MsoShape Type	Read-only. Returns the type of shape
VerticalFlip	MsoTriState	Read-only. Returns whether the shape has been vertically flipped
Vertices	Variant	Read-only. Returns a series of coordinate pairs describing the Freeform's vertices
Visible	MsoTriState	Set/Get whether the shape is visible
Width	Single	Read-only. Returns the type of shape
ZOrder Position	Long	Read-only. Returns where the shape is in the zorder of the collection (for example, front, back)

Shape Methods

Name	Returns	Parameters	Description
Apply			Activates the shape
Copy			Copies the shape to the clipboard
CopyPicture		[Appearance As XLPicture Appearance], [Format As XlCopyPicture Format]	Copies the range into the clipboard as a picture. The Appearance parameter can be used to specify whether the picture is copied as it looks on the screen or when printed. The Format parameter can specify the type of picture that will be put into the clipboard
Cut			Cuts the shape and places it in the clipboard
Delete			Deletes the shape
Duplicate	Shape		Duplicates the shape returning the new shape
Flip		FlipCmd As MsoFlipCmd	Flips the shape using the FlipCmd parameter

Name	Returns	Parameters	Description
`IncrementLeft`		`Increment As Single`	Moves the shape horizontally
`IncrementRotation`		`Increment As Single`	Rotates the shape using the `Increment` parameter as degrees
`IncrementTop`		`Increment As Single`	Moves the shape vertically
`PickUp`			Copies the format of the current shape so another shape can then apply the formats
`RerouteConnections`			Optimizes the route of the current connector shape connected between two shapes. Also, this method may be used to optimize all the routes of connectors connected to the current shape
`ScaleHeight`		`Factor As Single, RelativeToOriginalSize As MsoTriState, [Scale]`	Scales the height of the shape by the `Factor` parameter
`ScaleWidth`		`Factor As Single, RelativeToOriginalSize As MsoTriState, [Scale]`	Scales the width of the shape by the `Factor` parameter
`Select`		`[Replace]`	Selects the shape in the document
`SetShapesDefaultProperties`			Sets the formatting of the current shape as a default shape in Word
`Ungroup`	`ShapeRange`		Breaks apart the shapes that make up the `Shape` object
`ZOrder`		`ZOrderCmd As MsoZOrderCmd`	Changes the order of the shape object in the collection

Example: Shape Object and the Shapes Collection

The Shape object is a generic container object for other object types. Examples of using the Shapes collection and Shape object are included under the specific objects.

ShapeNode Object and the ShapeNodes Collection

The ShapeNodes collection has the list of nodes and curved segments that make up a free-form shape. The ShapeNode object specifies a single node or curved segment that makes up a free-form shape. The Nodes property of the Shape object is used to access the ShapeNodes collection.

The ShapeNodes collection has a few methods besides the typical collection attributes listed in the following table.

ShapeNodes Collection Properties and Methods

Name	Returns	Description
Count	Integer	Read-only. Returns the number of ShapeNode objects in the collection
Delete		Method. Parameters: Index As Integer. Deletes the node specified by the Index
Insert		Method. Parameters: Index As Integer, SegmentType As MsoSegmentType, EditingType As MsoEditingType, X1 As Single, Y1 As Single, X2 As Single, Y2 As Single, X3 As Single, Y3 As Single. Inserts a node or curved segment in the Nodes collection
SetEditing Type		Method. Parameters: Index As Integer, EditingType As MsoEditingType. Sets the editing type for a node
SetPosition		Method. Parameters: Index As Integer, X1 As Single, Y1 As Single. Moves the specified node
SetSegment Type		Method. Parameters: Index As Integer, SegmentType As MsoSegmentType. Changes the segment type following the node

ShapeNode Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ShapeNode Properties

Name	Returns	Description
`EditingType`	`MsoEditing Type`	Read-only. Returns the editing type for the node
`Points`	`Variant`	Read-only. Returns the positional coordinate pair
`SegmentType`	`MsoSegment Type`	Read-only. Returns the type of segment following the node

Example: ShapeNode Object and the ShapeNodes Collection

```
Sub ToggleArch()
   Dim oShp As Shape
   Dim oSN As ShapeNodes
   Set oShp = ActiveSheet.Shapes(1)
   'Is the Shape a freeform?
   If oShp.Type = msoFreeform Then

      'Yes, so get its nodes
      Set oSN = oShp.Nodes
      'Toggle segment 3 between a line and a curve
      If oSN.Item(3).SegmentType = msoSegmentCurve Then
         oSN.SetSegmentType 3, msoSegmentLine
      Else
         oSN.SetSegmentType 3, msoSegmentCurve
      End If
   End If
End Sub
```

ShapeRange Collection

The `ShapeRange` collection holds a collection of `Shape` objects for a certain range or selection in a document. Possible parent items are the `Range` and the `Selection` object. The `ShapeRange` collection has many properties and methods besides the typical collection attributes. These items are listed next.

However, some operations will cause an error if performed on a `ShapeRange` collection with multiple shapes.

ShapeRange Properties

Name	Returns	Description
`Adjustments`	`Adjustments`	Read-only. An object accessing the adjustments for a shape
`Alternative Text`	`String`	Set/Get the alternative text to appear if the image is not loaded. Used with a Web page

Continues

Name	Returns	Description
AutoShape Type	MsoAuto ShapeType	Set/Get the type of AutoShape used
BlackWhite Mode	MsoBlack WhiteMode	Property used for compatibility to other drawing packages only. Does not do anything
Callout	Callout Format	Read-only. An object accessing the callout properties of the shape
Child	MsoTriState	Read-only. Returns whether the specified shape is a child shape, or if all shapes in a shape range are child shapes of the same parent
Connection SiteCount	Long	Read-only. Returns the number of potential connection points (sites) on the shape for a connector
Connector	MsoTriState	Read-only. Returns whether the shape is a connector
Connector Format	Connector Format	Read-only. Returns an object containing formatting options for a connector shape. Shape must be a connector shape
Count	Long	Read-only. Returns the number of shapes in the collection
Diagram	Diagram	Read-only. Returns a Diagram object
DiagramNode	DiagramNode	Read-only. Returns a node in the diagram
Fill	FillFormat	Read-only. An object accessing the fill properties of the shape
GroupItems	GroupShapes	Read-only. Returns the shapes that make up the current shape
HasDiagram	MsoTriState	Read-only. Returns whether a shape or shape range contains a diagram
HasDiagram Node	MsoTriState	Read-only. Returns whether a diagram node exists in a given shape or shape range
Height	Single	Set/Get the height of the shape
Horizontal Flip	MsoTriState	Read-only. Returns whether the shape has been flipped
ID	Long	Read-only. Returns the type for the specified object
Left	Single	Set/Get the horizontal position of the shape
Line	LineFormat	Read-only. An object accessing the line formatting of the shape
LockAspect Ratio	MsoTriState	Set/Get whether the dimensional proportions of the shape are kept when the shape is resized
Name	String	Set/Get the name of the shape

Name	Returns	Description
`Nodes`	`ShapeNodes`	Read-only. Returns the nodes associated with the shape
`ParentGroup`	`Shape`	Read-only. Returns the common parent shape of a child shape or a range of child shapes
`Picture Format`	`Picture Format`	Read-only. An object accessing the picture format options
`Rotation`	`Single`	Set/Get the degrees rotation of the shape
`Shadow`	`Shadow Format`	Read-only. An object accessing the shadow properties
`TextEffect`	`TextEffect Format`	Read-only. An object accessing the text effect properties
`TextFrame`	`TextFrame`	Read-only. An object accessing the text frame properties
`ThreeD`	`ThreeD Format`	Read-only. An object accessing the 3D effect formatting properties
`Top`	`Single`	Set/Get the vertical position of the shape
`Type`	`MsoShape Type`	Read-only. Returns the type of shape
`VerticalFlip`	`MsoTriState`	Read-only. Returns whether the shape has been vertically flipped
`Vertices`	`Variant`	Read-only. Returns a series of coordinate pairs describing the `Freeform's` vertices
`Visible`	`MsoTriState`	Set/Get whether the shape is visible
`Width`	`Single`	Set/Get the width of the shape
`ZOrderPosition`	`Long`	Read-only. Changes the order of the object in the collection

ShapeRange Methods

Name	Returns	Parameters	Description
`Align`		`AlignCmd As MsoAlignCmd, RelativeTo As MsoTriState`	Aligns the shapes in the collection to the alignment properties set by the parameters
`Apply`			Applies the formatting that was set by the `PickUp` method

Continues

Name	Returns	Parameters	Description
Delete			Deletes the shape
Distribute		DistributeCmd As MsoDistributeCmd, RelativeTo As MsoTriState	Distributes the shapes in the collection evenly either horizontally or vertically
Duplicate	ShapeRange		Duplicates the shape and returns a new ShapeRange
Flip		FlipCmd As MsoFlipCmd	Flips the shape using the FlipCmd parameter
Group	Shape		Groups the shapes in the collection
IncrementLeft		Increment As Single	Moves the shape horizontally
IncrementRotation		Increment As Single	Rotates the shape using the Increment parameter as degrees
IncrementTop		Increment As Single	Moves the shape vertically
PickUp			Copies the format of the current shape so another shape can then apply the formats
Regroup	Shape		Regroups any previously grouped shapes
RerouteConnections			Optimizes the route of the current connector shape connected between two shapes. Also, this method may be used to optimize all the routes of connectors connected to the current shape
ScaleHeight		Factor As Single, RelativeToOriginalSize As MsoTriState, [Scale]	Scales the height of the shape by the Factor parameter
ScaleWidth		Factor As Single, RelativeToOriginalSize As MsoTriState, [Scale]	Scales the width of the shape by the Factor parameter
Select		[Replace]	Selects the shape in the document

Name	Returns	Parameters	Description
SetShapes Default Properties			Sets the formatting of the current shape as a default shape in Word
Ungroup	ShapeRange		Breaks apart the shapes that make up the Shape object
ZOrder		ZOrderCmd As MsoZOrderCmd	Changes the order of the shape object in the collection

Example: ShapeRange Collection

```
Sub AlignShapeRanges()
   Dim oSR As ShapeRange
   'Get the first two shapes on the sheet
   Set oSR = ActiveSheet.Shapes.Range(Array(1, 2))
   'Align the left-hand edges of the shapes
   oSR.Align msoAlignLefts, msoFalse
End Sub
```

Sheets Collection

The Sheets collection contains all of the sheets in the parent workbook. Sheets in a workbook consist of chart sheets and worksheets. Therefore, the Sheets collection holds both the Chart objects and Worksheet objects associated with the parent workbook. The parent of the Sheets collection is the Workbook object.

Sheets Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Sheets Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of sheets in the collection (and therefore workbook)
HPageBreaks	HPageBreaks	Read-only. Returns a collection holding all the horizontal page breaks associated with the Sheets collection
VPageBreaks	VpageBreaks	Read-only. Returns a collection holding all the vertical page breaks associated with the worksheets of the Sheets collection
Visible	Variant	Set/Get whether the sheets in the collection are visible. Also, can set this to xlVeryHidden to not allow a user to make the sheets in the collection visible

Sheets Methods

Name	Returns	Parameters	Description
`Add`		`[Before], [After], [Count], [Type]`	Adds a sheet to the collection. You can specify where the sheet goes by choosing which sheet object will be before the new sheet object (`Before` parameter) or after the new sheet (`After` parameter). The `Count` parameter decides how many sheets are created. The `Type` parameter can be used to specify the type of sheet using the `XLSheetType` constants
`Copy`		`[Before], [After]`	Adds a new copy of the currently active sheet to the position specified at the `Before` or `After` parameters
`Delete`			Deletes all the sheets in the collection. Remember a workbook must contain at least one sheet
`FillAcross Sheets`		`RangeAs Range, Type As XlFillWith`	Copies the values in the `Range` parameter to all the other sheets at the same location. The `Type` parameter can be used to specify whether cell contents, formulas or everything is copied
`Move`		`[Before], [After]`	Moves the current sheet to the position specified by the parameters. See the `Add` method
`PrintOut`		`[From], [To], [Copies], [Preview], [Active Printer], [Print ToFile], [Collate], [PrToFile Name]`	Prints out the sheets in the collection. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the `PrintToFile` and `PrToFileName` parameters. The `From` and `To` parameters can be used to specify the range of printed pages
`PrintPreview`		`[Enable Changes]`	Displays the current sheet in the collection in a print preview mode. Set the `EnableChanges` parameter to `False` to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup
`Select`		`[Replace]`	Selects the current sheet in the collection

SmartTag Object and the SmartTags Collection Object

The SmartTag object represents an identifier that is assigned to a cell. Excel comes with many SmartTags, such as the Stock Ticker or Date recognizer, built in. However, you may also write your own SmartTags in Visual Basic. SmartTags are covered in detail in Chapter 18, but note that a degree of familiarity with XML is required to work with SmartTags.

The SmartTags collection represents all the SmartTags assigned to cells in an application.

SmartTag Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

SmartTag Properties

Name	Returns	Description
DownloadURL	String	Read-only. Returns a URL to save along with the corresponding SmartTag
Name	String	Read-only. Returns name of the SmartTag
Properties	Custom Properties	Read-only. Returns the properties for the SmartTag
Range	Range	Read-only. Returns the range to which the specified SmartTag applies
SmartTag Actions	SmartTag Actions	Read-only. Returns the type of action for the selected SmartTag
XML	String	Read-only. Returns a sample of the XML that would be passed to the action handler

SmartTag Methods

Name	Returns	Parameters	Description
Delete			Deletes the object

Example: SmartTag Object

Note: This example is repeated in Chapter 18, in the *Remove a Tag from a Range* section.

One of the problems with SmartTags is the issue of false-positives, where a cell is erroneously tagged. An example is the standard Stock Symbol SmartTag that recognizes TRUE as a valid stock symbol, even if that TRUE is a Boolean True. The following code locates all of these false-positives and removes them:

```
Sub RemoveBooleanTrue()
   Dim oSht As Worksheet
   Dim oTag As SmartTag
   'This is the URI of the StockTicker SmartTag
   Const sTicker As String = _
                   "urn:schemas-microsoft-com:office:smarttags#stockticker"
   'Loop through all the worksheets in the active workbook
   For Each oSht In ActiveWorkbook.Worksheets
      'Loop through all the tags in the sheet
      For Each oTag In oSht.SmartTags
         'Is it a StockTicker tag with a Boolean value?
         If oTag.Name = sTicker And _
                        TypeName(oTag.Range.Value) = "Boolean" Then
            'Yes, so remove this SmartTag from the cell
            oTag.Delete
         End If
      Next
   Next
End Sub
```

SmartTagAction Object and the SmartTagActions Collection Object

The `SmartTagAction` object represents an action that can be performed by a SmartTag. This may involve displaying the latest price for a stock symbol, or setting up an appointment on a certain date.

The `SmartTagActions` collection represents all of the SmartTagAction objects in the application.

SmartTagAction Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

SmartTagAction Properties

Name	Returns	Description
`ActiveXControl`	`Object`	Reference to an ActiveX control that is currently in the Document Actions task pane
`CheckboxState`	`Boolean`	Returns `True` if the check box is checked, otherwise `False` is returned
`ExpandHelp`	`Boolean`	Returns `True` if the smart document help control is currently expanded. If not, `False` is returned
`ListSelection`	`Long`	Returns a index number for an item within a List control
`Name`	`String`	Read-only. Returns name of the SmartTag
`PresentInPane`	`Boolean`	Returns a Boolean value indicating if a smart document control is currently being shown in the Document Actions task pane

Name	Returns	Description
`RadioGroup Selection`	`Long`	Returns an index number to the currently selected radio button within a RadioGroup control
`TextboxText`	`String`	Returns the text within a TextBox control

SmartTagAction Methods

Name	Returns	Parameters	Description
`Execute`			Activates the SmartTag action

SmartTagOptions Collection Object

The `SmartTagOptions` collection represents all the options of a SmartTag. For instance, it holds whether SmartTags should be embedded in the worksheet, or if they should be displayed at all.

SmartTagOptions Collection Properties

Name	Returns	Description
`DisplaySmart Tags`	`XlSmartTag DisplayMode`	Set/Get the display features for SmartTags
`EmbedSmartTags`	`Boolean`	Set/Get whether to embed SmartTags on the specified workbook

SmartTagReconizer Object and the SmartTagRecognizers Collection Object

The `SmartTagReconizer` object represents the recognizer engines that label the data in the worksheet. These can be user-defined, and as such any kind of information can be identified by SmartTags. See Chapter 18 for more details.

The `SmartTagRecognizers` collection represents all of the `SmartTagRecognizer` objects in the application.

