Microsoft Graph Objects

- Application
  - DataSheet
  - Range
  - Font
- AutoCorrect
- CommandBars (CommandBar)
- Chart
  - ChartArea
  - PlotArea
  - Floor
  - Walls
  - Corners
  - ChartTitle
  - SeriesCollection (Series)
  - Trendlines (Trendline)
- Axes (Axis)
  - AxisTitle
  - DisplayUnitLabel
  - Gridlines
  - TickLabels

- DataTable
  - Border
  - Font
- Legend
  - LegendEntries (LegendEntry)
  - LegendKey
- ChartGroups (ChartGroup)

Legend

Object and collection
Object only

› Click arrow to expand chart
Axes Collection

A collection of all the Axis objects in the specified chart.
Using the Axes Collection

Use the **Axes** method to return the **Axes** collection. The following example displays the number of axes in the chart.

```vba
With myChart
    MsgBox .Axes.Count
End With
```

Use **Axes**(type, group), where type is the axis type and group is the axis group, to return a single **Axis** object.

*Type* can be one of the following **XlAxisType** constants.

**XlAxisType** can be one of these **XlAxisType** constants.

- **xlCategory**
- **xlSeries**
- **xlValue**

*Group* can be either of the following **XlAxisGroup** constants: **xlPrimary** or **xlSecondary**. For more information, see the **Axes** method.

The following example sets the title text for the category axis.

```vba
With myChart.Axes(xlCategory)
    .HasTitle = True
    .AxisTitle.Caption = "1994"
End With
```
ChartGroups Collection Object

ChartGroups	Collection	Object

Multiple objects

A collection of all the ChartGroup objects in the specified chart. Each ChartGroup object represents one or more series plotted with the same format in a chart. A chart contains one or more chart groups, each chart group contains one or more series, and each series contains one or more points. For example, a single chart might contain both a line chart group, containing all the series plotted with the line chart format, and a bar chart group, containing all the series plotted with the bar chart format.
Using the ChartGroups Collection

Use the **ChartGroups** method to return the **ChartGroups** collection. The following example displays the number of chart groups in the chart.

```vba
MsgBox myChart.ChartGroups.Count
```

Use **ChartGroups(index)**, where `index` is the chart group's index number, to return a single **ChartGroup** object. The following example adds drop lines to chart group one in the chart.

```vba
myChart.ChartGroups(1).HasDropLines = True
```

Because the index number for a particular chart group can change if the chart format used for that group is changed, it may be easier to use one of the named chart-group shortcut methods to return a particular chart group. The **PieGroups** method returns the collection of pie chart groups in the specified chart, the **LineGroups** method returns the collection of line chart groups, and so on. Each of these methods can be used with an index number to return a single **ChartGroup** object, or used without an index number to return a **ChartGroups** collection. The following methods are available for chart groups:

- **AreaGroups** method
- **BarGroups** method
- **ColumnGroups** method
- **DoughnutGroups** method
- **LineGroups** method
- **PieGroups** method
DataLabels Collection Object

DataLabels

Multiple objects

A collection of all the DataLabel objects for the specified series. Each DataLabel object represents a data label for a point or trendline. For a series without definable points (such as an area series), the DataLabels collection contains a single data label.
Using the DataLabels Collection

Use the DataLabels method to return the DataLabels collection. The following example sets the number format for data labels in series one in the chart.

```vba
With myChart.SeriesCollection(1)
    .HasDataLabels = True
    .DataLabels.NumberFormat = "##.##"
End With
```

Use DataLabels(index), where index is the data label's index number, to return a single DataLabel object. The following example sets the number format for the fifth data label in series one in the chart.

```vba
myChart.SeriesCollection(1).DataLabels(5).NumberFormat = "0.000"
```
HiLoLines Object

ChartGroup ▼ HiLoLines ▼ Border

Represents the high-low lines in the specified chart group. High-low lines connect the highest point with the lowest point in every category in the chart group. Only 2-D line groups can have high-low lines. This object isn't a collection. There's no object that represents a single high-low line; either you have high-low lines turned on for all points in a chart group or you have them turned off.
Using the HiLoLines Object

Use the HiLoLines property to return the HiLoLines object. The following example makes the high-low lines in chart group one in the chart blue.

```
myChart.ChartGroups(1).HiLoLines.Border.Color = RGB(0, 0, 255)
```
Remarks

If the HasHiLoLines property is False, most properties of the HiLoLines object are disabled.
LegendEntries Collection Object

A collection of all the LegendEntry objects in the specified chart legend. Each legend entry has two parts: the text of the entry, which is the name of the series or trendline associated with the entry; and the entry marker, which visually links the legend entry with its associated series or trendline in the chart. The formatting properties for the entry marker and its associated series or trendline are contained in the LegendKey object.
Using the LegendEntries Collection

Use the LegendEntries method to return the LegendEntries collection. The following example loops through the collection of legend entries in the chart and changes their font color to blue.

```vbscript
With myChart.Legend
    For i = 1 To .LegendEntries.Count
        .LegendEntries(i).Font.ColorIndex = 5
    Next
End With
```

Use `LegendEntries(index)`, where `index` is the legend entry's index number, to return a single LegendEntry object. You cannot return legend entries by name.

The index number represents the position of the legend entry in the legend. `LegendEntries(1)` is at the top of the legend; `LegendEntries(LegendEntries.Count)` is at the bottom. The following example changes the font style to italic for the text of the legend entry at the top of the legend (this is usually the legend for series one) in `myChart`.

```vbscript
myChart.Legend.LegendEntries(1).Font.Italic = True
```
Points Collection Object

A collection of all the **Point** objects in the specified series in a chart.
Using the Points Collection

Use the **Points** method to return the **Points** collection. The following example adds a data label to the last point in series one in the chart.

```vba
Dim pts As Points
Set pts = myChart.SeriesCollection(1).Points
pts(pts.Count).ApplyDataLabels Type:=xlShowValue
```

Use **Points(index)**, where `index` is the point's index number, to return a single **Point** object. Points are numbered from left to right in the series. **Points(1)** is the leftmost point, and **Points(Points.Count)** is the rightmost point. The following example sets the marker style for the third point in series one in the chart. The specified series must be a 2-D line, scatter, or radar series.

```vba
myChart.SeriesCollection(1).Points(3).MarkerStyle = xlDiamond
```
SeriesCollection Collection Object

A collection of all the Series objects in the specified chart or chart group.
Using the SeriesCollection Collection

Use the SeriesCollection method to return the SeriesCollection collection. The following example adjusts the interior color for each series in the collection:

For X = 1 To myChart.SeriesCollection.Count
    With myChart.SeriesCollection(X)
        .Interior.Color = RGB(X * 75, 50, X * 50)
    End With
Next X

Use SeriesCollection(index), where index is the series' index number or name, to return a single Series object. The following example sets the color of the interior for series one in the chart to red.

myChart.SeriesCollection(1).Interior.Color = RGB(255, 0, 0)
A collection of all the Trendline objects for the specified series. Each Trendline object represents a trendline in a chart. A trendline shows the trend, or direction, of data in a series.
Using the Trendlines Collection

Use the `Trendlines` method to return the `Trendlines` collection. The following example displays the number of trendlines for series one in the chart.

```
MsgBox myChart.SeriesCollection(1).Trendlines.Count
```

Use the `Add` method to create a new trendline and add it to the series. The following example adds a linear trendline to series one in the chart.

```
With myChart.SeriesCollection(1).Trendlines
    .Add Type:=xlLinear, Name:="Linear Trend"
End With
```

Use `Trendlines(index)`, where `index` is the trendline's index number, to return a single `TrendLine` object. The following example changes the trendline type for series one in the chart. If the series has no trendline, this example will fail.

```
myChart.SeriesCollection(1).Trendlines(1).Type = xlMovingAvg
```

The index number denotes the order in which the trendlines are added to the series. `Trendlines(1)` is the first trendline added to the series, and `Trendlines(Trendlines.Count)` is the last one added.
Application Object

Represented by the `Application` object, which represents the entire Microsoft Graph application. This object contains all of the objects, properties, and methods for the application.
Using the Application Object

Use the `Application` property to return the `Application` object. The following example applies the `DataSheet` property to the `Application` object.

```
```
AutoCorrect Object

Contains Microsoft Graph AutoCorrect attributes (capitalization of names of days, correction of two initial capital letters, automatic correction list, and so on).
Using the AutoCorrect Object

Use the **AutoCorrect** property to return the **AutoCorrect** object. The following example sets Microsoft Graph to correct words that begin with two initial capital letters.

```vbnet
With myChart.Application.AutoCorrect
  .TwoInitialCapitals = True
  .ReplaceText = True
End With
```
Axis Object

Multiple objects

Represents a single axis in a chart. The \textbf{Axis} object is a member of the \textbf{Axes} collection.
Using the Axis Object

Use **Axes**(*type, group*), where *type* is the axis type and *group* is the axis group, to return a single **Axis** object. *Type* can be one of the following **XLAxisType** constants: **xlCategory**, **xlSeries**, or **xlValue**. *Group* can be either of the following **XLAxisGroup** constants: **xlPrimary** or **xlSecondary**. For more information, see the **Axes** method.

The following example sets the text of the category axis title in the chart.

```vba
With myChart.Axes(xlCategory)
  .HasTitle = True
  .AxisTitle.Caption = "1994"
End With
```
AxisTitle Object

Represented by AxisTitle

Multiple objects

Represents the title of an axis in a chart.
Using the **AxisTitle** Object

Use the **AxisTitle** property to return an **AxisTitle** object. The following example sets the text of the value axis title and sets the font to 10-point Bookman.

```vba
With myChart.Axes(xlValue)
    .HasTitle = True
    With .AxisTitle
        .Caption = "Revenue (millions)"
        .Font.Name = "bookman"
        .Font.Size = 10
    End With
End With
```
Remarks

The **AxisTitle** object doesn't exist and cannot be used unless the **HasTitle** property for the specified axis is **True**.
**Border Object**

Multiple objects

*Border*

Represents the border of the specified object.
Using the Border Object

An object's border is treated as a single entity and is always returned as a unit (in its entirety), regardless of how many sides it has. Use the Border property to return the Border object. The following example places a dashed border around the chart area and places a dotted border around the plot area.

```
With myChart
    .ChartArea.Border.LineStyle = xlDash
    .PlotArea.Border.LineStyle = xlDot
End With
```
Chart Object

Multiple objects

Represents the specified Microsoft Graph chart.
Using the Chart Object

Use the Chart property to return a Chart object. Most of the time, you'll create a reference to a Microsoft Graph chart and then use the reference in your code.
ChartArea Object

Represents the chart area of the specified chart. The chart area in a 2-D chart contains the axes, the chart title, the axis titles, and the legend. The chart area in a 3-D chart contains the chart title and the legend; it doesn't include the plot area (the area within the chart area where the data is plotted). For information about formatting the plot area, see the PlotArea object.
Using the ChartArea Object

Use the ChartArea property to return the ChartArea object. The following example sets the pattern for the chart area.

\[ \text{myChart.ChartArea.Interior.Pattern} = \text{xllightDown} \]
ChartColorFormat Object

ChartFillFormat ← ChartColorFormat

Represents a foreground or background color.
Using the ChartColorFormat Object

Use the ForeColor property to return a ChartColorFormat object that represents the foreground fill color. Use the BackColor property to return the background fill color. Use the RGB property to return the color as an explicit red-green-blue value, and use the SchemeColor property to return or set the color as one of the colors in the current color scheme. The following example sets the foreground color, background color, and gradient for the chart area fill in myChart.

With myChart.ChartArea.Fill
  .Visible = True
  .ForeColor.SchemeColor = 15
  .BackColor.SchemeColor = 17
  .TwoColorGradient msoGradientHorizontal, 1
End With
ChartFillFormat Object

Multiple objects

- ChartFillFormat
- ChartColorFormat

Represents fill formatting.
Using the ChartFillFormat Object

Use the Fill property to return the ChartFillFormat object. The following example sets the foreground color, background color, and gradient for the chart area fill in myChart.

With myChart.ChartArea.Fill
    .Visible = True
    .ForeColor.SchemeColor = 15
    .BackColor.SchemeColor = 17
    .TwoColorGradient msoGradientHorizontal, 1
End With
ChartGroup Object

ChartGroups - ChartGroup
Multiple objects

Represents one or more series of points plotted in a chart with the same format. A chart contains one or more chart groups, each chart group contains one or more series, and each series contains one or more points. For example, a single chart might contain both a line chart group, which contains all the series plotted with the line chart format, and a bar chart group, which contains all the series plotted with the bar chart format. The ChartGroup object is a member of the ChartGroups collection.
**Using the ChartGroup Object**

Use `ChartGroups(index)`, where `index` is the chart group's index number, to return a single `ChartGroup` object. The following example adds drop lines to chart group one in the chart.

```
myChart.ChartGroups(1).HasDropLines = True
```

Because the index number for a particular chart group can change if the chart format used for that group is changed, it may be easier to use one of the named shortcut methods for chart groups to return a particular chart group. The `PieGroups` method returns the collection of pie chart groups in a chart, the `LineGroups` method returns the collection of all the line chart groups, and so on. You can use each of these methods with an index number to return a single `ChartGroup` object, or you can use each one without an index number to return a `ChartGroups` collection. The following methods are available for chart groups:

- `AreaGroups` method
- `BarGroups` method
- `ColumnGroups` method
- `DoughnutGroups` method
- `LineGroups` method
- `PieGroups` method
ChartTitle Object

Chart  ChartTitle
Multiple objects

Represents the title of the specified chart.
Using the ChartTitle Object

Use the **ChartTitle** property to return the **ChartTitle** object. The following example adds a title to the chart.

```vba
With myChart
    .HasTitle = True
    .ChartTitle.Text = "February Sales"
End With
```
Remarks

The ChartTitle object doesn't exist and cannot be used unless the HasTitle property for the chart is True.
Corners Object

Chart Corners

Represents the corners of the specified 3-D chart. This object isn't a collection.
Using the Corners Object

Use the `Corners` property to return the `Corners` object. The following example selects the corners of the chart.

```javascript
myChart.Corners.Select
```

If the chart isn't a 3-D chart, the `Corners` property fails.
DataLabel Object

Multiple objects ▼ DataLabel
▼ Multiple objects

Represents the data label for the specified point or trendline in a chart. For a series, the DataLabel object is a member of the DataLabels collection, which contains a DataLabel object for each point. For a series without definable points (such as an area series), the DataLabels collection contains a single DataLabel object.
Using the DataLabel Object

Use **DataLabels(index)**, where *index* is the data label's index number, to return a single **DataLabel** object. The following example sets the number format for the fifth data label in series one in the chart.

```vba
myChart.SeriesCollection(1).DataLabels(5).NumberFormat = "0.000"
```

Use the **DataLabel** property to return the **DataLabel** object for a single point. The following example turns on the data label for the second point in series one in the chart, and sets the data label text to "Saturday."

```vba
With myChart
    With .SeriesCollection(1).Points(2)
        .HasDataLabel = True
        .DataLabel.Text = "Saturday"
    End With
End With
```

For a trendline, the **DataLabel** property returns the text shown with the trendline. This can be the equation, the R-squared value, or both (if both are showing). The following example sets the trendline text to show only the equation and then places the data label text in cell A1 on the datasheet.

```vba
With myChart.SeriesCollection(1).Trendlines(1)
    .DisplayRSquared = False
    .DisplayEquation = True
    x = .DataLabel.Text
End With
With myChart.Application.DataSheet
    .Range("A1").Value = x
End With
```
DataSheet Object

Application DataSheet
Multiple objects

Represents the Microsoft Graph datasheet.
Using the DataSheet Object

After you've established a reference to a chart, you can use the Application property of the chart to retrieve the datasheet. The following example applies the DataSheet property to the Application object, and then it applies the Range property to the datasheet to set the value of cell A1 to 32.

Remarks

On the datasheet, the first column heading (starting on the left) is A, followed by B, C, D, and so on. The first row heading (starting on the left) is 1, followed by 2, 3, 4, and so on. Neither the leftmost column nor the top row has a heading. In other words, column A is actually the second column from the left; likewise, row 1 is the second row from the top. The leftmost column and the top row, which are commonly used for legend text or axis labels, are referred to as column 0 (zero) and row 0 (zero). Thus, the following example inserts the text "Annual Sales" in the top cell in column A (the second column).

\[
\text{myChart.Application.DataSheet.Range("A0").Value = "Annual Sales"}
\]

And the following example inserts the text "District 1" in the leftmost cell in row 2 (the third row).

\[
\text{myChart.Application.DataSheet.Range("02").Value = "District 1"}
\]
**DataTable Object**

Chart DataTable

Multiple objects

Represents a data table in the specified chart.
Using the DataTable Object

Use the **DataTable** property to return a **DataTable** object. The following example adds a data table with an outline border to the embedded chart.

```vba
With myChart
    .HasDataTable = True
    .DataTable.HasBorderOutline = True
End With
```
DisplayUnitLabel Object

Axis
  DisplayUnitLabel
  Multiple objects

Represents a unit label on the value axis in the specified chart. Unit labels are useful for charting large values— for example, numbers in the millions or billions. You can make the chart more readable by using a single unit label instead of large numbers with strings of zeros next to the tick marks on the axis. This way, you need never have numbers of more than one or two digits by the tick marks.
Using the DisplayUnitLabel Object

Use the `DisplayUnitLabel` property to return the `DisplayUnitLabel` object. The following example sets the caption for the value axis in `myChart` to “Millions” and turns off automatic font scaling.

```vba
With myChart.Axes(xlValue).DisplayUnitLabel
    .Caption = "Millions"
    .AutoScaleFont = False
End With
```
**DownBars Object**

*ChartGroup* ➔ *DownBars*

Multiple objects

Represents the down bars in the specified chart group. Down bars connect points in the first series in the chart group with lower values in the last series (the lines go down from the first series). Only 2-D line groups that contain at least two series can have down bars. This object isn't a collection. There's no object that represents a single down bar; either you have up bars and down bars turned on for all points in a chart group or you have them turned off.
Using the DownBars Object

Use the **DownBars** property to return the **DownBars** object. The following example turns on up and down bars for chart group one in the chart. The example then sets the up-bar color to blue and the down-bar color to red.

```vba
With myChart.ChartGroups(1)
    .HasUpDownBars = True
    .UpBars.Interior.Color = RGB(0, 0, 255)
    .DownBars.Interior.Color = RGB(255, 0, 0)
End With
```
Remarks

If the **HasUpDownBars** property is **False**, most properties of the **DownBars** object are disabled.
DropLines Object

ChartGroup ─ DropLines ─ Border

Represents the drop lines in the specified chart group. Drop lines connect the points in the chart with the x-axis. Only line and area chart groups can have drop lines. This object isn't a collection. There's no object that represents a single drop line; either you have drop lines turned on for all points in a chart group or you have them turned off.
Using the DropLines Object

Use the **DropLines** property to return the **DropLines** object. The following example turns on drop lines for chart group one in the chart and then sets the drop-line color to red.

```vba
myChart.ChartGroups(1).HasDropLines = True
myChart.ChartGroups(1).DropLines.Border.ColorIndex = 3
```
Remarks

If the HasDropLines property is False, most properties of the DropLines object are disabled.
ErrorBars Object

Series $\downarrow$ ErrorBars $\downarrow$ Border

Represents the error bars for the specified chart series. Error bars indicate the degree of uncertainty for chart data. Only series in area, bar, column, line, and scatter groups in a 2-D chart can have error bars. Only series in scatter groups can have x and y error bars.

This object isn't a collection. There's no object that represents a single error bar; either you have x error bars or y error bars turned on for all points in a series or you have them turned off.
Using the ErrorBars Object

Use the ErrorBars property to return the ErrorBars object. The following example turns on error bars for series one in myChart and then sets the end style for the error bars.

```plaintext
myChart.SeriesCollection(1).HasErrorBars = True
myChart.SeriesCollection(1).ErrorBars.EndStyle = xlNoCap
```
Remarks

The **ErrorBar** method changes the format and type of error bars.
Floor Object

Chart → Floor

Multiple objects

Represents the floor of the specified 3-D chart.
Using the Floor Object

Use the Floor property to return the Floor object. The following example sets the floor color for the chart to cyan. If the chart isn't a 3-D chart, this example will fail.

myChart.Floor.Interior.Color = RGB(0, 255, 255)
Font Object

Multiple objects

Contains the font attributes (font name, font size, color, and so on) for the specified object.
Using the Font Object

Use the **Font** property to return the **Font** object. The following example sets the title text for the value axis, sets the font to 10-point Bookman, and formats the word "millions" as italic.

```vba
With myChart.Axes(xlValue)
    .HasTitle = True
    With .AxisTitle
        .Caption = "Revenue (millions)"
        .Font.Name = "bookman"
        .Font.Size = 10
        .Characters(10, 8).Font.Italic = True
    End With
End With
```
Gridlines Object

Gridlines Object

Axis  Gridlines
Border

Gridlines Object

Represents major or minor gridlines on the specified chart axis. Gridlines extend the tick marks on a chart axis to make it easier to see the values associated with the data markers. This object isn't a collection. There's no object that represents a single gridline; either you have all gridlines for an axis turned on or you have them all turned off.
Using the Gridlines Object

Use the **MajorGridlines** property to return the **GridLines** object that represents the major gridlines for the axis. Use the **MinorGridlines** property to return the **GridLines** object that represents the minor gridlines for the axis. It's possible to return both major and minor gridlines at the same time.

The following example turns on major gridlines for the category axis on the chart and then formats the gridlines to be blue dashed lines.

```vba
With myChart.Axes(xlCategory)
    .HasMajorGridlines = True
    .MajorGridlines.Border.Color = RGB(0, 0, 255)
    .MajorGridlines.Border.LineStyle = xlDash
End With
```
**Interior Object**

Multiple objects $\text{Interior}$

Represents the interior of the specified object.
Using the Interior Object

Use the **Interior** property to return the **Interior** object. The following example sets the chart area color to gray and the plot area color to green.

```vbnet
With myChart
    .PlotArea.Interior.Color = RGB(0, 100, 150)
    .ChartArea.Interior.Color = RGB(50, 10, 50)
End With
```
LeaderLines Object

Series ▼ LeaderLines
  ▼ Border

Represents leader lines in the specified chart. Leader lines connect data labels to data points. This object isn't a collection; there's no object that represents a single leader line.
Using the LeaderLines Object

Use the LeaderLines property to return the LeaderLines object. The following example adds data labels and blue leader lines to series one in the chart.

