Help with the Graph control (Design-time)

Using the property pages

What are property pages?

Entering the property pages at design time

Giving your users access to property pages

Disabling selected property pages

Replacing property pages with your own dialogs

Reference

Graph control property pages

Graph control properties and events

Label formats

Graph control property pages

2D Gallery page	Select a 2D graph type
3D Gallery page	Select a 3D graph type
3D page	Set viewing angle and point of view. Adjust appearance of cage.
Axis page	Position axes. Add tick marks and grid lines.
Background page	Select colors for titles, legends, and the graphing window. Select an image for a backdrop. Select a color palette.
Bound data page	Select which fields in the bound database provide graph data, labels, and legend text.
Data page	Enter or adjust X, Y, and Z data. Screen out selected values.
Design page	Enable or disable property pages. Turn on hot-graphing. Select drawing style and initial drawing mode.
Error Bar page	Add and adjust error bars.
Fonts page	Select a font name, treatment, and size for titles, labels, and legend text.
Labels page	Select the type, orientation, and format for axis and data labels.
Legends page	Enter text for the legend. Set legend size and position.
Markers page	Select symbols and line styles. Adust color, size, and thickness.
Overlay page	Define a combination graph.
Style page	Select style options for the current graph type.
System page	Print a graph. Save a graph as an image or as a template. Load a graph previously saved in a template. Define image-mapping options.
Titles page	Enter text for titles. Orient left and right titles.
Trends page	Add statistical lines and limit lines. Select options for curve-fitting.

What are property pages?

The Graph control's property pages are a set of tabbed dialogs that let you design a graph interactively, without scrolling through a list in the property window. The property pages are always available at design time. By default, they are not available at run time. However, you can choose to make some or all pages available to your users at run time.

Entering the property pages (Design time)

You can enter the property pages in either of two ways:

- Click the right mouse button within a graphing window.

  The Graph control displays the first property page.   At design time, this is always the 2D Gallery property page.
- If the graph toolbar is enabled, click the left mouse button on a toolbar icon.

  The Graph control displays the property page associated with the toolbar icon.

Once you have entered the property pages, you can move from one property page to another by clicking on the tab for the desired page at the top of the property pages window.

▲▼

Topic

Entering the property pages

Related

Property pages

Giving your users access to property pages

By default, your users cannot enter the property pages. However, you can enable your users to participate in the design of a graph by making the property pages available at run time.

▶ To enable access through the right mouse button

1. Enter the property pages.
2. Choose the Design page.
3. In the Property Pages group, check Run Time on Right Click.

   At run time, the Graph control will display the first property page when a user clicks the right mouse button within the graphing window.

   You can also enable the property pages at run time by setting the PropertyPages property in code:

   ```
   Graph1.PropertyPages = 1 'Enabled at run time
   ```

▶ To enable access through the toolbar

1. Enter the property pages.
2. Choose the Design page.
3. In the Toolbar group, check the box labeled Run Time.

   At run time, the graph toolbar will appear at the top of the graphing window. When a user clicks one of the toolbar icons, the Graph control will display the property page associated with the icon.

   You can also enable the toolbar at run time by setting the Toolbar property in code:

   ```
   Graph1.Toolbar = 2 'Enabled at run time
   ```

▲▼

Topic

Giving your users access to property pages

Related

Disabling selected property pages

PropertyPages property

Toolbar property

ToolStat property

Disabling selected property pages

If you have enabled run-time access to the property pages, you can customize the editing interface by disabling some pages.   Any pages that you disable will not appear when the Graph control's tabbed dialog displays.

▶ To disable individual property pages

1. Enter the property pages.
2. Choose the Design page.
3. Do one or both of the following:
   - In the Property Pages group, check Run Time on Right Click.
   - In the Toolbar group, check the box labeled Run Time.
4. In the Run Time Options group, select Disable Page and Icon.
5. Check the box for each of the pages you wish to disable.

   At run time, your users will not be able to access the property pages you have disabled.   If you're using the toolbar at run time, the toolbar icons for the selected property pages are also disabled.

   You can also disable individual property pages in code. For details, see the ToolStat property.

▲▼

Topic

Disabling selected property pages

Related

Design property page

ToolStat property

Replacing property pages with your own dialogs

If you have enabled run-time access to the property pages, you can customize the editing interface by replacing the Graph control's property pages with your own dialogs.

▸ To replace the Graph control's property pages

1. Enter the property pages.
2. Choose the Design page.
3. Do one or both of the following:
   - In the Toolbar group, check the box labeled Run Time.

     This enables access to the property pages through the toolbar icons.
   - In the Property Pages group, check Run Time on Right Click.

     This enables access to the property pages through a right mouse click.
4. In the Run Time Options group, select Disable Page/Enable Event.
5. Check the box for each of the pages you wish to disable.

   In this case, disabling a property page enables a ToolHit event.   The event is fired:
   - If the toolbar is enabled and a user clicks the icon associated with a disabled property page
   - If right-mouse clicks are enabled, the Gallery page is disabled, and a user clicks the right mouse button within the graphing window
6. Write a ToolHit event procedure to display your own dialog when the event is fired.

   When a ToolHit event is fired, the Graph control returns a number indicating which property page was requested.   If you enable events for several pages, your ToolHit event procedure should check the value returned to determine how to respond. For details, see the entries for ToolStat and ToolHit in the alphabetical property reference.

▲▼

Topic

Replacing property pages

Related

Design property page

Disabling selected property pages

ToolStat property

ToolHit event

OK

When you click this button, the property pages disappear and the graph is redrawn with any changes you've made.

Related Property

DrawMode

Cancel/Close

This button can have two markings.

Cancel indicates that you have changed settings on the current property page but haven't yet applied the changes to the graph.   If you click the Cancel button at this point, the property pages disappear and the graph is redrawn without changes.

Close means you've already applied changes using the Apply Now button.   Click the Close button to exit the property pages.

Apply Now

Apply Now is similar to Ok. In both cases, changes are applied and the graph is redrawn. The one difference is that when you click Ok, the property pages disappear and you are returned to your project.   When you click Apply Now, the property pages remain active.

Note: When you turn from one property page to the next, any changes you've specified on the current page are applied to the graph before the next property page displays.

Related Property

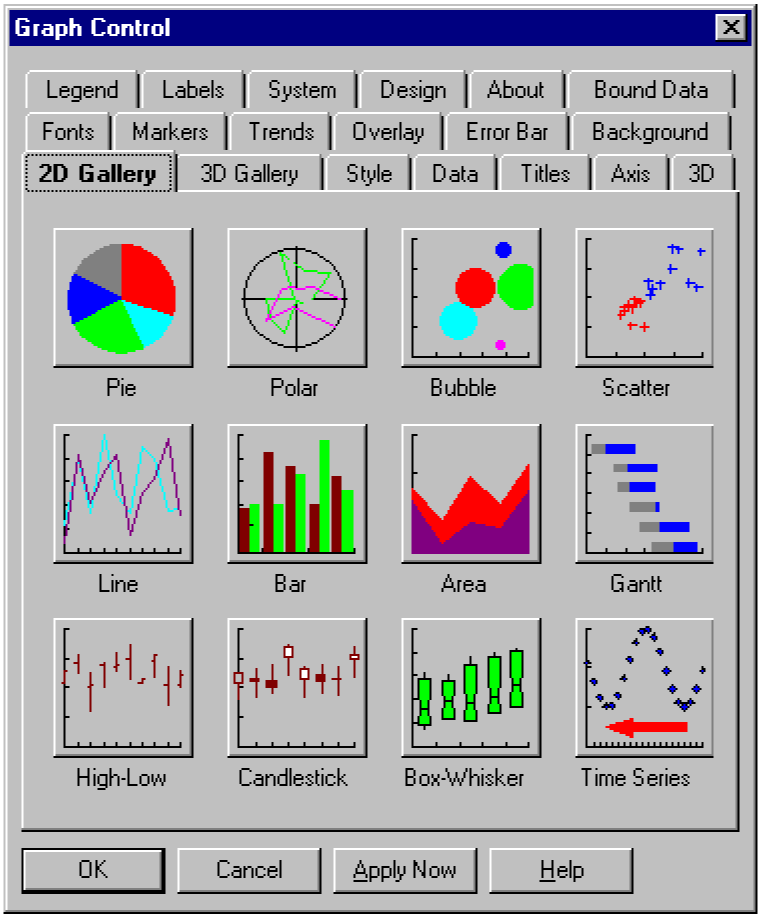
DrawMode

Help

When you click this button, the Graph control Help file opens.