SmartTagRecognizers Collection Properties

Name	Returns	Description
`Recognize`	`Boolean`	Set/Get whether data can be labeled with a SmartTag

SmartTagRecognizer Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

SmartTagRecognizer Properties

Name	Returns	Description
`Enabled`	`Boolean`	Set/Get whether the object is recognized
`FullName`	`String`	Read-only. Returns the name of the object, including its path on disk, as a string
`ProgId`	`String`	Read-only. Returns the programmatic identifiers for the object

SoundNote Object

The `SoundNote` object is not used in the current version of Excel. It is kept here for compatibility purposes only. The list of its methods is shown next.

SoundNote Methods

Name	Returns	Parameters
`Delete`	`Variant`	
`Import`	`Variant`	`Filename As String`
`Play`	`Variant`	
`Record`	`Variant`	

Speech Object

Represents the Speech recognition applet that comes with Office XP. This new Speech feature allows text to be read back on demand, or when you enter data on a document. For Excel, you have the option of having each cell's contents read back as they are entered on the worksheet. Use the `SpeakCellOnEnter` property of this object to enable this feature.

`Speech` is accessible through the `Application` object.

Speech Properties

Name	Returns	Description
`Direction`	`XlSpeak Direction`	Set/Get the order in which the cells will be spoken
`SpeakCellOn Enter`	`Boolean`	Set/Get whether to turn on Excel's mode where the active cell will be spoken when the *Enter* key is pressed, or when the active cell is finished being edited

Speech Methods

Name	Returns	Parameters	Description
Speak		Text As String, [SpeakAsync], [SpeakXML],[Purge]	The Text is spoken by Excel. If Purge is True the current speech will be terminated and any buffered text to be purged before Text is spoken

Example: Speech Object

The following routine reads off the expense totals for all items that are greater than a limit set in another cell on the sheet:

```
Sub ReadHighExpenses()

    Dim lTotal As Long
    Dim lLimit As Long
    Dim rng As Range

    'Grab the limitation amount
    lLimit = wksAllowEditRange.Range("Limit")

    'Loop through the expense totals
    For Each rng In wksAllowEditRange.Range("Expenses")
        'Store the current expense total
        lTotal = rng.Offset(0, 5).Value

        'If the current total is greater than
        ' the limit, read it off
        If lTotal > lLimit Then
            Application.Speech.Speak rng.Text
            Application.Speech.Speak lTotal
        End If
    Next rng

End Sub
```

SpellingOptions Collection Object

Represents the spelling options in Excel. These options can be found on the Spelling tab of the Tools ➪ Options command and are accessed through the Application object. Hence this object is accessible through the Application object.

SpellingOptions Collection Properties

Name	Returns	Description
ArabicModes	XlArabic Modes	Set/Get the mode for the Arabic spelling checker
DictLang	Long	Set/Get the dictionary language used by Excel for checking spelling

Continues

Name	Returns	Description
GermanPostReform	Boolean	Set/Get whether to check the spelling of words using the German post-reform rules
HebrewModes	XlHebrewModes	Set/Get the mode for the Hebrew spelling checker
IgnoreCaps	Boolean	Set/Get whether to check for uppercase words, or lowercase words during spelling checks
IgnoreFileNames	Boolean	Set/Get whether to check for Internet and file addresses during spelling checks
IgnoreMixedDigits	Boolean	Set/Get whether to check for mixed digits during spelling checks
KoreanCombineAux	Boolean	Set/Get whether to combine Korean auxiliary verbs and adjectives when using the spelling checker
KoreanProcessCompound	Boolean	Set/Get whether to process Korean compound nouns when using the spelling checker
KoreanUseAutoChangeList	Boolean	Set/Get whether to use the auto-change list for Korean words when using the spelling checker
SuggestMainOnly	Boolean	Set/Get whether to suggest words from only the main dictionary, for using the spelling checker
UserDict	String	Set/Get whether to create a custom dictionary to which new words can be added, when performing spelling checks

Example: SpellingOptions Collection Object

The following routine sets some spelling options and creates a new custom dictionary where added words during a spellcheck can be found:

```
Sub SetSpellingOptions()

    'This one is as simple as it gets
    With Application.SpellingOptions
        .IgnoreCaps = True
        .IgnoreFileNames = True
        .IgnoreMixedDigits = True
        .SuggestMainOnly = False

        'This property creates a custom dictionary
        ' called Wrox.dic, which can be found and directly edited
        ' in C:\WINDOWS\Application Data\Microsoft\Proof.
        'Added words during a spellcheck will now appear
        ' in this custom dictionary.
    End With
```

Style Object and the Styles Collection

The Styles collection holds the list of user-defined and built-in formatting styles, such as Currency and Normal, in a workbook or range. Each Style object represents formatting attributes associated with the parent object. There are some Excel built-in Style objects, such as Currency. Also, new styles can be created. Possible parents of the Styles collection are the Range and Workbook objects.

Styles can be accessed by the end user using the Style dialog box from the Format ➪ Style menu.

The Styles collection has three extra attributes besides the typical collection ones. The Count property returns the number of Style objects in the collection. The Add method uses the Name parameter to add a new style to the collection. The BasedOn parameter of the Add method can be used to specify a range that the new style will be based on. The Merge method merges the styles in the workbook specified by the Workbook parameter into the current parent workbook.

Style Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Style Properties

Name	Returns	Description
AddIndent	Boolean	Set/Get whether text associated with the style is automatically indented if the text alignment in a cell is set to equally distribute
Borders	Borders	Read-only. Returns the collection of borders associated with the style. Each border side can be accessed individually
BuiltIn	Boolean	Read-only. Returns whether the style is built-in
Font	Font	Read-only. Returns an object containing Font options for the associated style
FormulaHidden	Boolean	Set/Get whether formulas associated with the style will be hidden if the workbook/worksheet is protected
Horizontal Alignment	XlHAlign	Set/Get how the cells associated with the style are horizontally aligned. Use the XlHAlign constants
IncludeAlignment	Boolean	Set/Get whether the styles include properties associated with alignment (that is, AddIndent, HorizontalAlignment, Vertical Alignment, WrapText, and Orientation)
IncludeBorder	Boolean	Set/Get whether border attributes are included with the style (that is, Color, ColorIndex, LineStyle, and Weight)

Continues

Name	Returns	Description
IncludeFont	Boolean	Set/Get whether font attributes are included in the style (that is, Background, Bold, Color, ColorIndex, FontStyle, Italic, Name, OutlineFont, Shadow, Size, Strikethrough, Subscript, Superscript, and Underline)
IncludeNumber	Boolean	Set/Get whether the NumberFormat property is included in the style
Include Pattern	Boolean	Set/Get whether interior pattern related properties are included in the style (that is, Color, ColorIndex, InvertIfNegative, Pattern, PatternColor, and PatternColorIndex)
Include Protection	Boolean	Set/Get whether the locking related properties are included with the style (that is, FormulaHidden and Locked)
IndentLevel	Long	Set/Get the indent level for the style
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the style (for example, interior color)
Locked	Boolean	Set/Get whether the style properties can be changed if the workbook is locked
MergeCells	Variant	Set/Get whether the current style contains merged cells
Name	String	Read-only. Returns the name of the style
NameLocal	String	Read-only. Returns the name of the style in the language of the user's computer
NumberFormat	String	Set/Get the number format associated with the style
NumberFormat Local	String	Set/Get the number format associated with the style in the language of the end user
Orientation	Xl Orientation	Set/Get the text orientation for the cell text associated with the style. A value from –90 to 90 degrees can be specified or an XlOrientation constant
ReadingOrder	Long	Set/Get whether the text associated with the style is from right-to-left (xlRTL), left-to-right (xlLTR), or context sensitive (xlContext)
ShrinkToFit	Boolean	Set/Get whether the cell text associated with the style will automatically shrink to fit the column width

Name	Returns	Description
Value	String	Read-only. Returns the name of the style
Vertical Alignment	XlVAlign	Set/Get how the cells associated with the style are vertically aligned. Use the XlVAlign constants
WrapText	Boolean	Set/Get whether cell text wraps in cells associated with the style

Style Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the style from the collection

Example: Style Object and the Styles Collection

```
Sub UpdateStyles()
   Dim oStyle As Style
   Set oStyle = ActiveWorkbook.Styles("Editing")
   'Update the Editing style to be unlocked with a default background
   With oStyle
      .IncludePatterns = True
      .IncludeProtection = True
      .Locked = False
      .Interior.Pattern = xlNone
   End With
End Sub
```

Tab Object

Represents the Sheet tab at the bottom of an Excel chart sheet or worksheet. Excel 2003 now allows you to customize the sheet's tab color by using either the Color or ColorIndex properties of this object.

Note that when setting ColorIndex property of this object to xlColorIndexAutomatic (which appears on the AutoComplete list for the property), an error will occur.

Tab Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Tab Properties

Name	Returns	Description
Color	Variant	Set/Get the primary color of the Tab object. Use the RGB function to create a color value
ColorIndex	XlColor Index	Set/Get the color of the interior

Example: Tab Object

The following routine changes the tab color for all budget worksheet in a workbook based on a setting in a custom property for each worksheet:

```
Sub ColorBudgetTabs()
    Dim bBudget As Boolean
    Dim oCustomProp As CustomProperty
    Dim oCustomProps As CustomProperties
    Dim wks As Worksheet

    'Loop through each worksheet in this workbook
    For Each wks In ThisWorkbook.Worksheets
        'Loop through all of the custom properties
        ' for the current worksheet until the
        ' "IsBudget" proeprty name is found
        For Each oCustomProp In wks.CustomProperties
            If oCustomProp.Name = "IsBudget" Then
                'Grab its value and exit the loop
                bBudget = CBool(oCustomProp.Value)
                Exit For
            End If
        Next oCustomProp

        'Use the value in the custom property to determine
        ' whether the tab should be colored.
        If bBudget Then wks.Tab.ColorIndex = 20 'Light blue

    Next wks

End Sub
```

TextEffectFormat Object

The TextEffectFormat object contains all the properties and methods associated with WordArt objects. The parent object of the TextEffectFormat is always the Shape object.

TextEffectFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

TextEffectFormat Properties

Name	Returns	Description
Alignment	MsoText Effect Alignment	Set/Get the alignment of the WordArt
FontBold	MsoTriState	Set/Get whether the WordArt is bold
FontItalic	MsoTriState	Set/Get whether the WordArt is italic
FontName	String	Set/Get the font used in the WordArt

Name	Returns	Description
FontSize	Single	Set/Get the font size in the WordArt
KernedPairs	MsoTriState	Set/Get whether the characters are kerned in the WordArt
NormalizedHeight	MsoTriState	Set/Get whether both the uppercase and lowercase characters are the same height
PresetShape	MsoPresetTextEffectShape	Set/Get the shape of the WordArt
PresetTextEffect	MsoPresetTextEffect	Set/Get the effect associated with the WordArt
RotatedChars	MsoTriState	Set/Get whether the WordArt has been rotated by 90 degrees
Text	String	Set/Get the text in the WordArt
Tracking	Single	Set/Get the spacing ratio between characters

TextEffectFormat Methods

Name	Returns	Parameters	Description
ToggleVerticalText			Toggles the text from vertical to horizontal and back

Example: TextEffectFormat Object

```
Sub FormatTextArt()
   Dim oTEF As TextEffectFormat
   Dim oShp As Shape
   Set oShp = ActiveSheet.Shapes(1)
   If oShp.Type = msoTextEffect Then
      Set oTEF = oShp.TextEffect

      With oTEF
         .FontName = "Times New Roman"
         .FontBold = True
         .PresetTextEffect = msoTextEffect14
         .Text = "Hello World!"
      End With
   End If
End Sub
```

TextFrame Object

The TextFrame object contains the properties and methods that can manipulate text-frame shapes. Possible parent objects of the TextFrame object are the Shape and ShapeRange objects.

TextFrame Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

TextFrame Properties

Name	Returns	Description
`AutoMargins`	`Boolean`	Set/Get whether Excel will calculate the margins of the text frame automatically. Set this property to `False` to use the `MarginLeft`, `MarginRight`, `MarginTop`, and `MarginBottom` properties
`AutoSize`	`Boolean`	Set/Get whether the size of the text frame changes to match the text inside
`Horizontal Alignment`	`XlHAlign`	Set/Get how the text frame is horizontally aligned. Use the `XLHAlign` constants
`MarginBottom`	`Single`	Set/Get the bottom spacing in a text frame
`MarginLeft`	`Single`	Set/Get the left spacing in a text frame
`MarginRight`	`Single`	Set/Get the right spacing in a text frame
`MarginTop`	`Single`	Set/Get the top spacing in a text frame
`Orientation`	`MsoText Orientation`	Set/Get the orientation of the text in the text frame
`ReadingOrder`	`Long`	Set/Get whether the text in the frame is read from right-to-left (`xlRTL`), left-to-right (`xlLTR`), or context sensitive (`xlContext`)
`Vertical Alignment`	`XlVAlign`	Set/Get how the text frame is vertically aligned. Use the `XlVAlign` constants

TextFrame Methods

Name	Returns	Parameters	Description

`Characters`	`Characters`	`[Start],`
		`[Length]`
			Returns an object containing all the characters in the text frame. Allows manipulation on a character-by-character basis and to retrieve only a subset of text in the frame

Example: TextFrame Object

```
Sub SetShapeAutoSized()
   Dim oTF As TextFrame
   Dim oShp As Shape
   Set oShp = ActiveSheet.Shapes(1)
   Set oTF = oShp.TextFrame
   oTF.AutoSize = True
End Sub
```

ThreeDFormat Object

The `ThreeDFormat` object contains all of the three-dimensional formatting properties of the parent `Shape` object. The `ThreeD` property of the `Shape` object is used to access the `ThreeDFormat` object.

ThreeDFormat Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

ThreeDFormat Properties

Name	Returns	Description
`Depth`	`Single`	Set/Get the "depth" of a 3D shape
`ExtrusionColor`	`ColorFormat`	Read-only. An object manipulating the color of the extrusion
`ExtrusionColorType`	`MsoExtrusionColorType`	Set/Get how the color for the extrusion is set
`Perspective`	`MsoTriState`	Set/Get whether the shape's extrusion has perspective
`PresetExtrusionDirection`	`MsoPresetExtrusionDirection`	Read-only. Returns the direction of the extrusion
`PresetLightingDirection`	`MsoPresetLightingDirection`	Set/Get the directional source of the light source
`PresetLightingSoftness`	`MsoPresetLightingSoftness`	Set/Get the softness of the light source
`PresetMaterial`	`MsoPresetMaterial`	Set/Get the surface material of the extrusion
`PresetThreeDFormat`	`MsoPresetThreeDFormat`	Read-only. Returns the preset extrusion format
`RotationX`	`Single`	Set/Get how many degrees the extrusion is rotated
`RotationY`	`Single`	Set/Get how many degrees the extrusion is rotated
`Visible`	`MsoTriState`	Set/Get whether the 3D shape is visible

ThreeDFormat Methods

Name	Returns	Parameters	Description
`IncrementRotationX`		`Increment As Single`	Changes the `RotationX` property
`IncrementRotationY`		`Increment As Single`	Changes the `RotationY` property
`ResetRotation`			Resets the `RotationX` and `RotationY` to 0
`SetExtrusionDirection`		`PresetExtrusionDirection As MsoPresetExtrusionDirection`	Changes the extrusion direction
`SetThreeD Format`		`PresetThreeDFormat As MsoPresetThreeDFormat`	Sets the preset extrusion format

Example: ThreeDFormat Object

```
Sub SetShape3D()
   Dim o3DF As ThreeDFormat
   Dim oShp As Shape
   Set oShp = ActiveSheet.Shapes(1)
   Set o3DF = oShp.ThreeD
   With o3DF
      .Depth = 10
      .SetExtrusionDirection msoExtrusionBottomRight
   End With
End Sub
```

TickLabels Object

The `TickLabels` object contains the formatting options associated with the tick-mark labels for tick marks on a chart axis. The parent of the `TickLabels` object is the `Axis` object.

TickLabels Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

TickLabels Properties

Name	Returns	Description
Alignment	Long	Set/Get the alignment of the tick labels. Use the XlHAlign constants
AutoScaleFont	Variant	Set/Get whether the font size will change automatically if the parent chart changes sizes
Depth	Long	Read-only. Returns how many levels of category tick labels are on the axis
Font	Font	Read-only. Returns an object containing Font options for the tick label text
Name	String	Read-only. Returns the name of the TickLabels object
NumberFormat	String	Set/Get the numeric formatting to use if the tick labels are numeric values or dates
NumberFormat Linked	Boolean	Set/Get whether the same numerical format used for the cells containing the chart data is used by the tick labels
NumberFormat Local	Variant	Set/Get the name of the numeric format being used by the tick labels in the language being used by the user
Offset	Long	Set/Get the percentage distance between levels of labels as compared to the axis label’s font size
Orientation	XlTickLabel Orientation	Set/Get the angle of the text for the tick labels. The value can be in degrees (from –90 to 90) or one of the XlTickLabelOrientation constants
ReadingOrder	Long	Set/Get how the text is read (from left to right or right to left). Only applicable in appropriate languages

TickLabels Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the tick labels from the axis labels
Select	Variant		Selects the tick labels on the chart

Example: TickLabels Object

```
Sub FormatTickLabels()
   Dim oTL As TickLabels
   Set oTL = Charts(1).Axes(xlValue).TickLabels
   With oTL
      .NumberFormat = "#,##0"
      .Font.Size = 12
   End With
End Sub
```

TreeviewControl Object

The `TreeviewControl` object allows manipulation of the hierarchical member-selection of a cube field. This object is usually used by macro recordings and not when building VBA code. The parent of the `TreeviewControl` object is the `CubeField` object.