With myChart.SeriesCollection(1)
    .HasDataLabels = True
    .DataLabels.Position = xlLabelPositionBestFit
    .HasLeaderLines = True
    .LeaderLines.Border.ColorIndex = 5
End With
Legend Object

Chart ▼ Legend
  ▼ Multiple objects

Represents the legend in the specified chart. Each chart can have only one legend. The Legend object contains one or more LegendEntry objects; each LegendEntry object contains a LegendKey object.
Using the Legend Object

Use the **Legend** property to return the **Legend** object. The following example sets the font style for the legend to bold.

```javascript
myChart.Legend.Font.Bold = True
```
Remarks

The chart legend isn't visible unless the HasLegend property is True. If this property is False, properties and methods of the Legend object will fail.
LegendEntry Object

LegendEntries → LegendEntry
   Multiple objects

Represents a legend entry in the specified chart legend. The LegendEntry object is a member of the LegendEntries collection, which contains all the LegendEntry objects in the legend.

Each legend entry has two parts: the text of the entry, which is the name of the series associated with the entry; and an entry marker, which visually links the legend entry with its associated series or trendline in the chart. Formatting properties for the entry marker and its associated series or trendline are contained in the LegendKey object.

You cannot change the text of a legend entry. LegendEntry objects support font formatting, and they can be deleted. No pattern formatting is supported for legend entries. The position and size of entries is fixed.
Using the LegendEntry Object

Use **LegendEntries**(index), where index is the legend entry's index number, to return a single **LegendEntry** object. You cannot return legend entries by name.

The index number represents the position of the legend entry in the legend. LegendEntries(1) is at the top of the legend, and LegendEntries(LegendEntries.Count) is at the bottom. The following example changes the font style for the text of the legend entry at the top of the legend (this is usually the legend for series one).

myChart.Legend.LegendEntries(1).Font.Italic = True
Remarks

There's no direct way to return the series or trendline that corresponds to a particular legend entry.

After legend entries have been deleted, the only way to restore them is to remove and then recreate the legend that contained them by setting the HasLegend property for the chart to False and then back to True.
LegendKey Object

LegendEntry - LegendKey
Multiple objects

Represents a legend key in the specified chart legend. Each legend key is a graphic that visually links a legend entry with its associated series or trendline in the chart. The legend key is linked to its associated series or trendline in such a way that changing the formatting of one simultaneously changes the formatting of the other.
Using the LegendKey Object

Use the **LegendKey** property to return the **LegendKey** object. The following example changes the marker background color to blue for the legend entry at the top of the legend in the chart. This simultaneously changes the formatting of every point in the series associated with this legend entry (if, that is, the associated series supports data markers).

```csharp
myChart.Legend.LegendEntries(1).LegendKey.MarkerBackgroundColorIndex = 5
```
PlotArea Object

Multiple objects

Represents the plot area of the specified chart. This is the area where your chart data is plotted. The plot area in a 2-D chart contains the data markers, gridlines, data labels, trendlines, and optional chart items placed in the chart area. The plot area in a 3-D chart contains all the above items plus the walls, floor, axes, axis titles, and tick-mark labels in the chart.

The plot area is surrounded by the chart area. The chart area in a 2-D chart contains the axes, the chart title, the axis titles, and the legend. The chart area in a 3-D chart contains the chart title and the legend. For information about formatting the chart area, see the ChartArea object.
Using the PlotArea Object

Use the **PlotArea** property to return the **PlotArea** object. The following example places a dashed border around the chart area and places a dotted border around the plot area.

```vbnet
With myChart
  .ChartArea.Border.LineStyle = xlDash
  .PlotArea.Border.LineStyle = xlDot
End With
```
Point Object

Points ⊆ Point

Multiple objects

Represents a single point in a series on the specified chart. The Point object is a member of the Points collection, which contains all the points in the specified series.
Using the Point Object

Use **Points(index)**, where *index* is the point's index number, to return a single **Point** object. Points are numbered from left to right in the series. **Points(1)** is the leftmost point, and **Points(Points.Count)** is the rightmost point. The following example sets the marker style for the third point in series one. For this example to work, series one must be a 2-D line, scatter, or radar series.

```
myChart.SeriesCollection(1).Points(3).MarkerStyle = xlDiamond
```
Range Object

Multiple objects \(\text{Range}\)

\(\text{Range}\)

Represents a cell, a row, a column, or a selection of cells that contains one or more contiguous blocks of cells.
Using the Range Object

The following properties for returning a Range object are described in this section:

- Range property
- Cells property
Range Property

Use Range(arg), where arg is the name of the range, to return a Range object that represents a single cell or a range of cells. The following example places the value of cell A1 in cell A5.

myChart.Application.DataSheet.Range("A5").Value = __

The following example fills the range A1:H8 with the value 20.

Cells Property

Use `Cells(row, column)`, where `row` is the row's index number and `column` is the column's index number, to return a single cell. The following example sets the value of cell A1 to 24 (column A is the second column on the datasheet, and row 1 is the second row on the datasheet).

```vba
```

Although you can also use `Range("A1")` to return cell A1, there may be times when the `Cells` property is more convenient because you can use a variable for the row or column. The following example creates column and row headings on the datasheet.

```vba
Sub SetUpTable()
    With myChart.Application.DataSheet
        For theYear = 1 To 5
            .Cells(1, theYear + 1).Value = 1990 + theYear
        Next theYear
        For theQuarter = 1 To 4
            .Cells(theQuarter + 1, 1).Value = "Q" & theQuarter
        Next theQuarter
    End With
End Sub
```

Although you can use Visual Basic string functions to alter A1-style references, it's much easier (and much better programming practice) to use the `Cells(1, 1)` notation.

Use `expression.Cells(row, column)`, where `expression` is an expression that returns a `Range` object, and `row` and `column` are relative to the upper-left corner of the range, to return part of a range. The following example sets the value for cell C5.

```vba
myChart.Application.Range("C5:C10").Cells(1, 1).Value = 35
```
Series Object

SeriesCollection → Series → Multiple objects

Represents a series in the specified chart. The Series object is a member of the SeriesCollection collection.
**Using the Series Object**

Use `SeriesCollection(index)`, where `index` is the series' index number or name, to return a single `Series` object. The following example sets the color of the interior for series one in the chart.

```plaintext
myChart.SeriesCollection(1).Interior.Color = RGB(255, 0, 0)
```

The series index number indicates the order in which the series are added to the chart. `SeriesCollection(1)` is the first series added to the chart, and `SeriesCollection(SeriesCollection.Count)` is the last one added.
SeriesLines Object

ChartGroup ▼ SeriesLines
  ▼ Border

Represents series lines in the specified chart group. Series lines connect the data values in each series. Only 2-D stacked-bar or column chart groups can have series lines. This object isn't a collection. There's no object that represents a single series line; either you have series lines turned on for all points in a chart group or you have them turned off.
Using the SeriesLines Object

Use the **SeriesLines** property to return the **SeriesLines** object. The following example adds series lines to chart group one in the chart. The chart must be a 2-D stacked-bar or column chart.

```vbnet
With myChart.ChartGroups(1)
    .HasSeriesLines = True
    .SeriesLines.Border.Color = RGB(0, 0, 255)
End With
```
Remarks

If the HasSeriesLines property is False, most properties of the SeriesLines object are disabled.
**TickLabels Object**

Multiple objects

- **TickLabels**
- **Font**

Represents the tick-mark labels associated with tick marks on the specified chart axis. This object isn't a collection. There's no object that represents a single tick-mark label; you must return all the tick-mark labels as a unit.

Tick-mark label text for the category axis comes from the name of the associated category in the chart. The default tick-mark label text for the category axis is the number that indicates the position of the category relative to the left end of this axis. To change the number of unlabeled tick marks between tick-mark labels, you must change the **TickLabelSpacing** property for the category axis.

Tick-mark label text for the value axis is calculated based on the **MajorUnit**, **MinimumScale**, and **MaximumScale** properties of the value axis. To change the tick-mark label text for the value axis, you must change the values of these properties.
Using the TickLabels Object

Use the **TickLabels** property to return the **TickLabels** object. The following example sets the number format for the tick-mark labels on the value axis in the chart.

```plaintext
myChart.Axes(xlValue).TickLabels.NumberFormat = "0.00"
```
Trendline Object

Represents a trendline in the specified chart. A trendline shows the trend, or direction, of data in a series. The Trendline object is a member of the Trendlines collection, which contains all the Trendline objects for a single series.
Using the Trendline Object

Use **Trendlines(index)**, where *index* is the trendline's index number, to return a single **Trendline** object. The following example changes the trendline type for series one in the chart. If the series has no trendline, this example will fail.

```
myChart.SeriesCollection(1).Trendlines(1).Type = xlMovingAvg
```

The index number denotes the order in which the trendlines are added to the series. **Trendlines(1)** is the first trendline added to the series, and **Trendlines(Trendlines.Count)** is the last one added.
UpBars Object

ChartGroup ▼ UpBars
   ▼ Multiple objects

Represents the up bars in a chart group. Up bars connect points in series one with higher values in the last series in the chart group (the lines go up from series one). Only 2-D line groups that contain at least two series can have up bars. This object isn't a collection. There's no object that represents a single up bar; either you have up bars turned on for all points in a chart group or you have them turned off.
Using the UpBars Object

Use the **UpBars** property to return the **UpBars** object. The following example turns on up and down bars for chart group one in the chart. The example then sets the up-bar color to blue and sets the down-bar color to red.

```vba
With myChart.ChartGroups(1)
   .HasUpDownBars = True
   .UpBars.Interior.Color = RGB(0, 0, 255)
   .DownBars.Interior.Color = RGB(255, 0, 0)
End With
```
Remarks

If the HasUpDownBars property is False, most properties of the UpBars object are disabled.
Walls Object

Chart ▼ Walls
 ▼ Multiple objects

Represents the walls of the specified 3-D chart. This object isn't a collection. There's no object that represents a single wall; you must return all the walls as a unit.
Using the Walls Object

Use the **Walls** property to return the **Walls** object. The following example sets the pattern on the walls for the chart. If the chart isn’t a 3-D chart, this example will fail.

```plaintext
myChart.Walls.Interior.Pattern = xlGray75
```
Activate Method

Activates the object.

\textit{expression}\.\textbf{Activate}

\textit{expression} Required. An expression that returns an object in the Applies To list.
Example

This example activates the datasheet.

myChart.Application.DataSheet.Activate
Add Method

Creates a new trend line. Returns a Trendline object.

expression.Add(Type, Order, Period, Forward, Backward, Intercept, DisplayEquation, DisplayRSquared, Name)

expression Required. An expression that returns one of the objects in the Applies To list.

Type Optional XlTrendlineType. The type of trendline.

XlTrendlineType can be one of these XlTrendlineType constants.

xlExponential
xlLinear default
xlLogarithmic
xlMovingAvg
xlPolynomial
xlPower

Order Optional Variant. Required if Type is xlPolynomial. The trendline order. Must be an integer from 2 through 6.

Period Optional Variant. Required if Type is xlMovingAvg. The trendline period. Must be an integer greater than 1 and less than the number of data points in the series you're adding a trendline to.

Forward Optional Variant. The number of periods (or units on a scatter chart) that the trendline extends forward.

Backward Optional Variant. The number of periods (or units on a scatter chart) that the trendline extends backward.

Intercept Optional Variant. The trendline intercept. If this argument is omitted, the intercept is automatically set by the regression.
**DisplayEquation** Optional Variant. **True** to display the equation of the trendline on the chart (in the same data label as the R-squared value). The default value is **False**.

**DisplayRSquared** Optional Variant. **True** to display the R-squared value of the trendline on the chart (in the same data label as the equation). The default value is **False**.

**Name** Optional Variant. The name of the trendline, as text. If this argument is omitted, Microsoft Graph generates a name.
Example

This example creates a new linear trendline on the chart.

myChart.SeriesCollection(1).Trendlines.Add
AddChartAutoFormat Method

Adds a custom chart autoformat to the list of available chart autoformats.

expression.AddChartAutoFormat(Name, Description)

expression Required. An expression that returns an Application object.

Name Required String. The name of the autoformat.

Description Optional String. A description of the custom autoformat.
Example

This example adds a new autoformat.

```plaintext
gmyChart.Application.AddChartAutoFormat _
   Name:="Presentation Chart"
```
AddReplacement Method

Adds an entry to the array of AutoCorrect replacements.

expression.AddReplacement(What, Replacement)

expression Required. An expression that returns an AutoCorrect object.

What Required String. The text to be replaced. If this string already exists in the array of AutoCorrect replacements, the existing substitute text is replaced by the new text.

Replacement Required String. The replacement text.
Example

This example substitutes the word "Temp." for the word "Temperature" in the array of AutoCorrect replacements.

```vbnet
With myChart.Application.AutoCorrect
    .AddReplacement "Temperature", "Temp."
End With
```
ApplyCustomType Method

ApplyCustomType method as it applies to the Series object.

Applies a standard or custom chart type to a series.

expression.ApplyCustomType(ChartType)

expression Required. An expression that returns one of the above objects.

ChartType Required XLChartType. A standard chart type.

XLChartType can be one of these XLChartType constants.

xl3DAreaStacked
xlConeCol
xlConeColStacked
xlCylinderBarClustered
xlCylinderBarStacked100
xlCylinderColClustered
xlCylinderColStacked100
xlDoughnutExploded
xlLineMarkers
xlLineMarkersStacked100
xlLineStacked100
xlPieExploded
xlPyramidBarClustered
xlPyramidBarStacked100
xlPyramidColClustered
xlPyramidColStacked100
xlRadarFilled
xlStockHLC
xlStockVHLC
xlSurface
xlSurfaceTopViewWireframe
xlXYScatter
xlXYScatterLinesNoMarkers
xlXYScatterSmoothNoMarkers
xl3DArea
xl3DAreaStacked100
xl3DBarClustered
xl3DBarStacked
xl3DBarStacked100
xl3DColumn
xl3DColumnClustered
xl3DColumnStacked
xl3DColumnStacked100
xl3DLine
xl3DPie
xl3DPieExploded
xlArea
xlAreaStacked
xlAreaStacked100
xlBarClustered
xlBarOfPie
xlBarStacked
xlBarStacked100
xlBubble
xlBubble3DEffect
xlColumnClustered
xlColumnStacked
xlColumnStacked100
xlConeBarClustered
xlConeBarStacked
xlConeBarStacked100
xlConeColClustered
xlConeColStacked100
ApplyCustomType method as it applies to the Chart object.

Applies a standard or custom chart type to a chart.

expression.ApplyCustomType(ChartType, TypeName)

expression Required. An expression that returns one of the above objects.

ChartType Required XlChartType. A standard chart type.

XlChartType can be one of these XlChartType constants.

xl3DAreaStacked
xlConeCol
xlConeColStacked
xlArea
xlAreaStacked
xlAreaStacked100
xlBarClustered
xlBarOfPie
xlBarStacked
xlBarStacked100
xlBubble
xlBubble3DEffect
xlColumnClustered
xlColumnStacked
xlColumnStacked100
xlConeBarClustered
xlConeBarStacked
xlConeBarStacked100
xlConeColClustered
xlConeColStacked
xlConeColStacked100
xlCylinderBarStacked
xlCylinderCol
xlCylinderColStacked
xlDoughnut
xlLine
xlLineMarkersStacked
xlLineStacked
xlPie
xlPieOfPie
xlPyramidBarStacked
xlPyramidCol
xlPyramidColStacked
xlRadar
xlRadarMarkers
xlStockOHLC
xlStockVOHLC
xlSurfaceTopView
xlSurfaceWireframe
xlXYScatterLines
xlXYScatterSmooth

TypeName  Optional Variant. A String naming the custom chart type when ChartType specifies a custom chart gallery.
Example

This example applies the line with the markers chart type.

myChart.ApplyCustomType xlLineMarkers
ApplyDataLabels Method

ApplyDataLabels method as it applies to the Chart object.

Applies data labels to a point, a series, or all the series in a chart.

expression. ApplyDataLabels (Type, LegendKey, AutoText, HasLeaderLines)

expression Required. An expression that returns a Chart object.

Type Optional Variant. The data label type.

XlDataLabelsType can be one of these XlDataLabelsType constants.

xlDataLabelsShowBubbleSizes The bubble size for the data label.
xlDataLabelsShowLabel Category for the point.
xlDataLabelsShowLabelAndPercent Percentage of the total, and category for the point. Available only for pie charts and doughnut charts.
xlDataLabelsShowNone No data labels.
xlDataLabelsShowPercent Percentage of the total. Available only for pie charts and doughnut charts.
xlDataLabelsShowValue default. Value for the point.

LegendKey Optional Variant. True to show the legend key next to the point. The default value is False.

AutoText Optional Variant. True if the object automatically generates appropriate text based on content.

HasLeaderLines Optional Variant. True if the series has leader lines.

Applies data labels to a point, a series, or all the series in a chart.

expression. ApplyDataLabels (Type, LegendKey, AutoText, HasLeaderLines, ShowSeriesName, ShowCategoryName, ShowValue, ShowPercentage,
**ShowBubbleSize, Separator**

*expression* Required. An expression that returns one of the above objects.

*Type* Optional [xlDataLabelsType](#). The data label type.

XLDataLabelsType can be one of these XLDataLabelsType constants.

- **xlDataLabelsShowBubbleSizes** The bubble size for the data label.
- **xlDataLabelsShowLabel** Category for the point.
- **xlDataLabelsShowLabelAndPercent** Percentage of the total, and category for the point. Available only for pie charts and doughnut charts.
- **xlDataLabelsShowNone** No data labels.
- **xlDataLabelsShowPercent** Percentage of the total. Available only for pie charts and doughnut charts.
- **xlDataLabelsShowValue** default. Value for the point.

**LegendKey** Optional [Variant](#). True to show the legend key next to the point. The default value is False.

**AutoText** Optional [Variant](#). True if the object automatically generates appropriate text based on content.

**HasLeaderLines** Optional [Variant](#). True if the series has leader lines.

**ShowSeriesName** Optional [Variant](#). The series name for the data label.

**ShowCategoryName** Optional [Variant](#). The category name for the data label.

**ShowValue** Optional [Variant](#). The value for the data label.

**ShowPercentage** Optional [Variant](#). The percentage for the data label.

**ShowBubbleSize** Optional [Variant](#). The bubble size for the data label.

**Separator** Optional [Variant](#). The separator for the data label.
Example

ApplyDataLabels method as it applies to the Series object.

This example applies category labels to series one.

myChart.SeriesCollection(1). _
    ApplyDataLabels Type:=xlDataLabelsShowLabel
AreaGroups Method

On a 2-D chart, this method returns an object that represents a single area chart group or a collection of all the area chart groups.

\[ expression\.AreaGroups(\text{Index}) \]

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Index} Optional Variant. The index number of the specified chart group.
Example

This example turns on drop lines for the 2-D area chart group.

myChart.AreaGroups(1).HasDropLines = True
AutoFit Method

Changes the width of the columns in the specified range to achieve the best fit.

expression.AutoFit

expression  Required. An expression that returns a Range object. Must be a row or a range of rows, or a column or a range of columns. Otherwise, this method causes an error.
Remarks

One unit of column width is equal to the width of one character in the Normal style.
Example

This example changes the width of columns A through I on the datasheet to achieve the best fit.


This example changes the width of columns A through E on the datasheet to achieve the best fit, based only on the contents of cells A1:E1.

Axes Method

Returns an object that represents either a single axis or a collection of the axes on the chart.

\[expression.Axes\left(Type, \textit{AxisGroup}\right)\]

- \textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Type} Optional \texttt{XlAxisType}. Specifies the axis to return. The reference style of the formula.

\texttt{XlAxisType} can be one of these \texttt{XlAxisType} constants.

- \texttt{xlValue}
- \texttt{xlCategory}
- \texttt{xlSeriesAxis} (valid only for 3-D charts)

\textit{AxisGroup} Optional \texttt{XlAxisGroup}. The reference style of the formula.

\texttt{XlAxisGroup} can be one of these \texttt{XlAxisGroup} constants.

- \texttt{xlPrimary}
- \texttt{xlSecondary}

If this argument is omitted, the primary group is used. 3-D charts have only one axis group.
Example

This example adds an axis label to the category axis.

```vba
With myChart.Axes(xlCategory)
    .HasTitle = True
    .AxisTitle.Text = "July Sales"
End With
```

This example turns off major gridlines for the category axis.

```vba
myChart.Axes(xlCategory).HasMajorGridlines = False
```

This example turns off all gridlines for all axes.

```vba
For Each a In myChart.Axes
    a.HasMajorGridlines = False
    a.HasMinorGridlines = False
Next a
```
BarGroups Method

On a 2-D chart, this method returns an object that represents either a single bar chart group or a collection of all the bar chart groups.

expression.**BarGroups**(Index)

*expression* Required. An expression that returns one of the objects in the Applies To list.

*Index* Optional **Variant**. The index number of the specified bar chart group.
Example

This example sets the space between bar clusters in the 2-D bar chart group to be 50 percent of the bar width.

myChart.BarGroups(1).GapWidth = 50
Chart Method

Returns a Chart object that represents the Microsoft Graph chart.
ChartGroups Method

Returns an object that represents either a single chart group or a collection of all the chart groups in the chart. The returned collection includes every type of group.

\(expression.\text{ChartGroups}(\text{Index})\)

expression  Required. An expression that returns one of the objects in the Applies To list.

Index  Optional Variant. The chart group number.
**Example**

This example turns on up and down bars for chart group one and then sets their colors. The example should be run on a 2-D line chart containing two series that intersect at one or more data points.

```vbnet
With myChart.ChartGroups(1)
    .HasUpDownBars = True
    .DownBars.Interior.ColorIndex = 3
End With
```
Clear Method

Clear method as it applies to the ChartArea and Legend objects.

Clears the entire chart area.

**expression.Clear**

*expression* Required. An expression that returns one of the above objects.

Clear method as it applies to the Range object.

Clears the entire range.

**expression.Clear**

*expression* Required. An expression that returns one of the above objects.
Example

This example clears the formulas and formatting in cells A1:G37 on the datasheet.


This example clears the chart area (the chart data and formatting) of Chart1.

myChart.ChartArea.Clear
ClearContents Method

ClearContents method as it applies to the ChartArea object.

Clears the data from a chart but leaves the formatting.

expression.ClearContents

expression Required. An expression that returns one of the above objects.

ClearContents method as it applies to the Range object.

Clears the formulas from the range.

expression.ClearContents

expression Required. An expression that returns one of the above objects.
Example

This example clears the formulas from cells A1:G37 on the datasheet but leaves the formatting intact.