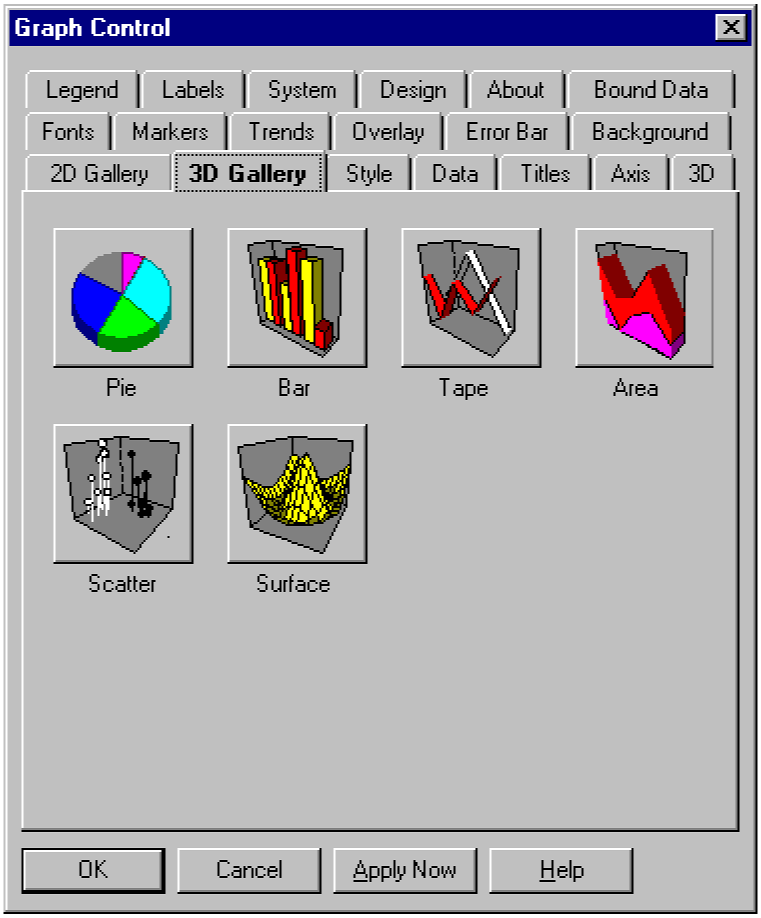
2D Gallery property page

Click where you need help



3D Gallery property page

Click where you need help



Pie chart

The pie chart, one of the simplest graph types, consists of a circle (or "pie") divided into two or more sections ("slices"). Pie charts show the proportion of parts to the whole. By labeling each pie slice with the quantity it represents, you can also allow a comparison of parts to each other, although not as effectively as you could with a bar graph or other graph type.

- Each pie chart can graph only one data set, with each data point represented by a pie slice.
- Negative data points are ignored and not shown.
- You can draw any pie chart in either 2D or 3D form.
- You can highlight any slice by "exploding" it (moving it slightly away from the center of the pie).

  To explode pie slices, go to the Data property page, click the Exploded Slices button (which appears only when you've selected a pie chart in the 2D Gallery or 3D Gallery property page), and enter a 1 for each data point (slice) you want to explode.

Related Property

GraphType = 1 (2D pie)
GraphType = 2 (3D pie)

Polar graph

The polar graph is essentially a line graph drawn on a circular grid. The line relates values to angles. Like logarithmic graphs, polar graphs are useful primarily in mathematical and statistical applications.

- In a polar graph, the independent variable is charted on the angular axis, based on an origin (zero point) of three o'clock.   The dependent variable is charted on the radial axis, with the origin at the center of the circle.
- Polar graphs can chart multiple data sets, each represented by a single line, with as many data points as are meaningful.
- If you don't supply an angular position for each data point, the Graph control automatically places the first point at an angle of 0, with subsequent points at increments of 360 (degrees) divided by the total number of points.
- In drawing a polar graph, you can use any combination of lines, symbols, and "sticks" drawn between points and the center origin.

Related Property

GraphType = 10

Bubble graph

The bubble graph lets you chart three variables in two dimensions.   It's a special form of the scatter graph in which the size of a circular marker (the *bubble*) for a data point is used to represent a value. For example, the size of a bubble might represent a product's percentage of gross sales; the bubble's position along the Y axis might represent market size; and the position along the X axis might represent the number of competing products.

- In a bubble graph, all three variables are independent.   You can choose which variable to show on the X axis, which to show on the Y axis, and which to show by the size of the bubble.
- You must supply values for the X position, Y position, and bubble size for each point.   If you have access to the Data property page, you can set these values by pressing the X and Y Position button (the top row of the dialog contains the X positions and the bottom row the Y positions) and the Bubble Size button.
- You can't draw curves on a bubble graph.

Related Property

GraphType = 12

2D scatter graph

The 2D scatter graph consists of plotted points "scattered" around an X-Y grid.   The pattern may reveal a relationship between the two variables measured by the X and Y axes.   You can illustrate trends in the plotted points by adding a curve (go to the Trends property page). You can also indicate the range of error or uncertainty in the data by applying error bars (go to the Error Bar property page).

- Scatter graphs can chart multiple data sets, each having any number of data points. Each set can be represented by a different symbol.
- You can display scatter plots alone, curves alone, or both together.   The combination is determined by your selections in the Style and Trends property pages.

Related Property

GraphType = 9

3D scatter graph

The 3D scatter graph consists of plotted points "scattered" around an X-Y-Z space. The pattern may reveal a relationship between the three variables measured by the X, Y, and Z axes.

- Scatter graphs can chart multiple data sets, each having any number of data points. Each set can be represented by a different symbol.
- In defining a scatter graph, you usually supply an X position for each data point. If you supply no X positions, the Graph control automatically places points at X increments of 1, starting at 0.
- You can emphasize the Y value of plotted points by selecting the Sticks option in the Style property page.
- You can link points in a set with a connecting line by selecting the Lines option in the Style property page. The effect is equivalent to a 3D line graph with symbols marking points.

Related Property

GraphType = 21

Line graph

The line graph consists of one or more lines (or sequences of symbols) drawn on an X-Y grid.   Lines graphs let you show trends in values along a continuous scale.

- In a line graph, the X axis usually represents an independent variable, which is most often a time scale.   The Y axis usually shows a dependent variable, such as a quantity or percentage.
- In drawing a line graph, you can use any combination of lines, symbols, and vertical "sticks."   You can choose this combination in the Style property page.
- In the Style property page, you can select options to apply logarithmic scaling to the X axis, Y axis, or both--creating a lin/log, log/lin, or log/log graph..
- You can create special graphing effects by drawing an "overlay" line graph on top of a another graph (bar, area, scatter, high-low-close, or another line graph).   To enable an overlay graph, go to the Overlay property page.

Related Property

GraphType = 6 (Line)
GraphType = 7 (Log/Lin)
GraphType = 15 (Log/Log)
GraphType = 16 (Lin/Log)

2D bar graph

A bar graph consists of two or more parallel bars of equal width drawn on an X-Y grid.   Bar graphs compare amounts to each other.   They can also suggest trends, especially in vertical form.

- 2D bar graphs are available in five styles: simple, stacked, stacked percentage, stacked floating, and Pareto.   You can choose a style on the Style property page.
- You can draw bars either vertically or horizontally.   In the vertical format, viewers tend to attribute a left-to-right sequence to the bars, whether you intend one or not.   To choose vertical or horizontal bars, go to the Style property page.

Related Property

GraphType = 3

3D bar graph

A bar graph consists of two or more parallel bars of equal width drawn on an X-Y grid. Bar graphs compare amounts to each other. They can also suggest trends, especially in vertical form.

- 3D bar graphs are available in six styles: simple, stacked, stacked percentage, stacked floating, Pareto, and z-clustered. You can choose a style on the Style property page.
- You can draw bars either vertically or horizontally. In the vertical format, viewers tend to attribute a left-to-right sequence to the bars, whether you intend one or not. To choose vertical or horizontal bars, go to the Style property page.

Related Property

GraphType = 4

Area graph

The area graph consists of one or more lines drawn on an X-Y grid, with the area between the line and the X axis filled in. Like line graphs, area graphs show trends in values but area graphs give greater emphasis to quantities.

- Area graphs are available in three forms: stacked, absolute, and stacked percentage. You can choose one of these forms on the Style property page.
- Negative data points are ignored and not shown.
- You can draw any area graph in either 2D or 3D form.

Related Property

GraphType = 8 (2D Area)
GraphType = 14 (3D Area)

Gantt chart

The Graph control's Gantt chart is a specialized version of the horizontal bar graph in simple or stacked form.   It's used almost exclusively to show a project schedule, with each bar or bar segment marking the start time, duration, and completion time of a task.