TreeviewControl Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

TreeviewControl Properties

Name	Returns	Description
`Drilled`	`Variant`	Set/Get a string array describing the drilled status of the members of the parent cube field
`Hidden`	`Variant`	Set/Get the hidden status of the members in a cube field

Trendline Object and the Trendlines Collection

The `Trendlines` collection holds the collection of trendlines in a chart. Each `TrendLine` object describes a trendline on a chart of a particular series. `Trendlines` are used to graphically show trends in the data and help predict future values. The parent of the `Trendlines` collection is the `Series` object.

The `Trendlines` collection has one property and one method besides the typical collection attributes. The `Count` property returns the number of `TrendLine` objects in the collection. The `Add` method adds a trendline to the current chart. The `Add` method has a `Type`, `Order`, `Period`, `Forward`, `Backward`, `Intercept`, `DisplayEquation`, `DispayRSquared`, and `Name` parameter. See the *Trendline Properties* section for more information.

Trendline Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Trendline Properties

Name	Returns	Description
Backward	Long	Set/Get how many periods the trendline extends back
Border	Border	Read-only. Returns the border's properties around the trendline
DataLabel	DataLabel	Read-only. Returns an object to manipulate the trendline's data label
Display Equation	Boolean	Set/Get whether the equation used for the trendline is displayed on the chart
Display RSquared	Boolean	Set/Get whether the R-squared value for the trendline is displayed on the chart
Forward	Long	Set/Get how many periods the trendline extends forward
Index	Long	Read-only. Returns the spot in the collection that the current object is
Intercept	Double	Set/Get at which point the trendline crosses the value (y) axis
InterceptIs Auto	Boolean	Set/Get whether the point the trendline crosses the value axis is automatically calculated with regression
Name	String	Set/Get the name of the Trendline object
NameIsAuto	Boolean	Set/Get whether Excel automatically chooses the trendline name
Order	Long	Set/Get the order of a polynomial trendline. The Type property must be xlPolynomial
Period	Long	Set/Get what the period is for the moving-average trendline
Type	XlTrendline Type	Set/Get the type of the trendline (for example, xlExponential, xlLinear, etc.)

Trendline Methods

Name	Returns	Parameters	Description
ClearFormats	Variant		Clears any formatting made on the trendlines
Delete	Variant		Deletes the trendlines
Select	Variant		Selects the trendlines on the chart

Example: Trendline Object and the Trendlines Collection

```
Sub AddTrendLine()
   Dim oSer As Series
   Dim oTL As Trendline
   Set oSer = Charts(1).SeriesCollection(1)
   Set oTL = oSer.Trendlines.Add(xlLinear)
   With oTL
      .DisplayEquation = True
      .DisplayRSquared = True
   End With
End Sub
```

UpBars Object

The UpBars object contains formatting options for up bars on a chart. The parent of the UpBars object is the ChartGroup object. To see if this object exists, use the HasUpDownBars property of the ChartGroup object.

UpBars Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

UpBars Properties

Name	Returns	Description
Border	Border	Read-only. Returns the border's properties around the up bars
Fill	ChartFill Format	Read-only. Returns an object containing fill formatting options for the up bars of a chart
Interior	Interior	Read-only. Returns an object containing options to format the inside area of the up bars (for example, interior color)
Name	String	Read-only. Returns the name of the up bars

UpBars Methods

Name	Returns	Parameters	Description
Delete	Variant		Deletes the up bars
Select	Variant		Selects the up bars in the chart

Example: UpBars Object

```
Sub AddAndFormatUpBars()
   Dim oUpBars As UpBars
   'Add Up/Down bars to the chart
   Charts(1).ChartGroups(1).HasUpDownBars = True
   'Get the collection of UpBars
   Set oUpBars = Charts(1).ChartGroups(1).UpBars
   'Format the up bars
   With oUpBars
      .Interior.ColorIndex = 3
      .Interior.Pattern = xlSolid
   End With
End Sub
```

UsedObjects Collection Object

Represents the total amount of objects currently being used in all open workbooks. Used objects can be worksheets, chart sheets, the workbook itself, and any ActiveX controls placed on worksheets. This object can be referenced through the `Application` object.

Note that in addition to the common collection properties defined earlier, `UsedObjects` has the `Item` and `Count` properties.

Example: UsedObjects Collection Object

The following routine lists all of the parent objects of the `UsedObjects` collection:

```
Sub CountUsedObjects()
    Dim lCount As Long
    Dim oUsedObjs As UsedObjects
    'Turn off the screen
    Application.ScreenUpdating = False

    'Store the used object collection
    Set oUsedObjs = Application.UsedObjects

    'Clear the old list
    wksUsedObjects.UsedRange.Offset(1, 0).Resize(, 1).ClearContents
    'Loop through and list the parents of all of the objects
    'Cannot seem to grab the name/caption/... of the object itself
    For lCount = 1 To oUsedObjs.Count
        wksUsedObjects.Range("ListStart").Cells(lCount, 1) =
oUsedObjs.Item(lCount).Parent.Name
    Next lCount
End Sub
```

UserAccess Collection Object

Represents one user within a possible group of users who have permission to access a range specified by the `AllowEditRange` object. You can refer to a user by using the `Item` property of the

UserAccessList object. Once referenced, you use the properties of this object to change the user's settings.

UserAccess Collection Properties

Name	Returns	Description
AllowEdit	Boolean	Set/Get whether the user is allowed access to the specified range on a protected worksheet
Name	String	Read-only. Returns the name of the UserAccess object

UserAccess Collection Methods

Name	Returns	Parameters	Description
Delete			Deletes the object

UserAccessList Collection Object

Represents a list of users who have access to a protected range on a worksheet. This object can be accessed via the AllowEditRange object after it's been created. Use the Add method of this object to add a user to the list, which contains an argument that determines whether or not they need a password to access the range.

Note that the password is set using the ChangePassword method of the AllowEditRange object. This means that all of the users for an AllowEditRange use the same password. Note that this collection only has Count and Item properties.

UserAccessList Methods

Name	Returns	Parameters	Description
Add	UserAccess	Name As String, AllowEdit As Boolean	Adds a user access list to the collection. Name is the name of the list, and if AllowEdit is True users on the access list are allowed to edit the editable ranges on a protected worksheet
DeleteAll			Removes all users associated with access to a protected range on a worksheet

Example: UserAccessList Object

The following routine loops through all of the AllowEditRange objects on a specified worksheet and removes all of the users except for the range pcNetSales:

```
Sub DeleteAllUsers()

    Dim oAllowRange As AllowEditRange

    'Loop through all of the AllowEditRange objects on the
    ' specified worksheet
    For Each oAllowRange In wksAllowEditRange.Protection.AllowEditRanges
        'Remove all names from all AllowEditRanges
        ' except for the range whose AllowEditRange Title
        ' is pcNetSales
        If oAllowRange.Title <> "pcNetSales" Then
            oAllowRange.Users.DeleteAll
        End If
    Next oAllowRange

End Sub
```

Validation Object

The Validation object contains properties and methods to represent validation for a range in a worksheet. The Range object is the parent of the Validation object.

Validation Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Validation Properties

Name	Returns	Description
AlertStyle	Long	Read-only. Returns how the user will be alerted if the range includes invalid data. Uses the XlDVAlertStyle constants
ErrorMessage	String	Set/Get the error message to show for data validation
ErrorTitle	String	Set/Get what the title is for the error data validation dialog box
Formula1	String	Read-only. Returns the value, cell reference, or formula used for data validation
Formula2	String	Read-only. Returns the second part of the value, cell reference, or formula used for data validation. The Operator property must be xlBetween or xlNotBetween

Continues

Name	Returns	Description
IgnoreBlank	Boolean	Set/Get whether a blank cell is always considered valid
IMEMode	Long	Set/Get how the Japanese input rules are described. Use the XlIMEMode constants
InCell Dropdown	Boolean	Set/Get whether a drop-down list of valid values is displayed in the parent range. Used when the Type property is xlValidateList
InputMessage	String	Set/Get the validation input message to prompt the user for valid data
InputTitle	String	Set/Get what the title is for the input data validation dialog box
Operator	Long	Read-only. Returns the operator describing how Formula1 and Formula2 are used for validation. Uses the XlFormatConditionOperator constants
ShowError	Boolean	Set/Get whether the error message will be displayed when invalid data is entered in the parent range
ShowInput	Boolean	Set/Get whether the input message will be displayed when the user chooses one of the cells in the parent range
Type	Long	Read-only. Returns the data validation type for the range. The XlDVType constants can be used (for example, xlValidateDecimal, xlValidateTime)
Value	Boolean	Read-only. Returns if the validation is fulfilled for the range

Validation Methods

Name	Returns	Parameters	Description
Add		Type As XlDVType, [Alert Style], [Operator], [Formula1], [Formula2]	Adds data validation to the parent range. The validation type (Type parameter) must be specified. The type of validation alert (AlertStyle) can be specified with the XlDVAlertStyle constants. The Operator parameter uses the XlFormatCondition Operator to pick the type of operator to use. The Formula1 and Formula2 parameters pick the data validation formula
Delete			Deletes the Validation method for the range

Name	Returns	Parameters	Description
Modify		[Type], [AlertStyle], [Operator], [Formula1], [Formula2]	Modifies the properties associated with the Validation. See the properties of the Validation object for a description of the parameters

Example: Validation Object

```
Sub AddValidation()
   Dim oValid As Validation
   Set oValid = Selection.Validation
   With oValid
      .Delete
      .Add Type:=xlValidateWholeNumber, AlertStyle:=xlValidAlertStop, _
        Operator:=xlBetween, Formula1:="10", Formula2:="20"
      .ShowInput = False
      .ShowError = True
      .ErrorTitle = "Error"
      .ErrorMessage = "Number must be between 10 and 20"
   End With
End Sub
```

VPageBreak Object and the VPageBreaks Collection

The VPageBreaks collection contains all of the vertical page breaks in the printable area of the parent object. Each VPageBreak object represents a single vertical page break for the printable area of the parent object. Possible parents of the VPageBreaks collection are the WorkSheet and the Chart objects.

The VPageBreaks collection contains one property and one method besides the typical collection attributes. The Count property returns the number of VPageBreak objects in the collection. The Add method is used to add a VPageBreak object to the collection (and vertical page break to the sheet). The Add method has a Before parameter to specify the range to the right of where the vertical page break will be added.

VPageBreak Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

VPageBreak Properties

Name	Returns	Description
Extent	XlPageBreak Extent	Read-only. Returns whether the vertical page break is full screen or only for the print area
Location	Range	Set/Get the cell where the vertical page break is located. The left edge of the cell is the location of the page break
Type	XlPageBreak	Set/Get whether the page break is automatic or manually set

VPageBreak Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the page break
`DragOff`		`Direction As XlDirection, RegionIndex As Long`	Drags the page break out of the printable area. The `Direction` parameter specifies the direction the page break is dragged. The `RegionIndex` parameter specifies which print region the page break is being dragged out of

Example: VPageBreak Object and the VPageBreaks Collection

```
Sub AddVPageBreaks()
   Dim oCell As Range
   'Loop through all the cells in the first column of the sheet
   For Each oCell In ActiveSheet.UsedRange.Rows(1).Cells
      'If the font size is 16, add a page break to the left of the cell
      If oCell.Font.Size = 16 Then
         ActiveSheet.VPageBreaks.Add oCell
      End If
   Next
End Sub
```

Walls Object

The `Walls` object contains formatting options for all the walls of a 3D chart. The walls of a 3D chart cannot be accessed individually. The parent of the `Walls` object is the `Chart` object.

Walls Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Walls Properties

Name	Returns	Description
`Border`	`Border`	Read-only. Returns the border's properties around the walls of the 3D chart
`Fill`	`ChartFill Format`	Read-only. Returns an object containing fill formatting options for the walls of a 3D chart
`Interior`	`Interior`	Read-only. Returns an object containing options to format the inside area of the walls (for example, interior color)
`Name`	`String`	Read-only. Returns the name of the `Walls` object

Name	Returns	Description
`PictureType`	`Variant`	Set/Get how an associated picture is displayed on the walls of the 3D chart (for example, stretched, tiled). Use the `XlPictureType` constants
`PictureUnit`	`Variant`	Set/Get how many units a picture represents if the `PictureType` property is set to `xlScale`

Walls Methods

Name	Returns	Parameters	Description
`ClearFormats`	`Variant`		Clears the formatting made on the `Walls` object
`Paste`			Deletes the `Walls` object
`Select`	`Variant`		Selects the walls on the parent chart

Example: Walls Object

```
Sub FormatWalls()
   Dim oWall As Walls
   Set oWall = Charts(1).Walls
   With oWall
      .Fill.PresetTextured msoTextureCork
      .Fill.Visible = True
   End With
End Sub
```

Watch Object and the Watches Collection Object

The `Watch` object represents one Watch in the Watch window (View ➪ Toolbars ➪ Watch Window). Each Watch can be a cell or cell range you need to keep track of as other data on the worksheet changes. A `Watch` object is an auditing tool similar to the watches you can create in the VBE. Watches do just that, they keep track of a cell or cell range, allowing you to study changes to those cells when other data on the worksheet changes.

The `Watches` collection contains all the `Watch` objects that have been set in the application.

Watches Collection Methods

Name	Returns	Parameters	Description
`Add`	`Watch`	`Source As Variant`	Adds a range which is tracked when the worksheet is recalculated
`Delete`			Deletes the object

Watch Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Watch Properties

Name	Returns	Description
`Source`	`Variant`	Read-only. Returns the unique name that identifies items that have a `SourceType` property value of `xlSourceRange`, `xlSourceChart`, `xlSourcePrintArea`, `xlSourceAutoFilter`, `xlSourcePivotTable`, or `xlSourceQuery`

Watch Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the object

Example: Watch Object

The following routine prompts the user for a range, then loops through each cell in the range and adds it to the Watch Window. It then displays the Watch Window:

```
Sub AddWatches()
    Dim oWatch As Watch
    Dim rng As Range
    Dim rngWatches As Range

    'Prompt the user for a range
    'Supress the error if they cancel
    On Error Resume Next
        Set rngWatches = Application.InputBox(_
          "Please select a cell or cell range to watch", "Add Watch", , , , , 8)
    On Error GoTo 0

    'If they selected a range
    If Not rngWatches Is Nothing Then
        'Loop through each cell and
        ' add it to the watch list
        For Each rng In rngWatches
            Application.Watches.Add rng
        Next rng
    End If
    'View the watch window based on their answer
    Application.CommandBars("Watch Window").Visible = (Not rngWatches
Is Nothing)

End Sub
```

WebOptions Object

The WebOptions object contains attributes associated with opening or saving Web pages. The parent of the WebOptions object is the Workbook object. The properties set in the WebOptions object override the settings of the DefaultWebOptions object.

WebOptions Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

WebOptions Properties

Name	Returns	Description
AllowPNG	Boolean	Set/Get whether Portable Network Graphics Format (PNG) is allowed as an output format. PNG is a file format for the lossless, portable, well-compressed storage of images
Download Components	Boolean	Set/Get whether Office components are downloaded to the end user's machine when viewing Excel files in a Web browser
Encoding	MsoEncoding	Set/Get the type of code page or character set to save with a document
FolderSuffix	String	Read-only. Returns what the suffix name is for the support directory created when saving an Excel document as a Web page. Language dependent
LocationOf Components	String	Set/Get the URL or path that contains the Office Web components needed to view documents in a Web browser
OrganizeIn Folder	Boolean	Set/Get whether supporting files are organized in a separate folder from the document
PixelsPer Inch	Long	Set/Get how dense graphics and table cells should be when viewed on a Web page
RelyOnCSS	Boolean	Set/Get whether Cascading Style Sheets (CSS) is used for font formatting
RelyOnVML	Boolean	Set/Get whether image files are not created when saving a document with drawn objects. Vector Markup Language is used to create the images on the fly. VML is an XML-based format for high-quality vector graphics on the Web
ScreenSize	MsoScreen Size	Set/Get the target monitor's screen size

Continues

Name	Returns	Description
TargetBrowser	MsoTargetBrowser	Set/Get the browser version
UseLongFileNames	Boolean	Set/Get whether links are updated every time the document is saved

WebOptions Methods

Name	Returns	Parameters	Description
UseDefaultFolderSuffix			Tells Excel to use its default naming scheme for creating supporting folders

Example: WebOptions Object

```
Sub SetWebOptions()
   Dim oWO As WebOptions
   Set oWO = ActiveWorkbook.WebOptions
   With oWO
      .ScreenSize = msoScreenSize800x600
      .RelyOnCSS = True
      .UseDefaultFolderSuffix
   End With
End Sub
```

Window Object and the Windows Collection

The `Windows` collection holds the list of windows used in Excel or in a workbook. Each `Window` object represents a single Excel window containing scrollbars and gridlines for the window. The parents of the `Windows` collection can be the `Application` object and the `Workbook` object.