This example clears the chart data from a chart but leaves the formatting intact.

myChart.ChartArea.ClearContents
ClearFormats Method

Clears the formatting of the object.

expression.ClearFormats

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example clears all formatting from cells A1:G37 on the datasheet.

```
```

This example clears the formatting from the chart.

```
myChart.ChartArea.ClearFormats
```
ColumnGroups Method

On a 2-D chart, returns an object that represents either a single column chart group or a collection of the column chart groups.

expression.ColumnGroups(Index)

expression Required. An expression that returns one of the objects in the Applies To list.

Index Optional Variant. The index number of the specified column chart group.
Example

This example sets the space between column clusters in the 2-D column chart group to be 50 percent of the column width.

myChart.ColumnGroups(1).GapWidth = 50
Copy Method

**Copy method as it applies to the ChartArea object.**

Copies a picture of the point or series to the Clipboard.

*expression*.**Copy**

*expression* Required. An expression that returns one of the above objects.

**Copy method as it applies to the Range object.**

Copies the **Range** to the specified range or to the Clipboard.

*expression*.**Copy**(Destination)

*expression* Required. An expression that returns one of the above objects.

**Destination** Optional **Variant**. Specifies the new range to which the specified range will be copied. If this argument is omitted, Microsoft Graph copies the range to the Clipboard.
Example

This example copies the formulas in cells A1:D4 on the datasheet into cells E5:H8.

Set mySheet = myChart.Application.DataSheet
mySheet.Range("A1:D4").Copy
    Destination:= mySheet.Range("E5")
Cut Method

Cuts the specified range to the Clipboard or pastes it into a specified destination.

\texttt{expression}.\texttt{Cut(Destination)}

\textit{expression} Required. An expression that returns a \textbf{Range} object.

\textit{Destination} Optional \textbf{Variant}. The range where the object should be pasted. If this argument is omitted, the object is cut to the Clipboard.
**Example**

This example cuts the range A1:G37 on the datasheet and places it on the Clipboard.

```vba
Set mySheet = myChart.Application.DataSheet
mySheet.Range("A1:G37").Cut
```
**DataLabels Method**

Returns an object that represents either a single data label or a collection of all the data labels for the series.

`expression.DataLabels(Index)`

*expression*  Required. An expression that returns one of the objects in the Applies To list.

*Index*  Optional *Variant*. The number of the data label.
Example

This example sets the data labels for series one to show their key, assuming that their values are visible when the example runs.

```vba
With myChart.SeriesCollection(1)
    .HasDataLabels = True
    With .DataLabels
        .ShowLegendKey = True
        .Type = xlValue
    End With
End With
```
Delete Method

Delete method as it applies to all objects in the Applies To list except the Range object.

Deletes the specified object.

expression.Delete

expression Required. An expression that returns one of the above objects.

Delete method as it applies to the Range object.

Deletes the specified object.

expression.Delete(Shift)

expression Required. An expression that returns one of the above objects.

Shift Optional XlDeleteShiftDirection. Used only with Range objects. Specifies how to shift cells to replace deleted cells.

XlDeleteShiftDirection can be one of these XlDeleteShiftDirection constants.

xlShiftToLeft
xlShiftUp

If this argument is omitted, Microsoft Graph decides how to shift cells based on the shape of the specified range.
Remarks

Deleting a Point or LegendKey object deletes the entire series.
Example

This example deletes cells A1:D10 on the datasheet and shifts the remaining cells to the left.

Set mySheet = myChart.Application.DataSheet
mySheet.Range("A1:D10").Delete Shift:=xlShiftToLeft

This example deletes the chart title.

myChart.ChartTitle.Delete
DeleteChartAutoFormat Method

Removes a custom chart autoformat from the list of available chart autoformats.

\[\text{expression} . \text{DeleteChartAutoFormat} (\text{Name})\]

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

*Name*  Required **String**. The name of the custom autoformat to be removed.
Example

This example deletes the custom autoformat named "Presentation Chart."

myChart.Application.DeleteChartAutoFormat 
  name:="Presentation Chart"
DeleteReplacement Method

 Deletes an entry from the array of AutoCorrect replacements.

```
expression.DeleteReplacement(What)
```

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

*What* Required *String*. The text to be replaced, as it appears in the row to be deleted from the array of AutoCorrect replacements. If this string doesn't exist in the array of AutoCorrect replacements, this method fails.
Example

This example removes the word "Temperature" from the array of AutoCorrect replacements.

    With myChart.Application.AutoCorrect
        .DeleteReplacement "Temperature"
    End With
Deselect Method

Cancels the selection for the chart.

\textit{expression.Deselect}

\textit{expression} Required. An expression that returns a \texttt{Chart} object.
Example

This example is equivalent to pressing ESC while working on the chart. The example should be run on a chart that has a component (such as an axis) selected.

myChart.Deselect
DoughnutGroups Method

On a 2-D chart, returns an object that represents either a single doughnut chart group or a collection of the doughnut chart groups.

\[ expression.DoughnutGroups(Index) \]

*expression*  Required. An expression that returns one of the objects in the Applies To list.

*Index*  Optional Variant. Specifies the chart group.
Example

This example sets the starting angle for doughnut group one.

\texttt{myChart.DoughnutGroups(1).FirstSliceAngle = 45}
ErrorBar Method

Applies error bars to the specified series. Variant.

expression.ErrorBar(Direction, Include, Type, Amount, MinusValues)

expression Required. An expression that returns one of the objects in the Applies To list.

Direction Required XlErrorBarDirection. The error bar direction.

XlErrorBarDirection can be one of these XlErrorBarDirection constants. xlX Can only be used with scatter charts. xlY default.

Include Required XlErrorBarInclude. The error bar parts to be included.

XlErrorBarInclude can be one of these XlErrorBarInclude constants. xlErrorBarIncludeBoth default. xlErrorBarIncludeMinusValues xlErrorBarIncludeNone xlErrorBarIncludePlusValues

Type Required XlErrorBarType. The error bar type.

XlErrorBarType can be one of these XlErrorBarType constants. xlErrorBarTypeCustom xlErrorBarTypeFixedValue xlErrorBarTypePercent xlErrorBarTypeStDev xlErrorBarTypeStError

Amount Optional Variant. The error amount. Used for only the positive error amount when Type is xlErrorBarTypeCustom.
*MinusValues* Optional *Variant*. The negative error amount when *Type* is `xlErrorBarTypeCustom`. 
Example

This example applies standard error bars in the Y direction for series one. The error bars are applied in the positive and negative directions. The example should be run on a 2-D line chart.

myChart.SeriesCollection(1).ErrorBar__
   Direction:=xlY, Include:=xlErrorBarIncludeBoth, _
   Type:=xlErrorBarTypeStError
Evaluate Method

Converts a Microsoft Graph name to an object or a value.

\textit{expression}.\texttt{Evaluate(}\textit{Name})\texttt{)

\textit{expression}  Required. An expression that returns a Microsoft Graph \texttt{Application} object.

\textit{Name}  Required \texttt{String}. The name of the specified object, using the Microsoft Graph naming convention.
Remarks

You can use the following types of names in Microsoft Graph with this method:

- A1-style references. You can use any reference to a single cell in A1-style notation. All references are considered to be absolute references.
- Ranges. You can use the range, intersect, and union operators (colon, space, and comma, respectively) with references.
- Defined names. You can specify any name in the language of the macro.

Note  Using square brackets (for example, "[A1:C5]") is identical to calling the Evaluate method with a string argument. For example, the following expressions are equivalent:

```vba
myChart.Application.[a1].Value = 25
myChart.Application.Evaluate("A1").Value = 25
```

The advantage of using square brackets is that the code is shorter. The advantage of using Evaluate is that the argument is a string, so you can either construct the string in your code or use a Visual Basic variable.
Example

This example clears cell A1 on the datasheet.

clearCell = "A1"
myChart.Application.Evalute(clearCell).Clear
Export Method

Exports the chart in a graphic format. Returns a value of type **Boolean**.

\[ expression . Export(FileName, FilterName, Interactive) \]

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

*FileName* Required **String**. The name of the exported file.

*FilterName* Optional **Variant**. The language-independent name of the graphic filter as it appears in the registry.

*Interactive* Optional **Variant**. **True** to display the dialog box that contains the filter-specific options. If this argument is **False**, Microsoft Graph uses the default values for the filter. The default value is **False**.
Example

This example exports the chart as a GIF file.

```javascript
myChart.Export(
    FileName:="current_sales.gif", FilterName:="GIF"
)
```
FileImport Method

Imports a specified file or range, or an entire sheet of data.

expression.FileImport(FileName, Password, ImportRange, WorksheetName, OverwriteCells)

expression  Required. An expression that returns an Application object.

FileName  Required String. The file that contains the data to be imported.

Password  Optional Variant. The password for the file to be imported, if the file is password protected.

ImportRange  Optional Variant. The range of cells to be imported, if the file to be imported is a Microsoft Excel worksheet or workbook. If this argument is omitted, the complete contents of the worksheet are imported.

WorksheetName  Optional Variant. The name of the worksheet to be imported, if the file to be imported is a Microsoft Excel workbook.

OverwriteCells  Optional Variant. True to specify that the user be notified before imported data overwrites existing data on the specified datasheet. The default value is True.
Example

This example imports data from the range A2:D5 on the worksheet named "MySheet" in the Microsoft Excel workbook named "mynums.xls."

With myChart.Application
    .FileImport FileName:="C:\mynums.xls", _
    ImportRange:="A2:D5", WorksheetName:="MySheet", _
    OverwriteCells:=False
End With
Insert Method

Inserts a cell or a range of cells into the datasheet and shifts other cells away to make space.

\textit{expression}.\texttt{Insert(Shift)}

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Shift} Optional \texttt{XlInsertShiftDirection}. Specifies which way to shift the cells.

\texttt{XlInsertShiftDirection} can be one of these \texttt{XlInsertShiftDirection} constants. \\
\texttt{xlShiftToRight} \\
\texttt{xlShiftDown}

If this argument is omitted, Microsoft Graph decides based on the shape of the range.
Example

This example inserts a new row before row four on the datasheet.

```vba
```

This example inserts new cells at the range A1:C5 on the datasheet and shifts cells downward.

```vba
Set mySheet = myChart.Application.DataSheet
```
Show All
Item Method

Item method as it applies to the **Axes** object.

Returns a single **Axis** object from an **Axes** collection.

```expression.Item(Type, AxisGroup)```

**expression** Required. An expression that returns an **Axes** collection.

**Type** Required **XlAxisType**. The axis type.

XlAxisType can be one of these XlAxisType constants.
- xlCategory
- xlSeriesAxis Valid only for 3-D charts.
- xlValue

**AxisGroup** Optional **XlAxisGroup**. The axis group.

XlAxisGroup can be one of these XlAxisGroup constants.
- xlSecondary
- xlPrimary *default*

Item method as it applies to the **ChartGroups** object.

Returns a single **ChartGroup** object from a **ChartGroups** collection.

```expression.Item(Index)```

**expression** Required. An expression that returns a **ChartGroups** collection.

**Index** Required **Variant**. The index number of the chart group.

Item method as it applies to the **DataLabels** object.

Returns a single **DataLabel** object from a **DataLabels** collection.
expression.Item(\textit{Index})

\textit{expression} Required. An expression that returns a \texttt{DataLabels} collection.

\textit{Index} Required \texttt{Variant}. The name or index number of the data label.

\texttt{Item method as it applies to the LegendEntries object.}

Returns a single \texttt{LegendEntry} object from a \texttt{LegendEntries} collection.

expression.Item(\textit{Index})

\textit{expression} Required. An expression that returns a \texttt{LegendEntries} collection.

\textit{Index} Required \texttt{Variant}. The index number of the legend entry.

\texttt{Item method as it applies to the Points object.}

Returns a single \texttt{Point} object from a \texttt{Points} collection.

expression.Item(\textit{Index})

\textit{expression} Required. An expression that returns a \texttt{Points} collection.

\textit{Index} Required \texttt{Long}. The index number of the point.

\texttt{Item method as it applies to the SeriesCollection object.}

Returns a single \texttt{Series} object from a \texttt{SeriesCollection} collection.

expression.Item(\textit{Index})

\textit{expression} Required. An expression that returns a \texttt{SeriesCollection} collection.

\textit{Index} Required \texttt{Variant}. The name or index number of the series.

\texttt{Item method as it applies to the Trendlines object.}

Returns a single \texttt{Trendline} object from a \texttt{Trendlines} collection.
expression.Item(\textit{Index})

\textit{expression} Required. An expression that returns a \textbf{Trendlines} collection.

\textit{Index} Optional \textbf{Variant}. The name or index number of the trendline.
Example

As it applies to the **Axes object**.

This example sets the title text for the category axis on Chart1.

```vba
With Charts("chart1").Axes.Item(xlCategory)
    .HasTitle = True
    .AxisTitle.Caption = "1994"
End With
```

As it applies to the **ChartGroups object**.

This example adds drop lines to chart group one on chart sheet one.

```vba
Charts(1).ChartGroups.Item(1).HasDropLines = True
```

As it applies to the **DataLabels object**.

This example sets the number format for the fifth data label in series one in embedded chart one on worksheet one.

```vba
Worksheets(1).ChartObjects(1).Chart.SeriesCollection(1).DataLabels.Item(5).NumberFormat = "0.000"
```

As it applies to the **LegendEntries object**.

This example changes the font for the text of the legend entry at the top of the legend (this is usually the legend for series one) in embedded chart one on Sheet1.

```vba
```

As it applies to the **Points object**.
This example sets the marker style for the third point in series one in embedded chart one on worksheet one. The specified series must be a 2-D line, scatter, or radar series.

```vba
```

As it applies to the **SeriesCollection** object.

This example provides two lines of code that are equivalent:

```vba
myChart.SeriesCollection.Item(1)
myChart.SeriesCollection(1)
```

As it applies to the **Trendlines** object.

This example sets the number of units that the trendline on Chart1 extends forward and backward. The example should be run on a 2-D column chart that contains a single series with a trendline.

```vba
With Charts("Chart1").SeriesCollection(1).Trendlines.Item(1)
    .Forward = 5
    .Backward = .5
End With
```
LegendEntries Method

Returns an object that represents either a single legend entry or a collection of legend entries for the legend.

expression.LegendEntries(Index)

expression  Required. An expression that returns one of the objects in the Applies To list.

Index  Optional Variant. The number of the legend entry.
Example

This example sets the font for legend entry one.

```csharp
myChart.Legend.LegendEntries(1).Font.Name = "Arial"
```
LineGroups Method

On a 2-D chart, returns an object that represents either a single line chart group or a collection of the line chart groups.

\[\text{expression}.\text{LineGroups(\text{Index})}\]

*expression*  Required. An expression that returns one of the objects in the Applies To list.

*Index*  Optional *Variant*. Specifies the chart group.
Example

This example sets line group one to use a different color for each data marker. The example should be run on a 2-D chart.

myChart.LineGroups(1).VaryByCategories = True
OneColorGradient Method

Sets the specified fill to a one-color gradient.

\textit{expression.\texttt{OneColorGradient(Style, Variant, Degree)}}

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Style} Required \texttt{MsoGradientStyle}. The gradient style for the specified fill.

MsoGradientStyle can be one of these MsoGradientStyle constants.
\texttt{msoGradientDiagonalDown}
\texttt{msoGradientDiagonalUp}
\texttt{msoGradientFromCenter}
\texttt{msoGradientFromCorner}
\texttt{msoGradientFromTitle}
\texttt{msoGradientHorizontal}
\texttt{msoGradientMixed}
\texttt{msoGradientVertical}

\textit{Variant} Required \texttt{Long}. The gradient variant for the specified fill. Can be a value from 1 through 4, corresponding to the four variants listed on the Gradient tab in the Fill Effects dialog box. If \textit{Style} is \texttt{msoGradientFromCenter}, the \textit{Variant} argument can only be 1 or 2.

\textit{Degree} Required \texttt{Single}. The gradient degree for the specified fill. Can be a value from 0.0 (dark) through 1.0 (light).
Example

This example sets the chart's fill format.

With myChart.ChartArea.Fill
   If .Type = msoFillGradient Then
     If .GradientColorType = msoGradientOneColor Then
       .OneColorGradient Style:=msoGradientFromCorner, _
       Variant:=1, Degree:=0.3
     End If
   End If
End With
Paste Method

Pastes the contents of the Clipboard into the specified range on the datasheet.

\( \text{expression}.\text{Paste(} \text{Link} \text{)} \)

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

**Link**  Optional **Variant**. **True** to establish a link to the source of the pasted data. The default value is **False**.
Example

This example pastes the contents of the Clipboard into cell A1 on the datasheet.

```vbnet
```
Patterned Method

Sets a pattern for the specified fill.

expression.Patterned(Pattern)

expression   Required. An expression that returns one of the objects in the Applies To list.

Pattern   Required MsoPatternType. The type of pattern.

MsoPatternType can be one of these MsoPatternType constants.

msoPattern10Percent
msoPattern20Percent
msoPattern25Percent
msoPattern30Percent
msoPattern40Percent
msoPattern50Percent
msoPattern5Percent
msoPattern60Percent
msoPattern70Percent
msoPattern75Percent
msoPattern80Percent
msoPattern90Percent
msoPatternDarkDownwardDiagonal
msoPatternDarkHorizontal
msoPatternDarkUpwardDiagonal
msoPatternDarkVertical
msoPatternDashedDownwardDiagonal
msoPatternDashedHorizontal
msoPatternDashedUpwardDiagonal
msoPatternDashedVertical
msoPatternDiagonalBrick
msoPatternDivot
msoPatternDottedDiamond
msoPatternDottedGrid
msoPatternHorizontalBrick
msoPatternLargeCheckerBoard
msoPatternLargeConfetti
msoPatternLargeGrid
msoPatternLightDownwardDiagonal
msoPatternLightHorizontal
msoPatternLightUpwardDiagonal
msoPatternLightVertical
msoPatternMixed
msoPatternNarrowHorizontal
msoPatternNarrowVertical
msoPatternOutlinedDiamond
msoPatternPlaid
msoPatternShingle
msoPatternSmallCheckerBoard
msoPatternSmallConfetti
msoPatternSmallGrid
msoPatternSolidDiamond
msoPatternSphere
msoPatternTrellis
msoPatternWave
msoPatternWeave
msoPatternWideDownwardDiagonal
msoPatternWideUpwardDiagonal
msoPatternZigZag
Example

This example sets the fill pattern.

With myChart.ChartArea.Fill
  .Patterned msoPatternDiagonalBrick
  .Visible = True
End With
PieGroups Method

On a 2-D chart, returns an object that represents either a single pie chart group or a collection of the pie chart groups.

*expression*.PieGroups(*Index*)

*expression*  Required. An expression that returns one of the objects in the Applies To list.

*Index*  Optional Variant. Specifies the chart group.
Example

This example sets pie group one to use a different color for each data marker. The example should be run on a 2-D chart.

```csharp
myChart.PieGroups(1).VaryByCategories = True
```
Points Method

Returns an object that represents a single point or a collection of all the points in the series. Read-only.

\[ expression.\text{Points}(Index) \]

- \textit{expression} Required. An expression that returns one of the objects in the Applies To list.
- \textit{Index} Optional \textit{Variant}. The name or number of the point.
Example

This example applies a data label to point one in series one.

myChart.SeriesCollection(1).Points(1).ApplyDataLabels
PresetGradient Method

Sets the specified fill to a preset gradient.

\[ expression . \text{PresetGradient}(Style, Variant, PresetGradientType) \]

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Style} Required \textbf{MsoGradientStyle}. The gradient style for the specified fill.

\text{MsoGradientStyle} can be one of these \text{MsoGradientStyle} constants.
\begin{itemize}
  \item \texttt{msoGradientDiagonalDown}
  \item \texttt{msoGradientDiagonalUp}
  \item \texttt{msoGradientFromCenter}
  \item \texttt{msoGradientFromCorner}
  \item \texttt{msoGradientFromTitle}
  \item \texttt{msoGradientHorizontal}
  \item \texttt{msoGradientMixed}
  \item \texttt{msoGradientVertical}
\end{itemize}

\textit{Variant} Required \textbf{Long}. The gradient variant for the specified fill. Can be a value from 1 through 4, corresponding to the four variants listed on the Gradient tab in the Fill Effects dialog box. If \textit{Style} is \texttt{msoGradientFromCenter}, the \textit{Variant} argument can only be 1 or 2.

\textit{PresetGradientType} Required \textbf{MsoPresetGradientType}. The gradient type for the specified fill.

\text{MsoPresetGradientType} can be one of these \text{MsoPresetGradientType} constants.
\begin{itemize}
  \item \texttt{msoGradientBrass}
  \item \texttt{msoGradientChrome}
  \item \texttt{msoGradientDaybreak}
  \item \texttt{msoGradientEarlySunset}
\end{itemize}
**Example**

This example sets the chart's fill format to the preset brass color.

```vba
With myChart.ChartArea.Fill
    .Visible = True
    .PresetGradient msoGradientDiagonalDown, 3, msoGradientBrass
End With
```
PresetTextured Method

Sets the format of the specified fill to a preset texture.

expression.PresetTextured(PresetTexture)

expression  Required. An expression that returns one of the objects in the Applies To list.

PresetTexture  Required MsoPresetTexture. The preset texture for the specified fill.

MsoPresetTexture can be one of these MsoPresetTexture constants.
msopresettexturedmixed
msotexturebouquet
msotexturecanvas
msotexturedenim
msotexturegranite
msotexturemediumwood
msotextureoak
msotexturepapyrus
msotexturepinktissuepaper
msotexturerecycledpaper
msotexturestationery
msotexturewaterdroplets
msotexturewovenmat
msotexturebluetissuepaper
msotexturebrownmarble
msotexturecork
msotexturefishfossil
msotexturegreenmarble
msotexturenewsprint
msotexturepaperbag
**Example**

This example changes the chart's textured fill format from oak to walnut.

```vba
With myChart.ChartArea.Fill
    If .Type = msoFillTextured Then
        If .TextureType = msoTexturePreset Then
            If .PresetTexture = msoTextureOak Then
                .PresetTextured msoTextureWalnut
            End If
        End If
    End If
End With
```
Quit Method

Quits Microsoft Graph.

expression.Quit

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

This example quits Microsoft Graph.

myChart.Application.Quit
RadarGroups Method

On a 2-D chart, returns an object that represents either a single radar chart group or a collection of the radar chart groups.