- Depending on your needs, you can have each bar represent either a single task (one solid bar) or a sequence of tasks (stacked bar).
- Gantt charts are made up of at least two data sets.   The first set contains the values for the start point of each bar, and subsequent sets contain the end points of each bar segment.
- Unlike bar graphs, Gantt charts are always drawn horizontally and only in 2D form.
- The Graph control automatically places Gantt bars along the Y axis at increments of 1, starting at 1.
- In default form, Gantt chart bars are drawn with no spaces between them.   You can add spaces if you choose.   Go to the Style property page and select the Spaced option.

  You can't show negative data points on a Gantt chart.

Related Property

GraphType = 5

High-low-close graph

The high-low-close (HLC) graph lets you chart a range of values on an X-Y grid. The range is shown as a vertical bar, with horizontal crossbars for the high, the low, and a normative value usually called the close. An alternate version, the open-high-low-close (OHLC) graph, adds a fourth crossbar for another normative value usually called the open.

- When you click the High-Low icon on the 2D Gallery property page, you get an HLC graph by default. To get an OHLC graph, go to the Style property page and select the 'Open' Values option.
- An HLC graph must have three data sets (high, low, and close values), and an OHLC graph must have four data sets (open, high, low, and close values). There's no limit on the number of data points you can graph, but each data set should have the same number.
- You can optionally draw the graph without the open and close bars, without the high and low bars, or with no bars at all. These options are available in the Style property page.

Related Property

GraphType = 11 (HLC)
GraphType = 18 (OHLC)

Candlestick graph

The candlestick graph is an alternative to the open-high-low-close graph. It consists of a series of boxes, with lines extending up and down from the ends, drawn on an X-Y grid. The top and bottom of each box indicate the open and close values. If the open value is higher, the box is filled with a color; if the close value is higher, the box is filled with white. The ascending and descending lines indicate the high and low values for that point.

- The candlestick graph requires four data sets (open, high, low, and close values), each of which should have the same number of data points.
- If you don't supply an X position for each data point, the Graph control automatically places points at increments of 1, starting at 0.
- No style variants are available.

Related Property

GraphType = 19

Box-whisker graph

The box-whisker graph illustrates the spread of data groups around their medians, using a "box" and "whiskers" to break down each data group by percentile.

In creating a box-whisker graph, you can either specify the seven percentile parameters for each symbol (provide "parametric" data) or supply a group of "raw" data for the Graph control to process and graph.

- With raw data, you supply as many sets of data as you want and the Graph control computes percentile parameters. For example, if you are graphing the scores achieved by 24 students on five tests, you need 24 sets of data with five points per set. The control will analyze the data and draw five box-whiskers, one for each test. Each box-whisker will show the percentile distribution of scores for a single test.
- With parametric data, you compute percentile data yourself and send it to the Graph control as exactly seven data sets, which specify the values at percentiles of 5, 10, 25, 50 (the median), 75, 90, and 95. The number of points within each set determines how many box-whiskers are drawn.

Related Property

GraphType = 17

Time series graph

Unlike other graphs, the time series graph shows open-ended streams of data, rather than finite data sets. This graph is ideal for plotting real-time data.

A time series graph is drawn on a dynamic X-Y grid. Points are added one at a time to the right-hand edge. When the graph reaches the limit of points it can show, the oldest data begins to drop off the left edge.   As a result, the graph appears to move on the screen.

- Time series graphs can chart multiple data sets, each represented by a single sequence of symbols.
- Because time series graphs represent continuous streams of data, they must be displayed on screen to show all the data.   Printouts of time series graphs can show only freeze-frames of the graph captured at particular times.
- The data for a time series graph must be provided by the application. You cannot enter data for this type of graph from the property pages.

Related Property

GraphType = 22

Tape graph

The tape graph is a 3D form of the line graph.   It gives you only one styling option--"tapes" drawn between data points--but is otherwise the same as the line graph.

Related Property

GraphType = 13

Surface graph

The surface graph lets you represent data topographically in three dimensions. The graph uses an X-Z grid drawn at regular increments in the X and Z directions, with one Y value for each X-Z intersection. The color scale of the graph is automatically keyed to the height of points, helping the viewer differentiate between higher and lower values.

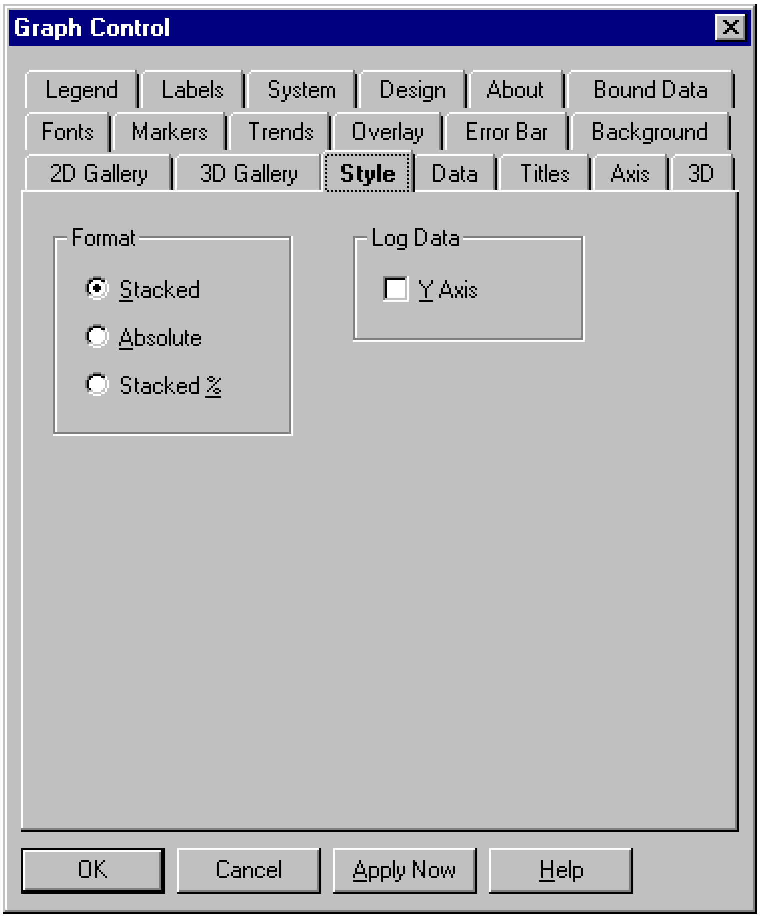
- A surface graph represents a minimum of two data sets and usually at least three. Each data set holds the Y values of a row of points along the X axis. The first set applies to the row of points perpendicular to the Z origin (the "back" of the graph), and subsequent sets apply to additional rows.
- All panels of the surface graph (the rectangles formed by the X and Z grids) are colored according to their height. You specify the colors at the maximum and minimum points of the axis, and the Graph control interpolates colors between these points.
- In drawing a surface graph, you can use lines to show the edges of each panel, fill each panel with a solid color, or use both lines and fills. You can also add side walls to the front and right edges of the graph if you choose. All of these options are available in the Style property page.

Related Property

GraphType = 20

Style page (Area graph--2D)

Click where you need help



Format

● Stacked

○ Absolute

○ Stacked %

Log Data

□ Y Axis

▲▼

Topic

Style page (Area graph--2D)

Related

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

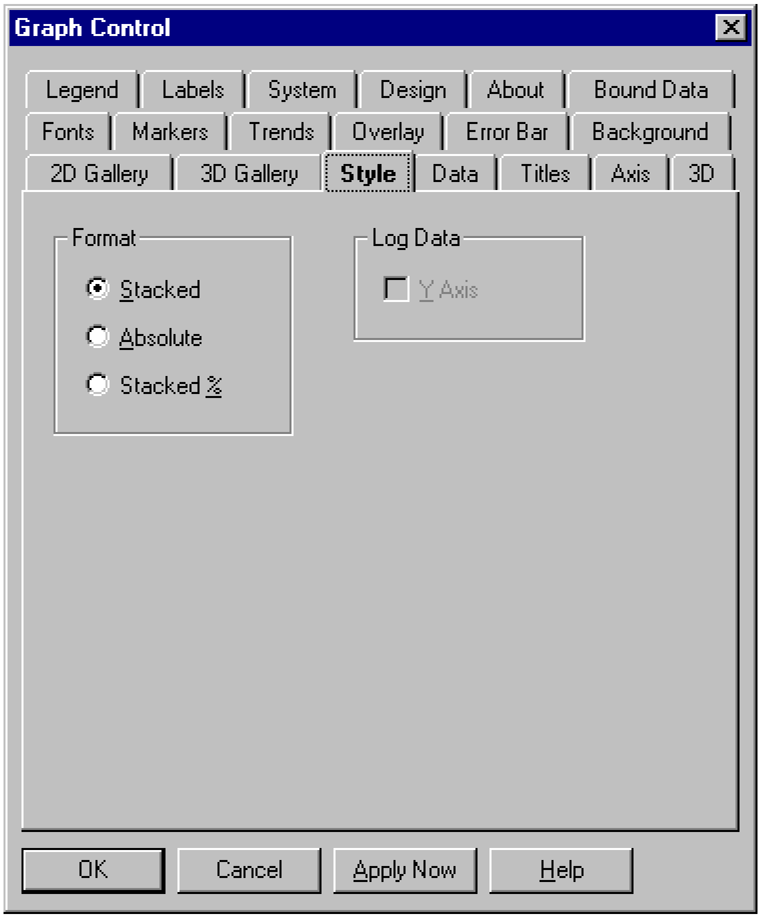
Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Style property page (Area graph--3D)