The Windows collection has a `Count` property and an `Arrange` method besides the typical collection attributes. The `Count` property returns the number of `Window` objects in the collection. The `Arrange` method arranges the windows in the collection in the manner specified by the `ArrangeStyle` parameter. Use the `XlArrangeStyle` constants to set the `ArrangeStyle` parameter. Set the `ActiveWorkbook` parameter to `True` to arrange only the windows associated with the open workbook. Set the `SyncHorizontal` parameter or the `SyncVertical` parameter to `True` so the windows will scroll horizontally or vertically together, respectively.

Window Common Properties

The `Application`, `Creator`, and `Parent` properties are defined at the beginning of this Appendix.

Window Properties

Name	Returns	Description
`ActiveCell`	`Range`	Read-only. Returns the cell in the window where the cursor is
`ActiveChart`	`Chart`	Read-only. Returns the currently selected chart in the window. If no chart is currently selected, nothing is returned
`ActivePane`	`Pane`	Read-only. Returns the active pane in the window
`ActiveSheet`		Read-only. Returns the active sheet in the window
`Caption`	`Variant`	Set/Get the caption that appears in the window
`Display Formulas`	`Boolean`	Set/Get whether formulas are displayed in the window. Not valid in a `Chart` sheet
`Display Gridlines`	`Boolean`	Set/Get whether worksheet gridlines are displayed
`Display Headings`	`Boolean`	Set/Get whether row and column headings are displayed. Not valid in a `Chart` sheet
`Display Horizontal ScrollBar`	`Boolean`	Set/Get whether the horizontal scrollbar is displayed in the window
`Display Outline`	`Boolean`	Set/Get whether outline symbols are displayed
`Display RightToLeft`	`Boolean`	Set/Get whether the window contents are displayed from right to left. Valid only with languages that support right-to-left text
`Display Vertical ScrollBar`	`Boolean`	Set/Get whether the vertical scrollbar is displayed in the window
`Display WorkbookTabs`	`Boolean`	Set/Get whether workbook tabs are displayed
`DisplayZeros`	`Boolean`	Set/Get whether zero values are displayed. Not valid with `Chart` sheets
`EnableResize`	`Boolean`	Set/Get whether a user can resize the window
`FreezePanes`	`Boolean`	Set/Get whether split panes are frozen. Not valid with `Chart` sheets
`GridlineColor`	`Long`	Set/Get the color of the gridlines. Use the `RGB` function to create the color value

Continues

Name	Returns	Description
`GridlineColor Index`	`XlColor Index`	Set/Get the color of the gridlines. Use the `XlColorIndex` constants or an index value in the current color palette
`Height`	`Double`	Set/Get the height of the window
`Index`	`Long`	Read-only. Returns the spot in the collection where the current object is located
`Left`	`Double`	Set/Get the distance from the left edge of the client area to the window's left edge
`OnWindow`	`String`	Set/Get the name of the procedure to run whenever a window is activated
`Panes`	`Panes`	Read-only. Returns the panes that are contained in the window
`RangeSelection`	`Range`	Read-only. Returns the selected range of cells or object in the window
`ScrollColumn`	`Long`	Set/Get the column number of the left-most column in the window
`ScrollRow`	`Long`	Set/Get the row number of the top-most row in the window
`SelectedSheets`	`Sheets`	Read-only. Returns all the selected sheets in the window
`Selection`	`Object`	Read-only. Returns the selected object in the window
`Split`	`Boolean`	Set/Get whether the window is split into panes
`SplitColumn`	`Long`	Set/Get at which column number the window split is going to be located
`Split Horizontal`	`Double`	Set/Get where the horizontal split of window will be located, in points
`SplitRow`	`Long`	Set/Get at which row number the window split is going to be located
`SplitVertical`	`Double`	Set/Get where the vertical split of window will be located, in points
`SyncScrolling SideBySide`	`Boolean`	When documents are compared, setting this to `True` will allow both documents to be scrolled at the same time
`TabRatio`	`Double`	Set/Get how big a workbook's tab is as a ratio of a workbook's tab area width to the window's horizontal scrollbar width

Name	Returns	Description
`Top`	`Double`	Set/Get the distance from the top edge of the client area to the window's top edge
`Type`	`XlWindow Type`	Read-only. Returns the window type
`UsableHeight`	`Double`	Read-only. Returns the maximum height that the window can be
`UsableWidth`	`Double`	Read-only. Returns the maximum width that the window can be
`View`	`XlWindow View`	Set/Get the view in the window (for example,`xlNormalView`, `xlPageBreakPreview`)
`Visible`	`Boolean`	Set/Get whether the window is visible
`VisibleRange`	`Range`	Read-only. Returns the range of cells that are visible in the current window
`Width`	`Double`	Set/Get the width of the window
`WindowNumber`	`Long`	Read-only. Returns the number associated with a window. Typically used when the same workbook is opened twice (for example, `MyBook.xls:1` and `MyBook.xls:2`)
`WindowState`	`XlWindow State`	Set/Get the state of window: minimized, maximized, or normal
`Zoom`	`Variant`	Set/Get the percentage window zoom

Window Methods

Name	Returns	Parameters	Description
`Activate`	`Variant`		Sets focus to the window
`ActivateNext`	`Variant`		Activates the next window in the z-order
`Activate Previous`	`Variant`		Activates the previous window in the z-order
`BreakSideBySide`	`Boolean`		Disables side-by-side mode. The return value indicates if the operation was successful

Continues

Name	Returns	Parameters	Description
Close	Boolean	[SaveChanges], [Filename], [Route Workbook]	Closes the window. Set SaveChanges to True to automatically save changes in the window's workbook. If SaveChanges is False then all changes are lost. The Filename parameter can be used to specify the filename to save to. RouteWorkbook is used to automatically route the workbook onto the next recipient, if applicable
CompareSideBy SideWith	Boolean		Opens two windows in side-by-side mode
LargeScroll	Variant	[Down],[Up], [ToRight], [ToLeft]	Causes the document to scroll a certain direction a screen-full at a time, as specified by the parameters
NewWindow	Window		Creates and returns a new window
PointsTo Screen PixelsX	Long	Points As Long	Converts the horizontal document coordinate Points parameter to screen coordinate pixels
PointsTo Screen PixelsY	Long	Points As Long	Converts the vertical document coordinate Points parameter to screen coordinate pixels
PrintOut	Variant	[From],[To], [Copies], [Preview], [Active Printer], [PrintToFile], [Collate], [PrToFile Name]	Prints out the document in the window. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the PrintToFile and PrToFileName parameters. The From and To parameters can be used to specify the range of printed pages
PrintPreview	Variant	[Enable Changes]	Displays the current workbook in the window in a print preview mode. Set the EnableChanges parameter to False to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup
RangeFrom Point	Object	x As Long, y As Long	Returns the shape or range located at the x and y coordinates. Returns nothing if there is no object at the x, y coordinates

Name	Returns	Parameters	Description
ResetPositions SideBySide			Resets the positions of two windows that are currently in side-by-side mode
ScrollInto View		Left As Long, Top As Long, Width As Long, Height As Long, [Start]	Scrolls the spot specified by the Left, Top, Width, and Height parameters to either the upper-left corner of the window (Start = True) or the lower-right corner of the window (Start = False). The Left, Top, Width, and Height parameters are specified in points
Scroll Workbook Tabs	Variant	[Sheets], [Position]	Scrolls through the number of sheets specified by the Sheets parameter or goes to the sheet specified by the position parameter (xlFirst or xlLast)
SmallScroll	Variant	[Down], [Up], [ToRight], [ToLeft]	Causes the document to scroll a certain direction one document line at a time, as specified by the parameters

Example: Window Object and the Windows Collection

```
Sub MinimiseAllWindows()
   Dim oWin As Window
   For Each oWin In Windows
      oWin.WindowState = xlMinimized
   Next
End Sub
```

Workbook Object and the Workbooks Collection

The Workbooks collection contains the list of open workbooks. A Workbook object represents a single workbook. The parent of the Workbook is the Application object.

Workbooks Properties

Name	Returns	Description
Count	Long	Read-only. Returns the number of Workbook objects in the collection
DisplayInk Comments	Boolean	Setting this to True will display ink comments

Continues

Name	Returns	Description
DocumentLibraryVersions		Returns a collection that represents all versions of shared workbooks that have versioning enabled
InactiveListBorderVisible	Boolean	Determines if the border of a list control is visible when the list control isn't active
Permission		Returns a Permission object that represents all permission settings
SharedWorkspace		A link to the document workspace in which a document is located
SmartDocument		Returns all settings for a smart document
Sync		The Sync property provides access to all methods and properties for documents that are part of a document workspace
XmlMaps		Represents all schema maps that have been added to the document workbook
XmlNamespaces		Represents all XML namespaces that have been added to the document workbook

Workbooks Methods

Name	Returns	Parameters	Description
Add	Workbook	[Template]	Adds a new workbook to the collection. Using a template name in the Template parameter can specify a template. Also, the XlWBATemplate constants can be used to open up a type of workbook
CanCheckOut	Boolean	Filename As String	Returns whether Excel can check out a specified workbook from a server
CheckOut		Filename As String	Returns a specified workbook from a server for editing

Name	Returns	Parameters	Description
`Close`			Closes the workbook
`DiscardConflict`		`Filename As String`	This keyword is reserved for future use
`OfflineConflict`	`Boolean`	`Filename As String`	This keyword is reserved for future use
`Open`	`Workbook`	`Filename As String, [UpdateLinks], [ReadOnly], [Format], [Password], [WriteResPassword], [IgnoreReadOnlyRecommended], [Origin], [Delimiter], [Editable], [Notify], [Converter], [AddToMru], [Local], [CorruptLoad], [OpenConflictDocument]`	Opens a workbook specified by the `Filename` parameter and adds it to the collection. Use the `UpdateLinks` parameter to choose how links in the file are updated. Set `ReadOnly` to `True` to open up the workbook in read-only mode. If the file requires a password, use the `Password` or `WriteResPassword` parameters. Set `AddToMru` to `True` to add the opening workbook to the recently used files list. If the file to open is a delimited text file then there are some parameters that can be used. Use the `Format` parameter to choose the text delimiter character if opening a text file. Use the `Origin` parameter to choose the code page style of the incoming `delimited` text file. Use the Delimiter parameter to specify a delimiter if 6 (custom) was chosen for the `Format` parameter
`OpenDatabase`	`Workbook`	`Filename As String, [CommandText], [CommandType], [BackgroundQuery], [ImportDataAs]`	Returns a `Workbook` representing a database specified by the `Filename` parameter. The `CommandText` and `CommandType` parameters set the text and the type of the query

Continues

Name	Returns	Parameters	Description
OpenText		Filename As String, [Origin], [StartRow], [DataType], [Text Qualifier As XlText Qualifier], [Consecutive Delimiter], [Tab], [Semicolon], [Comma], [Space], [Other], [OtherChar], [FieldInfo], [TextVisual Layout], [Decimal Separator], [Thousands Separator], [Trailing MinusNumbers], [Local]	Opens the text file in Filename and parses it into a sheet on a new workbook. Origin is used to choose the code page style of the file (XlPlatform constant). StartRow decides the first row to parse. DataType decides if the file is xlDelimited or xlFixedWidth. Set ConsecutiveDelimiter to True to treat consecutive delimiters as one. Set Tab, Semicolon, Comma, Space, or Other to True to pick the delimiter character. Use the DecimalSeparator and ThousandsSeparator to pick the numeric characters to use
OpenXML	Workbook	Filename As String, [Stylesheets]	Returns an XML file in Microsoft Excel. Use the Stylesheets parameter to specify which XSLT stylesheet processing instructions to apply

Workbook Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Workbook Properties

Name	Returns	Description
AcceptLabelsIn Formulas	Boolean	Set/Get whether labels can be used in worksheet formulas
ActiveChart	Chart	Read-only. Returns the active chart in the workbook

Name	Returns	Description
ActiveSheet		Read-only. Returns the active sheet (chart or workbook) in the workbook
AutoUpdate Frequency	Long	Set/Get how often a shared workbook is updated automatically, in minutes
AutoUpdate SaveChanges	Boolean	Set/Get whether changes made to a shared workbook are visible to other users whenever the workbook is automatically updated
Builtin Document Properties	Document Properties	Read-only. Returns a collection holding all the built-in properties of the workbook. Things like title, subject, author, and number of words of the workbook can be accessed from this object
Calculation Version	Long	Read-only. Returns the version number of Excel that was last used to recalculate the Excel spreadsheet
ChangeHistory Duration	Long	Set/Get how far back, in days, a shared workbook's change history is visible
Charts	Sheets	Read-only. Returns the charts in the workbook
CodeName	String	Read-only. Returns the name of the workbook that was set at design time in the VBE
Colors	Variant	Parameters: `[Index]`. Set/Get the color palette colors for the workbook. There are 56 possible colors in the palette
CommandBars	CommandBars	Read-only. Returns an object to manipulate the commandbars in Excel
Conflict Resolution	XlSave Conflict Resolution	Set/Get how shared workbook conflicts are resolved when they are being updated (for example, `xlLocalSessionChanges` means that the local user's changes are always accepted)
Container		Read-only. Returns the object that contains the workbook, if applicable
CreateBackup	Boolean	Read-only. Returns whether a backup file is created whenever the workbook is saved
Custom Document Properties	Document Properties	Read-only. Returns a collection holding all the user-defined properties of the workbook
CustomViews	CustomViews	Read-only. Returns the collection of custom views in a workbook

Continues

Name	Returns	Description
`Date1904`	`Boolean`	Set/Get whether the 1904 date system is used in the workbook
`DisplayDrawingObjects`	`xlDisplayDrawingObjects`	Set/Get if shapes are displayed, placeholders are displayed or shapes are hidden
`EnableAutoRecover`	`Boolean`	Set/Get whether the option to save changed files, of all formats, on a timed interval, is switched on
`EnvelopeVisible`	`Boolean`	Set/Get whether the envelope toolbar and e-mail composition header are visible
`Excel4IntlMacroSheets`	`Sheets`	Read-only. Returns the collection of Excel 4.0 international macro sheets in the workbook
`Excel4MacroSheets`	`Sheets`	Read-only. Returns the collection of Excel 4.0 macro sheets in the workbook
`FileFormat`	`XlFileFormat`	Read-only. Returns the file format of the workbook
`FullName`	`String`	Read-only. Returns the path and file name of the workbook
`FullNameURLEncoded`	`String`	Read-only. Returns the name of the object, including its path on disk, as a string
`HasPassword`	`Boolean`	Read-only. Returns whether the workbook has a protection password
`HasRoutingSlip`	`Boolean`	Set/Get whether the workbook has a routing slip. Use with the `RoutingSlip` object
`HighlightChangesOnScreen`	`Boolean`	Set/Get whether changes in a shared workbook are visibley highlighted
`HTMLProject`	`HTMLProject`	Read-only. Returns an object to access the project explorer of the script editor
`IsAddin`	`Boolean`	Set/Get whether the current workbook is running as an Addin
`IsInplace`	`Boolean`	Read-only. Returns whether the workbook is being edited as an object (`True`) or in Microsoft Excel (`False`)
`KeepChangeHistory`	`Boolean`	Set/Get whether changes are tracked in a shared workbook
`ListChangesOnNewSheet`	`Boolean`	Set/Get whether a separate worksheet is used to display changes of a shared workbook
`Mailer`	`Mailer`	Read-only