*expression*.RadarGroups(*Index*)

*expression* Required. An expression that returns one of the objects in the Applies To list.

*Index* Optional Variant. Specifies the chart group.
Example

This example sets radar group one to use a different color for each data marker. The example should be run on a 2-D chart.

```
myChart.RadarGroups(1).VaryByCategories = True
```
Refresh Method

Causes the specified chart to be redrawn immediately.

$expression$.Refresh

$expression$  Required. An expression that returns a Chart object.
Example

This example refreshes the first chart in the application. This example assumes a chart exists in the application.

Sub RefeshChart()
    Application.Charts(1).Refresh
End Sub
SaveAs Method

Saves changes to the graph in a different file.

expression.SaveAs(FileName)

expression  Required. An expression that returns one of the objects in the Applies To list.

FileName  Required String. A string that indicates the name of the file to be saved. You can include a full path; if you don't, Microsoft Excel saves the file in the current folder.
Example

This example creates a new workbook, prompts the user for a file name, and then saves the workbook.

Set NewBook = Workbooks.Add
Do
    fName = Application.GetSaveAsFilename
Loop Until fName <> False
NewBook.SaveAs Filename:=fName
SaveAsOldFileFormat Method

In a host application such as Microsoft PowerPoint, saves a chart in the specified older file format.

expression.SaveAsOldFileFormat(MajorVersion, MinorVersion)

expression Required. An expression that returns an Application object.

MajorVersion Optional Variant. Specifies the major version number of the file format you want to use.

MinorVersion Optional Variant. Specifies the minor version number of the file format you want to use.
Example

This example saves the chart in Microsoft Graph version 5.0 file format.

myChart.Application.SaveAsOldFileFormat MajorVersion:=5
SeriesCollection Method

Returns an object that represents either a single series or a collection of all the series in the chart or chart group.

expression.SeriesCollection(Index)

expression Required. An expression that returns one of the objects in the Applies To list.

Index Optional Variant. The name or number of the series.
Example

This example turns on data labels for series one.

```
myChart.SeriesCollection(1).HasDataLabels = True
```
SetDefaultChart Method

Specifies the name of the chart template that Microsoft Graph will use when creating new charts.

(expression).SetDefaultChart(FormatName, Gallery)

expression  Required. An expression that returns one of the objects in the Applies To list.

FormatName  Optional Variant. The name of the specified custom autoformat. This name can be a string that denotes the custom autoformat, or it can be the special constant xlBuiltIn to specify the built-in chart template.

Gallery  Optional Variant.
Example

This example sets the default chart template to the custom autoformat named "Monthly Sales."

`myChart.Application.SetDefaultChart FormatName:="Monthly Sales"`
SetEchoOn Method

Returns a **Chart** object.

`expression.SetEchoOn(EchoOn)`

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

*EchoOn*  Optional  **Variant**.
Example

This example sets the echo on for the first object in the application.

Sub UseEchoOn()
    Dim grpOne As Graph.Chart
    Set grpOne = Application.ActiveSheet.OLEObjects(1).Object
    grpOne.SetEchoOn
End Sub
**Solid Method**

Sets the specified fill to a uniform color. Use this method to convert a gradient, textured, patterned, or background fill back to a solid fill.

*expression*. **Solid**

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Example

This example converts the chart area fill to a solid color.

`myChart.ChartArea.Fill.Solid`
**Trendlines Method**

Returns an object that represents a single trendline or a collection of all the trendlines for the series.

`expression.Trendlines(Index)`

*expression* Required. An expression that returns one of the objects in the Applies To list.

*Index* Optional Variant. The name or number of the trendline.
**Example**

This example adds a linear trendline to series one.

```csharp
myChart.SeriesCollection(1).Trendlines.Add Type:=xlLinear
```
TwoColorGradient Method

Sets the specified fill to a two-color gradient.

\textit{expression}.TwoColorGradient(\textit{Style, Variant})

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.

\textit{Style} Required \textbf{MsoGradientStyle}. Specifies the gradient style.

\textbf{MsoGradientStyle} can be one of these \textbf{MsoGradientStyle} constants.
\begin{itemize}
  \item \textbf{msoGradientDiagonalDown}
  \item \textbf{msoGradientDiagonalUp}
  \item \textbf{msoGradientFromCenter}
  \item \textbf{msoGradientFromCorner}
  \item \textbf{msoGradientFromTitle}
  \item \textbf{msoGradientHorizontal}
  \item \textbf{msoGradientMixed}
  \item \textbf{msoGradientVertical}
\end{itemize}

\textit{Variant} Required \textbf{Long}. Specifies the gradient variant. Can be a value from 1 through 4, corresponding to the four variants on the \textbf{Gradient} tab in the \textbf{Fill Effects} dialog box. If \textit{Style} is \textbf{msoGradientFromCenter}, the \textbf{Variant} argument can only be either 1 or 2.
Example

This example sets the gradient, background color, and foreground color for the chart area fill on the chart.

With myChart.ChartArea.Fill
    .Visible = True
    .ForeColor.SchemeColor = 15
    .BackColor.SchemeColor = 17
    .TwoColorGradient msoGradientHorizontal, 1
End With
Update Method

Updates the specified embedded object in the host file.

\textit{expression}.\texttt{Update}

\textit{expression} Required. An expression that returns an \texttt{Application} object.
Example

This example updates the application.

Sub UseUpdate()
    Application.Update
End Sub
UserPicture Method

Fills the specified shape with an image.

\[ \text{expression}.\text{UserPicture}(	ext{PictureFile}, \text{PictureFormat}, \text{PictureStackUnit}, \text{PicturePlacement}) \]

\text{expression} Required. An expression that returns one of the objects in the Applies To list.

\textbf{PictureFile} Required \textbf{Variant}. The name of the specified picture file.

\textbf{PictureFormat} Optional \textbf{XlChartPictureType}. The format of the specified picture.

\text{XlChartPictureType} can be one of these \text{XlChartPictureType} constants.

- \text{xlScale}
- \text{xlStackScale}
- \text{xlStack}
- \text{xlStretch}

\textbf{PictureStackUnit} Optional \textbf{Variant}. The stack or scale unit for the specified picture (depends on the \text{PictureFormat} argument).

\textbf{PicturePlacement} Optional \textbf{XlChartPicturePlacement}. The placement of the specified picture.

\text{XlChartPicturePlacement} can be one of these \text{XlChartPicturePlacement} constants.

- \text{xlSides}
- \text{xlEnd}
- \text{xlEndSides}
xlFront
xlFrontSides
xlFrontEnd
xlAllFaces
Example

This example sets the chart's fill format so that it's based on a user-supplied picture.

```vba
With myChart.ChartArea.Fill
    .UserPicture PictureFile:="C:\My Documents\brick.bmp"
    .Visible = True
End With
```
UserTextured Method

Fills the specified shape with small tiles of an image. If you want to fill the shape with one large image, use the UserPicture method.

\[ \text{expression}.\text{UserTextured(\text{TextureFile})} \]

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

**TextureFile**  Required **String**. The name of the specified picture file.
Example

This example changes the user-defined texture type for the chart's fill format.

With myChart.ChartArea.Fill
    If .Type = msoFillTextured Then
        If .TextureType = msoTextureUserDefined Then
            If .TextureName = "C:\brick.bmp" Then
                .UserTextured "C:\stone.bmp"
            End If
        End If
    End If
End With
**XYGroups Method**

On a 2-D chart, returns an object that represents either a single scatter chart group or a collection of the scatter chart groups.

\( expression.XYGroups(Index) \)

- **expression**  Required. An expression that returns one of the objects in the Applies To list.
- **Index**  Optional **Variant**. Specifies the chart group.
Example

This example sets X-Y group (scatter group) one to use a different color for each data marker. The example should be run on a 2-D chart.

myChart.XYGroups(1).VaryByCategories = True
Application Property

Returns an Application object that represents the Microsoft Graph application. Read-only Application object.

expression.Application

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example substitutes the word "Temp." for the word "Temperature" in the array of AutoCorrect replacements.

With myChart.Application.AutoCorrect
    .AddReplacement "Temperature", "Temp."
End With
ApplyPictToEnd Property

**True** if a picture is applied to the end of the point or all points in the series. Read/write **Boolean**.

*expression*.ApplyPictToEnd

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Example

This example applies pictures to the end of all points in series one. The series must already have pictures applied to it (setting this property changes the picture orientation).

myChart.SeriesCollection(1).ApplyPictToEnd = True
ApplyPictToFront Property

True if a picture is applied to the front of the point or all points in the series. Read/write Boolean.

expression.ApplyPictToFront

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example applies pictures to the front of all points in series one. The series must already have pictures applied to it (setting this property changes the picture orientation).

myChart.SeriesCollection(1).ApplyPictToFront = True
ApplyPictToSides Property

True if a picture is applied to the sides of the point or all points in the series. Read/write Boolean.
Example

This example applies pictures to the sides of all points in series one. The series must already have pictures applied to it (setting this property changes the picture orientation).

myChart.SeriesCollection(1).ApplyPictToSides = True
**Area3DGroup Property**

Returns a `ChartGroup` object that represents the specified area chart group on a 3-D chart. Read-only `ChartGroup` object.

`expression.Area3DGroup`

`expression`  Required. An expression that returns one of the objects in the Applies To list.
Example

This example turns on drop lines for the 3-D area chart group.

myChart.Area3DGroup.HasDropLines = True
AutoCorrect Property

Returns an AutoCorrect object that represents the Microsoft Graph AutoCorrect attributes. Read-only.
Example

This example substitutes the word "Temp." for the word "Temperature" in the array of AutoCorrect replacements.

With myChart.Application.AutoCorrect
    .AddReplacement "Temperature", "Temp."
End With
**AutoScaleFont Property**

*True* if the text in the object changes font size when the object size changes. The default value is *True*. Read/write *Variant*. 
**Example**

This example adds a title to the chart, and it causes the title font to remain the same size whenever the chart size changes.

```
With myChart
    .HasTitle = True
    .ChartTitle.Text = "1996 sales"
    .ChartTitle.AutoScaleFont = False
End With
```
AutoScaling Property

True if Microsoft Graph scales a 3-D chart so that it's closer in size to the equivalent 2-D chart. The RightAngleAxes property must be True. Read/write Boolean.

expression.AutoScaling

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example automatically scales the chart. The example should be run on a 3-D chart.

With myChart
    .RightAngleAxes = True
    .AutoScaling = True
End With
AutoText Property

True if the object automatically generates appropriate text based on context. Read/write Boolean.
Example

This example sets the data labels for series one to automatically generate appropriate text.

myChart.SeriesCollection(1).DataLabels.AutoText = True
**AxisBetweenCategories Property**

**True** if the value axis crosses the category axis between categories. Read/write **Boolean**.

*expression*.\textit{AxisBetweenCategories}  

*expression*  
Required. An expression that returns one of the objects in the Applies To list.
Remarks

This property applies only to category axes, and it doesn't apply to 3-D charts.
Example

This example causes the value axis to cross the category axis between categories.

`myChart.Axes(xlCategory).AxisBetweenCategories = True`
**AxisGroup Property**

AxisGroup property as it applies to the ChartGroup and Series objects.

Returns the group for the specified chart group or series. Read/write XlAxisGroup.

XlAxisGroup can be one of these XlAxisGroup constants.
- xlPrimary
- xlSecondary

expression.AxisGroup

(expression) Required. An expression that returns one of the above objects.

AxisGroup property as it applies to the Axis object.

Returns the group for the specified axis. Read-only XlAxisGroup.

XlAxisGroup can be one of these XlAxisGroup constants.
- xlPrimary
- xlSecondary

expression.AxisGroup

(expression) Required. An expression that returns one of the above objects.
Remarks

For 3-D charts, only xlPrimary is valid.
Example

This example deletes the value axis if it's in the secondary group.

With myChart.Axes(xlValue)
  If .AxisGroup = xlSecondary Then .Delete
End With
AxisTitle Property

Returns an AxisTitle object that represents the title of the specified axis. Read-only AxisTitle object.

expression.AxisTitle

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example adds an axis label to the category axis in myChart.

With myChart.Axes(xlCategory)
    .HasTitle = True
    .AxisTitle.Text = "July Sales"
End With
BackColor Property

Returns a ChartColorFormat object that represents the fill background color.
Example

This example sets the gradient, background color, and foreground color for the chart area fill.

```
With myChart.ChartArea.Fill
    .Visible = True
    .ForeColor.SchemeColor = 15
    .BackColor.SchemeColor = 17
    .TwoColorGradient msoGradientHorizontal, 1
End With
```
Background Property

Returns or sets the text background type. This property is used only for text on charts. Read/write XlBackground.

XlBackground can be one of these XlBackground constants.

- xlBackgroundAutomatic
- xlBackgroundOpaque
- xlBackgroundTransparent

expression. Background

expression  Required. An expression that returns one of the objects in the Applies To list.
**Example**

This example adds a chart title and then sets the font size and background type for the title.

```vba
With myChart
  .HasTitle = True
  .ChartTitle.Text = "1995 Rainfall Totals by Month"
  With .ChartTitle.Font
    .Size = 10
    .Background = xlBackgroundTransparent
  End With
End With
```
Backward Property

Returns or sets the number of periods (or units on a scatter chart) that the trendline extends backward. Read/write Long.
**Example**

This example sets the number of units that the trendline extends forward and backward. The example should be run on a 2-D column chart that contains a single series with a trendline.

```vbnet
With myChart.SeriesCollection(1).Trendlines(1)
    .Forward = 5
    .Backward = .5
End With
```
Bar3DGroup Property

Returns a ChartGroup object that represents the specified bar chart group on a 3-D chart. Read-only.
Example

This example sets the space between bar clusters in the 3-D bar chart group to be 50 percent of the bar width.

myChart.BarGroup3DGroup.GapWidth = 50
BarShape Property

Returns or sets the shape used with the specified 3-D bar or column chart. Read/write **XlBarShape**.

XlBarShape can be one of these XlBarShape constants.

- xlConeToMax
- xlCylinder
- xlPyramidToPoint
- xlBox
- xlConeToPoint
- xlPyramidToMax

*expression*.**BarShape**

*expression*  Required. An expression that returns one of the objects in the Applies To list.
**Example**

This example sets the shape used with series one on the chart.

```csharp
myChart.SeriesCollection(1).BarShape = xlConeToPoint
```
BaseUnit Property

Returns or sets the base unit for the specified category axis Read/write XlTimeUnit.

XlTimeUnit can be one of these XlTimeUnit constants.
  xlDays
  xlMonths
  xlYears

expression.BaseUnit

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

Setting this property has no visible effect if the CategoryType property for the specified axis is set to xlCategoryScale. The set value is retained, however, and takes effect when the CategoryType property is set to xlTimeScale.

You cannot set this property for a value axis.
**Example**

This example sets the category axis on the chart to use a time scale, with months as the base unit.

```vba
With myChart
    With .Axes(xlCategory)
        .CategoryType = xlTimeScale
        .BaseUnit = xlMonths
    End With
End With
```
BaseUnitIsAuto Property

**True** if Microsoft Graph chooses appropriate base units for the specified category axis. The default value is **True**. Read/write **Boolean**.

*expression*.BaseUnitIsAuto

**expression**  Required. An expression that returns one of the objects in the Applies To list.
Remarks

You cannot set this property for a value axis.
Example

This example sets the category axis on the chart to use a time scale with automatic base units.

With myChart
    With .Axes(xlCategory)
        .CategoryType = xlTimeScale
        .BaseUnitIsAuto = True
    End With
End With
**Bold Property**

**True** if the font style is bold. Read/write **Variant**.

*expression*.**Bold**

*expression* Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the font style to bold for the chart title.

`myChart.ChartTitle.Font.Bold = True`
Border Property

Returns a Border object that represents the border of the specified object. Read-only Border object.

expression.Border

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the color of the chart area border to red.

myChart.ChartArea.Border.ColorIndex = 3
BubbleScale Property

Returns or sets the scale factor for bubbles in the specified chart group. Can be an integer value from 0 (zero) to 300, corresponding to a percentage of the default size. Applies only to bubble charts. Read/write Long.

expression.BubbleScale

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the bubble size in chart group one to 200 percent of the default size.

With myChart
    .ChartGroups(1).BubbleScale = 200
End With
CapitalizeNamesOfDay Property

**True** if the first letter of day names is capitalized automatically. Read/write **Boolean**.

\[ expression.CapitalizeNamesOfDay \]

\[ expression \] Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets Microsoft Graph to capitalize the first letter of the names of days.

    With myChart.Application.AutoCorrect
        .CapitalizeNamesOfDays = True
        .ReplaceText = True
    End With
Caption Property

Returns or sets the title text for the object. Read/write **String**.

*expression*.**Caption**

*expression*  Required. An expression that returns one of the above objects.
Example

This example adds the title "Annual Salary Figures" to the chart.

```csharp
myChart.HasTitle = True
myChart.ChartTitle.Caption = "Annual Salary Figures"
```
CategoryType Property

Returns or sets the category axis type. Read/write `XlCategoryType`.

`XlCategoryType` can be one of these `XlCategoryType` constants:
- `xlCategoryScale`
- `xlAutomaticScale`
- `xlTimeScale`

`expression.CategoryType`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Remarks

You cannot set this property for a value axis.
Example

This example sets the category axis on the chart to use a time scale, with months as the base unit.

With myChart
    With .Axes(xlCategory)
        .CategoryType = xlTimeScale
        .BaseUnit = xlMonths
    End With
End With
CellDragAndDrop Property

True if dragging and dropping cells is enabled. Read/write Boolean.
Example

This example enables dragging and dropping cells.

myChart.Application.CellDragAndDrop = True
Cells Property

Returns a Range object that represents the cells in the specified range, as it applies to the Range object. Also, returns a Range object that represents all the cells on the datasheet (not just the cells that are currently in use), as it applies to the DataSheet object. Read-only Range object.

expression.Cells

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

This example clears the formula in cell A1 on the datasheet. Note that on the datasheet, column A is the second column and row 1 is the second row.

```
```

This example loops through cells A1:I3 on the datasheet. If any of these cells contains a value less than 0.001, the example replaces that value with 0 (zero).

```
Set mySheet = myChart.Application.DataSheet
For rwIndex = 2 To 4
    For colIndex = 2 To 10
        If mySheet.Cells(rwIndex, colIndex) < .001 Then
            mySheet.Cells(rwIndex, colIndex).Value = 0
        End If
    Next colIndex
Next rwIndex
```
ChartArea Property

Returns a ChartArea object that represents the complete chart area for the chart. Read-only.
Example

This example sets the chart area interior color of myChart to red and sets the border color to blue.

With myChart.ChartArea
  .Interior.ColorIndex = 3
  .Border.ColorIndex = 5
End With
ChartTitle Property

Returns a ChartTitle object that represents the title of the specified chart. Read-only.
Example

This example sets the text for the title of the chart.

With myChart
  .HasTitle = True
  .ChartTitle.Text = "First Quarter Sales"
End With
Show All
ChartType Property

Returns or sets the chart type. Read/write XlChartType.

XlChartType can be one of these XlChartType constants.

- xl3DArea. 3-D Area
- xl3DAreaStacked. 3-D Stacked Area
- xl3DAreaStacked100. 3-D Stacked Area
- xl3DBarClustered. 3-D Clustered Bar
- xl3DBarStacked. 3-D Stacked Bar
- xl3DBarStacked100. 3-D 100% Stacked Bar
- xl3DColumn. 3-D Column
- xl3DColumnClustered. 3-D Clustered Column
- xl3DColumnStacked. 3-D Stacked Column
- xl3DColumnStacked100. 3-D 100% Stacked Column
- xl3DLine. 3-D Line
- xl3DPie. 3-D Pie
- xl3DPieExploded. Exploded 3-D Pie
- xlArea. Area
- xlAreaStacked. Stacked Area
- xlAreaStacked100. 100% Stacked Area
- xlBarClustered. Clustered Bar
- xlBarOfPie. Bar of Pie
- xlBarStacked. Stacked Bar
- xlBarStacked100. 100% Stacked Bar
- xlBubble. Bubble
- xlBubble3DEffect. Bubble with 3-D Effects
- xlColumnClustered. Clustered Column
- xlColumnStacked. Stacked Column
- xlColumnStacked100. 100% Stacked Column
- xlConeBarClustered. Clustered Cone Bar
- xlConeBarStacked. Stacked Cone Bar
xlConeBarStacked100. 100% Stacked Cone Bar
xlConeCol. 3-D Cone Column
xlConeColClustered. Clustered Cone Column
xlConeColStacked. Stacked Cone Column
xlConeColStacked100. 100% Stacked Cone Column
xlCylinderBarStacked. Stacked Cylinder Bar
xlCylinderCol. 3-D Cylinder Column
xlCylinderColStacked. Stacked Cylinder Column
xlCylinderBarClustered. Clustered Cylinder Bar
xlCylinderBarStacked100. 100% Stacked Cylinder Bar
xlCylinderColClustered. Clustered Cylinder Column
xlCylinderColStacked100. 100% Stacked Cylinder Column
xlDoughnut. Doughnut
xlDoughnutExploded. Exploded Doughnut
xlLineMarkers. Line with Data Markers
xlLineMarkersStacked100. 100% Stacked Line with Markers
xlLineStacked100. 100% Stacked Line
xlLine. Line
xlLineMarkersStacked. Stacked Line with Data Markers
xlLineStacked. Stacked Line
xlPie. Pie
xlPieExploded. Exploded Pie
xlPieOfPie. Pie of Pie
xlPyramidBarClustered. Clustered Pyramid Bar
xlPyramidBarStacked. Stacked Pyramid Bar
xlPyramidBarStacked100. 100% Stacked Pyramid Bar
xlPyramidCol. 3-D Pyramid Column
xlPyramidColStacked. Stacked Pyramid Column
xlPyramidColClustered. Clustered Pyramid Column
xlPyramidColStacked100. 100% Stacked Pyramid Column
xlRadar. Radar
xlRadarFilled. Filled Radar
xlRadarMarkers. Radar with Data Markers
xlStockHLC. High-Low-Close
xlStockOHLC. Open-High-Low-Close
xlStockVHLC. Volume-High-Low-Close
xlStockVOHLC. Volume-Open-High-Low-Close
xlSurface. 3-D Surface
xlSurfaceTopView. Surface (Top View)
xlSurfaceTopViewWireframe. Surface (Top View wire-frame)
xlSurfaceWireframe. 3-D Surface(wire-frame)
xlXYScatter. Scatter
xlXYScatterLines. Scatter with Lines
xlXYScatterLinesNoMarkers. Scatter with Lines and No Data Markers
xlXYScatterSmooth. Scatter with SmoothedLines
xlXYScatterSmoothNoMarkers. Scatter with Smoothed Lines and No Data Markers

expression.ChartType

type expression Required. An expression that returns one of the objects in the
Applies To list.
Example

This example sets the bubble size in chart group one to 200 percent of the default size if the chart is a 2-D bubble chart.