Click where you need help



Format

● Stacked

○ Absolute

○ Stacked %

Log Data

□ Y Axis

▲▼

Topic

Style page (Area graph--3D

Related

Style page (Area graph--2D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Stacked (default)

Select this option for "stacked"--cumulative--layers. Each data set is drawn on top of any previous ones.

Related Property

GraphStyle = 0

Absolute

Select this option to draw each data set from the Y origin. This style is more readable in 3D, where the graph can be rotated to reveal data sets obscured by those in front.

Related Property

GraphStyle = 1

Stacked %

Select this option to draw each data set as a percentage of a whole.

Related Property

GraphStyle = 2

Log Y

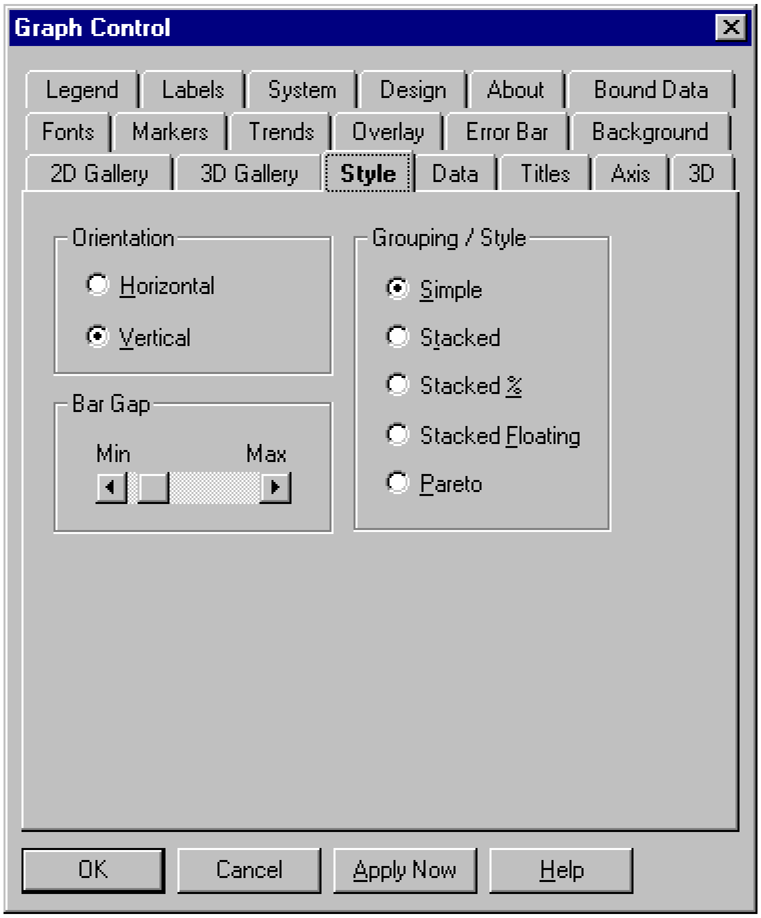
Select this option for a logarithmic Y axis.

Related Property

GraphStyle = 3 (semi-log stacked)
GraphStyle = 4 (semi-log absolute)

Style page (Bar graph--2D)

Click where you need help



Orientation

○ Horizontal

● Vertical

Bar Gap

Grouping / Style

● Simple

○ Stacked

○ Stacked %

○ Stacked Floating

○ Pareto

▲▼

Topic

Style page (Bar graph--2D)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

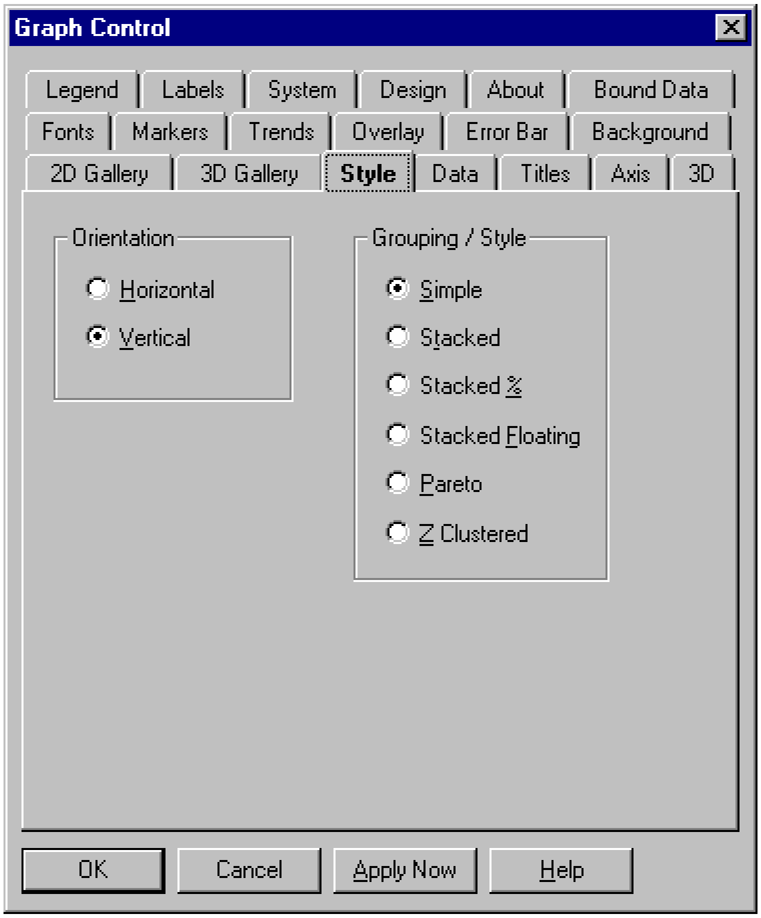
Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Style page (Bar graph--3D)

Click where you need help



Orientation

○ Horizontal

● Vertical

Grouping / Style

● Simple

○ Stacked

○ Stacked %

○ Stacked Floating

○ Pareto

○ Z Clustered

▲▼

Topic

Style page (Bar graph--3D)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Horizontal

Select this option for a graph with bars drawn horizontally.

Related Property

GraphStyle = 1 (Horizontal)
GraphStyle = 1 + 2 (Horizontal stacked)
GraphStyle = 1 + 4 (Horizontal stacked percentage)
GraphStyle = 1 + 8 (Horizontal stacked floating)

Vertical (default)

Select this option for a graph with bars drawn vertically, which is sometimes called a column graph.

Related Property

GraphStyle = 0 (Vertical)
GraphStyle = 0 + 2 (Vertical stacked)
GraphStyle = 0 + 4 (Vertical stacked percentage)
GraphStyle = 0 + 8 (Vertical stacked floating)

Bar Gap

Adjusts the space between adjacent bars.

Related Property

Bar2DGap = percentage

Simple (default)

Select this option to draw one vertical bar per data point. If you have more than one data set, bars for corresponding points from each set are grouped together.

Related Property

GraphStyle = 0

Stacked

Select this option to divide bars into segments, showing multiple data sets at corresponding data points.

Related Property

GraphStyle = 0 + 2   (Vertical stacked)
GraphStyle = 1 + 2 (Horizontal stacked)

Stacked Floating

Select this option to make the first segment of a bar invisible so that the bar appears to float above the axis.

Note: This style requires at least two data sets. The values in the first data set specify the length of the invisible segment.

Related Property

GraphStyle = 0 + 8   (Vertical floating)
GraphStyle = 1 + 8 (Horizontal floating)

Stacked %

Select this option to divide bars into segments representing percentages of a whole. Each complete bar will be the same length, but the breakdown of segments will vary according to their percentages at each data point.