Name	Returns	Description
`MultiUser Editing`	`Boolean`	Read-only. Returns whether a workbook is being shared
`Name`	`String`	Read-only. Returns the file name of the workbook
`Names`	`Names`	Read-only. Returns the collection of named ranges in the workbook
`Password`	`String`	Set/Get the password that must be supplied to open the specified workbook
`Password Encryption Algorithm`	`String`	Read-only. Returns the algorithm used by Excel to encrypt passwords for the specified workbook
`Password EncryptionFile Properties`	`Boolean`	Read-only. Returns whether Excel encrypts file properties for the specified password-protected workbook
`Password EncryptionKey Length`	`Long`	Read-only. Returns the key length of the algorithm that Excel uses when encrypting passwords for the specified workbook
`Password Encryption Provider`	`String`	Read-only. Returns the name of the algorithm encryption provider that Excel uses when encrypting passwords for the specified workbook
`Path`	`String`	Read-only. Returns the file path of the workbook
`PersonalViewList Settings`	`Boolean`	Set/Get whether a user's view of the workbook includes filters and sort settings for lists
`PersonalView PrintSettings`	`Boolean`	Set/Get whether a user's view of the workbook includes print settings
`PrecisionAs Displayed`	`Boolean`	Set/Get whether the precision of numbers in the workbook are as displayed in the cells. Used for calculations
`Protect Structure`	`Boolean`	Read-only. Returns whether the sheet order cannot be changed in the workbook
`Protect Windows`	`Boolean`	Read-only. Returns whether the workbook windows are protected
`Publish Objects`	`Publish Objects`	Read-only. Returns access to an object used to publish objects in the workbook as Web pages
`ReadOnly`	`Boolean`	Read-only. Returns whether the workbook is in read-only mode
`ReadOnly Recommended`	`Boolean`	Read-only. Returns whether the user is prompted with a message recommending that you open the workbook as read-only

Continues

Name	Returns	Description
RemovePersonalInformation	Boolean	Set/Get whether personal information can be removed from the specified workbook
RevisionNumber	Long	Read-only. Returns how many times a shared workbook has been saved while open
Routed	Boolean	Read-only. Returns whether a workbook has been routed to the next recipient
RoutingSlip	RoutingSlip	Read-only. Returns access to a RoutingSlip object that can be used to add a routing slip for the workbook. Use with the HasRoutingSlip property
Saved	Boolean	Set/Get whether a workbook does not have changes that need saving
SaveLinkValues	Boolean	Set/Get whether values linked from external sources are saved with the workbook
Sheets	Sheets	Read-only. Returns the collection of sheets in a workbook (Chart or Worksheet)
ShowConflictHistory	Boolean	Set/Get whether the sheet containing conflicts related to shared workbooks are displayed
ShowPivotTableFieldList	Boolean	Set/Get whether the PivotTable field list can be shown
SmartTagOptions	SmartTagOptions	Read-only. Returns the options that can be performed with a SmartTag
Styles	Styles	Read-only. Returns the collection of styles associated with the workbook
TemplateRemoveExtData	Boolean	Set/Get whether all the external data references are removed after a workbook is saved as a template
UpdateLinks	XlUpdateLinks	Set/Get the workbook's setting for updating embedded OLE links
UpdateRemoteReferences	Boolean	Set/Get whether remote references are updated for the workbook
UserStatus	Variant	Read-only. Returns the name of the current user
VBASigned	Boolean	Read-only. Returns whether the VBA Project for the workbook has been digitally signed
VBProject	VBProject	Read-only. Returns access to the VBE and associated project

Name	Returns	Description
WebOptions	WebOptions	Read-only. Returns an object allowing manipulation of Web related properties of the workbook
Windows	Windows	Read-only. Returns the collection of windows that make up the workbook
Worksheets	Sheets	Read-only. Returns the collection of worksheets that make up the workbook
WritePassword	String	Set/Get the write password of a workbook
WriteReserved	Boolean	Read-only. Returns whether the workbook can be modified
Write ReservedBy	String	Read-only. Returns the name of the person with write permission to the workbook

Workbook Methods

Name	Returns	Parameters	Description
Accept AllChanges		[When], [Who], [Where]	Accepts all the changes made by other people in a shared workbook
Activate			Activates the workbook
AddTo Favorites			Adds the workbook shortcut to the Favorites folder
BreakLink		Name As String, Type As XlLinkType	Converts formulas linked to other Excel sources or OLE sources to values
CanCheckIn	Boolean		Set/Get whether Excel can check in a specified workbook to a server
ChangeFile Access		Mode As Xl FileAccess, [Write Password], [Notify]	Changes access permissions of the workbook to the one specified by the Mode parameter. If necessary, the WritePassword can be specified. Set Notify to True to have the user notified if the file cannot be accessed
ChangeLink		Name As String, NewName As String, Type As XlLinkType	Changes the link from the workbook specified by the Name parameter to the NewName workbook. Type chooses the type of link (for example, OLE, Excel)

Continues

Name	Returns	Parameters	Description
CheckIn		[SaveChanges], [Comments], [MakePublic]	Performs a check-in or undo-check-out of the working copy on the server
Close		[SaveChanges], [Filename], [Route Workbook]	Closes the workbook. Set SaveChanges to True to automatically save changes in the workbook. If SaveChanges is False then all changes are lost. The Filename parameter can be used to specify the filename to save to. RouteWorkbook is used to automatically route the workbook onto the next recipient, if applicable
DeleteNumber Format		NumberFormat As String	Deletes the number format in the NumberFormat parameter from the workbook
EndReview			Ends the review of a file that has been sent for review
Exclusive Access	Boolean		Gives the current user exclusive access to a shared workbook
Follow Hyperlink		Address As String, [SubAddress], [NewWindow], [AddHistory], [ExtraInfo], [Method], [HeaderInfo]	Opens up the appropriate application with the URL specified by the Address parameter. Set NewWindow to True to open up a new window for the hyperlink. Use the ExtraInfo and Method parameters to send more information to the hyperlink (say, for an ASP page). The Method parameter uses the MsoExtraInfoMethod constants
Highlight Changes Options		[When], [Who], [Where]	Set/Get when changes are viewed in a shared workbook (When), whose workbook changes can be viewed (Who), and the range that the changes should be put in (Where). Use the XlHighlighChangesTime constants with the When parameter

Name	Returns	Parameters	Description
LinkInfo	Variant	Name As String, LinkInfo As XlLinkInfo, [Type], [EditionRef]	Returns the link details mentioned in the LinkInfo parameter for the link specified by the Name parameter. Use the Type parameter with the XlLinkInfoType constants to pick the type of link that will be returned
LinkSources	Variant	[Type]	Returns the array of linked documents, editions, DDE and OLE servers in a workbook. Use the Type parameter with the XlLinkInfoType constants to pick the type of link that will be returned
Merge Workbook		Filename	Merges the changes from the Filename workbook into the current workbook
NewWindow	Window		Opens up a new window with the current workbook
OpenLinks		Name As String, [ReadOnly], [Type]	Opens the Name link and supporting documents. Set ReadOnly to True to open the documents as read-only. Use the Type parameter with the XlLinkInfoType constants to pick the type of link that will be returned
PivotCaches	PivotCaches		Returns the collection of PivotTable caches in the workbook
Post		[DestName]	Posts the workbook into a Microsoft Exchange public folder
PrintOut		[From], [To], [Copies], [Preview], [Active Printer], [PrintToFile], [Collate], [PrToFile Name]	Prints out the workbook. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the PrintToFile and PrToFileName parameters. The From and To parameters can be used to specify the range of printed pages

Continues

Name	Returns	Parameters	Description
PrintPreview		[Enable Changes]	Displays the current workbook in a print preview mode. Set the EnableChanges parameter to False to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup
Protect		[Password], [Structure], [Windows]	Protects the workbook from user changes. A protect Password can be specified. Set the Structure parameter to True to protect the relative position of the sheets. Set the Windows to True to protect the workbook windows
Protect Sharing		[Filename], [Password], [WriteRes Password], [ReadOnly Recommended], [Create Backup], [Sharing Password]	Protects and saves the workbook for sharing. The file is saved to the Filename parameter with the optional passwords in Password, WriteResPassword, and SharingPassword parameters. Set ReadOnlyRecommended to True to display a message to the user every time the workbook is opened. Set CreateBackup to True to create a backup of the saved file
PurgeChange HistoryNow		Days As Long, [Sharing Password]	Deletes the entries in the change log for the shared workbook. The Days parameter specifies how many days back to delete the entries. A SharingPassword may be required
RecheckSmart Tags			Does a foreground SmartTag check. Any data that was not annotated before will now be annotated
RefreshAll			Refreshes any external data source's data into the workbook
RejectAll Changes		[When], [Who], [Where]	Rejects all the changes in a shared workbook
ReloadAs		Encoding As MsoEncoding	Reopens the workbook using the Web page related Encoding parameter

Name	Returns	Parameters	Description
`RemoveUser`		`Index As Long`	Disconnects the user (specified by the user index in the `Index` parameter) from a shared workbook
`ReplyAll`			Replies to all recipients of the sent workbook. Valid only in the Macintosh Edition of Excel
`ReplyWithChanges`		`[ShowMessage]`	E-mails a notification to the author of a workbook telling them that a reviewer has completed review of the workbook
`ResetColors`			Resets the colors in the color palette to the default colors
`Route`			Routes the workbook using the routing slip
`RunAutoMacros`		`Which As XlRunAutoMacro`	Runs the auto macro specified by the `Which` parameter
`Save`			Saves the workbook
`SaveAs`		`Filename, FileFormat, Password, WriteResPassword, ReadOnlyRecommended, CreateBackup, AccessMode As XlSaveAsAccessMode, [ConflictResolution], [AddToMru], [TextCodepage], [TextVisualLayout], [Local]`	Saves the workbook as `FileName`. The type of file to be saved can be specified with the `FileFormat` parameter. The file can be saved with the optional passwords in the `Password` and `WriteResPassword` parameters. Set `ReadOnlyRecommended` to `True` to display a message to the user every time the workbook is opened. Set `CreateBackup` to `True` to create a backup of the saved file. Use the `AccessMode` to choose how the workbook is accessed (for example, `xlShared`, `xlExclusive`). Use the `ConflictResolution` parameter to decide how shared workbooks resolve change conflicts. Set the `AddToMru` parameter to `True` to add the workbook to the recently opened files list

Continues

Name	Returns	Parameters	Description
SaveCopyAs		[Filename]	Saves a copy of the workbook as the FileName
SendForReview		[Recipients], [Subject], [ShowMessage], [Include Attachment]	Sends a workbook in an e-mail message for review to the specified recipients
SendMail		Recipients, [Subject], [Return Receipt]	Sends the workbook through the default mail system. The recipient or recipients and subject can be specified with the parameters. Set ReturnReceipt to True to request a return receipt
SendMailer		FileFormat, Priority As XlPriority	
SetLinkOnData		Name As String, [Procedure]	Runs the procedure in the Procedure parameter whenever the DDE or OLE link in the Name parameter is updated
SetPassword Encryption Options		[Password Encryption Provider], [Password Encryption Algorithm], [Password EncryptionKey Length], [Password Encryption File Properties]	Sets the options for encrypting workbooks using passwords
Unprotect		[Password]	Unprotects the workbook with the password, if necessary
Unprotect Sharing		[Sharing Password]	Unprotects the workbook from sharing and saves the workbook
UpdateFrom File			Reloads the current workbook from the file if the file is newer then the workbook

Name	Returns	Parameters	Description
UpdateLink		[Name], [Type]	Updates the link specified by the Name parameter. Use the Type parameter with the XlLinkInfoType constants to pick the type of link that will be returned
WebPagePreview			Previews the workbook as a Web page

Workbook Events

Name	Parameters	Description
Activate		Triggered when the workbook is activated
AddinInstall		Triggered when the workbook is opened as an Addin
Addin Uninstall		Triggered when the workbook opened as an Addin is uninstalled
BeforeClose	Cancel As Boolean	Triggered just before the workbook closes. Set the Cancel parameter to True to cancel the closing
BeforePrint	Cancel As Boolean	Triggered just before the workbook is printed. Set the Cancel parameter to True to cancel the printing
BeforeSave	SaveAsUI As Boolean, Cancel As Boolean	Triggered just before the workbook is saved. Set the Cancel parameter to True to cancel the saving. Set the SaveAsUI to True for the user to be prompted with the Save As dialog box
Deactivate		Triggered when the workbook loses focus
NewSheet	Sh As Object	Triggered when a new sheet is created in the workbook. The Sh parameter passes in the new sheet
Open		Triggered when the workbook is opened

Continues

Name	Parameters	Description
`PivotTableCloseConnection`	`ByVal Target As PivotTable`	Triggered when a `PivotTable` report closes the connection to its data source. `Target` is the selected `PivotTable`
`PivotTableOpenConnection`	`ByVal Target As PivotTable`	Triggered when a PivotTable report opens the connection to its data source. `Target` is the selected PivotTable
`SheetActivate`	`Sh As Object`	Triggered when a sheet is activated in the workbook. The `Sh` parameter passes in the activated sheet
`SheetBeforeDoubleClick`	`Sh As Object, Target As Range, Cancel As Boolean`	Triggered when a sheet is about to be double-clicked. The sheet and the potential double-click spot are passed into the event. The double-click action can be canceled by setting the `Cancel` parameter to `True`
`SheetBeforeRightClick`	`Sh As Object, Target As Range, Cancel As Boolean`	Triggered when a sheet is about to be right-clicked. The sheet and the potential right-click spot are passed into the event. The right-click action can be canceled by setting the `Cancel` parameter to `True`
`SheetCalculate`	`Sh As Object`	Triggered when a sheet is recalculated passing in the recalculated sheet
`SheetChange`	`Sh As Object, Target As Range`	Triggered when the contents of a cell are changed in any worksheet in the workbook, for example, triggered by entering new data, clearing the cell, deleting a row/column. *Not* triggered when inserting rows/columns
`SheetDeactivate`	`Sh As Object`	Triggered when a sheet loses focus. Passes in the sheet
`SheetFollowHyperlink`	`Sh As Object, Target As Hyperlink`	Triggered when the user clicks on a hyperlink on a sheet. Passes in the sheet and the clicked hyperlink
`SheetPivotTableUpdate`	`ByVal Sh As Object, Target As PivotTable`	Triggered when the sheet of the `PivotTable` report has been updated
`SheetSelectionChange`	`Sh As Object, Target As Range`	Triggered when the user selects a different cell on the sheet. Passes in the new range and the sheet where the change occurred

Name	Parameters	Description
WindowActivate	Wn As Window	Triggered when a workbook window is activated (brought up to the front of other workbook windows). The workbook and the window are passed in
WindowDeactivate	Wn As Window	Triggered when a workbook window loses focus. The related workbook and the window are passed in
WindowResize	Wn As Window	Triggered when a workbook window is resized. The resized workbook and window are passed into the event

Example: Workbook Object and the Workbooks Collection

Please refer to Chapter 4 for Workbook object examples.

Worksheet Object and the Worksheets Collection

The Worksheets collection holds the collection of worksheets in a workbook. The Workbook object is always the parent of the Worksheets collection. The Worksheets collection only holds the worksheets. The Worksheet objects in the Worksheets collection can be accessed using the Item property. Either the name of the worksheet can be specified as a parameter to the Item's parameter or an index number describing the position of the worksheet in the workbook (from left to right).

The Worksheet object allows access to all of the attributes of a specific worksheet in Excel. This includes worksheet formatting and other worksheet properties. The Worksheet object also exposes events that can be used programmatically.

The Worksheets collection has a few properties and methods besides the typical collection attributes. These are listed in the following table.