With myChart
    If .ChartType = xlBubble Then
        .ChartGroups(1).BubbleScale = 200
    End If
End With
Color Property

Returns or sets the primary color of the **Border** object, **Font** object, or the **Interior** object. Use the **RGB** function to create a color value. Read/write **Variant**.

*expression*.Color

*expression* Required. An expression that returns an object in the Applies To List.
Example

This example sets the color of the tick-mark labels on the value axis.

```javascript
myChart.Axes(xlValue).TickLabels.Font.Color = RGB(0, 255, 0)
```
## ColorIndex Property

Returns or sets the color of the border, font or interior, as shown in the following table. The color is specified as an index value into the current color palette, or as one of the following `XlColorIndex` constants: `xlColorIndexAutomatic` or `xlColorIndexNone`. Read/write Variant.

<table>
<thead>
<tr>
<th>Object</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Border</strong></td>
<td>The color of the border.</td>
</tr>
<tr>
<td><strong>Font</strong></td>
<td>The color of the font.</td>
</tr>
<tr>
<td></td>
<td>The color of the interior fill.</td>
</tr>
<tr>
<td><strong>Interior</strong></td>
<td>Set <code>ColorIndex</code> to <code>xlColorIndexNone</code> to specify that you don't want an interior fill. Set <code>ColorIndex</code> to <code>xlColorIndexAutomatic</code> to specify the automatic fill (for drawing objects).</td>
</tr>
</tbody>
</table>

*expression*.ColorIndex

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Remarks

This property specifies a color as an index into the color palette. The following illustration shows the color-index values in the default color palette.

1  2  3  4  5  6  7
8  9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31 32 33 34 35
36 37 38 39 40 41 42
43 44 45 46 47 48 49
50 51 52 53 54 55 56
**Example**

The following examples assume that you're using the default color palette.

This example sets the color of the major gridlines for the value axis.

```vba
With myChart.Axes(xlValue)
    If .HasMajorGridlines Then
        'Set color to blue
    End If
End With
```

This example sets the color of the chart area interior to red and sets the border color to blue.

```vba
With myChart.ChartArea
    .Interior.ColorIndex = 3
    .Border.ColorIndex = 5
End With
```
Colors Property

Returns or sets colors in the palette for a Chart object. The palette has 56 entries, each represented by an RGB value. Read/write Variant.

expression.Session.Chart.Colors(Index)

expression Required. An expression that returns one of the objects in the Applies To list.

Index Optional Variant. The color number (from 1 to 56). If this argument isn’t specified, this method returns an array that contains all 56 of the colors in the palette.
Example

This example sets color five in the color palette for the active chart.

`ActiveChart.Colors(5) = RGB(255, 0, 0)`
Column3DGroup Property

Returns a ChartGroup object that represents the specified column chart group on a 3-D chart. Read-only ChartGroup object.

expression.

expression. Column3DGroup

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the space between column clusters in the 3-D column chart group to be 50 percent of the column width.

\texttt{myChart.Column3DGroup.GapWidth = 50}
Columns Property

Returns a **Range** object that represents the columns in the specified range or all the columns on the datasheet. Read-only **Range** object.

**expression.Range**

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

For information about returning a single member of a collection, see [Returning an Object from a Collection](#).
Example

This example clears column A of the datasheet.

ColumnWidth Property

Returns or sets the width of all columns in the specified range. Read/write Variant.

expression.ColumnWidth

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

One unit of column width is equal to the width of one character in the Normal style. For proportional fonts, the width of the character 0 (zero) is used.

If all columns in the range have the same width, the ColumnWidth property returns the width. If columns in the range have different widths, this property returns Null.
Example

This example doubles the width of column A on the datasheet.

```vbscript
With myChart.Application.DataSheet.Columns("A")
    .ColumnWidth = .ColumnWidth * 2
End With
```
CommandBars Property

Returns a CommandBars object that represents the Microsoft Graph command bars. Read-only CommandBars object.

`expression.CommandBars`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example deletes all custom command bars that aren't visible.

For Each bar In myChart.Application.CommandBars
    If Not bar.BuiltIn And Not bar.Visible Then bar.Delete
Next
Corners Property

Returns a `Corners` object that represents the corners of the specified 3-D chart. Read-only.
CorrectCapsLock Property

True if Microsoft Graph automatically corrects accidental use of the CAPS LOCK key. Read/write Boolean.
Example

This example enables Microsoft Graph to automatically correct accidental use of the CAPS LOCK key.

myChart.Application.AutoCorrect.CorrectCapsLock = True
CorrectSentenceCap Property

**True** if Microsoft Graph automatically corrects sentence (first word) capitalization. Read/write **Boolean**.
Example

This example enables Microsoft Graph to automatically correct sentence capitalization.

myChart.Application.AutoCorrect.CorrectSentenceCap = True
Count Property

Returns the number of objects in the specified collection. Read-only Long.
Example

This example displays the number of chart groups in the chart.

MsgBox "The chart contains " & _
       myChart.ChartGroups.Count & _
       " chart groups."
Show All
Creator Property

Returns a 32-bit integer that indicates the application in which the specified object was created. If the object was created in Microsoft Graph, this property returns the string MSGR, which is equivalent to the hexadecimal number 4D534752. Read-only **XICreator**.

XICreator can be one of these XICreator constants.

**xlCreatorCode**

*expression*.Creator

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Example

This example displays a message about the creator of myChart.

If myChart.Creator = &h4D534752 Then
    MsgBox "This is a Microsoft Graph object"
Else
    MsgBox "This is not a Microsoft Graph object"
End If
Crosses Property

Returns or sets the point on the specified axis where the other axis crosses. Read/write **XlAxisCrosses**.

XlAxisCrosses can be one of these XlAxisCrosses constants.

**xlAxisCrossesCustom**. The **CrossesAt** property specifies the axis crossing point.

**xlAxisCrossesMinimum**. The axis crosses at the minimum value.

**xlAxisCrossesAutomatic**. Microsoft Graph sets the axis crossing point.

**xlAxisCrossesMaximum**. The axis crosses at the maximum value.

_expression.Crosses_

_expression_ Required. An expression that returns one of the objects in the Applies To list.
Remarks

This property isn't available for radar charts. For 3-D charts, this property indicates where the plane defined by the category axis crosses the value axis.

This property can be used for both category and value axes. On the category axis, \texttt{xlMinimum} sets the value axis to cross at the first category, and \texttt{xlMaximum} sets the value axis to cross at the last category.

Note that \texttt{xlMinimum} and \texttt{xlMaximum} can have different meanings, depending on the axis.
**Example**

This example sets the value axis to cross the category axis at the maximum x value.

```plaintext
myChart.Axes(xlCategory).Crosses = xlMaximum
```
CrossesAt Property

Returns or sets the point on the value axis where the category axis crosses it. Applies only to the value axis. Read/write Double.
Remarks

Setting this property causes the Crosses property to change to xlAxisCrossesCustom.

This property cannot be used on 3-D charts or radar charts.
Example

This example sets the category axis in the ActiveChart to cross the value axis at value 3.

Sub Chart()
    ' Create a sample source of data.
    Range("A1") = "2"
    Range("A2") = "4"
    Range("A3") = "6"
    Range("A4") = "3"

    ' Create a chart based on the sample source of data.
    Charts.Add

    With ActiveChart
        .ChartType = xlLineMarkersStacked
        .SetSourceData Source:=Sheets("Sheet1").Range("A1:A4"), Plot
        .Location Where:=xlLocationAsObject, Name:="Sheet1"
    End With

    ' Set the category axis to cross the value axis at value 3.
    ActiveChart.Axes(xlValue).Select
    Selection.CrossesAt = 3
End Sub
DataLabel Property

Returns a DataLabel object that represents the data label associated with the specified point or trendline. Read-only.
Example

This example turns on the data label for point seven in series three, and then it sets the data label color to blue.

With myChart.SeriesCollection(3).Points(7)
  .HasDataLabel = True
  .ApplyDataLabels type:=xlValue
    .DataLabel.Font.ColorIndex = 5
End With
DataSheet Property

Returns the `DataSheet` object. Read-only.
Example

This example sets the value of cell A1 on the datasheet to 3.14159.

With myChart.Application
  .DataSheet.Range("A1").Value = 3.14159
End With
DataTable Property

Returns a DataTable object that represents the chart data table. Read-only.
Example

This example adds a data table with an outline border to the chart.

With myChart
    .HasDataTable = True
    .DataTable.HasBorderOutline = True
End With
DepthPercent Property

Returns or sets the depth of a 3-D chart as a percentage of the chart width (between 20 and 2000 percent). Read/write Long.
**Example**

This example sets the depth of the chart to be 50 percent of its width. The example should be run on a 3-D chart (the `DepthPercent` property fails on 2-D charts).

```plaintext
myChart.DepthPercent = 50
```
DisplayAlerts Property

True if Microsoft Graph displays certain alerts and messages while a macro is running. Read/write Boolean.
Remarks

The default value is True. Set this property to False if you don't want to be disturbed by prompts and alert messages while a macro is running; any time a message requires a response, Microsoft Graph chooses the default response.

If you set this property to False, Microsoft Graph doesn't automatically set it back to True when your macro stops running. Write your macro such that it always sets this property back to True when it stops running.
DisplayAutoCorrectOptions Property

Allows the user to display or hide the AutoCorrect Options button. The default value is True. Read/write Boolean.

expression.DisplayAutoCorrectOptions

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

The `DisplayAutoCorrectOptions` property is a Microsoft Office-wide setting. Changing this property in Microsoft Graph will affect the other Office applications also.
Example

This example determines if the **AutoCorrect Options** button can be displayed and notifies the user.

Sub CheckDisplaySetting()

    'Determine setting and notify user.
    If Application.AutoCorrect.**DisplayAutoCorrectOptions** = True Then
        MsgBox "The AutoCorrect Options button can be displayed."
    Else
        MsgBox "The AutoCorrect Options button cannot be displayed."
    End If

End Sub
DisplayBlanksAs Property

Returns or sets the way that blank cells are plotted on a chart. Read/write XlDisplayBlanksAs.

XlDisplayBlanksAs can be one of these XlDisplayBlanksAs constants.
- xlInterpolated
- xlNotPlotted
- xlZero

expression.DisplayBlanksAs

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets Microsoft Graph to not plot blank cells.

myChart.DisplayBlanksAs = xlNotPlotted
DisplayEquation Property

**True** if the equation for the trendline is displayed on the chart (in the same data label as the R-squared value). Setting this property to **True** automatically turns on data labels. Read/write **Boolean**.
Example

This example displays the R-squared value and equation for trendline one. The example should be run on a 2-D column chart that has a trendline for the first series.

With myChart.SeriesCollection(1).Trendlines(1)
  .DisplayRSquared = True
  .DisplayEquation = True
End With
DisplayRSquared Property

**True** if the R-squared value of the trendline is displayed on the chart (in the same data label as the equation). Setting this property to **True** automatically turns on data labels. Read/write **Boolean**.
Example

This example displays the R-squared value and equation for trendline one. The example should be run on a 2-D column chart that has a trendline for the first series.

```vbnet
With myChart.SeriesCollection(1).Trendlines(1)
    .DisplayRSquared = True
    .DisplayEquation = True
End With
```
DisplayUnit Property

Returns or sets the units displayed for the value axis in the specified chart. If the value is xlCustom, the DisplayUnitCustom property returns or sets the value of the units displayed for the value axis. Read/write XlDisplayUnit.

XlDisplayUnit can be one of these XlDisplayUnit constants.
- xlHundredMillions
- xlHundreds
- xlHundredThousands
- xlMillionMillions
- xlMillions
- xlTenMillions
- xlTenThousands
- xlThousandMillions
- xlThousands

expression.DisplayUnit

expression  Required. An expression that returns one of the objects in the Applies To list.
Remarks

Using unit labels for the value axis when charting large values makes the incremental labels on the axis more readable and the data easier to track. In other words, if you label your value axis in thousands (for example), you can use smaller numeric values next to the tick marks on the axis.
Example

This example sets the units displayed on the value axis in myChart to hundreds.

With myChart.Axes(xlValue)
    .DisplayUnit = xlHundreds
    .HasTitle = True
    .AxisTitle.Caption = "Rebate Amounts"
End With
DisplayUnitCustom Property

If the value returned or set by the DisplayUnit property is xlCustom, the DisplayUnitCustom property returns or sets the value of the units displayed for the value axis in the specified chart. The value must be a number from 0 through 10E307. Read/write Double.
Remarks

Using unit labels for the value axis when charting large values makes the incremental labels on the axis more readable and the data easier to track. In other words, if you label your value axis in thousands (for example), you can use smaller numeric values next to the tick marks on the axis.
Example

This example sets the units displayed on the value axis in myChart to increments of 500.

```
With myChart.Axes(xlValue)
    .DisplayUnit = xlCustom
    .DisplayUnitCustom = 500
    .HasTitle = True
    .AxisTitle.Caption = "Rebate Amounts"
End With
```
DisplayUnitLabel Property

Returns the DisplayUnitLabel object for the value axis in the specified chart. Returns Null if the HasDisplayUnitLabel property is False. Read-only.
Example

This example sets the caption for the value axis in myChart to “Millions” and turns off automatic font scaling.

With myChart.Axes(xlValue).DisplayUnitLabel
  .Caption = "Millions"
  .AutoScaleFont = False
End With
DoughnutHoleSize Property

Returns or sets the size of the hole in a doughnut chart group. The hole size is expressed as a percentage of the chart size, between 10 and 90 percent. Read/write Long.
Example

This example sets the hole size for doughnut group one. The example should be run on a 2-D doughnut chart.

myChart.DoughnutGroups(1).DoughnutHoleSize = 10
DownBars Property

Returns a DownBars object that represents the down bars on a line chart. Applies only to line charts. Read-only.
Example

This example turns on up bars and down bars for chart group one and then sets their colors. The example should be run on a 2-D line chart that has two series that cross each other at one or more data points.

With myChart.ChartGroups(1)
  .HasUpDownBars = True
  .DownBars.Interior.ColorIndex = 3
End With
DropLines Property

Returns a DropLines object that represents the drop lines for a series on a line chart or area chart. Applies only to line charts or area charts. Read-only.
Example

This example turns on drop lines for chart group one and then sets their line style, weight, and color. The example should be run on a 2-D line chart that has one series.

With myChart.ChartGroups(1)
  .HasDropLines = True
  With .DropLines.Border
    .LineStyle = xlThin
    .Weight = xlMedium
    .ColorIndex = 3
  End With
End With
Elevation Property

Returns or sets the elevation of the 3-D chart view, in degrees. Read/write Long.
Remarks

The chart elevation is the height at which you view the chart, in degrees. The default is 15 for most chart types. The value of this property must be between -90 and 90, except for 3-D bar charts, where it must be between 0 and 44.
Example

This example sets the chart elevation to 34 degrees. The example should be run on a 3-D chart (the Elevation property fails on 2-D charts).

myChart.Elevation = 34
EndStyle Property

Returns or sets the end style for the error bars. Read/write XlEndStyleCap.

XlEndStyleCap can be one of these XlEndStyleCap constants.

- xlCap
- xlNoCap

expression.EndStyle

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the end style for the error bars for series one. The example should be run on a 2-D line chart that has Y error bars for the first series.

myChart.SeriesCollection(1).ErrorBars.EndStyle = xlCap
ErrorBars Property

Returns an ErrorBars object that represents the error bars for the series. Read-only.
Example

This example sets the error bar color for series one. The example should be run on a 2-D line chart that has error bars for series one.

```vbnet
With myChart.SeriesCollection(1)
    .ErrorBars.Border.ColorIndex = 8
End With
```
Explosion Property

Returns or sets the explosion value for a pie-chart or doughnut-chart slice. Returns 0 (zero) if there's no explosion (the tip of the slice is in the center of the pie). Read/write Long.
Example

This example sets the explosion value for point two. The example should be run on a pie chart.

myChart.SeriesCollection(1).Points(2).

Explosion = 20
Fill Property

Returns a ChartFillFormat object that contains fill formatting properties for the specified chart. Read-only.
Example

This example sets the fill format for the chart to the preset brass color.

With myChart.ChartArea.Fill
  .Visible = True
  .PresetGradient msoGradientDiagonalDown, 3, msoGradientBrass
End With
**FirstSliceAngle Property**

Returns or sets the angle of the first pie-chart or doughnut-chart slice, in degrees (clockwise from vertical). Applies only to pie, 3-D pie, and doughnut charts. Read/write **Long**.
Example

This example sets the angle for the first slice in chart group one. The example should be run on a 2-D pie chart.

```
myChart.ChartGroups(1).FirstSliceAngle = 15
```
Floor Property

Returns a Floor object that represents the floor of the 3-D chart. Read-only.
Example

This example sets the floor color to blue. The example should be run on a 3-D chart (the `Floor` property fails on 2-D charts).

```
myChart.Floor.Interior.ColorIndex = 5
```
Font Property

Returns a Font object that represents the font of the specified object. Read/write Font object only for the DataSheet object, for all other objects, read-only Font object.

expression.Font

expression  Required. An expression that returns one of the above objects.
Example

This example sets the font in the chart title to 14-point bold italic.

With myChart.ChartTitle.Font
    .Size = 14
    .Bold = True
    .Italic = True
End With
FontStyle Property

Returns or sets the font style. Read/write Variant.

*expression.FontStyle*

*expression*     Required. An expression that returns one of the objects in the Applies To list.
Remarks

Changing this property may affect other **Font** properties (such as **Bold** and **Italic**).
Example

This example sets the font style for the chart title to bold and italic.

myChart.ChartTitle.Font.FontStyle = "Bold Italic"
ForeColor Property

Returns a **ChartColorFormat** object that represents the foreground fill color.
Example

This example sets the gradient, background color, and foreground color for the chart area fill.

With myChart.ChartArea.Fill
  .Visible = True
  .ForeColor.SchemeColor = 15
  .BackColor.SchemeColor = 17
  .TwoColorGradient msoGradientHorizontal, 1
End With
Forward Property

Returns or sets the number of periods (or units on a scatter chart) that the trendline extends forward. Read/write Long.
Example

This example sets the number of units that the trendline extends forward and backward. The example should be run on a 2-D column chart that contains a single series with a trendline.

```vbnet
With myChart.SeriesCollection(1).Trendlines(1)
  .Forward = 5
  .Backward = .5
End With
```
GapDepth Property

Returns or sets the distance between the data series on a 3-D chart, as a percentage of the marker width. The value of this property must be between 0 and 500. Read/write **Long**.
Example

This example sets the distance between the data series to 200 percent of the marker width. The example should be run on a 3-D chart (the `GapDepth` property fails on 2-D charts).

```
myChart.GapDepth = 200
```
**GapWidth Property**

Bar and Column charts: Returns or sets the space between bar or column clusters, as a percentage of the bar or column width. The value of this property must be between 0 and 500. Read/write **Long**.

Pie of Pie and Bar of Pie charts: Returns or sets the space between the primary and secondary sections of the specified chart. The value of this property must be between 5 and 200. Read/write **Long**.
Example

This example sets the space between column clusters to be 50 percent of the column width.

`myChart.ChartGroups(1).GapWidth = 50`
GradientColorType Property

Returns the gradient color type for the specified fill. Read-only MsoGradientColorType.

MsoGradientColorType can be one of these MsoGradientColorType constants.
msoGradientColorMixed
msoGradientOneColor
msoGradientPresetColors
msoGradientTwoColors

expression.GradientColorType

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the fill format for the chart if its chart area has a one-color gradient fill.

```vba
With myChart.ChartArea.Fill
    If .Type = msoFillGradient Then
        If .GradientColorType = msoGradientOneColor Then
            .OneColorGradient.Style := msoGradientFromCorner,
            Variant := 1, Degree := 0.3
        End If
    End If
End With
```
GradientDegree Property

Returns the gradient degree of the specified one-color shaded fill as a floating-point value from 0.0 (dark) through 1.0 (light). Read-only Single.

This property is read-only. Use the OneColorGradient method to set the gradient degree for the fill.

expression.GradientDegree

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the chart's fill format so that its gradient degree is at least 0.3.

With myChart.ChartArea.Fill
    If .Type = msoFillGradient Then
        If .GradientColorType = msoGradientOneColor Then
            If .GradientDegree < 0.3 Then
                .OneColorGradient .GradientStyle, _
                .GradientVariant, 0.3
            End If
        End If
    End If
End With
GradientStyle Property

Returns the gradient style for the specified fill. Read-only MsoGradientStyle.

MsoGradientStyle can be one of these MsoGradientStyle constants.
- msoGradientDiagonalDown
- msoGradientDiagonalUp
- msoGradientFromCenter
- msoGradientFromCorner
- msoGradientFromTitle
- msoGradientHorizontal
- msoGradientMixed
- msoGradientVertical

This property is read-only. Use the OneColorGradient or TwoColorGradient method to set the gradient style for the fill.

expression.GradientStyle

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the chart's fill format so that its gradient style is changed to \texttt{msoGradientDiagonalUp} if it was originally \texttt{msoGradientDiagonalDown}.

\begin{verbatim}
With myChart.ChartArea.Fill
    If .Type = msoFillGradient Then
        If .GradientColorType = msoGradientOneColor Then
            If .GradientStyle = msoGradientDiagonalDown Then
                .OneColorGradient msoGradientDiagonalUp, _
                .GradientVariant, .GradientDegree
            End If
        End If
    End If
End With
\end{verbatim}
GradientVariant Property

Returns the shade variant for the specified fill as an integer value from 1 through 4. The values for this property correspond to the gradient variants (numbered from left to right and from top to bottom) listed on the Gradient tab in the Fill Effects dialog box. Read-only Long.

This property is read-only. Use the OneColorGradient or TwoColorGradient method to set the gradient variant for the fill

expression.GradientVariant

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the chart's fill format so that it's displayed using the second shade variant if it's currently using the first shade variant.

With myChart.ChartArea.Fill
    If .Type = msoFillGradient Then
        If .GradientColorType = msoGradientOneColor Then
            If .GradientVariant = 1 Then
                .OneColorGradient .GradientStyle, 2,
                .GradientDegree
            End If
        End If
    End If
End With
Has3DEffect Property

**True** if the series has a three-dimensional appearance. Applies only to bubble charts. Read/write **Boolean**.
Example

This example gives series one on the bubble chart a three-dimensional appearance.