Related Property

GraphStyle = 0 + 4 (Vertical stacked percentage)
GraphStyle = 1 + 4 (Horizontal stacked percentage)

Pareto

Select this option to sort bars by size. If the bars are vertical, they will be sorted in descending order, from left to right. If they are horizontal, bars will be sorted in descending order from top to bottom. Label text moves with the bar to which it is assigned.

Related Property

GraphStyle = 0 + 10 (Vertical Pareto)
GraphStyle = 1 + 10 (Horizontal Pareto)

Z-Clustered

Select this option to cluster multiple sets of bars in rows ranked from back to front. That is, the first set of bars is drawn at the back of the 3D cage. The second set is drawn in front of the first. The third is drawn in front of the second...and so on.

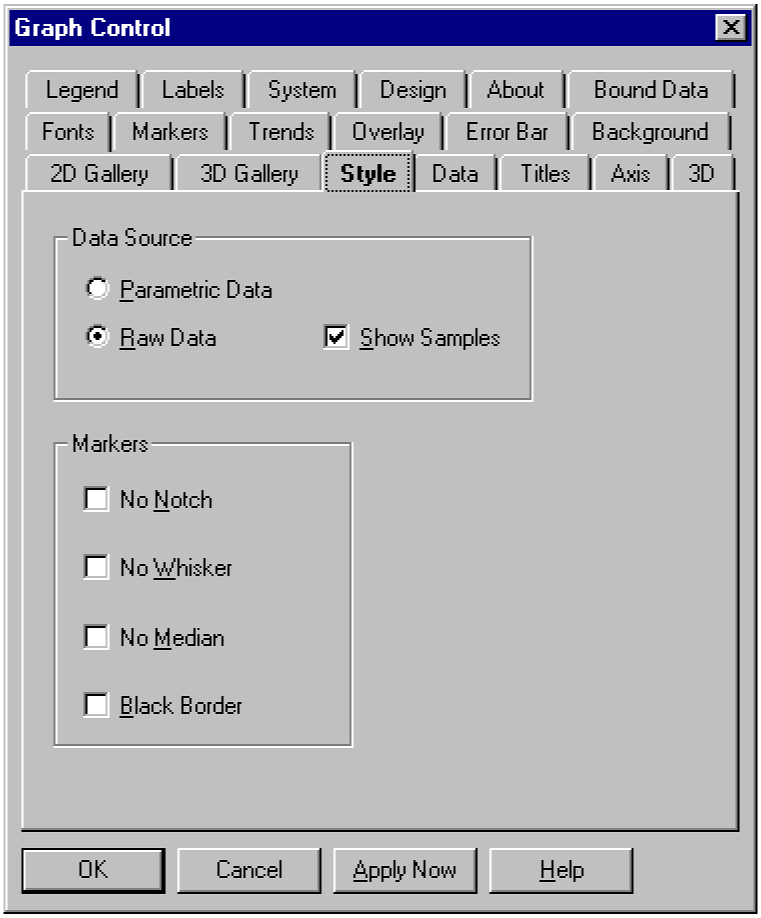
With this style, small bars may be hidden behind larger values. You may need to rotate the graph in order to see all bars. (To rotate a graph, go to the 3D property page.)

Related Property

GraphStyle = 0 + 6 (Vertical clustered)
GraphStyle = 1 + 6 (Horizontal clustered)

Style page (Box-whisker graph)

Click where you need help



Data Source

○ Parametric Data

● Raw Data ☑ Show Samples

Markers

□ No Notch

□ No Whisker

□ No Median

□ Black Border

▲▼

Topic

Style page (Box-whisker graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Parametric Data

Select this option if you want to compute percentile data yourself. You will need exactly seven data sets, which specify the values at percentiles of 5, 10, 25, 50 (the median), 75, 90, and 95. The number of points within each set determines how many box-whiskers are drawn.

Related Property

GraphStyle = 2

Raw Data (default)

Select this option if you want the Graph control to process a group of data and produce the seven values making up the box-whisker.

The number of data sets you use is up to you to decide. For example, if you are graphing the scores achieved by 24 students on five tests, you need 24 sets of data with five points per set. The control will analyze the data and draw five box-whiskers, one for each test. Each box-whisker will show the percentile distribution of scores for a single test.

Related Property

GraphStyle = 0

Show Samples (default is on)

Select this option to show the actual data values (samples) from which the box-whisker graph is produced, superimposed as symbols over the box-whisker graphics. Clear the box to show no samples.

This option is available only when you are graphing Raw Data.

Related Property

GraphStyle =   0 (Raw data with samples)
GraphStyle = 1 (Raw data without samples)

No notch

Select this option to draw box markers without a notch at the median. Clear the checkbox to draw a notch.

This is an additive property designed to be used in combination with other GraphStyle settings. Additive settings cannot be made through the property window. Set these combinations either in code or through the property pages.

Related Property

GraphStyle = 0 + 4 (Raw data with samples, no notch)
GraphStyle = 1 + 4 (Raw data without samples, no notch)
GraphStyle = 2 + 4 (Parametric data, no notch)
GraphStyle = 2 + 4 + 8 +16 +32 (Parametric data, no notch, no whiskers, no median, black border)

No whiskers

Select this option to omit whiskers. Clear the checkbox to draw whiskers.

This is an additive property designed to be used in combination with other GraphStyle settings. Additive settings cannot be made through the property window. Set these combinations either in code or through the property pages.

Related Property

GraphStyle = 0 + 8 (Raw data with samples, no whiskers)
GraphStyle = 1 + 8 (Raw data without samples, no whiskers)
GraphStyle = 2 + 8 (Parametric data, no whiskers)
GraphStyle = 0 + 4 + 8 +16 +32 (Raw data with samples, no notch, no whiskers, no median, black border)

No median

Select this option to omit a median line. Clear the checkbox to draw a median line.

Note: By default, median lines are drawn in the same color as the marker. This means they will not be visible when the fill pattern is solid.

This is an additive property designed to be used in combination with other GraphStyle settings. Additive settings cannot be made through the property window. Set these combinations either in code or through the property pages.

Related Property

GraphStyle = 0 + 16 (Raw data with samples, no median)
GraphStyle = 1 + 16 (Raw data without samples, no median)
GraphStyle = 2 + 16 (Parametric data, no median)
GraphStyle = 1 + 4 + 8 +16 +32 (Raw data without samples, no notch, no whiskers, no median, black border)

Black border

Select this option to outline the box-whisker symbols with a black border line. Clear the checkbox to omit the border.

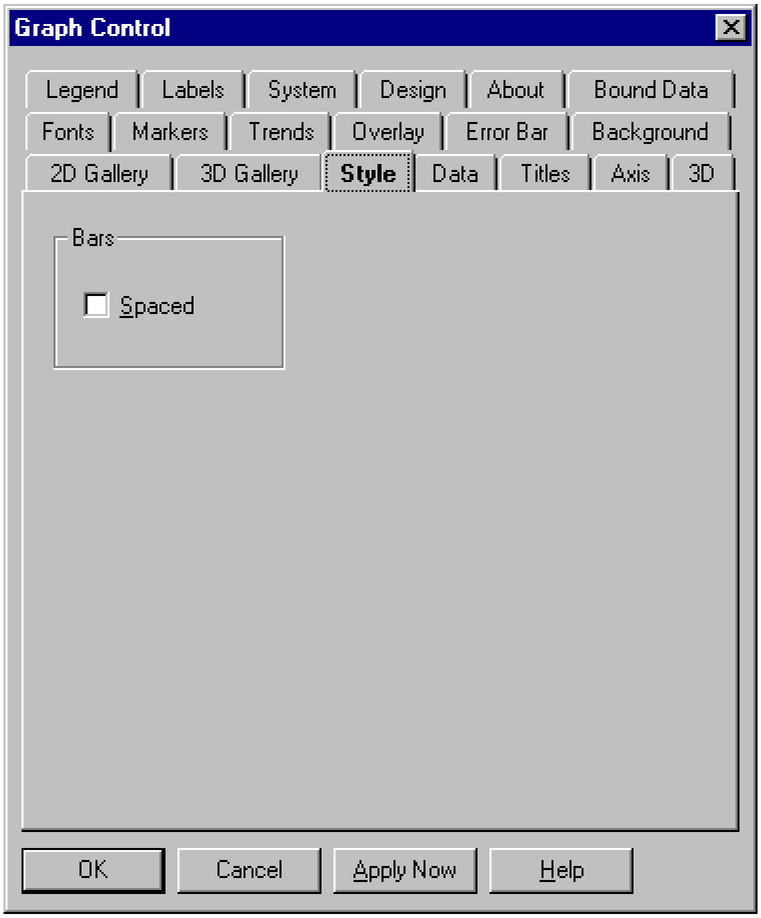
This is an additive property designed to be used in combination with other GraphStyle settings. Additive settings cannot be made through the property window. Set these combinations either in code or through the property pages.