Worksheets Collection Properties and Methods

Name	Returns	Description
Count	Long	Read-only. Returns the number of worksheets in the collection
HPageBreaks	HPageBreaks	Read-only. Returns a collection holding all the horizontal page breaks associated with the Worksheets collection

Continues

Name	Returns	Description
`VpageBreaks`	`VPage Breaks`	Read-only. Returns a collection holding all the vertical page breaks associated with the `Worksheets` collection
`Visible`	`Variant`	Set/Get whether the worksheets in the collection are visible. Also can set this to `xlVeryHidden` to not allow a user to make the worksheets in the collection visible
`Add`		Method. Parameters: `[Before], [After], [Count], [Type]`. Adds a worksheet to the collection. You can specify where the worksheet goes by choosing which sheet object will be before the new worksheet object (`Before` parameter) or after the new worksheet (`After` parameter). The `Count` parameter decides how many worksheets are created
`Copy`		Method. Parameters: `[Before], [After]`. Adds a new copy of the currently active worksheet to the position specified at the `Before` or `After` parameters
`Delete`		Method. Deletes all the worksheets in the collection
`FillAcross Sheets`		Method. Parameters: `Range As Range, Type As XlFillWith`. Copies the range specified by the `Range` parameter across all the other worksheets in the collection. Use the `Type` parameter to pick what part of the range is copied (for example, `xlFillWithContents, xlFillWithFormulas`)
`Move`		Method. Parameters: `[Before], [After]`. Moves the current worksheet to the position specified by the parameters
`PrintPreview`		Method. Parameters: `[EnableChanges]`. Displays the current worksheet in the collection in a print preview mode. Set the `EnableChanges` parameter to `False` to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup

Name	Returns	Description
PrintOut		Method. Parameters: [From],[To],Copies],[Preview],[ActivePrinter],[PrintToFile],[Collate],[PrToFileName]. Prints out the worksheets in the collection. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the PrintToFile and PrToFileName parameters. The From and To parameters can be used to specify the range of printed pages
Select		Method. Parameters: [Replace]. Selects the current worksheet in the collection

Worksheet Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

Worksheet Properties

Name	Returns	Description
AutoFilter	AutoFilter	Read-only. Returns an AutoFilter object if filtering is turned on
AutoFilter Mode	Boolean	Set/Get whether AutoFilter drop-down arrows are currently displayed on the worksheet
Cells	Range	Read-only. Returns the cells in the current worksheet
Circular Reference	Range	Read-only. Returns the cell range that contains the first circular reference on the worksheet
CodeName	String	Read-only. Returns the name of the worksheet set at design time in the VBE
Columns	Range	Read-only. Returns a range of the columns in the current worksheet
Comments	Comments	Read-only. Returns the collection of comments in the worksheet
Consolidation Function	Xl Consolidation Function	Read-only. Returns the type of consolidation being used in the worksheet (for example, xlSum, xlMax, xlAverage)

Continues

Name	Returns	Description
ConsolidationOptions	Variant	Read-only. Returns a one-dimensional array containing three elements of Booleans. The first element describes whether the labels in the top row are used; the second element describes whether the labels in the left-most column are used; and the third element describes whether links are created to the source data
ConsolidationSources	Variant	Read-only. Returns the array of strings that describe the source sheets for the current worksheet's consolidation
CustomProperties	CustomProperties	Read-only. Returns the identifier information associated with a worksheet
DisplayPageBreaks	Boolean	Set/Get whether page breaks are displayed
DisplayRightToLeft	Boolean	Set/Get whether the worksheet contents are displayed from right to left. Valid only with languages that support right-to-left text
EnableAutoFilter	Boolean	Set/Get whether the AutoFilter arrows are enabled when a worksheet is user interface-only protected
EnableCalculation	Boolean	Set/Get whether Excel will automatically recalculate the worksheet as necessary
EnableOutlining	Boolean	Set/Get whether outlining symbols are enabled when a worksheet is user interface-only protected
EnablePivotTable	Boolean	Set/Get whether PivotTable controls and related actions are enabled when a worksheet is user interface-only protected
EnableSelection	XlEnableSelection	Set/Get what objects can be selected when a worksheet is protected (for example, xlNoSelection, xlNoRestrictions)
FilterMode	Boolean	Read-only. Returns whether a worksheet is in a filter mode
HPageBreaks	HPageBreaks	Read-only. Returns a collection holding all the horizontal page breaks associated with the Worksheet
Hyperlinks	Hyperlinks	Read-only. Returns the collection of hyperlinks in the worksheet
Index	Long	Read-only. Returns the spot in the parent collection where the current worksheet is located
ListObjects	Range	Returns a Range object

Name	Returns	Description
`MailEnvelope`	`MsoEnvelope`	Set/Get the e-mail header for a document
`Name`	`String`	Set/Get the name of the worksheet
`Names`	`Names`	Read-only. Returns the collection of ranges with names in the worksheet
`Next`		Read-only. Returns the next sheet in the workbook (from left to right) as an object
`Outline`	`Outline`	Read-only. Returns an object to manipulate an outline in the worksheet
`PageSetup`	`PageSetup`	Read-only. Returns an object to manipulate the page setup properties for the worksheet
Previous		Read-only. Returns the previous sheet in the workbook (from right to left) as an object
`Protect Contents`	`Boolean`	Read-only. Returns whether the worksheet and everything in it is protected from changes
`Protect Drawing Objects`	`Boolean`	Read-only. Returns whether the shapes in the worksheet can be modified (`ProtectDrawingObjects = False`)
`Protection`	`Protection`	Read-only. Returns the protection options of the worksheet
`Protection Mode`	`Boolean`	Read-only. Returns whether protection has been applied to the user interface. Even if a worksheet has user interface protection on, any VBA code associated with the worksheet can still be accessed
`Protect Scenarios`	`Boolean`	Read-only. Returns whether the worksheet scenarios are protected
`QueryTables`	`QueryTables`	Read-only. Returns the collection of query tables associated with the worksheet
`Range`	`Range`	Read-only. Parameters: `Cell1`, `[Cell2]`. Returns a `Range` object as defined by the `Cell1` and optionally `Cell2` parameters
`Rows`	`Range`	Read-only. Returns a `Range` object containing the rows of the current worksheet
`Scripts`	`Scripts`	Read-only. Returns the collection of VBScript code associated with a worksheet (typically to later use on Web pages)
`ScrollArea`	`String`	Sets the A1-style reference string describing the range in the worksheet that can be scrolled. Cells not in the range cannot be selected

Continues

Name	Returns	Description
Shapes	Shapes	Read-only. Returns all the shapes contained by the worksheet
SmartTags	SmartTags	Read-only. Returns the identifier for the specified cell
Standard Height	Double	Read-only. Returns the default height of the rows in the worksheet, in points
Standard Width	Double	Read-only. Returns the default width of the columns in the worksheet, in points
Tab	Tab	Read-only. Returns the Tab object for the selected chart or worksheet
TransitionExp Eval	Boolean	Set/Get whether evaluates expressions using Lotus 1-2-3 rules in the worksheet
TransitionForm Entry	Boolean	Set/Get whether formula entries can be entered using Lotus 1-2-3 rules
Type	XlSheetType	Read-only. Returns the worksheet type (for example, xlWorksheet, xlExcel4MacroSheet, xlExcel4IntlMacroSheet)
UsedRange	Range	Read-only. Returns the range in the worksheet that is being used
Visible	XlSheet Visibility	Set/Get whether the worksheet is visible. Also can set this to xlVeryHidden to not allow a user to make the worksheet visible
VPageBreaks	VPageBreaks	Read-only. Returns a collection holding all the vertical page breaks associated with the worksheet

Worksheet Methods

Name	Returns	Parameters	Description
Activate			Activates the worksheet
Calculate			Calculates all the formulas in the worksheet
ChartObjects		[Index]	Returns either a chart object (ChartObject) or a collection of chart objects (ChartObjects) in a worksheet

Name	Returns	Parameters	Description
CheckSpelling		[Custom Dictionary], [Ignore Uppercase], [Always Suggest], [SpellLang]	Checks the spelling of the text in the worksheet. A custom dictionary can be specified (CustomDictionary), all uppercase words can be ignored (IgnoreUppercase), and Excel can be set to display a list of suggestions (AlwaysSuggest)
CircleInvalid			Circles the invalid entries in the worksheet
ClearArrows			Clears out all the tracer arrows in the worksheet
ClearCircles			Clears all the circles around invalid entries in a worksheet
Copy		[Before], [After]	Adds a new copy of the worksheet to the position specified at the Before or After parameters
Delete			Deletes the worksheet
Evaluate	Variant	Name	Evaluates the Name string expression as if it were entered into a worksheet cell
Move		[Before], [After]	Moves the worksheet to the position specified by the parameters
OLEObjects		[Index]	Returns either a single OLE object (OLEObject) or a collection of OLE objects (OLEObjects) for a worksheet
Paste		[Destination], [Link]	Pastes the contents of the clipboard into the worksheet. A specific destination range can be specified with the Destination parameter. Set Link to True to establish a link to the source of the pasted data. Either the Destination or the Link parameter can be used

Continues

Name	Returns	Parameters	Description
PasteSpecial		[Format], [Link], [DisplayAs Icon], [IconFile Name], [IconIndex], [IconLabel], [NoHTML Formatting]	Pastes the clipboard contents into the current worksheet. The format of the clipboard data can be specified with the string Format parameter. Set Link to True to establish a link to the source of the pasted data. Set DisplayAsIcon to True to display the pasted data as an icon and the IconFileName, IconIndex, and IconLabel to specify the icon and label. A destination range must be already selected in the worksheet
PivotTables		[Index]	Returns either a single PivotTable report (PivotTable) or a collection of PivotTable reports (PivotTables) for a worksheet
PivotTable Wizard	Pivot Table	[SourceType], [SourceData], [Table Destination], [TableName], [RowGrand], [ColumnGrand], [SaveData], [HasAuto Format], [AutoPage], [Reserved], [Background Query], [Optimize Cache], [PageField Order], [PageField WrapCount], [ReadData], [Connection]	Creates a PivotTable report. The SourceType uses the XLPivotTableSourceType constants to specify the type of SourceData being used for the PivotTable. Table Destination holds the range in the parent worksheet that report will be placed. TableName holds the name of the new report. Set RowGrand or ColumnGrand to True to show grand totals for rows and columns, respectively. Set HasAutoFormat to True for Excel to format the report automatically when it is refreshed or changed. Use the AutoPage parameter to set if a page field is created automatically for consolidation.

Name	Returns	Parameters	Description
			Set BackgroundQuery to True for Excel to query the data source asynchronously. Set OptimizeCache to True for Excel to optimize the cache when it is built. Use the PageFieldOrder with the xlOrder constants to set how new page fields are added to the report. Use the PageFieldWrapCount to set the number of page fields in each column or row. Set ReadData to True to copy the data from the external database into a cache. Finally, use the Connection parameter to specify an ODBC connection string for the PivotTable's cache
PrintOut		[From], [To], [Copies], [Preview], [Active Printer], [PrintToFile], [Collate], [PrToFile Name]	Prints out the worksheet. The printer, number of copies, collation, and whether a print preview is desired can be specified with the parameters. Also, the sheets can be printed to a file using the PrintToFile and PrToFileName parameters. The From and To parameters can be used to specify the range of printed pages
PrintPreview		[Enable Changes]	Displays the worksheet in a print preview mode. Set the EnableChanges parameter to False to disable the Margins and Setup buttons, hence not allowing the viewer to modify the page setup
Protect		[Password], [Drawing Objects], [Contents], [Scenarios], [User Interface Only], [Allow Formatting Cells], [Allow Formatting	Protects the worksheet from changes. A case-sensitive Password can be specified. Also, whether shapes are protected (DrawingObjects), the entire contents are protected (Contents), and whether only the user interface is protected (UserInterfaceOnly)

Continues

Name	Returns	Parameters	Description
		Columns], [Allow Formatting Rows], [Allow Inserting Columns], [Allow Inserting Rows], [Allow Inserting Hyperlinks], [Allow Deleting Columns], [Allow DeletingRows], [Allow Sorting], [Allow Filtering], [AllowUsing PivotTables]	
ResetAllPage Breaks			Resets all the page breaks in the worksheet
SaveAs		Filename As String, [FileFormat], [Password], [WriteRes Password], [ReadOnly Recommended], [Create Backup], [AddToMru], [Text Codepage], [TextVisual Layout] , [Local]	Saves the worksheet as FileName. The type of file to be saved can be specified with the FileFormat parameter. The file can be saved with the optional passwords in the Password and WriteResPassword parameters. Set ReadOnly Recommended to True to display a message to the user every time the worksheet is opened. Set CreateBackup to True to create a backup of the saved file. Set the AddToMru parameter to True to add the worksheet to the recently-opened files list
Scenarios		[Index]	Returns either a single scenario (Scenario) or a collection of scenarios (Scenarios) for a worksheet

Name	Returns	Parameters	Description
`Select`		`[Replace]`	Selects the worksheet
`SetBackground Picture`		`Filename As String`	Sets the worksheet's background to the picture specified by the `FileName` parameter
`ShowAllData`			Displays all of the data that is currently filtered
`ShowDataForm`			Displays the data form that is part of the worksheet
`Unprotect`		`[Password]`	Deletes the protection set up for a worksheet. If the worksheet was protected with a password, the password must be specified now

`XmlDataQuery`	`Range`	`[XPath]`
		`[Selection Namespaces]`
		`[Map]`
			Represents cells mapped to a particular XPath

`XmlMapQuery`	`Range`	`[XPath]`
		`[Selection Namespaces]`
		`[Map]`
			Represents cells mapped to a particular XPath

Worksheet Events

Name	Parameters	Description
`Activate`		Triggered when a worksheet is made to have focus
`BeforeDouble Click`	`Target As Range, Cancel As Boolean`	Triggered just before a user double-clicks a worksheet. The cell closest to the point double-clicked in the worksheet is passed in to the event procedure as `Target.` The double-click action can be canceled by setting the `Cancel` parameter to `True`
`BeforeRight Click`	`Cancel As Boolean`	Triggered just before a user right-clicks a worksheet. The cell closest to the point right-clicked in the worksheet is passed in to the event procedure as `Target.` The right-click action can be canceled by setting the `Cancel` parameter to `True`
`Calculate`		Triggered after the worksheet is recalculated

Continues

Name	Parameters	Description
Change	Target As Range	Triggered when the worksheet cell values are changed. The changed range is passed into the event procedure as Target
Deactivate		Triggered when the worksheet loses focus
Follow Hyperlink	Target As Hyperlink	Triggered when a hyperlink is clicked on the worksheet. The hyperlink that was clicked is passed into the event procedure as Target
PivotTable Update	ByVal Target As Pivot Table	Triggered when a PivotTable report is updated on a worksheet
Selection Change	Target As Range	Triggered when the selection changes in a worksheet. The new selected range is passed into the event procedure as Target

Example: Worksheet Object and the Worksheets Collection

Please refer to Chapter 4 for Worksheet object examples.

WorksheetFunction Object

The WorksheetFunction object contains all of the Excel worksheet function. The WorksheetFunction object allows access to Excel worksheet function in Visual Basic code. The parent of the WorksheetFunction object is the Application object.

WorksheetFunction Common Properties

The Application, Creator, and Parent properties are defined at the beginning of this Appendix.

WorksheetFunction Methods

Name	Returns	Parameters	Description
Acos	Double	Arg1 As Double	Returns the arccosine of the Arg1 number. Arg1 must be between –1 to 1
Acosh	Double	Arg1 As Double	Returns the inverse hyperbolic cosine of the Arg1 number. Arg1 must be >= 1
And	Boolean	Arg1, [Arg2], ... [Arg30]	Returns True if all the arguments (from Arg1 up to Arg30) evaluate to True

Name	Returns	Parameters	Description
Asc	String	Arg1 As String	Returns the half-width character equivalent of the full width characters in the Arg1 string. Not the same as the VBA Asc function which returns the ASCII code of the first character in the string
Asin	Double	Arg1 As Double	Returns the arcsine of the Arg1 number. Arg1 must be between –1 and 1
Asinh	Double	Arg1 As Double	Returns the inverse hyperbolic sine of the Arg1 number
Atan2	Double	Arg1 As Double, Arg2 As Double	Returns the arctangent of the x and y coordinates specified in the Arg1 and Arg2 parameters, respectively
Atanh	Double	Arg1 As Double	Returns the inverse hyperbolic tangent of the Arg1 number. Arg1 must be between –1 and 1
AveDev	Double	Arg1, [Arg2], ... [Arg30]	Returns the average of the absolute deviation from the mean of the Arg1 to Arg30 number parameters
Average	Double	Arg1, [Arg2], ... [Arg30]	Returns the average of the numbers in Arg1 to Arg30
BahtText	String	Arg1 As Double	Returns a number in Thai text with a "Baht." suffix. Arg1 is the number to be converted
BetaDist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, [Arg4], [Arg5]	Returns the cumulative beta probability. Arg1 is the number to evaluate. Arg2 is the Alpha part of the distribution. Arg3 is the Beta part of the distribution. Arg4 and Arg5 can be the lower and upper bounds of the interval in Arg1

Continues

Name	Returns	Parameters	Description
`BetaInv`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double, [Arg4], [Arg5]`	Returns the inverse of the cumulative beta probability density. `Arg1` is the probability of the distribution. `Arg2` is the Alpha part of the distribution. `Arg3` is the Beta part of the distribution. `Arg4` and `Arg5` can be the lower and upper bounds of the evaluated number
`BinomDist`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Boolean`	Returns the individual term binomial distribution probability. `Arg1` is the number of successes in the trials. `Arg2` holds the total number of trials. `Arg3` is the probability of success on a single trial. `Arg4` sets whether the method returns the cumulative distribution function (`True`) or the probability mass function (`False`)
`Ceiling`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the nearest number to `Arg1` that is a multiple of `Arg2,` rounded positively
`ChiDist`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the one-tail probability of the chi-squared distribution. `Arg1` is the number to evaluate. `Arg2` is the number of degrees of freedom
`ChiInv`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the inverse of the one-tail probability of the chi-squared distribution. `Arg1` is the probability. `Arg2` is the number of degrees of freedom
`ChiTest`	`Double`	`Arg1, Arg2`	Returns the chi-squared distribution test for independence. `Arg1` holds the range of data that will be tested against the expected values. `Arg2` holds the range of expected data
`Choose`	`Variant`	`Arg1, Arg2, [Arg3], ... [Arg30]`	Returns one of the parameter values `(Arg2 to Arg30)` given the index value in Arg1. For example, if `Arg1` is 2 then the value in `Arg3` is returned
`Clean`	`String`	`Arg1 As String`	Returns the string in `Arg1` without any nonprintable characters