With myChart
    .SeriesCollection(1).Has3DEffect = True
End With
Has3DShading Property

True if the chart group has three-dimensional shading. Read/write Boolean.
Example

This example adds three-dimensional shading to chart group one on the chart.

Charts(1).ChartGroups(1).Has3DShading = True
HasAxis Property

Returns or sets which axes exist on the chart. Read/write **Variant**.

`expression.HasAxis(Index1, Index2)`

- **expression** Required. An expression that returns one of the objects in the Applies To list.
- **Index1** Optional **XlAxisType**. The type of axis.
  - `XlAxisType` can be one of these `XlAxisType` constants:
    - `xlCategory`
    - `xlValue`
    - `xlSeriesAxis`. Series axes apply only to 3-D charts.
- **Index2** Optional **XlAxisGroup**. The axis priority.
  - `XlAxisGroup` can be one of these `XlAxisGroup` constants:
    - `xlPrimary`
    - `xlSecondary`
  - 3-D charts have only one set of axes.
Remarks

Microsoft Graph may create or delete axes if you change the chart type or change the AxisGroup property.
Example

This example turns on the primary value axis.

myChart.HasAxis(xlValue, xlPrimary) = True
HasBorderHorizontal Property

True if the chart data table has horizontal cell borders. Read/write Boolean.
**Example**

This example causes the chart data table to be displayed with an outline border and no cell borders.

```vba
With myChart
    .HasDataTable = True
    With .DataTable
        .HasBorderHorizontal = False
        .HasBorderVertical = False
        .HasBorderOutline = True
    End With
End With
```

HasBorderOutline Property

True if the chart data table has outline borders. Read/write Boolean.
Example

This example causes the chart data table to be displayed with an outline border and no cell borders.

With myChart
  .HasDataTable = True
With .DataTable
  .HasBorderHorizontal = False
  .HasBorderVertical = False
  .HasBorderOutline = True
End With
End With
HasBorderVertical Property

True if the chart data table has vertical cell borders. Read/write Boolean.
**Example**

This example causes the chart data table to be displayed with an outline border and no cell borders.

```vba
With myChart
    .HasDataTable = True
    With .DataTable
        .HasBorderHorizontal = False
        .HasBorderVertical = False
        .HasBorderOutline = True
    End With
End With
```
HasDataLabel Property

True if the point has a data label. Read/write Boolean.
Example

This example turns on the data label for point seven in series three, and then it sets the data label color to blue.

With myChart.SeriesCollection(3).Points(7)
   .HasDataLabel = True
   .ApplyDataLabels Type:=xlValue
   .DataLabel.Font.ColorIndex = 5
End With
HasDataLabels Property

*True* if the series has data labels. Read/write *Boolean*. 
**Example**

This example turns on data labels for series three.

```vba
With myChart.SeriesCollection(3)
   .HasDataLabels = True
   .ApplyDataLabels Type:=xlValue
End With
```
HasDataTable Property

True if the chart has a data table. Read/write Boolean.
**Example**

This example causes the chart data table to be displayed with an outline border and no cell borders.

```vbnet
With myChart
    .HasDataTable = True
    With .DataTable
        .HasBorderHorizontal = False
        .HasBorderVertical = False
        .HasBorderOutline = True
    End With
End With
end with
```
HasDisplayUnitLabel Property

True if the label specified by the DisplayUnit or DisplayUnitCustom property is displayed on the value axis. False if no units are displayed. The default value is True. Read/write Boolean.
**Example**

This example sets the units on the value axis in myChart to increments of 500 but hides the unit label itself.

```vba
With myChart.Axes(xlValue)
    .DisplayUnit = xlCustom
    .DisplayUnitCustom = 500
    .AxisTitle.Caption = "Rebate Amounts"
    .HasDisplayUnitLabel = False
End With
```
HasDropLines Property

True if the line chart or area chart has drop lines. Applies only to line and area charts. Read/write Boolean.
Example

This example turns on drop lines for chart group one and then sets their line style, weight, and color. The example should be run on a 2-D line chart that has one series.

```vba
With myChart.ChartGroups(1)
    .HasDropLines = True
    With .DropLines.Border
        .LineStyle = xlThin
        .Weight = xlMedium
        .ColorIndex = 3
    End With
End With
```
HasErrorBars Property

True if the series has error bars. This property isn't available for 3-D charts. Read/write Boolean.
Example

This example removes error bars from series one. The example should be run on a 2-D line chart that has error bars for series one.

```csharp
myChart.SeriesCollection(1).HasErrorBars = False
```
HasHiLoLines Property

**True** if the line chart has high-low lines. Applies only to line charts. Read/write **Boolean**.
Example

This example turns on high-low lines for chart group one and then sets line style, weight, and color. The example should be run on a 2-D line chart that has three series of stock-quote-like data (high-low-close).

With myChart.ChartGroups(1)
  .HasHiLoLines = True
  With .HiLoLines.Border
    .LineStyle = xlThin
    .Weight = xlMedium
    .ColorIndex = 3
  End With
End With
HasLeaderLines Property

True if the series has leader lines. Read/write Boolean.
Example

This example adds data labels and blue leader lines to series one on the pie chart.

With myChart.SeriesCollection(1)
    .HasDataLabels = True
    .DataLabels.Position = xlLabelPositionBestFit
    .HasLeaderLines = True
    .LeaderLines.Border.ColorIndex = 5
End With
HasLegend Property

**True** if the chart has a legend. Read/write **Boolean**.
Example

This example turns on the legend for the chart and then sets the legend font color to blue.

With myChart
    .HasLegend = True
    .Legend(Font.ColorIndex = 5
End With
HasLinks Property

True if the specified chart has links to an external data source. Read-only Boolean.
Example

This example clears cells A1:D4 on the datasheet if the chart has no links.

With myChart.Application
    If .HasLinks = False Then
        .DataSheet.Range("A1:D4").Clear
    End If
End With
HasMajorGridlines Property

True if the axis has major gridlines. Only axes in the primary axis group can have gridlines. Read/write Boolean.
Example

This example sets the color of the major gridlines for the value axis.

With myChart.Axes(xlValue)
    If .HasMajorGridlines Then
        .MajorGridlines.Border.ColorIndex = 3  'set color to red
    End If
End With
HasMinorGridlines Property

**True** if the axis has minor gridlines. Only axes in the primary axis group can have gridlines. Read/write **Boolean**.
Example

This example sets the color of the minor gridlines for the value axis.

With myChart.Axes(xlValue)
    If .HasMinorGridlines Then
        .MinorGridlines.Border.ColorIndex = 4
        ' Set color to green.
    End If
End With
HasRadarAxisLabels Property

True if a radar chart has axis labels. Applies only to radar charts. Read/write Boolean.
Example

This example turns on radar axis labels for chart group one and sets their color. The example should be run on a radar chart.

```vba
With myChart.ChartGroups(1)
    .HasRadarAxisLabels = True
End With
```
HasSeriesLines Property

**True** if a stacked column chart or bar chart has series lines or if a Pie of Pie chart or Bar of Pie chart has connector lines between the two sections. Applies only to stacked column charts, bar charts, Pie of Pie charts, or Bar of Pie charts. Read/write **Boolean**.
Example

This example turns on series lines for chart group one and then sets their line style, weight, and color. The example should be run on a 2-D stacked column chart that has two or more series.

With myChart.ChartGroups(1)
    .HasSeriesLines = True
    With .SeriesLines.Border
        .LineStyle = xlThin
        .Weight = xlMedium
        .ColorIndex = 3
    End With
End With
HasTitle Property

True if the axis or chart has a visible title. Read/write Boolean.
Remarks

An axis title is represented by an `AxisTitle` object.

A chart title is represented by a `ChartTitle` object.
Example

This example adds an axis label to the category axis.

With myChart.Axes(xlCategory)
    .HasTitle = True
    .AxisTitle.Text = "July Sales"
End With
HasUpDownBars Property

True if the specified line chart has up and down bars. Applies only to line charts. Read/write Boolean.
**Example**

This example turns on up and down bars for chart group one and then sets their colors. The example should be run on a 2-D line chart containing two series that cross each other at one or more data points.

```vba
With myChart.ChartGroups(1)
  .HasUpDownBars = True
  .DownBars.Interior.ColorIndex = 3
End With
```
Height Property

The height of the main application window or the object. If the window is minimized, this property is read-only and refers to the height of the icon. If the window is maximized, this property cannot be set. Use the WindowState property to determine the window state. Read/write Double for all objects, except for the Chart object which is read/write Variant.

expression.Height

expression Required. An expression that returns one of the above objects.
Example

This example sets the height of the chart legend to 1 inch (72 points).

myChart.Legend.Height = 72
HeightPercent Property

Returns or sets the height of a 3-D chart as a percentage of the chart width (between 5 and 500 percent). Read/write Long.
**Example**

This example sets the height of the chart to 80 percent of its width. The example should be run on a 3-D chart.

```javascript
myChart.HeightPercent = 80
```
HiLoLines Property

Returns a HiLoLines object that represents the high-low lines for the specified series on a line chart. Applies only to line charts. Read-only.
Example

This example turns on high-low lines for chart group one on the chart and then sets their line style, weight, and color. The example should be run on a 2-D line chart that has three series of stock-quote-like data (high-low-close).

```vba
With myChart.ChartGroups(1)
    .HasHiLoLines = True
    With .HiLoLines.Border
        .LineStyle = xlThin
        .Weight = xlMedium
        .ColorIndex = 3
    End With
End With
```
HorizontalAlignment Property

Returns or sets the horizontal alignment for the specified object. Read/write XlHAlign.

XlHAlign can be one of these XlHAlign constants.
- xlHAlignCenter
- xlHAlignCenterAcrossSelection
- xlHAlignDistributed
- xlHAlignFill
- xlHAlignGeneral
- xlHAlignJustify
- xlHAlignLeft
- xlHAlignRight

expression.HorizontalAlignment

eexpression Required. An expression that returns one of the objects in the Applies To list.
Remarks

Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.
Example

This example centers the chart title.

myChart.ChartTitle.HorizontalAlignment = xlCenter
Include Property

**True** if the data in the specified row or column is included in the chart. Read/write **Variant**.

expression.**Include**

**expression** Required. An expression that returns one of the objects in the Applies To list.
Example

This example causes the data in the second row on the datasheet to be excluded from the chart.

With myChart.Application.DataSheet
    .Rows(2).Include = False
End With
Index Property

Returns the index number of the object within the collection of similar objects. Read-only Long.
Example

This example displays the index number of an object passed to this procedure.

MsgBox "The index number of this object is " & obj.Index
InsideHeight Property

Returns the inside height of the plot area, in points. Read-only Double.
Remarks

The plot area used for this measurement doesn't include the axis labels. The **Height** property for the plot area uses the bounding rectangle that includes the axis labels.
InsideLeft Property

Returns the distance from the chart edge to the inside left edge of the plot area, in points. Read-only Double.
Remarks

The plot area used for this measurement doesn't include the axis labels. The **Left** property for the plot area uses the bounding rectangle that includes the axis labels.
InsideTop Property

Returns the distance from the chart edge to the inside top edge of the plot area, in points. Read-only Double.
Remarks

The plot area used for this measurement doesn't include the axis labels. The `Top` property for the plot area uses the bounding rectangle that includes the axis labels.
InsideWidth Property

Returns the inside width of the plot area, in points. Read-only Double.
Remarks

The plot area used for this measurement doesn't include the axis labels. The `Width` property for the plot area uses the bounding rectangle that includes the axis labels.
Intercept Property

Returns or sets the point where the trendline crosses the value axis. Read/write Double.
Remarks

Setting this property sets the **InterceptIsAuto** property to **False**.
Example

This example sets trendline one to cross the value axis at 5. The example should be run on a 2-D column chart that contains a single series with a trendline.

```
myChart.SeriesCollection(1).Trendlines(1).Intercept = 5
```
InterceptIsAuto Property

*True* if the point where the trendline crosses the value axis is automatically determined by the regression. Read/write *Boolean*. 
Remarks

Setting the `Intercept` property sets this property to `False`. 
Example

This example sets Microsoft Graph to automatically determine the trendline intercept point. The example should be run on a 2-D column chart that contains a single series with a trendline.

```csharp
myChart.SeriesCollection(1).Trendlines(1) _
    .InterceptIsAuto = True
```
Interior Property

Returns an Interior object that represents the interior of the specified object. Read-only.
Example

This example sets the interior color of the chart title.

\[\text{myChart.ChartTitle.Interior.ColorIndex} = 8\]
InvertIfNegative Property

True if Microsoft Graph inverts the pattern in the item when it corresponds to a negative number. Read/write Boolean for all objects, except for the Interior object, which is read/write Variant.

expression.InvertIfNegative

expression Required. An expression that returns one of the above objects.
**Example**

This example inverts the pattern for negative values in series one. The example should be run on a 2-D column chart.

```
myChart.SeriesCollection(1).InvertIfNegative = True
```
Italic Property

**True** if the font style is italic. Read/write **Variant**.

`expression.Italic`

`expression` Required. An expression that returns one of the objects in the Applies To list.
**Example**

This example sets the font style to italic for the chart title.

```plaintext
myChart.ChartTitle.Font.Italic = True
```
**Item Property**

Returns a **Range** object that represents a range that's offset from the specified range. Read/write **Variant**.

*expression*.Item(*RowIndex*, *ColumnIndex*)

*expression*  Required. An expression that returns one of the objects in the Applies To list.

*RowIndex*  Optional **Variant**. The row number of the cell you want to work with (the first row in the range is 1).

*ColumnIndex*  Optional **Variant**. A number or string that indicates the column number of the cell you want to work with (the first column in the range is either 1 or A).
Remarks

Syntax 1 uses a row number and either a column number or a letter as index arguments. For more information about this syntax, see the Range object. The RowIndex and ColumnIndex arguments are relative offsets. In other words, specifying 1 for RowIndex returns cells in the first row in the range, not the first row on the datasheet.
Example

This example clears cell B2 on the datasheet.

LeaderLines Property

Returns a LeaderLines object that represents the leader lines for the specified series. Read-only.
Example

This example adds data labels and blue leader lines to series one on the pie chart.

With myChart.SeriesCollection(1)
    .HasDataLabels = True
    .DataLabels.Position = xlLabelPositionBestFit
    .HasLeaderLines = True
    .LeaderLines.Border.ColorIndex = 5
End With
Left Property

`Left property as it applies to the Application, and DataSheet object.`

Returns or sets the distance from the left edge of the screen to the left edge of the main Microsoft Graph window. Read/write `Double`.

`expression.Left`

`expression` Required. An expression that returns one of the above objects.
Remarks

If the window is maximized, Application.Left returns a negative number that varies based on the width of the window border. Setting Application.Left to 0 (zero) will make the window a tiny bit smaller than it would be if the application window were maximized. In other words, if Application.Left is 0, the left border of the main Microsoft Graph window will just barely be visible on the screen.

If the Microsoft Graph window is minimized, Application.Left controls the position of the window icon.

Left property as it applies to the **AxisTitle, ChartArea, ChartTitle, DataLabel, DisplayUnitLabel, Legend, and PlotArea objects**.

Returns or sets the distance from the left edge of the object to the left edge of the chart area. Read/write **Double**.

expression.Left

expression  Required. An expression that returns one of the above objects.

Left property as it applies to the **Axis, LegendEntry, and LegendKey objects**.

Returns or sets the distance from the left edge of the object to the left edge of the chart area. Read-only **Double**.

expression.Left

expression  Required. An expression that returns one of the above objects.

Left property as it applies to the **Chart object**.

Returns or sets the distance from the left edge of the object to the left edge of the Microsoft Graph window. Read/write **Variant**.

expression.Left
expression  Required. An expression that returns a Chart object.
Example

As it applies to the ChartTitle object.

This example aligns the left edge of the chart title with the left edge of the chart area.

myChart.ChartTitle.Left = 0
Legend Property

Returns a Legend object that represents the legend for the specified chart. Read-only.
Example

This example turns on the legend for the chart and then sets the font color for the legend to blue.

```csharp
myChart.HasLegend = True
myChart.Legend.Font.ColorIndex = 5
```
LegendKey Property

Returns a LegendKey object that represents the legend key associated with the entry.
Example

This example sets the legend key for legend entry one to be a triangle. The example should be run on a 2-D line chart.

```plaintext
myChart.Legend.LegendEntries(1).LegendKey .MarkerStyle = xlMarkerStyleTriangle
```
Line3DGroup Property

Returns a ChartGroup object that represents the line chart group on a 3-D chart. Read-only.
Example

This example sets the 3-D line group to use a different color for each data marker.

```
myChart.Line3DGroup.VaryByCategories = True
```
LineStyle Property

Returns or sets the line style for the border. Read/write XlLineStyle.

XlLineStyle can be one of these XlLineStyle constants.
xlContinuous
xlDash
xlDashDot
xlDashDotDot
xlDot
xlDouble
xlSlantDashDot
xlLineStyleNone

expression.LineStyle

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example puts a border around the chart area and the plot area.

With myChart
    .ChartArea.Border.LineStyle = xlDashDot
With .PlotArea.Border
    .LineStyle = xlDashDotDot
    .Weight = xlThick
End With
End With
MajorGridlines Property

Returns a `Gridlines` object that represents the major gridlines for the specified axis. Only axes in the primary axis group can have gridlines. Read-only.
Example

This example sets the color of the major gridlines for the value axis in the chart.

With myChart.Axes(xlValue)
    If .HasMajorGridlines Then
    End If
End With
**MajorTickMark Property**

Returns or sets the type of major tick mark for the specified axis. Read/write **XlTickMark**.

XlTickMark can be one of these XlTickMark constants.

- xlTickMarkCross
- xlTickMarkInside
- xlTickMarkNone
- xlTickMarkOutside

expression.**MajorTickMark**

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the major tick marks for the value axis to be outside the axis.

`myChart.Axes(xlValue).MajorTickMark = xlTickMarkOutside`
MajorUnit Property

Returns or sets the major units for the axis. Read/write Double.
Remarks

Setting this property sets the `MajorUnitIsAuto` property to **False**.

Use the `TickMarkSpacing` property to set tick-mark spacing on the category axis.
Example

This example sets the major and minor units for the value axis.

With myChart.Axes(xlValue)
    .MajorUnit = 100
    .MinorUnit = 20
End With
**MajorUnitIsAuto Property**

*True* if Microsoft Graph calculates the major units for the axis. Read/write *Boolean*. 
Remarks

Setting the \texttt{MajorUnit} property sets this property to \texttt{False}. 
Example

This example automatically sets the major and minor units for the value axis.

```vba
With myChart.Axes(xlValue)
    .MajorUnitIsAuto = True
    .MinorUnitIsAuto = True
End With
```
MajorUnitScale Property

Returns or sets the major unit scale value for the category axis when the CategoryType property is set to xlTimeScale. Read/write Xl TimeUnit.

XlTimeUnit can be one of these XlTimeUnit constants.
- xlDays
- xlMonths
- xlYears

expression.MajorUnitScale

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the category axis to use a time scale and sets the major and minor units.

With myChart.Axes(xlCategory)
    .CategoryType = xlTimeScale
    .MajorUnit = 5
    .MajorUnitScale = xlDays
    .MinorUnit = 1
    .MinorUnitScale = xlDays
End With
MarkerBackgroundColor Property

Returns or sets the marker background color as an RGB value. Applies only to line, scatter, and radar charts. Read/write Long.
Example

This example sets the marker background and foreground colors for the second point in series one.

```vbnet
With myChart.SeriesCollection(1).Points(2)
    .MarkerBackgroundColor = RGB(0, 255, 0) ' green
    .MarkerForegroundColor = RGB(255, 0, 0) ' red
End With
```
**MarkerBackgroundColorIndex Property**

Returns or sets the marker background color as an index into the current color palette, or as one of the following `XlColorIndex` constants. Read/write `XlColorIndex`.

`XlColorIndex` can be one of these `XlColorIndex` constants.

- `xlColorIndexNone`
- `xlColorIndexAutomatic`

`expression.MarkerBackgroundColorIndex`  

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the marker background and foreground colors for the second point in series one.

```vbnet
With myChart.SeriesCollection(1).Points(2)
    .MarkerBackgroundColorIndex = 4 'green
    .MarkerForegroundColorIndex = 3 'red
End With
```
MarkerForegroundColor Property

Returns or sets the foreground color of the marker as an RGB value. Applies only to line, scatter, and radar charts. Read/write Long.
Example

This example sets the marker background and foreground colors for the second point in series one.

With myChart.SeriesCollection(1).Points(2)
   .MarkerBackgroundColor = RGB(0,255,0)  ' green
   .MarkerForegroundColor = RGB(255,0,0)  ' red
End With
MarkerForegroundColorIndex Property

Returns or sets the marker foreground color as an index into the current color palette, or as one of the following **XlColorIndex** constants. Read/write **XlColorIndex**.

XlColorIndex can be one of these XlColorIndex constants.
- **xlColorIndexNone**
- **xlColorIndexAutomatic**

```
expression.MarkerForegroundColorIndex
```

*expression* Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the marker background and foreground colors for the second point in series one.

```vbnet
With myChart.SeriesCollection(1).Points(2)
  .MarkerBackgroundColorIndex = 4 'green
  .MarkerForegroundColorIndex = 3 'red
End With
```
MarkerSize Property

Returns or sets the data-marker size, in points. Read/write Long.
Example

This example sets the data-marker size for all data markers in series one.

MyChart.SeriesCollection(1).MarkerSize = 10
MarkerStyle Property

Returns or sets the marker style for a point or series in a line chart, scatter chart, or radar chart. Read/write XlMarkerStyle.

XlMarkerStyle can be one of these XlMarkerStyle constants.

- **xlMarkerStyleCircle**. Circular markers
- **xlMarkerStyleDiamond**. Diamond-shaped markers
- **xlMarkerStyleNone**. No markers
- **xlMarkerStylePlus**. Square markers with a plus sign
- **xlMarkerStyleStar**. Square markers with an asterisk
- **xlMarkerStyleX**. Square markers with an X
- **xlMarkerStyleAutomatic**. Automatic markers
- **xlMarkerStyleDash**. Long bar markers
- **xlMarkerStyleDot**. Short bar markers
- **xlMarkerStylePicture**. Picture markers
- **xlMarkerStyleSquare**. Square markers
- **xlMarkerStyleTriangle**. Triangular markers

expression.MarkerStyle

*expression*. Required. An expression that returns one of the objects in the Applies To list.
**Example**

This example sets the marker style for series one. The example should be run on a 2-D line chart.

```csharp
myChart.SeriesCollection(1).MarkerStyle = xlMarkerStyleCircle
```
**MaximumScale Property**

Returns or sets the maximum value on the axis. Read/write *Double*. 
Remarks

Setting this property sets the MaximumScaleIsAuto property to False.
Example

This example sets the minimum and maximum values for the value axis.