Related Property

GraphStyle = 0 + 32 (Raw data with samples, black border)
GraphStyle = 1 + 32 (Raw data, no samples, black border)
GraphStyle = 2 + 32 (Parametric data, black border)

Style page (Gantt chart)

Click where you need help



Bars

☐ Spaced

▲▼

Topic

Style page (Gantt chart)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Spaced (default is off)

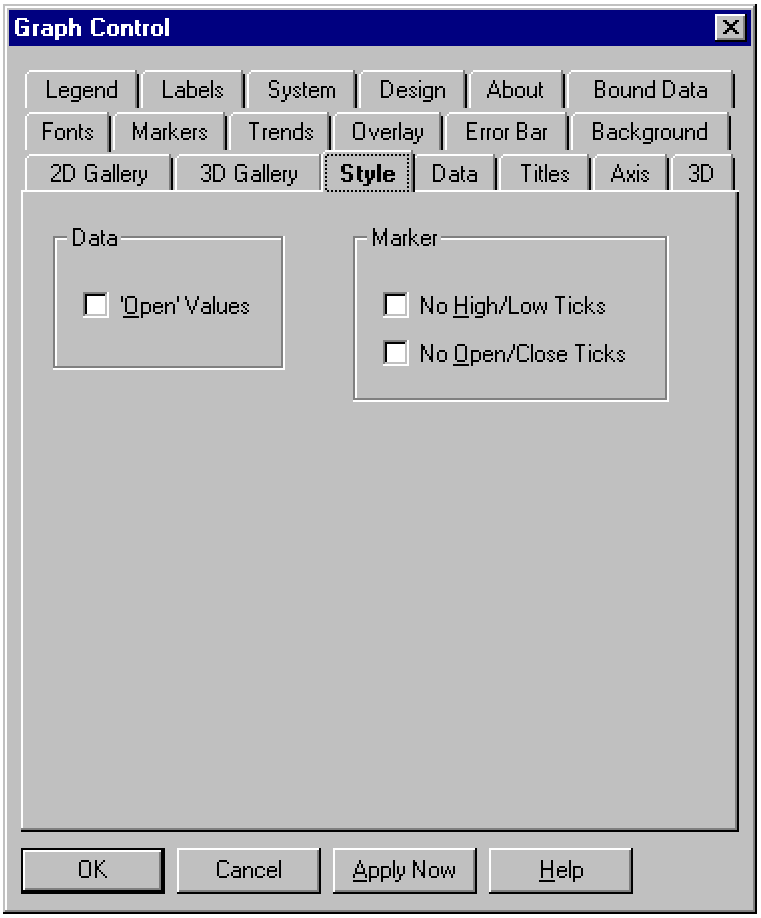
Select this option to place spaces between Gantt bars.   Clear the box to draw all bars adjacent to each other.

Related Property

GraphStyle = 0 (Adjacent)
GraphStyle = 1 (Spaced)

Style page (High-low-close graph)

Click where you need help



Data

□ 'Open' Values

Marker

□ No High/Low Ticks

□ No Open/Close Ticks

▲▼

Topic

Style page (HLC graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Open Values (default is off)

Select this option for an open-high-low-close graph. Clear the box for a high-low-close graph.

Note: A high-low-close graph has three data sets: high values, low values, and closing values. An open-high-low-close graph has a fourth data set for opening values.

Related Property

GraphType = 11 (HLC)	Box is clear.
GraphType = 18 (OHLC)	Box is checked.

No High/Low Ticks (default is off)

Select this option to disable high and low ticks.   Clear the box to draw ticks.

Related Property

GraphStyle = 2 (No high or low ticks)
GraphStyle = 3 (No open, close, high or low ticks)

No Open/Close Ticks (default is off)

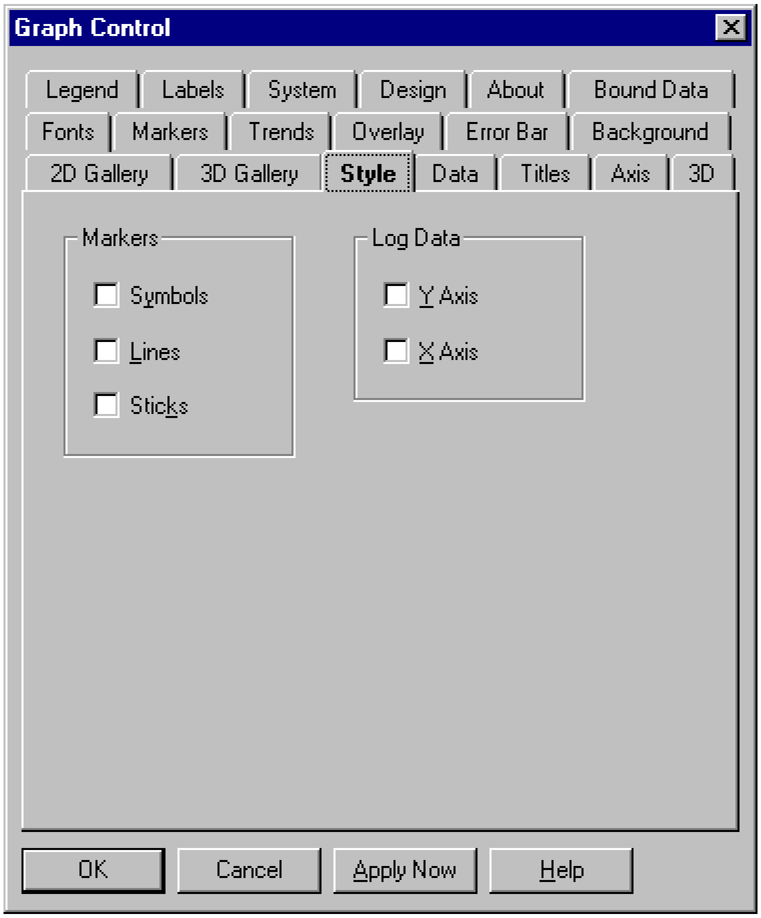
Select this option to disable open and close ticks.   Clear the box to draw ticks.

Related Property

GraphStyle = 1 (No open or close ticks)
GraphStyle = 3 (No open, close, high or low ticks)

Style page (Line graph--including log/lin, lin/log and log/log)

Click where you need help



Graph Control

Legend | Labels | System | Design | About | Bound Data

Fonts | Markers | Trends | Overlay | Error Bar | Background

2D Gallery | 3D Gallery | **Style** | Data | Titles | Axis | 3D

Markers

□ Symbols

□ Lines

□ Sticks

Log Data

□ Y Axis

□ X Axis

OK | Cancel | Apply Now | Help

▲▼

Topic

Style page (Line graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Symbols (default is off)

Select this option to draw a symbol at the position of each data point. Clear the box for no symbols.

Related Property

GraphStyle = 1 (Symbols)
GraphStyle = 3 (Sticks and symbols)
GraphStyle = 7 (Lines, sticks, and symbols)

Lines (default is off)

Select this option to draw lines between data points. Clear the box for no lines.

Note that if *all* options in the Markers group (Symbols, Lines, and Sticks) are deselected, lines are drawn in any case.

Related Property

GraphStyle = 4 (Lines)
GraphStyle = 0 (Lines only)
GraphStyle = 7 (Lines, sticks, and symbols)

Sticks (default is off)

Select this option to draw a vertical line between each data point and the X axis. Clear the box for no sticks.

Related Property

GraphStyle = 2 (Sticks)
GraphStyle = 3 (Sticks and symbols)
GraphStyle = 7 (Lines, sticks, and symbols)

Y Axis (default is off)

Select this option for a logarithmic Y axis. If you select Y Axis and leave X Axis off, you'll get a log/lin graph; if you select both, you'll get a log/log graph.