Name	Returns	Parameters	Description
Combin	Double	Arg1 As Double, Arg2 As Double	Returns the total possible number of combinations of a group ofArg2 items in a total number of Arg1 items
Confidence	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns a range on either side of a sample mean for a population mean. Arg1 is the Alpha value used to determine the confidence level. Arg2 is the standard deviation for the data range. Arg3 is the sample size
Correl	Double	Arg1, Arg2	Returns the correlation coefficient of the arrays in Arg1 and Arg2 . The parameters can also be cell ranges
Cosh	Double	Arg1 As Double	Returns the hyperbolic cosine of the Arg1 number
Count	Double	Arg1, [Arg2], ... [Arg30]	Returns the number of numeric values in the arguments. Arg1 to Arg30 can be values or range references
CountA	Double	Arg1, [Arg2], ... [Arg30]	Returns the number of non-empty values in the arguments. Arg1 to Arg30 can be values or range references
CountBlank	Double	Arg1 As Range	Returns the number of empty values in the range in Arg1
CountIf	Double	Arg1 As Range, Arg2	Counts the number of cells in the Arg1 range that meet the criteria in Arg2
Covar	Double	Arg1, Arg2	Returns the covariance of the arrays or ranges in Arg1 and Arg2
CritBinom	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the smallest value where the cumulative binomial distribution is greater than or equal to the criterion value. Arg1 is the number of Bernoulli trials. Arg2 is the probability of success for each trial. Arg3 is the criterion value

Continues

Name	Returns	Parameters	Description
DAverage	Double	Arg1 As Range, Arg2, Arg3	Returns the average of the column specified by Arg2 in the range of cells in Arg1. Arg3 contains the criteria used to choose rows of records to be averaged
Days360	Double	Arg1, Arg2, [Arg3]	Returns the difference of days between the Arg1 and Arg2 dates (Arg1 – Arg2). If the Arg3 method is set to True then the European method of calculation is used. If Arg3 is set to False or omitted then the US method of calculation is used
Db	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Double, [Arg5]	Returns depreciation for a specified period using the fixed-declining balance method. Arg1 is the initial cost to depreciate. Arg2 is the final salvage cost (cost at end of depreciation). Arg3 is the number of periods to depreciate. Arg4 is the specific period from Arg3 to depreciation. Arg5 can be the number of months in the first year that depreciation will start
Dbcs	String	Arg1 As String	Returns the Double-Byte-Character-Set string of the given ASCII string. Opposite of the Asc function
DCount	Double	Arg1 As Range, Arg2, Arg3	Returns the number of cells that match the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DCountA	Double	Arg1 As Range, Arg2, Arg3	Returns the number of non-blank cells that match the criteria in Arg3. Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count

Name	Returns	Parameters	Description
Ddb	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Double, [Arg5]	Returns depreciation for a specified period using the double-declining balance method. Arg1 is the initial cost to depreciate. Arg2 is final salvage cost (cost at end of depreciation). Arg3 is the number of periods to depreciate. Arg4 is the specific period from Arg3 to depreciation. Arg5 can be the rate at which the balance declines
Degrees	Double	Arg1 As Double	Converts the radians in Arg1 into degrees and returns the degrees
DevSq	Double	Arg1, [Arg2], ... [Arg30]	Returns the sum of the squares of deviations of the Arg1 to Arg30 from their mean
DGet	Variant	Arg1 As Range, Arg2, Arg3	Returns the cell value that matches the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count. Criteria must match a single cell
DMax	Double	Arg1 As Range, Arg2 , Arg3	Returns the largest value that matches the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DMin	Double	Arg1 As Range, Arg2, Arg3	Returns the smallest value that matches the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
Dollar	String	Arg1 As Double, [Arg2]	Returns a currency-type string of the number in Arg1 with the decimal points specified in Arg2

Continues

Name	Returns	Parameters	Description
DProduct	Double	Arg1 As Range, Arg2, Arg3	Returns the multiplication product of the values that match the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DStDev	Double	Arg1 As Range, Arg2, Arg3	Returns the estimated standard deviation of the values that match the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DStDevP	Double	Arg1 As Range, Arg2, Arg3	Returns the standard deviation of the values that match the criteria in Arg3 assuming the entire population is given. Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DSum	Double	Arg1 As Range, Arg2, Arg3	Returns the sum of the values that match the criteria in Arg3 . Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DVar	Double	Arg1 As Range, Arg2, Arg3	Returns the variance of the values that match the criteria in Arg3. Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
DVarP	Double	Arg1 As Range, Arg2, Arg3	Returns the variance of the values that match the criteria in Arg3 assuming that the entire population is given. Arg1 specifies the range of rows and columns to count and Arg2 is used to choose the field name or number to count
Even	Double	Arg1 As Double	Converts the Arg1 number into the nearest even whole number, rounded up, and returns it

Name	Returns	Parameters	Description
ExponDist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Boolean	Returns the exponential distribution of a value. Arg1 is the value of the function. Arg2 is the Lambda parameter value. Set Arg3 to True for the method to return the cumulative distribution. Set Arg3 to False to return the probability density
Fact	Double	Arg1 As Double	Returns the factorial of Arg1
FDist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the F probability distribution of a value. Arg1 is the value to evaluate the function. Arg2 is the numerator degrees of freedom and Arg3 is the denominator degrees of freedom
Find	Double	Arg1 As String, Arg2 As String, [Arg3]	Finds the text in Arg1 from the text in Arg2 and returns the starting position of the found text. Arg3 can specify the starting position to search in Arg2
FindB	Double	Arg1 As String, Arg2 As String, [Arg3]	Finds the text in Arg1 from the text in Arg2 and returns the starting position of the found text. Arg3 can specify the starting byte position to search in Arg2
FInv	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the inverse of the F probability distribution. Arg1 is the probability that is associated with the F cumulative distribution. Arg2 is the numerator degrees of freedom and Arg3 is the denominator degrees of freedom
Fisher	Double	Arg1 As Double	Returns the Fisher transformation at the Arg1 value
FisherInv	Double	Arg1 As Double	Returns the inverse of the Fisher transformation given the value of Arg1
Fixed	String	Arg1 As Double, [Arg2], [Arg3]	Rounds the number Arg1 to the decimal points Arg2 and returns the value as a string. Set Arg3 to True to put commas in the returned text. Set Arg3 to False to not put commas in the returned text

Continues

Name	Returns	Parameters	Description
Floor	Double	Arg1 As Double, Arg2 As Double	Returns the nearest number to Arg1 that is a multiple of Arg2, rounded down towards 0
Forecast	Double	Arg1 As Double, Arg2, Arg3	Returns a predicted y value for a given x value (Arg1) by using the existing value pairs. Arg2 is an array or range corresponding to the y's known values. Arg3 is an array or range corresponding to the x's known values
Frequency	Variant	Arg1, Arg2	Returns an array of numbers describing the frequency of values in the array Arg1 that are in the intervals specified by the Arg2 array. Arg1 and Arg2 can also be references to a range
FTest	Double	Arg1, Arg2	Returns the result of an F-test of the arrays in Arg1 and Arg2. Arg1 and Arg2 can also be references to a range
Fv	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, [Arg4], [Arg5]	Returns the future value of an investment for a time period. Arg1 is the interest rate per period. Arg2 is the total number of payment periods. Arg3 is the payment per period. Arg4 is the initial value. Arg5 determines if payments are due at the end of the period (0) or the beginning of the period (1)
GammaDist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Boolean	Returns the gamma distribution. Arg1 is the value to evaluate. Arg2 is the alpha parameter to the distribution. Arg3 is the beta parameter to the distribution. Set Arg4 to True for the method to return the cumulative distribution. Set Arg4 to False to return the probability density
GammaInv	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the inverse of the gamma cumulative distribution. Arg1 is the gamma distribution probability. Arg2 is the alpha parameter to the distribution. Arg3 is the beta parameter to the distribution
GammaLn	Double	Arg1 As Double	Returns the natural logarithm of the gamma function with the Arg1 number

Name	Returns	Parameters	Description
GeoMean	Double	Arg1, [Arg2], ... [Arg30]	Returns the geometric mean of the numbers in Arg1 to Arg30. Arg1 to Arg30 can also be a reference to a range
Growth	Variant	Arg1, [Arg2], [Arg3], [Arg4]	Returns the predicted exponential growth of y values (Arg1) for a series of new x values (Arg3).Arg2 can be used to set the series of existing x values. Arg4 can be set to False to make the 'b' part of the equation equal to one
HarMean	Double	Arg1, [Arg2], ... [Arg30]	Returns the harmonic mean of the numbers in Arg1 to Arg30. Arg1 to Arg30 can also be a reference to a range
HLookup	Variant	Arg1, Arg2, Arg3, [Arg4]	Looks up the value specified by Arg1 in the table array (or range reference); Arg2 first row. Arg3 specifies the row number in the table array that contains the matching value. Set Arg4 to True to find approximate data or set Arg4 to False to only lookup exact values
HypGeom Dist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Double	Returns the hypergeometric distribution probability. Arg1 is the number of successes in the trials. Arg2 holds the total number of trials. Arg3 is the number of successes in the trial. Arg4 is the size of the population
Index	Variant	Arg1, Arg2 As Double, [Arg3], [Arg4]	May return the cell or array of cells from Arg1 that has a row number of Arg2 and a column number of Arg3. May also return the cell or range of cells that have a row number of Arg2 and a column number of Arg3. If Arg1 contains many areas then Arg4 can be used to specify the area

Continues

Name	Returns	Parameters	Description

`Intercept`	`Double`	`Arg1,`
		`Arg2`
			Returns the point where the x-axis and y-axis coordinates intersect. `Arg1` represents the array of known y values and `Arg2` represents the array of known x values

`Ipmt`	`Double`	`Arg1 As Double,`
		`Arg2 As Double,`
		`Arg3 As Double,`
		`Arg4 As Double,`
		`[Arg5],`
		`[Arg6]`
			Returns the interest amount paid for an investment for a time period. `Arg1` is the interest rate per period. `Arg2` is the period that you want to find the amount of interest for. `Arg3` is the total number of payment periods. `Arg4` is the initial value. `Arg5` is the future value that is wanted to be attained. `Arg6` determines if payments are due at the end of the period (0) or the beginning of the period (1)

`Irr`	`Double`	`Arg1,`
		`[Arg2]`
			Returns the rate of return for an array of values in `Arg1`. `Arg2` can be used to specify a guess of the `Irr` result

`IsErr`	`Boolean`	`Arg1`	Returns whether the cell `Arg1` contains an error value (except #N/A)
`IsError`	`Boolean`	`Arg1`	Returns whether the cell `Arg1` contains any error value
`IsLogical`	`Boolean`	`Arg1`	Returns whether the cell or value `Arg1` contains a logical value
`IsNA`	`Boolean`	`Arg1`	Returns whether the cell `Arg1` contains the #N/A value
`IsNonText`	`Boolean`	`Arg1`	Returns whether the cell `Arg1` does not contain text
`IsNumber`	`Boolean`	`Arg1`	Returns whether the cell `Arg1` contains a numeric value

`Ispmt`	`Double`	`Arg1 As Double,`
		`Arg2 As Double,`
		`Arg3 As Double,`
		`Arg4 As Double`
			Returns the interest amount paid for an investment at a particular period. Used for compatibility purposes. `Arg1` is the interest rate per period. `Arg2` is the period that you want to find the amount of interest for. `Arg3` is the total number of payment periods. `Arg4` is the initial value

Name	Returns	Parameters	Description
IsText	Boolean	Arg1	Returns whether the cell Arg1 contains a text value
Kurt	Double	Arg1, Arg2], ... [Arg30]	Returns the kurtosis of the values in Arg1 to Arg30. Also, Arg1 can be a reference to a cell range
Large	Double	Arg1, Arg2 As Double	Returns the Arg2 largest value in the array or cell reference specified by Arg1 (for example, second largest, third largest)
LinEst	Variant	Arg1, [Arg], [Arg3], [Arg4]	Returns an array describing a straight line that best fits the data of known y values (Arg1) and known x values (Arg2). Set Arg3 to False to make the 'b' part of the calculations equal to 0. Set Arg4 to True to return additional statistics
Ln	Double	Arg1 As Double	Returns the natural logarithm of the Arg1 number
Log	Double	Arg1 As Double, [Arg2]	Returns the logarithm of the Arg1 number to the base specified in Arg2. Arg2 is 10 by default
Log10	Double	Arg1 As Double	Returns the base-10 logarithm of the Arg1 number
LogEst	Variant	Arg1, [Arg2], [Arg3], [Arg4]	Returns an array describing the curved line that best fits the data of known y values (Arg1) and known x values (Arg2). Set Arg3 to False to make the 'b' part of the calculations equal to 0. Set Arg4 to True to return additional statistics
LogInv	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the inverse of the lognormal cumulative distribution of a value. Arg1 is the probability that will have to be inversed. Arg2 is the mean of ln(value). Arg3 is the standard deviation of ln(value)

Continues

Name	Returns	Parameters	Description
`LogNormDist`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double`	Returns the cumulative lognormal distribution of `Arg1`. `Arg2` is the mean of ln(`Arg1`) and `Arg3` is the standard deviation of ln(`Arg1`)
`Lookup`	`Variant`	`Arg1, Arg2, [Arg3]`	The value `Arg1` is searched for in the single row or column in `Arg2`. A value from the matching spot in another array, `Arg3`, is returned
`Match`	`Double`	`Arg1, Arg2, [Arg3]`	Returns the relative position of an item in an array, `Arg2`, which matches a specific value, `Arg1`. Use `Arg3` to set the type of match

`Max`	`Double`	`Arg1,`
		`[Arg2],`
		`...`
		`[Arg30]`
			Returns the largest value in the numbers `Arg1` to `Arg30`. `Arg1` can also be a cell range

`MDeterm`	`Double`	`Arg1`	Returns the matrix determinant of the matrix array specified by `Arg1`. `Arg1` can also be a cell range. Cell elements cannot contain text or be empty. There must be an equal amount of rows to columns

`Median`	`Double`	`Arg1,`
		`[Arg2],`
		`...`
		`[Arg30]`
			Returns the median value in the numbers `Arg1` to `Arg30`. `Arg1` can also be a cell range

`Min`	`Double`	`Arg1,`
		`[Arg2],`
		`...`
		`[Arg30]`
			Returns the smallest value in the numbers `Arg1` to `Arg30`. `Arg1` can also be a cell range

`MInverse`	`Variant`	`Arg1`	Returns the inverse matrix for the matrix array in `Arg1`. `Arg1` can also be a cell range. Cell elements cannot contain text or be empty. There must be an equal amount of rows to columns

Name	Returns	Parameters	Description
`MIrr`	`Double`	`Arg1, Arg2 As Double, Arg3 As Double`	Returns the modified rate or return for a series of values in `Arg1.Arg2` is the interest rate paid. `Arg3` is the interest rate received as the cash flow values are reinvested
`MMult`	`Variant`	`Arg1, Arg2`	Returns the matrix product of the two matrix arrays `Arg1` and `Arg2`. The number of columns in `Arg1` must be the same as the number of rows in `Arg2. Arg1` and `Arg2` can also be a cell range. Cell elements cannot contain text or be empty
`Mode`	`Double`	`Arg1, [Arg2], ... [Arg30]`	Returns the most frequently occurring number in `Arg1` to `Arg30. Arg1` can also be a cell range
`NegBinom Dist`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double`	Returns the negative binomial distribution of the arguments. `Arg1` is the number of failures in the trials. `Arg2` holds the threshold number of successes. `Arg3` is the probability of success on a single trial
`NormDist`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Boolean`	Returns the normal cumulative distribution for the value to distribute (`Arg1`), the mean (`Arg2`), and the standard deviation (`Arg3`). Set `Arg4` to `True` to return the cumulative distribution and `False` to return the probability mass
`NormInv`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double`	Returns the inverse of the normal cumulative distribution given a probability (`Arg1`), the mean (`Arg2`), and the standard deviation (`Arg3`)
`NormSDist`	`Double`	`Arg1 As Double`	Returns the standard normal cumulative distribution for the value to distribute (`Arg1`)