With myChart.Axes(xlValue)
    .MinimumScale = 10
    .MaximumScale = 120
End With
MaximumScaleIsAuto Property

True if Microsoft Graph calculates the maximum value for the axis. Read/write Boolean.
Remarks

Setting the `MaximumScale` property sets this property to `False`. 
Example

This example automatically calculates the minimum scale and the maximum scale for the value axis.

With myChart.Axes(xlValue)
    .MinimumScaleIsAuto = True
    .MaximumScaleIsAuto = True
End With
MinimumScale Property

Returns or sets the minimum value on the axis. Read/write Double.
Remarks

Setting this property sets the **MinimumScaleIsAuto** property to **False**.
Example

This example sets the minimum and maximum values for the value axis.

With myChart.Axes(xlValue)
    .MinimumScale = 10
    .MaximumScale = 120
End With
MinimumScaleIsAuto Property

**True** if Microsoft Graph calculates the minimum value for the axis. Read/write **Boolean**.
Remarks

Setting the `MinimumScale` property sets this property to `False`.
**Example**

This example automatically calculates the minimum scale and the maximum scale for the value axis.

```vba
With myChart.Axes(xlValue)
    .MinimumScaleIsAuto = True
    .MaximumScaleIsAuto = True
End With
```
MinorGridlines Property

Returns a Gridlines object that represents the minor gridlines for the specified axis. Only axes in the primary axis group can have gridlines. Read-only.
Example

This example sets the color of the minor gridlines for the value axis in the chart to blue.

With myChart.Axes(xlValue)
    If .HasMinorGridlines Then
    End If
End With
MinorTickMark Property

Returns or sets the type of minor tick mark for the specified axis. Read/write XlTickMark.

XlTickMark can be one of these XlTickMark constants.

- xlTickMarkCross
- xlTickMarkInside
- xlTickMarkNone
- xlTickMarkOutside

expression.MinorTickMark

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the minor tick marks for the value axis to be inside the axis.

\[
\text{myChart.Axes(xlValue).MinorTickMark} = \text{xlTickMarkInside}
\]
**MinorUnit Property**

Returns or sets the minor units on the axis. Read/write **Double**.

`expression.MinorUnit`

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Remarks

Setting this property sets the **MinorUnitIsAuto** property to **False**.

Use the **TickMarkSpacing** property to set tick-mark spacing on the category axis.
Example

This example sets the major and minor units for the value axis.

```vba
With myChart.Axes(xlValue)
    .MajorUnit = 100
    .MinorUnit = 20
End With
```
**MinorUnitIsAuto Property**

*True* if Microsoft Graph calculates minor units for the axis. Read/write *Boolean*. 
Remarks

Setting the MinorUnit property sets this property to False.
Example

This example automatically calculates major and minor units for the value axis.

With myChart.Axes(xlValue)
    .MajorUnitIsAuto = True
    .MinorUnitIsAuto = True
End With
**MinorUnitScale Property**

Returns or sets the minor unit scale value for the category axis when the `CategoryType` property is set to `xlTimeScale`. Read/write `XlTimeUnit`.

`XlTimeUnit` can be one of these `XlTimeUnit` constants.

- `xlDays`
- `xlMonths`
- `xlYears`

`expression.MinorUnitScale`

`expression`  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the category axis to use a time scale and sets the major and minor units.

```vba
With myChart.Axes(xlCategory)
    .CategoryType = xlTimeScale
    .MajorUnit = 5
    .MajorUnitScale = xlDays
    .MinorUnit = 1
    .MinorUnitScale = xlDays
End With
```
MoveAfterReturn Property

True if the active cell will be moved as soon as the ENTER (RETURN) key is pressed. Read/write Boolean.
Example

This example sets the **MoveAfterReturn** property to **True**.

```
myChart.Application.MoveAfterReturn = True
```
Name Property

Name property as it applies to the Application and Trendline objects.
Returns or sets the name of the object. Read/write String.

expression.Name

expression   Required. An expression that returns one of the above objects.

Name property as it applies to the Font object.
Returns or sets the name of the object. Read/write Variant.

expression.Name

expression   Required. An expression that returns a Font object.

Name property as it applies to the all other objects.
Returns or sets the name of the object. Read-only String.

expression.Name

expression   Required. An expression that returns one of the above objects.
**Example**

This example assigns the name of the first trendline to the variable `myTrendname`.

```vba
myTrendname = myChart.SeriesCollection(1).Trendlines(1).Name
```
**NameIsAuto Property**

*True* if Microsoft Graph automatically determines the name of the trendline. Read/write *Boolean*. 
Example

This example sets Microsoft Graph to automatically determine the name for trendline one. The example should be run on a 2-D column chart that contains a single series with a trendline.

myChart.SeriesCollection(1).Trendlines(1).NameIsAuto = True
NumberFormat Property

Returns or sets the format code for the object. Returns Null if the cells in the specified range don't all have the same number format. Read/write String for all objects, except for the Range object, which is read/write Variant.

expression.NumberFormat

expression Required. An expression that returns one of the above objects.
Example

This example sets the number format for the data labels for series one.

```csharp
myChart.SeriesCollection(1).DataLabels.NumberFormat = "General"
```
**NumberFormatLocal Property**

Returns or sets the format code for the specified object as a string in the language of the user. Read/write **Variant**.
Remarks

The **Format** function uses different format code strings than do the **NumberFormat** and **NumberFormatLocal** properties.
Example

This example displays the number format for the data labels in the first series on the chart, in the language of the user.

MsgBox "The number format for the first series is " & _
myChart.SeriesCollection(1).DataLabels.NumberFormatLocal
Offset Property

Returns or sets the distance between each of the levels of labels, and the distance between the first level and the axis line. The default is 100, which represents the spacing between the axis labels and axis line. The value can be an integer from 0 to 1000, relative to the size of the font of the axis label. Read/write Long.
Example

This example doubles the existing tick-mark spacing on the value axis in myChart, if the offset is less than 500.

With myChart.Axes(xlCategory).TickLabels
    If .Offset < 500 then
        .Offset = .Offset * 2
    End If
End With
Order Property

Returns or sets the trendline order (an integer greater than 1) when the trendline type is xlPolynomial. Read/write Long.
Example

This example sets the order of the first trendline for series one if it's polynomial.

With myChart.SeriesCollection(1).Trendlines(1)
    If .Type = xlPolynomial Then .Order = 3
End With
Orientation Property

Returns or sets the text orientation. Can be an integer value from –90 degrees to 90 degrees or one of the following XLOrientation constants. Read/write XITickLabelOrientation for all objects, except for the TickLabels object, which is read/write Variant.

XITickLabelOrientation can be one of these XITickLabelOrientation constants.
xlTickLabelOrientationAutomatic
xlTickLabelOrientationDownward
xlTickLabelOrientationHorizontal
xlTickLabelOrientationUpward
xlTickLabelOrientationVertical

expression.Orientation

expression Required. An expression that returns one of the above objects.
Example

This example sets the orientation for the chart title.

myChart.ChartTitle.\textit{Orientation} = XLHorizontal
OutlineFont Property

**True** if the font is an outline font. Read/write **Variant**.

`expression.OutlineFont`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Remarks

This property has no effect in Windows, but its value is retained (it can be set and returned).
Example

This example sets the font for the chart title to an outline font.

\texttt{myChart.ChartTitle.Font.OutlineFont = True}
Overlap Property

Specifies how bars and columns are positioned. Can be a value between – 100 and 100. Applies only to 2-D bar and 2-D column charts. Read/write Long.
Remarks

If this property is set to – 100, bars are positioned so that there's one bar width between them. If the overlap is 0 (zero), there's no space between bars (one bar starts immediately after the preceding bar). If the overlap is 100, bars are positioned on top of each other.
Example

This example sets the overlap for chart group one to –50. The example should be run on a 2-D column chart that has two or more series.

\[ \text{myChart.ChartGroups(1).Overlap} = -50 \]
Parent Property

Returns the parent object.

expression:`.Parent`

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example returns the parent object of the application.

Sub UseParent()
    Application.Paren
End Sub
Pattern Property

For the ChartFillFormat object, returns or sets the fill pattern, read-only MsoPatternType. For the Interior object, returns or sets the interior pattern, read/write Variant.

MsoPatternType can be one of these MsoPatternType constants.

- msoPattern10Percent
- msoPattern20Percent
- msoPattern25Percent
- msoPattern30Percent
- msoPattern40Percent
- msoPattern50Percent
- msoPattern5Percent
- msoPattern60Percent
- msoPattern70Percent
- msoPattern75Percent
- msoPattern80Percent
- msoPattern90Percent
- msoPatternDarkDownwardDiagonal
- msoPatternDarkHorizontal
- msoPatternDarkUpwardDiagonal
- msoPatternDarkVertical
- msoPatternDashedDownwardDiagonal
- msoPatternDashedHorizontal
- msoPatternDashedUpwardDiagonal
- msoPatternDashedVertical
- msoPatternDiagonalBrick
- msoPatternDivot
- msoPatternDottedDiamond
- msoPatternDottedGrid
- msoPatternHorizontalBrick
msoPatternLargeCheckerBoard
msoPatternLargeConfetti
msoPatternLargeGrid
msoPatternLightDownwardDiagonal
msoPatternLightHorizontal
msoPatternLightUpwardDiagonal
msoPatternLightVertical
msoPatternMixed
msoPatternNarrowHorizontal
msoPatternNarrowVertical
msoPatternOutlinedDiamond
msoPatternPlaid
msoPatternShingle
msoPatternSmallCheckerBoard
msoPatternSmallConfetti
msoPatternSmallGrid
msoPatternSolidDiamond
msoPatternSphere
msoPatternTrellis
msoPatternWave
msoPatternWeave
msoPatternWideDownwardDiagonal
msoPatternWideUpwardDiagonal
msoPatternZigZag

expression.Pattern

expression Required. An expression that returns one of the above objects.
Example

This example adds a crisscross pattern to the interior of the plot area.

```plaintext
myChart.PlotArea.Interior.Pattern = xlPatternCrissCross
```
PatternColor Property

Returns or sets the color of the interior pattern as an RGB value. Read/write Variant.

expression.PatternColor

expression Required. An expression that returns an Interior object.
Example

This example sets the color of the interior pattern for the chart area.

`myChart.ChartArea.Interior.PatternColor = RGB(255,0,0)`
PatternColorIndex Property

Returns or sets the color of the interior pattern as an index into the current color palette, or as one of the following XlColorIndex constants. Read/write Variant.

XlColorIndex can be one of these XlColorIndex constants.
- xlColorIndexAutomatic
- xlColorIndexNone

```
expression.PatternColorIndex
```

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

Set this property to `xlColorIndexAutomatic` to specify the automatic pattern for cells or the automatic fill style for drawing objects. Set this property to `xlColorIndexNone` to specify that you don't want a pattern (this is the same as setting the `Pattern` property of the `Interior` object to `xlPatternNone`).
Example

This example sets the color of the interior pattern for the chart area.

myChart.ChartArea.Interior.PatternColorIndex = 5
Period Property

Returns or sets the period for the moving-average trendline. Read/write Long.
Example

This example sets the period for the moving-average trendline. The example should be run on a 2-D column chart with a single series that contains 10 data points and a moving-average trendline.

With myChart.SeriesCollection(1).Trendlines(1)
    If .Type = xlMovingAvg Then .Period = 5
End With
Perspective Property

Returns or sets the perspective for the 3-D chart view. Must be from 0 through 100. This property is ignored if the RightAngleAxes property is True. Read/write Long.
Example

This example sets the perspective of `myChart` to 70. The example should be run on a 3-D chart.

```plaintext
myChart.RightAngleAxes = False
myChart.Perspective = 70
```
PictureType Property

Returns or sets the way pictures are displayed on a column or bar picture chart or on the walls and faces of a 3-D chart. For the Point and Series objects, read/write XlChartPictureType. For the LegendKey object, read/write Long. For the Floor and Walls objects, read/write Variant.

XlChartPictureType can be one of these XlChartPictureType constants.

xlStack. Stacks the pictures to reach the necessary value.
xlStretch. Stretches the picture to reach the necessary value.
xlStackScale. Stacks the pictures; use the PictureUnit property to determine what unit each picture represents.

expression.PictureType

expression Required. An expression that returns one of the above objects.
Example

This example sets series one to stretch pictures. The example should be run on a 2-D column chart with picture data markers.

myChart.SeriesCollection(1).

**PictureType** = xlStretch
**PictureUnit Property**

Returns or sets the unit for each picture on the chart if the `PictureType` property is set to `xlScale` (otherwise, this property is ignored). Read/write `Long` for all objects, except for the `Walls` object, which is read/write `Variant`.

`expression.PictureUnit`

`expression` Required. An expression that returns one of the above objects.
Example

This example sets series one to stack pictures and uses each picture to represent five units. The example should be run on a 2-D column chart with picture data markers.

With myChart.SeriesCollection(1)
    .PictureType = xlScale
    .PictureUnit = 5
End With
Pie3DGroup Property

Returns a ChartGroup object that represents the pie chart group on a 3-D chart. Read-only.
Example

This example sets the 3-D pie group to use a different color for each data marker.

\texttt{myChart.Pie3DGroup.VaryByCategories = True}
PlotArea Property

Returns a PlotArea object that represents the plot area of a chart. Read-only.
Example

This example sets the color of the plot area interior of `myChart` to cyan.

`myChart.PlotArea.Interior.ColorIndex = 8`
PlotBy Property

Returns or sets the way columns or rows are used as data series on the chart. Read/write XlRowCol.

XlRowCol can be one of these XlRowCol constants.
- xlColumns
- xlRows

expression.PlotBy

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example causes the embedded chart to plot data by columns.

myChart.PlotBy = xlColumns
PlotOnX Property

Returns or sets the index of the data sheet row whose contents are to be used as the X-axis values in the specified X-Y scatter chart. Read/write Long.
Example

This example sets row 0 as the row whose contents will be plotted as values on the X-axis in myChart.

myChart.PlotOnX = 0
Position Property

Position property as it applies to the DataLabel and DataLabels objects.

Returns or sets the position of the data label. Read/write XlDataLabelPosition.

XlDataLabelPosition can be one of these XlDataLabelPosition constants.
- xlLabelPositionBelow
- xlLabelPositionCenter
- xlLabelPositionInsideBase
- xlLabelPositionInsideEnd
- xlLabelPositionLeft
- xlLabelPositionMixed
- xlLabelPositionOutsideEnd
- xlLabelPositionRight
- xlLabelPositionAbove
- xlLabelPositionBestFit
- xlLabelPositionCustom

expression. Position

expression Required. An expression that returns one of the above objects.

Position property as it applies to the Legend object.

Returns or sets the position of the legend on the chart. Read/write XlLegendPosition.

XlLegendPosition can be one of these XlLegendPosition constants.
- xlLegendPositionBottom
- xlLegendPositionCorner
- xlLegendPositionLeft
- xlLegendPositionRight
**xlLegendPositionTop**

`expression.Position`

`expression`  Required. An expression that returns one of the above objects.
Example

This example sets the position of the legend to the top of the chart.

`myChart.Legend.Position = xlLegendPositionTop`
**PresetGradientType Property**

Returns the preset gradient type for the specified fill. Read-only `MsoPresetGradientType`.

`MsoPresetGradientType` can be one of these `MsoPresetGradientType` constants:

- `msoGradientBrass`
- `msoGradientChrome`
- `msoGradientDaybreak`
- `msoGradientEarlySunset`
- `msoGradientFog`
- `msoGradientGoldII`
- `msoGradientLateSunset`
- `msoGradientMoss`
- `msoGradientOcean`
- `msoGradientPeacock`
- `msoGradientRainbowII`
- `msoGradientSilver`
- `msoGradientWheat`
- `msoPresetGradientMixed`
- `msoGradientCalmWater`
- `msoGradientChromeII`
- `msoGradientDesert`
- `msoGradientFire`
- `msoGradientGold`
- `msoGradientHorizon`
- `msoGradientMahogany`
- `msoGradientNightfall`
- `msoGradientParchment`
- `msoGradientRainbow`
- `msoGradientSapphire`
expression.PresetGradientType

expression  Required. An expression that returns one of the objects in the Applies To list.

This property is read-only. Use the PresetGradient method to set the preset gradient type for the fill.
**Example**

This example changes the chart's preset gradient fill format from silver to gold.

```
With myChart.ChartArea.Fill
    If .Type = msoFillGradient Then
        If .GradientColorType = msoGradientPresetColors Then
            If .PresetGradientType = msoGradientSilver Then
                .PresetGradient.GradientStyle, _
                .GradientVariant, msoGradientGold
            End If
        End If
    End If
End With
```
PresetTexture Property

Returns the preset texture for the specified fill. Read-only MsoPresetTexture.

MsoPresetTexture can be one of these MsoPresetTexture constants.

msoPresetTextureMixed
msoTextureBouquet
msoTextureCanvas
msoTextureDenim
msoTextureGranite
msoTextureMediumWood
msoTextureOak
msoTexturePapyrus
msoTexturePinkTissuePaper
msoTextureRecycledPaper
msoTextureStationery
msoTextureWaterDroplets
msoTextureWovenMat
msoTextureBlueTissuePaper
msoTextureBrownMarble
msoTextureCork
msoTextureFishFossil
msoTextureGreenMarble
msoTextureNewsprint
msoTexturePaperBag
msoTextureParchment
msoTexturePurpleMesh
msoTextureSand
msoTextureWalnut
msoTextureWhiteMarble

expression.PresetTexture
expression  Required. An expression that returns one of the objects in the Applies To list.

This property is read-only. Use the **PresetTextured** method to set the preset texture for the fill.
Example

This example changes the chart's textured fill format from oak to walnut.

With myChart.ChartArea.Fill
  If .Type = msoFillTextured Then
    If .TextureType = msoTexturePreset Then
      If .PresetTexture = msoTextureOak Then
        .PresetTextured = msoTextureWalnut
        End If
      End If
    End If
  End If
End With
RadarAxisLabels Property

Returns a `TickLabels` object that represents the radar axis labels for the specified chart group. Read-only.
Example

This example turns on radar axis labels for chart group one on the chart and then sets the color for the labels. The example should be run on a radar chart.

With myChart.ChartGroups(1)
    .HasRadarAxisLabels = True
End With
Range Property

Returns a `Range` object that represents the specified cell or range of cells. Read-only `Range` object.

`expression.Range(Range1, Range2)`

`expression` Required. An expression that returns a `DataSheet` object.

**Range 1** Required for a single cell. The name of the specified range. This must be an A1-style reference in the language the macro is written in. It can include the range operator (a colon), the intersection operator (a space), or the union operator (a comma). It can also include dollar signs, but they're ignored.

OR

**Range1 , Range2** Required for a range of cells. The cells in the upper-left and lower-right corners of the specified range. Each argument can be a `Range` object that contains a single cell (or an entire column or entire row), or the argument can be a string that names a single cell in the language the macro is written in.
Remarks

On the datasheet, the first column heading (starting on the left) is A, followed by B, C, D, and so on. The first row heading (starting at the top) is 1, followed by 2, 3, 4, and so on. Neither the leftmost column nor the top row has a heading. In other words, column A is actually the second column from the left; likewise, row 1 is the second row from the top. The leftmost column and the top row, which are commonly used for legend text or axis labels, are referred to as column 0 (zero) and row 0 (zero). Thus, the following example inserts the text "Annual Sales" in the top cell in column A (the second column).

```vba
myChart.Application.DataSheet.Range("A0").Value = "Annual Sales"
```

And the following example inserts the text "District 1" in the leftmost cell in row 2 (the third row).

```vba
myChart.Application.DataSheet.Range("02").Value = "District 1"
```
Example

This example sets the value of cell A1 on the datasheet to 3.14159.

```
myChart.DataSheet.Range("A1").Value = 3.14159
```

This example loops on cells A1:C3 on the datasheet. If one of the cells has a value less than 0.001, the example replaces that value with 0 (zero).

```
With myChart.Application.DataSheet
    For Each c in .Range("A1:C3")
        If c.Value < .001 Then
            c.Value = 0
        End If
    Next c
End With
```
**ReadingOrder Property**

Returns or sets the reading order for the specified object. Can be one of the following **constants**. Read/write **Long**.

- **xlContext**
- **xlLTR** (left-to-right)
- **xlRTL** (right-to-left)

`expression`.**CharacterType**

`expression`  Required. An expression that returns one of the objects in the Applies To list.
Remarks

Some of these constants may not be available to you, depending on the language support (U.S. English, for example) that you've selected or installed.
Example

This example sets the chart title’s reading order to right-to-left.

myChart.ChartTitle.ReadingOrder = xlRTL
ReplacementList Property

Returns the array of AutoCorrect replacements.

```
expression.ReplacementList(Index)
```

- **expression** Required. An expression that returns an **AutoCorrect** object.
- **Index** Optional **Variant**. The row index of the array of AutoCorrect replacements to be returned. The row is returned as a one-dimensional array with two elements: The first element is the text in column 1, and the second element is the text in column 2.
Remarks

Use the `AddReplacement` method to add an entry to the replacement list.
Example

This example searches the replacement list for "Temperature" and displays the replacement entry if it exists.

repl = Application.AutoCorrect.ReplacementList
For x = 1 To UBound(repl)
    If repl(x, 1) = "Temperature" Then MsgBox repl(x, 2)
Next
ReplaceText Property

**True** if text in the list of AutoCorrect replacements is replaced automatically. Read/write **Boolean**.
Example

This example turns off automatic text replacement for the chart.

With myChart.Application.AutoCorrect
    .CapitalizeNamesOfDays = True
    .ReplaceText = False
End With
ReversePlotOrder Property

True if Microsoft Graph plots data points from last to first. Read/write Boolean.
Remarks

This property cannot be used on radar charts.
Example

This example plots data points from last to first on the value axis.

myChart.Axes(xlValue).ReversePlotOrder = True
RGB Property

Returns the red-green-blue value of the specified color. Read-only Long.
Example

This example sets the color of the legend font to the foreground fill color of the plot area.

```csharp
myChart.Legend.Font.Color = __
    myChart.PlotArea.Fill.ForeColor.RGB
```
**RightAngleAxes Property**

*True* if the chart axes are at right angles, independent of chart rotation or elevation. Applies only to 3-D line, column, and bar charts. Read/write *Variant*.