Related Property

GraphType = 7 (Log/Lin)
GraphType = 15 (Log/Log)

X Axis (default is off)

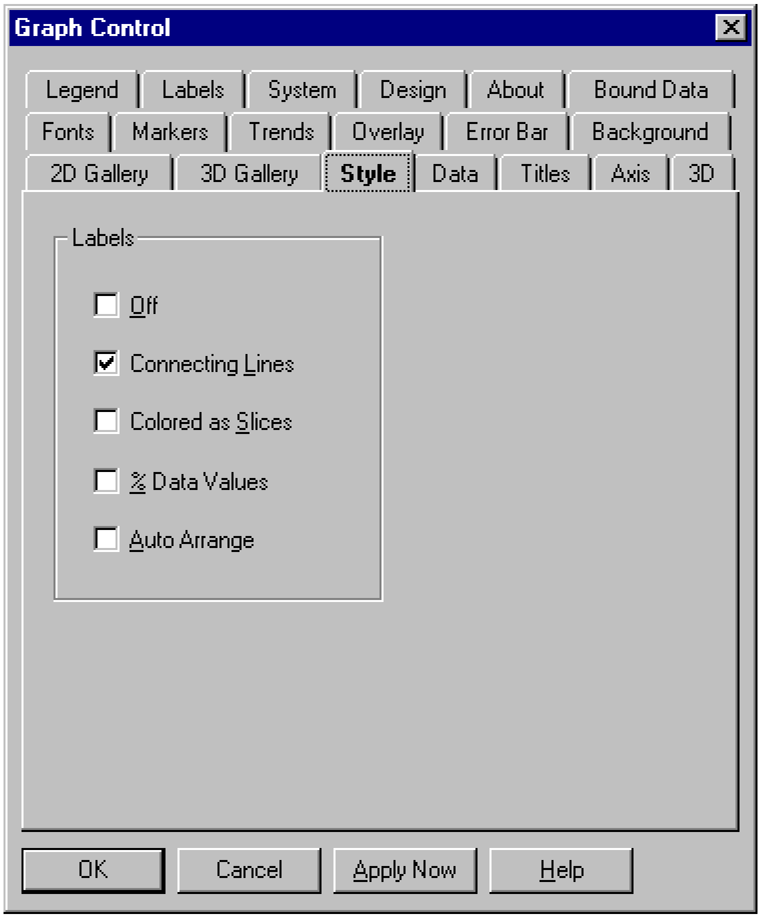
Select this option for a logarithmic X axis.   If you select X Axis and leave Y Axis off, you'll get a lin/log graph; if you select both, you'll get a log/log graph.

Related Property

GraphType = 16 (Lin/Log)
GraphType = 15 (Log/Log)

Style page (Pie chart)

Click where you need help



Labels

□ Off

☑ Connecting Lines

□ Colored as Slices

□ % Data Values

□ Auto Arrange

▲▼

Topic

Style page (Pie chart)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Off

Select this option to draw a pie chart with no labels for the slices.   Clear the box to draw labels.

Related Property

GraphStyle = 0 (No labels)   Box is checked.

Connecting Lines (default is on)

Select this option to have connecting lines drawn between pie slices and their labels. Clear the box for no connecting lines.

Related Property

GraphStyle = 0 (Lines) Box is checked.
GraphStyle = 1 (No lines) Box is clear.

Colored as Slices (default is off)

Select this option to have pie labels drawn in the same colors as their corresponding slices. Clear the box to have every label drawn in a single color (black by default, or as specified on the Fonts property page).

Related Property

GraphStyle = 2 (Colored labels)
GraphStyle = 3 (Colored labels, no lines)
GraphStyle = 6 (Colored percentage labels)
GraphStyle = 7 (Colored percentage labels, no lines)

% Data Values (default is off)

Select this option to have pie labels show the percentage values of their slices (as related to the whole). Clear the box to have labels show the actual values of their slices.

Related Property

GraphStyle = 4 (Percentage labels)
GraphStyle = 6 (Colored percentage labels)
GraphStyle = 7 (Colored percentage labels, no lines)

Auto Arrange (default is off)

Select this option to have the Graph control position labels so that they do not overlap. Some labels may be drawn farther from the pie than others.

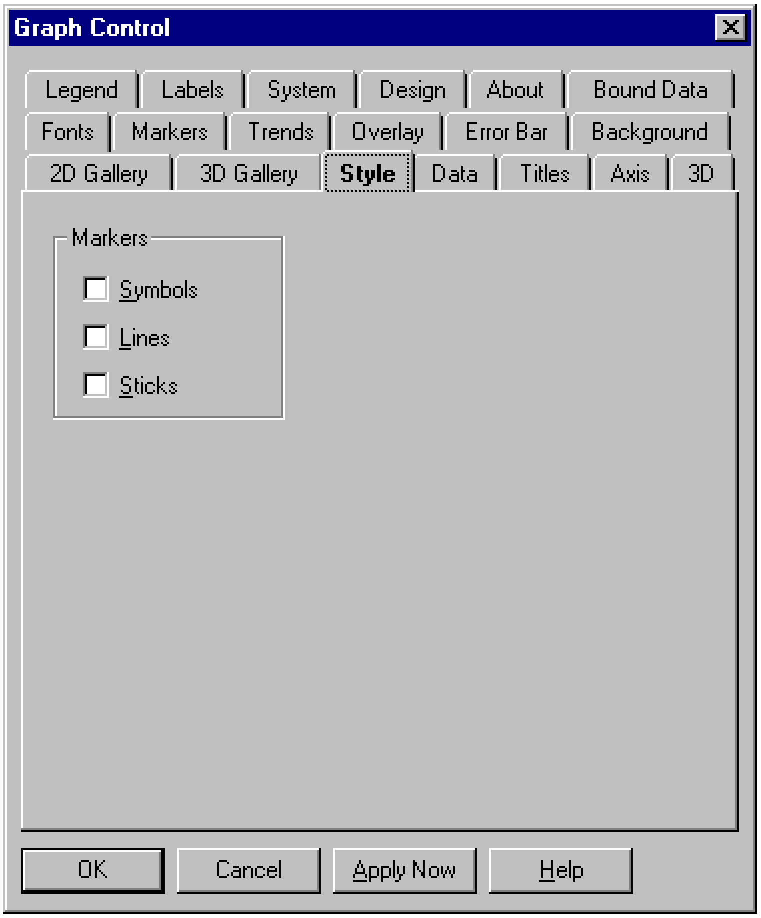
If your pie has many small segments, Auto Arrange can help prevent labels from being drawn on top of one another.

Related Property

SmartLabels = 1 (On)

Style page (Polar graph)

Click where you need help



Markers

□ Symbols

□ Lines

□ Sticks

▲▼

Topic

Style page (Polar graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Symbols (default is off)

Select this option to draw a symbol at the position of each data point.   Clear the box for no symbols.

Related Property

GraphStyle = 1 (Symbols)
GraphStyle = 3 (Sticks and symbols)
GraphStyle = 5 (Lines and symbols)
GraphStyle = 7 (Lines, sticks, and symbols)

Lines (default is off)

Select this option to draw lines between data points.   Clear the box for no lines.

Note that if *all* options in the Markers group (Symbols, Lines, and Sticks) are deselected, lines are drawn in any case.

Related Property

GraphStyle = 0 (Lines only)
GraphStyle = 4 (Lines)
GraphStyle = 5 (Lines and symbols)
GraphStyle = 6 (Lines and sticks)
GraphStyle = 7 (Lines, sticks, and symbols)

Sticks (default is off)

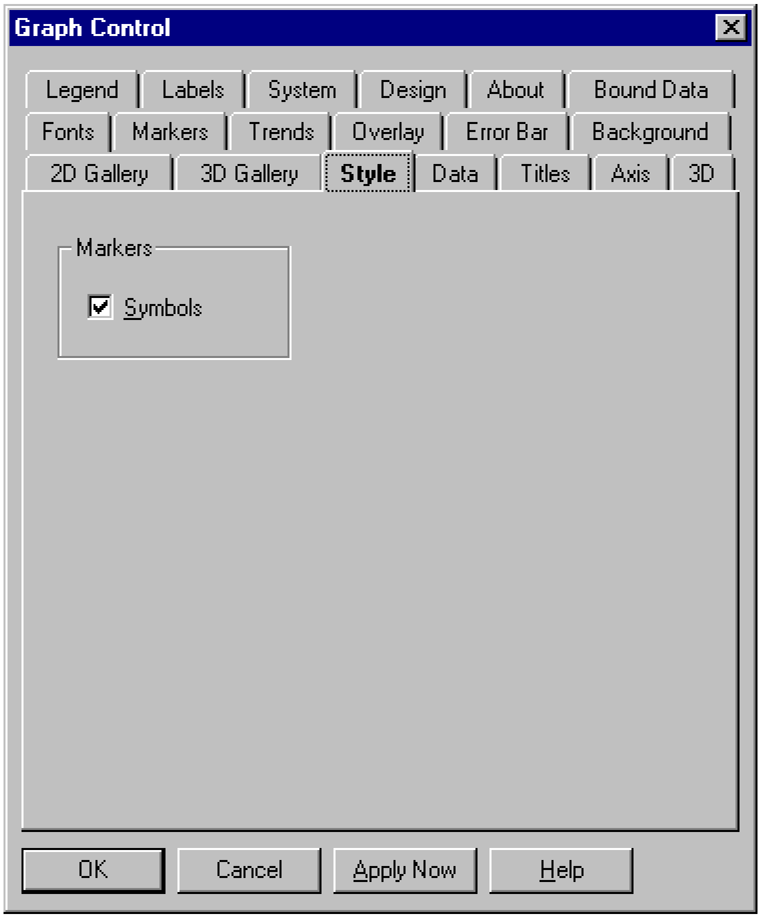
Select this option to draw a "stick" between each data point and the origin.   Clear the box for no sticks.