Continues

Name	Returns	Parameters	Description
NormSInv	Double	Arg1 As Double	Returns the inverse of the standard normal cumulative distribution for a given probability (Arg1)
NPer	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, [Arg4], [Arg5]	Returns the number of periods for an investment. Arg1 is the interest rate per period. Arg2 is the payment amount made each period. Arg3 is the initial value. Arg4 is the future value that is wanted to be attained. Arg5 determines if payments are due at the end of the period (0) or the beginning of the period (1)
Npv	Double	Arg1 As Double, Arg2, [Arg3], ... [Arg30]	Returns the net present value of an investment using a discount rate (Arg1) and many future payments and income (Arg2 to Arg30)
Odd	Double	Arg1 As Double	Converts the Arg1 number into the nearest odd whole number, rounded up, and returns it
Or	Boolean	Arg1, [Arg2], ... [Arg30]	Returns True if any of the expressions in Arg1 to Arg30 returns True
Pearson	Double	Arg1, Arg2	Returns the Pearson product moment correlation coefficient containing an array of values. Arg1 is the array of independent values and Arg2 is the array of dependent values. Arg1 and Arg2 can also be cell references
Percentile	Double	Arg1, Arg2 As Double	Returns the Arg2 percentile of values in the Arg1 range of cells or array
Percent Rank	Double	Arg1, Arg2 As Double, [Arg3]	Returns how the value Arg2 ranks in the Arg1 range of cells or array. Arg3 can specify the number of significant digits for the returned percentage

Name	Returns	Parameters	Description
`Permut`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the total possible number of permutations of a group of `Arg2` items in a total number of `Arg1` items
`Phonetic`	`String`	`Arg1 As Range`	Returns the phonetic characters from the `Arg1` text string
`Pi`	`Double`		Returns pi (3.14) to 15 decimal places
`Pmt`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double, [Arg4], [Arg5]`	Returns the payment for a loan. `Arg1` is the interest rate per period. `Arg2` is the number of payments for the loan. `Arg3` is the initial value. `Arg4` is the future value that is wanted to be attained. `Arg5` determines if payments are due at the end of the period (0) or the beginning of the period (1)
`Poisson`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Boolean`	Returns the Poisson distribution given the number of events (`Arg1`) and the expected numeric value (`Arg2`). Set `Arg3` to `True` to return the cumulative probability and `False` to return the probability mass
`Power`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the base number `Arg1` raised to the power of `Arg2`
`Ppmt`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3  As Double, Arg4 As Double, [Arg5], [Arg6]`	Returns the payment on the principal of an investment for a given period of time. `Arg1` is the interest rate per period. `Arg2` specifies the period to look at. `Arg3` is the total number of payments. `Arg4` is the initial value. `Arg5` is the future value that is wanted to be attained. `Arg6` determines if payments are due at the end of the period (0) or the beginning of the period (1)
`Prob`	`Double`	`Arg1, Arg2, Arg3 As Double, [Arg4]`	Returns the probability that the values in the `Arg1` array and associated `Arg2` array are within the lower limit (`Arg3`) and upper limit (`Arg4`)

Continues

Name	Returns	Parameters	Description

`Product`	`Double`	`Arg1,`
		`[Arg2],`
		`...`
		`[Arg30]`
			Returns the multiplication product of all the values in `Arg1` to `Arg30`

`Proper`	`String`	`Arg1 As String`	Capitalizes the start of every word in `Arg1` and makes everything else lowercase

`Pv`	`Double`	`Arg1 As Double,`
		`Arg2 As Double,`
		`Arg3 As Double,`
		`[Arg4],`
		`[Arg5]`
			Returns the present value of an investment. `Arg1` is the interest rate per period. `Arg2` is the number of payments for the loan. `Arg3` is the payment amount made per period. `Arg4` is the future value that is wanted to be attained. `Arg5` determines if payments are due at the end of the period (0) or the beginning of the period (1)

`Quartile`	`Double`	`Arg1,`
		`Arg2 As Double`
			Returns the quartile specified by `Arg2` of the array in `Arg1`. `Arg2` can be 0 (Minimum value), 1 (first quartile), 2 (second quartile), 3 (third quartile), or 4 (maximum value)

`Radians`	`Double`	`Arg1 As Double`	Converts the `Arg1` number from degrees to radians and returns the new value

`Rank`	`Double`	`Arg1 As Double,`
		`Arg2 As Range,`
		`[Arg3]`
			Returns the rank of `Arg1` in the range `Arg2`. `Arg3` can be used to set how to rank `Arg1`

`Rate`	`Double`	`Arg1 As Double,`
		`Arg2 As Double,`
		`Arg3 As Double,`
		`[Arg4],`
		`[Arg5],`
		`[Arg6]`
			Returns the interest rate per period for a value. `Arg1` is the total number of payments. `Arg2` is the payment amount per period. `Arg3` is the initial value. `Arg4` is the future value that is wanted to be attained. `Arg5` determines if payments are due at the end of the period (0) or the beginning of the period (1)

Name	Returns	Parameters	Description
`Replace`	`String`	`Arg1 As String, Arg2 As Double, Arg3 As Double, Arg4 As String`	Replaces part of the text in `Arg1` with the text in `Arg4`. The starting character of the replacement is at the number `Arg2` and `Arg3` specifies the number of replaced characters in `Arg1`
`ReplaceB`	`String`	`Arg1 As String, Arg2 As Double, Arg3 As Double, Arg4 As String`	Replaces part of the text in `Arg1` with the text in `Arg4`. The starting character of the replacement is at the number `Arg2` and `Arg3` specifies the number of replaced bytes in `Arg1`
`Rept`	`String`	`Arg1 As String, Arg2 As Double`	Repeats the string in `Arg1` by `Arg2` number of times and returns that new string
`Roman`	`String`	`Arg1 As Double, [Arg2]`	Returns the number in `Arg1` to a Roman numeral equivalent. `Arg2` can specify the style of Roman numerals. 0 or `True` is the classic style. 4 or `False` is the simplified style. The other options are 2, 3, and 4 that set the style to varying degrees of simplification
`Round`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the `Arg1` number rounded to the number of digits specified in `Arg2`
`RoundDown`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the `Arg1` number rounded to the number of digits specified in `Arg2`. The number is rounded down towards 0
`RoundUp`	`Double`	`Arg1 As Double, Arg2 As Double`	Returns the `Arg1` number rounded to the number of digits specified in `Arg2`. The number is rounded up towards 0

Continues

Name	Returns	Parameters	Description
RSq	Double	Arg1, Arg2	Returns the square of the Pearson product moment correlation coefficient containing an array of values. Arg1 is the array of y values and Arg2 is the array of x values. Arg1 and Arg2 can also be cell references
RTD	Variant	progID As Variant, server As Variant, topic1 As Variant, [topic2], [topic3], [topic4], [topic5], [topic6], [topic7], [topic8], [topic9], [topic10], [topic11], [topic12], [topic13], [topic14], [topic15], [topic16], [topic17], [topic18], [topic19], [topic20], [topic21], [topic22], [topic23], [topic24], [topic25], [topic26], [topic27], [topic28]	Connects to a source to receive RTD. progID is the real-time server programmatic identifier, server is the server name, and topic1 is the topic to look up
Search	Double	Arg1 As String, Arg2 As String, [Arg3]	Finds the text in Arg1 from the text in Arg2 and returns the starting position of the found text. Arg3 can specify the starting position to search in Arg2

Name	Returns	Parameters	Description
`SearchB`	`Double`	`Arg1 As String, Arg2 As String, [Arg3]`	Finds the text in `Arg1` from the text in `Arg2` and returns the starting position of the found text. `Arg3` can specify the starting byte position to search in `Arg2`
`Sinh`	`Double`	`Arg1 As Double`	Returns the hyperbolic sine of the `Arg1` number
`Skew`	`Double`	`Arg1, [Arg2], ... [Arg30]`	Returns how skewed the numbers in `Arg1` to `Arg30` are. The arguments can also be a range reference
`Sln`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double`	Returns the simple straight-line depreciation of an asset costing `Arg1` with a salvage value of `Arg3` over `Arg2` number of periods
`Slope`	`Double`	`Arg1, Arg2`	Returns the slope of the linear regression line through the data points of the x values `(Arg1)` and y values `(Arg2)`
`Small`	`Double`	`Arg1, Arg2 As Double`	Returns the `Arg2` smallest value in the array or cell reference specified by `Arg1` (for example, second smallest, third smallest)
`Standardize`	`Double`	`Arg1 As Double, Arg2 As Double, Arg3 As Double`	Returns the normalized value from a distribution given a value `(Arg1)`, a mean `(Arg2)`, and a standard deviation `(Arg3)`
`StDev`	`Double`	`Arg1, [Arg2], ... [Arg30]`	Returns the estimated standard deviation of the values in `Arg1` to `Arg30`. The arguments can also be a range reference
`StDevP`	`Double`	`Arg1, [Arg2], ... [Arg30]`	Returns the standard deviation of the values in `Arg1` to `Arg30` based on all the values. The arguments can also be a range reference

Continues

Name	Returns	Parameters	Description
StEyx	Double	Arg1, Arg2	Returns the standard error of the predicted y values for each of the x values in the regression given some know y values (Arg1) and x values (Arg2)
Substitute	String	Arg1 As String, Arg2 As String, Arg3 As String, [Arg4]	Substitutes all occurrences of the Arg2 text with the Arg3 text in the original Arg1 text string. Arg4 can be used to specify which occurrence to replace
Subtotal	Double	Arg1 As Double, Arg2 As Range, [Arg3], ... [Arg30]	Returns subtotals for the ranges or references specified in the Arg2 to Arg30 parameters. Arg1 is a number describing what type of function to use for calculating the subtotal. Valid function numbers are from 1 to 10 representing Average, Count, CountA, Max, Min, Product, StcDev, StDevP, Sum, Var, and VarP in numerical order
Sum	Double	Arg1, [Arg2], ... [Arg30]	Returns the sum of all the numbers in Arg1 to Arg30
SumIf	Double	Arg1 As Range, Arg2, [Arg3]	Returns the sum of the cells in the range Arg1 with the criteria matching Arg2. A different range to sum can be specified with Arg3. The columns or rows in Arg1 and Arg3 have to be the same
SumProduct	Double	Arg1, [Arg2], ... [Arg30]	Multiplies each corresponding element in the arrays Arg1 to Arg30 and returns the sum of the products. The arrays in Arg1 to Arg30 must have the same dimension
SumSq	Double	Arg1, [Arg2], ... [Arg30]	Returns the sum of the square roots of the number in Arg1 to Arg30

Name	Returns	Parameters	Description
SumX2MY2	Double	Arg1, Arg2	Subtracts the squares of the corresponding elements in the arrays Arg1 and Argc2 and returns the sum of all the new elements
SumX2PY2	Double	Arg1, Arg2	Adds the squares of the corresponding elements in the arrays Arg1 and Arg2 and returns the sum of all the new elements
SumXMY2	Double	Arg1, Arg2	Subtracts the corresponding elements in the arrays Arg1 and Arg2, squares the difference and returns the sum of all the new elements
Syd	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Double	Returns the sum-of-years digits depreciation of an asset over a specified period. Arg1 is the initial cost. Arg2 is the salvage cost. Arg3 is the number of periods to depreciate the asset over. Arg4 is the specified period to return
Tanh	Double	Arg1 As Double	Returns the hyperbolic tangent of the Arg1 number
Tdist	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double	Returns the probability for the student t-distribution for a value, Arg1, is a calculated value of ‘t’. Arg2 indicates the number of degrees of freedom. Set Arg3 to 1 to return a one-tailed distribution. Set Arg3 to 2 to return a two-tailed distribution
Text	String	Arg1, Arg2 As String	Converts the value in Arg1 into text using the formatting in Arg2
Tinv	Double	Arg1 As Double, Arg2 As Double	Returns the t-value of the Student’s t-distribution given the probability (Arg1) and the degrees of freedom (Arg2)
Transpose	Variant	Arg1	Transposes the range specified by Arg1 from column to row or vice versa and returns the new range

Continues

Name	Returns	Parameters	Description
Trend	Variant	Arg1, [Arg2], [Arg3], [Arg4]	Returns the values associated with a linear trend given some y values (Arg1), some x values (Arg2), and some new x values (Arg3). Set Arg4 to False to make 'b' equal to 0 in the equation
Trim	String	Arg1 As String	Returns the string in Arg1 without leading and trailing spaces
TrimMean	Double	Arg1, Arg2 As Double	Returns the mean of the interior of the data array in Arg1. Use Arg2 to specify the fractional part of the data array to exclude
Ttest	Double	Arg1, Arg2, Arg3 As Double, Arg4 As Double	Returns the probability associated with a student's t-Test given the two sets of data in Arg1 and Arg12. Set Arg3 to 1 to return a one-tailed distribution. Set Arg3 to 2 to return a two-tailed distribution. Set Arg4 to 1, 2, or 3 to set the type of t-Test to paired, two-sample equal variance, or two-sample unequal variance, respectively
USDollar	String	Arg1 As Double, Arg2 As Double	Returns a currency-type string of the number in Arg1 with the decimal points specified in Arg2
Var	Double	Arg1, [Arg2] ... [Arg30]	Returns the estimated variance based on the numbers in Arg1 to Arg30. The arguments can also be range references
VarP	Double	Arg1, [Arg2] ... [Arg30]	Returns the variance based on the numbers in Arg1 to Arg30 as an entire population. The arguments can also be range references
Vdb	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Double, Arg5 As Double, [Arg6], [Arg7]	Returns the double-declining balance method depreciation (unless otherwise specified) of an asset for specified periods. Arg1 is the initial cost. Arg2 is the salvage cost. Arg3 is the number of periods to depreciate the asset over. Arg4 is the starting period to calculate depreciation. Arg5 is the ending period to calculate depreciation. Arg6 is the rate at which the balance declines. Set Arg7 to True to keep the calculation using the double-declining balance method. Set Arg7 to False to have Excel switch to straight-line depreciation, when necessary

Name	Returns	Parameters	Description
Vlookup	Variant	Arg1, Arg2, Arg3, [Arg4]	Looks up the value specified by Arg1 in the table array (or range reference) Arg2 first column Arg3 specifies the column number in the table array that contains the matching value. Set the Arg4 to True to find approximate data or set Arg4 to False to only lookup exact values
Weekday	Double	Arg1, [Arg2]	Returns the numerical day of the week for the date in Arg1. Arg2 specifies which day is the start of the week
Weibull	Double	Arg1 As Double, Arg2 As Double, Arg3 As Double, Arg4 As Boolean	Returns the Weibull distribution using the value (Arg1). Arg2 is the alpha parameter to the distribution. Arg3 is the beta parameter to the distribution. Set Arg4 to True to return the cumulative probability and False to return the probability mass
Ztest	Double	Arg1, Arg2 As Double, [Arg3]	Returns the two-tailed P-value of a z-test. Arg1 is the array or range of data to test against Arg2. Arg2 is the value to test. Arg3 is the population standard deviation

Example: WorksheetFunction Object

```
Sub GetBiggest()
   Dim oWSF As WorksheetFunction
   Dim vaArray As Variant
   Set oWSF = Application.WorksheetFunction
   vaArray = Array(10, 20, 13, 15, 56, 12, 8, 45)
   MsgBox "Biggest is " & oWSF.Max(vaArray)
End Sub
```

XmlDataBinding Object

Represents the connection to a data soure for an XML Map.

Name	Returns	Description
Application	Object	Returns a reference to the Microsoft Excel application
Creator	Long	Returns a long integer which denotes the application that created this object
Parent	Object	Reference to the parent object
SourceUrl	String	Returns a string representing the XML data file

XmlDataBinding Methods

Name	Returns	Parameters	Description
ClearSettings	Boolean		Clears all settings for the current object
LoadSettings	Boolean		Loads a set of settings in
Refresh			Refreshes all data

XmlMap Object

The XMLMap object represents an XML Map that has been added to a workbook.

XmlMap Properties

Name	Returns	Description
AdjustColumn Width	Boolean	Set this to True to automatically adjust column widths when data is refreshed
AppendOnImport	Boolean	Set this to True and imported data will be appended to current data. Otherwise imported data will overwritten
Application	Long	Reference to the Excel Application
Creator	Long	Value representing the creator of the object
DataBinding	XmlDataBinding	
IsExportable	Boolean	Determines if the current data is exportable
Name	String	Name of this object
Parent	Object	Reference to the parent of this object
PreserveColumn Filter	Boolean	Determines if column filters are persisted
PreserveNumber Formatting	Boolean	Determines if number formatting is persisted
RootElement Name	String	Returns a string representing the root element name in the XML tree
RootElement Namespace	XmlNamespace	Returns a XmlNamespace object that represents the namespace of the current root element
SaveDataSource Definition	Boolean	Determines if the data source information is saved

Name	Returns	Description
`Schemas`	`XmlSchemas`	Returns a collection of `XmlSchema` objects that have been applied to the current workbook
`ShowImportExportValidationErrors`	`Boolean`	Set this to `True` and during import and export operations—any validation errors will be displayed

XmlMap Methods

Name	Returns	Parameters	Description
`Delete`			Deletes the current `XmlMap` object
`Export`			Exports the current `XmlMap` object
`ExportXml`			Exports the current `XmlMap` object as XML which can be persisted to a file
`Import`			Imports an `XmlMap` object
`ImportXml`			Imports an `XmlMap` object from XML