*expression*.**RightAngleAxes**

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Remarks

If this property is True, the Perspective property is ignored.
Example

This example sets the axes to intersect at right angles. The example should be run on a 3-D chart.

\texttt{myChart.RightAngleAxes = True}
Rotation Property

Returns or sets the rotation of the 3-D chart view (the rotation of the plot area around the z-axis, in degrees). The value of this property must be from 0 to 360, except for 3-D bar charts, where the value must be from 0 to 44. The default value is 20. Applies only to 3-D charts. Read/write Variant.
Example

This example sets the rotation of myChart to 30 degrees. The example should be run on a 3-D chart.

myChart.\texttt{Rotation} = 30
Rows Property

Returns a Range object that represents the rows in the specified Range or DataSheet object. Read-only.

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

This example deletes row three on the datasheet.

`myChart.Application.DataSheet.Rows(3).Delete`
ScaleType Property

Returns or sets the value axis scale type. Applies only to the value axis. Read/write XlScaleType.

XlScaleType can be one of these XlScaleType constants.
   xlScaleLinear
   xlScaleLogarithmic

expression.ScaleType

expression  Required. An expression that returns one of the objects in the Applies To list.
Remarks

A logarithmic scale uses base 10 logarithms.
Example

This example sets the value axis to use a logarithmic scale.

```plaintext
myChart.Axes(xlValue).ScaleType = xlScaleLogarithmic
```
SecondaryPlot Property

**True** if the point is in the secondary section of either a pie of pie chart or a bar of pie chart. Applies only to points on pie of pie charts or bar of pie charts. Read/write **Boolean**.
**Example**

This example must be run on either a pie of pie chart or a bar of pie chart. The example moves point four to the secondary section of the chart.

```vbnet
With myChart.SeriesCollection(1)
    .Points(4).SecondaryPlot = True
End With
```
SecondPlotSize Property

Returns or sets the size of the secondary section of either a pie of pie chart or a bar of pie chart, as a percentage of the size of the primary pie. Can be a value from 5 through 200. Read/write Long.
Example

This example must be run on either a pie of pie chart or a bar of pie chart. The example splits the two sections of the chart by value, combining all values under 10 in the primary pie and displaying them in the secondary section. The secondary section is 50 percent of the size of the primary pie.

```vba
With myChart.ChartGroups(1)
    .SplitType = xlSplitByValue
    .SplitValue = 10
    .VaryByCategories = True
    .SecondPlotSize = 50
End With
```
Separator Property

Allows the user to set or return the separator used for the data labels on a chart. Read/write Variant.

expression.Separator

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
**Example**

This example sets the data label separator, for the first series, on the first chart, to a semi-colon.

Sub ChangeSeparator()
    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1).DataLabels.Separator = ";"
End Sub
**SeriesLines Property**

Returns a [SeriesLines](#) object that represents the series lines for the specified stacked bar chart or stacked column chart. Applies only to stacked bar and stacked column charts. Read-only.
Example

This example turns on series lines for chart group one on the chart and then sets their line style, weight, and color. The example should be run on a 2-D stacked column chart that has two or more series.

With myChart.ChartGroups(1)
    .HasSeriesLines = True
    With .SeriesLines.Border
        .LineStyle = xlThin
        .Weight = xlMedium
        .ColorIndex = 3
    End With
End With
Shadow Property

Shadow property as it applies to the AxisTitle, ChartArea, ChartTitle, DataLabel, DataLabels, DisplayUnitLabel, Legend, LegendKey, Point, and Series objects.

True if the font is a shadow font or if the specified object has a shadow. Read/write Boolean.

expression.Shadow

expression Required. An expression that returns one of the above objects.

Shadow property as it applies to the Font object.

True if the font is a shadow font or if the specified object has a shadow. Read/write Variant.

expression.Shadow

expression Required. An expression that returns one of the above objects.
Remarks

For the **Font** object, this property has no effect in Microsoft Windows, but its value is retained (it can be set and returned).
Example

This example adds a shadow to the title of myChart.

myChart.ChartTitle.Shadow = True
ShowBubbleSize Property

Allows the user to show the bubble size for the data labels on a chart. Read/write Boolean.

expression.ShowBubbleSize

expression  Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
Example

This example enables the bubble size to be shown for the data labels of the first series on the first chart.

Sub UseBubbleSize()

    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1).DataLabels.ShowBubbleSize = True

End Sub
ShowCategoryName Property

Allows the user to show the category name for the data labels on a chart. Read/write **Boolean**.

`expression.ShowCategoryName`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
Example

This example enables the category name to be shown for the data labels of the first series on the first chart.

Sub UseCategoryName()
    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1).DataLabels.ShowCategoryName = True
End Sub
ShowChartTipNames Property

True if charts show chart tip names. The default value is True. Read/write Boolean.
Example

This example turns off chart tip names and values.

With myChart.Application
    .ShowChartTipNames = False
    .ShowChartTipValues = False
End With
ShowChartTipValues Property

True if charts show chart tip values. The default value is True. Read/write Boolean.
Example

This example turns off chart tip names and values.

With myChart.Application
    .ShowChartTipNames = False
    .ShowChartTipValues = False
End With
ShowLegendKey Property

True if the data label legend key is visible. Read/write Boolean.
Example

This example sets the data labels for series one to show values and the legend key.

With myChart.SeriesCollection(1).DataLabels
    .ShowLegendKey = True
    .Type = xlShowValue
End With
ShowNegativeBubbles Property

True if negative bubbles are shown for the chart group. Valid only for bubble charts. Read/write Boolean.
Example

This example makes negative bubbles visible for chart group one.

myChart.ChartGroups(1).ShowNegativeBubbles = True
ShowPercentage Property

Allows the user to show the percentage value for the data labels on a chart. Read/write Boolean.

expression.ShowPercentage

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
Example

This example enables the percentage value to be shown for the data labels of the first series on the first chart.

Sub UsePercentage()
    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1) .DataLabels.ShowPercentage = True
End Sub
ShowSeriesName Property

Allows the user to show the series name for the data labels on a chart. Read/write Boolean.

expression.ShowSeriesName

expression Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
Example

This example enables the series name to be shown for the data labels of the first series on the first chart.

Sub UseSeriesName()
    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1).DataLabels.ShowSeriesName = True
End Sub
ShowValue Property

Allows the user to show the value for the data labels on a chart. Read/write Boolean.

expression.ShowValue

expression  Required. An expression that returns one of the objects in the Applies To list.
Remarks

The chart must first be active before you can access the data labels programmatically.
Example

This example enables the value to be shown for the data labels of the first series on the first chart.

Sub UseValue()
    ActiveSheet.ChartObjects(1).Activate
    ActiveChart.SeriesCollection(1)._DataLabels.ShowValue = True
End Sub
Size Property

Returns or sets the size of the font. Read/write Variant.
Example

This example sets the font size for the chart title.

myChart.ChartTitle.Font.Size = 12
SizeRepresents Property

Returns or sets what the bubble size represents on a bubble chart. Read/write XlSizeRepresents.

XlSizeRepresents can be one of these XlSizeRepresents constants.

- xlSizeIsArea
- xlSizeIsWidth

expression.SizeRepresents

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets what the bubble size represents for the chart. (The example assumes that the chart is a bubble chart.)

```plaintext
myChart.ChartGroups(1).SizeRepresents = xlSizeIsWidth
```
Smooth Property

**True** if curve smoothing is turned on for the line chart or scatter chart. Applies only to line and scatter charts. Read/write **Boolean**.

`expression.Smooth`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example turns on curve smoothing for series one. The example should be run on a 2-D line chart.

```csharp
myChart.SeriesCollection(1).Smooth = True
```
SplitType Property

Returns or sets the way the two sections of either a pie of pie chart or a bar of pie chart are split. Read/write XlChartSplitType.

XlChartSplitType can be one of these XlChartSplitType constants.

- xlSplitByPercentValue
- xlSplitByValue
- xlSplitByCustomSplit
- xlSplitByPosition

expression.SplitType

description Required. An expression that returns one of the objects in the Applies To list.
Example

This example must be run on either a pie of pie chart or a bar of pie chart. The example splits the two sections of the chart by value, combining all values under 10 in the primary pie and displaying them in the secondary section.

With myChart.ChartGroups(1)
   .SplitType = xlSplitByValue
   .SplitValue = 10
   .VaryByCategories = True
End With
SplitValue Property

Returns or sets the threshold value separating the two sections of either a pie of pie chart or a bar of pie chart. Read/write Variant.
**Example**

This example must be run on either a pie of pie chart or a bar of pie chart. The example splits the two sections of the chart by value, combining all values under 10 in the primary pie and displaying them in the secondary section.

```vba
With myChart.ChartGroups(1)
    .SplitType = xlSplitByValue
    .SplitValue = 10
    .VaryByCategories = True
End With
```
Strikethrough Property

**True** if the font is struck through with a horizontal line. Read/write **Variant**.

`expression.Strikethrough`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the font in the chart title to strikethrough.

myChart.ChartTitle.Font.Strikethrough = True
Subscript Property

**True** if the specified font is formatted as subscript. The default value is **False**. Read/write **Variant**.
Superscript Property

**True** if the specified font is formatted as superscript. The default value is **False**. Read/write **Variant**.
SurfaceGroup Property

Returns a `ChartGroup` object that represents the surface chart group of a 3-D chart. Read-only `ChartGroup` object.

`expression.SurfaceGroup`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the 3-D surface group to use a different color for each data marker. The example should be run on a 3-D chart.

myChart.SurfaceGroup.VaryByCategories = True
Text Property

Returns or sets the text for the specified object. Read/write String.

expression.Text

expression  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the text for the title of the chart.

With myChart
    .HasTitle = True
    .ChartTitle.Text = "First Quarter Sales"
End With

This example sets the axis title text for the category axis.

With myChart.Axes(xlCategory)
    .HasTitle = True
    .AxisTitle.Text = "Month"
End With
TextureName Property

Returns the name of the custom texture file for the specified fill. Read-only String.

This property is read-only. Use the UserPicture or UserTextured method to set the texture file for the fill.
Example

This example changes the user-defined texture type for the chart's fill format.

With myChart.ChartArea.Fill
    If .Type = msoFillTextured Then
        If .TextureType = msoTextureUserDefined Then
            If .TextureName = "brick.bmp" Then
                .UserTextured "stone.bmp"
            End If
        End If
    End If
End With
TextureType Property

Returns the texture type for the specified fill. Read-only MsoTextureType.

MsoTextureType can be one of these MsoTextureType constants.

msoTexturePreset
msoTextureTypeMixed
msoTextureUserDefined

This property is read-only. Use the UserTextured method to set the texture type for the fill.

expression.TextureType

expression Required. An expression that returns one of the objects in the Applies To list.
Example

This example changes the user-defined texture type for the chart's fill format.

With myChart.ChartArea.Fill
    If .Type = msoFillTextured Then
        If .TextureType = msoTextureUserDefined Then
            If .TextureName = "C:\brick.bmp" Then
                .UserTextured "C:\stone.bmp"
            End If
        End If
    End If
End With
**TickLabelPosition Property**

Describes the position of tick-mark labels on the specified axis. Read/write **XlTickLabelPosition**.

XlTickLabelPosition can be one of these XlTickLabelPosition constants.

- **xlTickLabelPositionHigh**
- **xlTickLabelPositionLow**
- **xlTickLabelPositionNextToAxis**
- **xlTickLabelPositionNone**

*expression*. **TickLabelPosition**

*expression*  Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets tick-mark labels on the category axis to the high position (above the chart).

```csharp
myChart.Axes(xlCategory) _
    .TickLabelPosition = xlTickLabelPositionHigh
```
TickLabels Property

Returns a TickLabels collection that represents the tick-mark labels for the specified axis. Read-only.
Example

This example sets the color of the tick-mark label font for the value axis.

```vbnet
```
TickLabelSpacing Property

Returns or sets the number of categories or series between tick-mark labels. Applies only to category and series axes. Read/write `Long`. 
Remarks

Tick-mark label spacing on the value axis is always calculated by Microsoft Graph.
Example

This example sets the number of categories between tick-mark labels on the category axis.

myChart.Axes(xlCategory).TickLabelSpacing = 10
TickMarkSpacing Property

Returns or sets the number of categories or series between tick marks. Applies only to category and series axes. Read/write Long.
**Remarks**

Use the `MajorUnit` and `MinorUnit` properties to set tick-mark spacing on the value axis.
Example

This example sets the number of categories between tick marks on the category axis.

`myChart.Axes(xlCategory).TickMarkSpacing = 10`
Top Property

Top property as it applies to the Application object.

Returns or sets the position of the Application object. The distance from the top edge of the screen to the top edge of the main Microsoft Graph window. In Windows, if the application window is minimized, this property controls the position of the window icon (anywhere on the screen). Read/write Double.

expression.Top

expression Required. An expression that returns one of the above objects.

Top property as it applies to the AxisTitle, ChartArea, ChartTitle, DataLabel, DataSheet, DisplayUnitLabel, Legend, and PlotArea objects.

The distance from the top edge of the object to the top of row 1 (on a datasheet) or the top of the chart area (on a chart). Read/write Double.

expression.Top

expression Required. An expression that returns one of the above objects.

Top property as it applies to the Axis, LegendEntry, and LegendKey objects.

The distance from the top edge of the object to the top of row 1 (on a datasheet) or the top of the chart area (on a chart). Read-only Double.

expression.Top

expression Required. An expression that returns one of the above objects.

Top property as it applies to the Chart object.

The distance from the top edge of the object to the top of row 1 (on a datasheet) or the top of the chart area (on a chart). Read/write Variant.
expression.Top

expression Required. An expression that returns one of the above objects.
Example

This example sets the position of the top of the chart title.

myChart.ChartTitle.Top = 10
TwoInitialCapitals Property

**True** if words that begin with two capital letters are corrected automatically. Read/write **Boolean**.
Example

This example sets Microsoft Graph to automatically correct words that begin with two capital letters.

With myChart.Application.AutoCorrect
    .TwoInitialCapitals = True
    .ReplaceText = True
End With
Type Property

Type property as it applies to the Axis object.

Returns or sets the axis type. Read/write XlAxisType.

XlAxisType can be one of these XlAxisType constants.
- xlSeriesAxis
- xlCategory
- xlValue

expression.Type

expression  Required. An expression that returns an Axis object.

Type property as it applies to the ChartColorFormat object.

Returns the color type. Read-only Long.

expression.Type

expression  Required. An expression that returns a ChartColorFormat object.

Type property as it applies to the ChartFillFormat object.

Returns the fill type. Read-only MsoFillType.

MsoFillType can be one of these MsoFillType constants.
- msoFillGradient
- msoFillBackground
- msoFillMixed
- msoFillPatterned
- msoFillPicture
- msoFillSolid
**msoFillTextured**

*expression*.Type

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

**Type property as it applies to the DataLabel and DataLabels objects.**

Returns or sets the data label type. Read/write **Variant**.

*expression*.Type

*expression* Required. An expression that returns one of the above objects.

**Type property as it applies to the Series object.**

Returns or sets the series type. Read/write **Long**.

*expression*.Type

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

**Type property as it applies to the Trendline object.**

Returns or sets the trendline type. Read/write **XlTrendlineType**.

XlTrendlineType can be one of these XlTrendlineType constants.

- xlExponential
- xlLinear
- xlLogarithmic
- xlMovingAvg
- xlPolynomial
- xlPower

*expression*.Type

*expression* Required. An expression that returns a **Trendline** object.
Example

As it applies to the Trendline object.

This example changes the trendline type for the first series in the chart. If the series has no trendline, this example fails.

myChart.SeriesCollection(1).Trendlines(1).Type = xlMovingAvg
Underline Property

Returns or sets the type of underline applied to the font. Required `XlUnderlineStyle`.

`XlUnderlineStyle` can be one of these `XlUnderlineStyle` constants.

- `xlUnderlineStyleNone`
- `xlUnderlineStyleSingle`
- `xlUnderlineStyleDouble`
- `xlUnderlineStyleSingleAccounting`
- `xlUnderlineStyleDoubleAccounting`

`expression.Underline`

`expression` Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the font in the chart title to single underline.

myChart.ChartTitle.Font.Underline = xlUnderlineStyleSingle
UpBars Property

Returns an **UpBars** object that represents the up bars on a line chart. Applies only to line charts. Read-only.
Example

This example turns on up and down bars for chart group one and then sets their colors. The example should be run on a 2-D line chart containing two series that cross each other at one or more data points.

With myChart.ChartGroups(1)
  .HasUpDownBars = True
  .DownBars.Interior.ColorIndex = 3
End With
Value Property

Returns the value of the specified cell. If the cell is empty, Value returns the value Empty (use the IsEmpty function to test for this case). If the Range object contains more than one cell, this property returns an array of values (use the IsArray function to test for this case). Read/write Variant.

expression. Value(RangeValueDataType)

expression  Required. An expression that returns one of the objects in the Applies To list.

RangeValueDataType  Optional Variant.
Example

This example sets the value of cell A1 on the datasheet to 3.14159.

\[
\text{myChart.Application.DataSheet.Range("A1").Value} = 3.14159
\]
VaryByCategories Property

**True** if Microsoft Graph assigns a different color or pattern to each data marker. The chart must contain only one series. Read/write **Boolean**.
Example

This example assigns a different color or pattern to each data marker in chart group one. The example should be run on a 2-D line chart that has data markers on a series.

```
myChart.ChartGroups(1).VaryByCategories = True
```
Version Property

Returns the Microsoft Graph version number. Read-only String.
Example

This example displays a message box that contains the Microsoft Graph version number.

MsgBox "Welcome to Microsoft Graph version " & _
myChart.Application.Version
**VerticalAlignment Property**

Returns or sets the vertical alignment of the specified object. Required **XlVAlign**.

XlVAlign can be one of these XlVAlign constants.

- **xlVAlignBottom**
- **xlVAlignCenter**
- **xlVAlignDistributed**
- **xlVAlignJustify**
- **xlVAlignTop**

`expression.VerticalAlignment`

*expression* Required. An expression that returns one of the objects in the Applies To list.
Remarks

Some of these constants may not be available to you depending on the language support (U.S. English, for example) that you've selected or installed.
Example

This example centers the chart title vertically.

myChart.ChartTitle.VerticalAlignment = xlCenter
Visible Property

Visible property as it applies to the Application object.

Determines whether the object is visible. Read/write Boolean.

expression.Visible

expression Required. An expression that returns an Application object.

Visible property as it applies to the ChartFillFormat object.

Determines whether the application is visible. Read/write MsoTriState.

MsoTriState can be one of these MsoTriState constants.
msCTrue
msoFalse
msoTriStateMixed
msoTriStateToggle
msoTrue The object is visible.

expression.Visible

expression Required. An expression that returns a ChartFillFormat object.
Example

As it applies to the **ChartFillFormat** object.

This example formats the chart's fill with a preset gradient and then makes the fill visible.

```vba
With myChart.ChartArea.Fill
    .Visible = msoTrue
    .PresetGradient msoGradientDiagonalDown, 3, msoGradientBrass
End With
```
Walls Property

Returns a Walls collection that represents the walls of the 3-D chart. Read-only.
Remarks

This property doesn't apply to 3-D pie charts.
Example

This example sets the color of the wall border of the chart to red. The example should be run on a 3-D chart.

myChart.Walls.Border.ColorIndex = 3
WallsAndGridlines2D Property

**True** if gridlines are drawn two-dimensionally on a 3-D chart. Read/write Boolean.
Example

This example causes Microsoft Graph to draw 2-D gridlines for the chart.

myChart.WallsAndGridlines2D = True
Weight Property

Returns or sets the weight of the border. Read/write XlBorderWeight.

XlBorderWeight can be one of these XlBorderWeight constants. 

- xlHairline 
- xlThin 
- xlMedium 
- xlThick 

**expression**.Weight

**expression** Required. An expression that returns one of the objects in the Applies To list.
Example

This example sets the border weight for the chart area.

```plaintext
myChart.ChartArea.Border.Weight = xlMedium
```
Width Property

As it applies to the Application object, the Width property determines the distance from the left edge of the application window to the right edge of the application window. For all other objects, the Width property, determines the width of the object. Read/write Double for all objects, except for the Chart object, which is read/write Variant.

expression.Width

expression   Required. An expression that returns one of the above objects.

If the window is minimized, Application.Width is read-only and returns the width of the window icon.
Example

This example sets the width of the chart.

\texttt{myChart.Width = 360}
WindowState Property

Returns or sets the state of the window. Read/write \texttt{XLWindowState}.

\texttt{XLWindowState} can be one of these \texttt{XLWindowState} constants.
\begin{itemize}
  \item \texttt{xlMaximized}
  \item \texttt{xlNormal}
  \item \texttt{xlMinimized}
\end{itemize}

\textit{expression}.\texttt{WindowState}

\textit{expression} Required. An expression that returns one of the objects in the Applies To list.
Example

This example maximizes the Microsoft Graph application window.

myChart.Application.WindowState = xlMaximized
Microsoft Graph Objects
(ChartGroup)

ChartGroups (ChartGroup)
- DownBars
- UpBars
- HiLoLines
- SeriesLines
- DropLines
- TickLabels
- SeriesCollection (Series)
  - ErrorBars
  - Border
- DataLabels (DataLabel)
- ChartFillFormat
  - Interior
  - LeaderLines
- Points (Point)
  - DataLabel
  - Trendlines (Trendline)

Legend
Object and collection
Object only
SchemeColor Property

Returns or sets the color of the specified ChartColorFormat object as an index in the current color scheme. Read/write Long.
Example

This example sets the foreground color, background color, and gradient for the chart area fill on the chart.

With myChart.ChartArea.Fill
    .Visible = True
    .ForeColor.SchemeColor = 15
    .BackColor.SchemeColor = 17
    .TwoColorGradient msoGradientHorizontal, 1
End With
Returning an Object from a Collection

The **Item** method returns a single object from a collection. The following example sets the `firstEntry` variable to a `LegendEntry` object that represents the first legend entry.

```
Set firstEntry = myChart.Legend.LegendEntries.Item(1)
```

The **Item** method is the default method for most collections, so you can write the same statement more concisely by omitting the **Item** keyword.

```
Set firstEntry = myChart.Legend.LegendEntries(1)
```

For more information about a specific collection, see the Help topic for that collection.