Related Property

GraphStyle = 2 (Sticks)
GraphStyle = 3 (Sticks and symbols)
GraphStyle = 6 (Lines and sticks)
GraphStyle = 7 (Lines, sticks, and symbols)

Style page (Scatter graph--2D)

Click where you need help



Markers

☑ Symbols

▲▼

Topic

Style page (Scatter graph--2D)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--3D)

Style page (Surface graph)

Style page (Time series graph)

Symbols (default is on)

This option matters only if you've enabled curve fitting for the graph in the Trends property page. In that case, select Symbols to draw symbols along with curves and deselect it to draw curves only. If you haven't enabled a curve, symbols are drawn in any case.

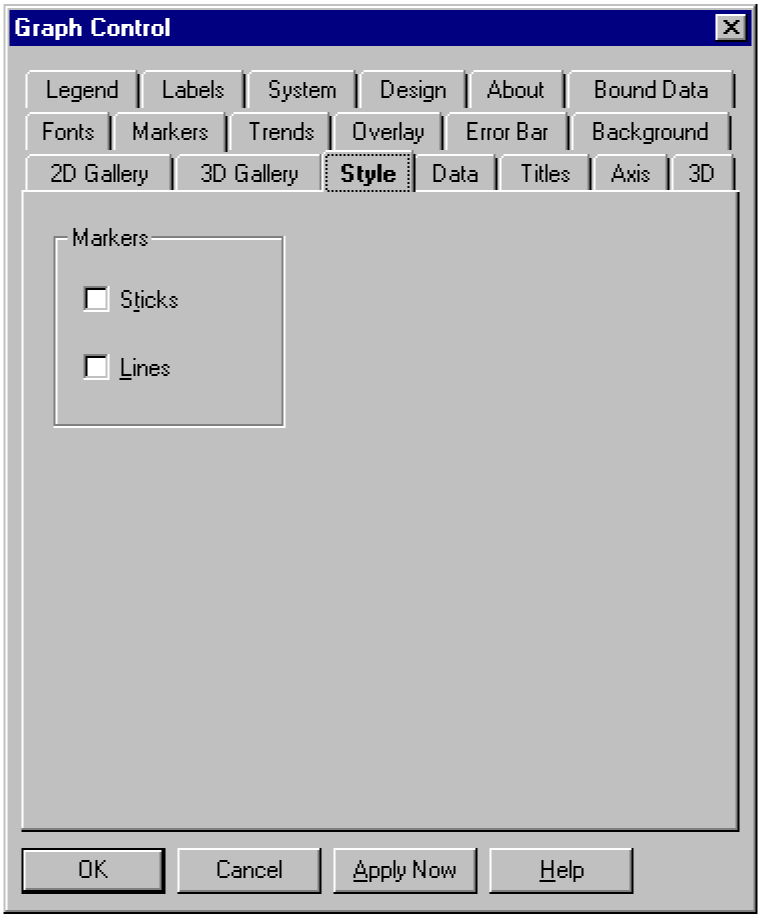
Related Property

GraphStyle = 0 (Symbols only)
GraphStyle = 1 (Curve)
GraphStyle = 2 (Symbols)
GraphStyle = 3 (Curve and symbols)

Note: To get a curve, you must set the LineStats property.

Style page (Scatter graph--3D)

Click where you need help



Markers

☐ Sticks

☐ Lines

▲▼

Topic

Style page (Scatter graph--3D)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Surface graph)

Style page (Time series graph)

Sticks (default is off)

Select this option to draw a vertical stick between each data point and the Y origin plane.   Clear the box for no sticks.

Related Property

GraphStyle = 0 (Symbols only)
GraphStyle = 1 (Symbols and sticks)
GraphStyle = 3 (Sticks, symbols, and lines)

Lines (default is off)

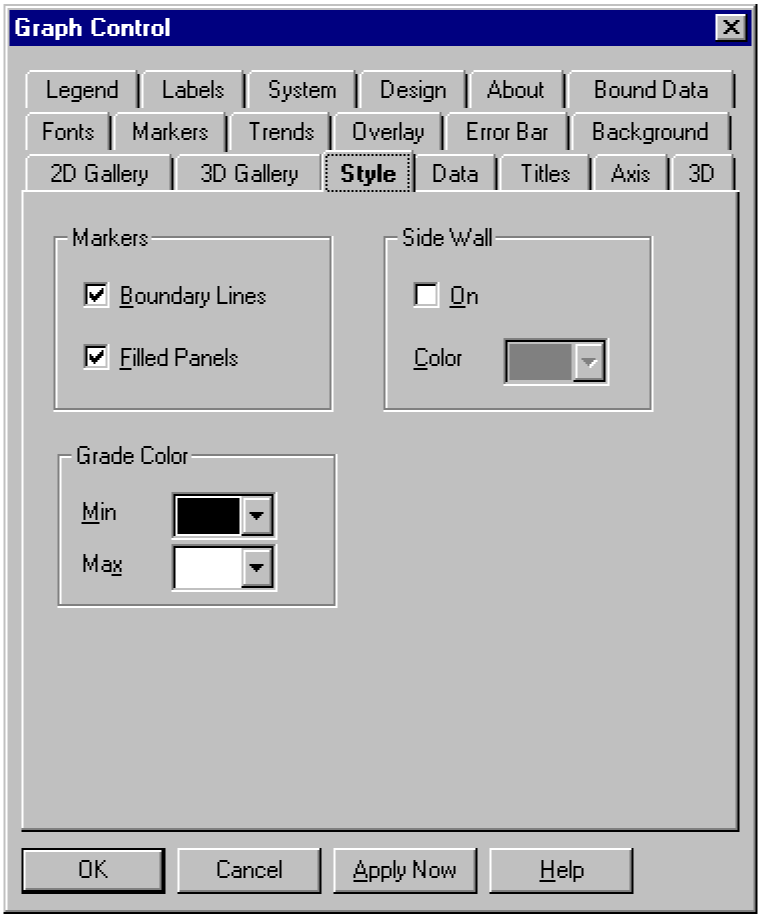
Select this option to draw connecting lines between points in a set.

Related Property

GraphStyle = 2 (Symbols and lines)
GraphStyle = 3 (Sticks, symbols, and lines)

Style page (Surface graph)

Click where you need help



Markers

☑ Boundary Lines

☑ Filled Panels

Side Wall

□ On

Color

Grade Color

Min

Max

▲▼

Topic

Style page (Surface graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Time series graph)

Filled Panels (default is on)

Select this option to fill in the cells of a surface graph grid with "panels" (actually paired triangles) of color.   Deselect it for no panels.

The color range of panels is determined by your settings in the Grade Color group.

Related Property

GraphStyle = 2 (Filled panels only)
GraphStyle = 0 (Filled panels and boundary lines)
GraphStyle = 3 (Filled panels, boundary lines, and sidewall)
GraphStyle = 5 (Filled panels and sidewall)

Boundary Lines (default is on)

This option lets you draw lines along the edges of each cell of the surface graph grid.

- **If Filled Panels is selected,** select Boundary Lines to draw lines (which are always black) and deselect it for no lines.
- **If Filled Panels is deselected**, lines are drawn in any case. Their color range is determined by your settings in the Grade Color group.

Related Property

GraphStyle = 1 (Boundary lines only)
GraphStyle = 0 (Boundary lines and filled panels)
GraphStyle = 3 (Boundary lines, filled panels, and sidewall)

Min

This box sets the color for the lowest point on the surface graph.

Colors vary according to the color palette youve selected.   To choose a palette, go to the Background property page.

Related Property

SurfaceColorMin = ColorIndex

Max

This box sets the color for the highest point on the surface graph.

Colors vary according to the color palette youve selected.   To choose a palette, go to the Background property page.

Related Property

SurfaceColorMax = ColorIndex

On (default is off)

Select this option to draw side walls around the perimeter of the graph in the X and Z planes. Deselect it for no side walls.

Related Property

GraphStyle = 3 (Filled panels, boundary lines, and sidewall)
GraphStyle = 5 (Filled panels and sidewall)

Color

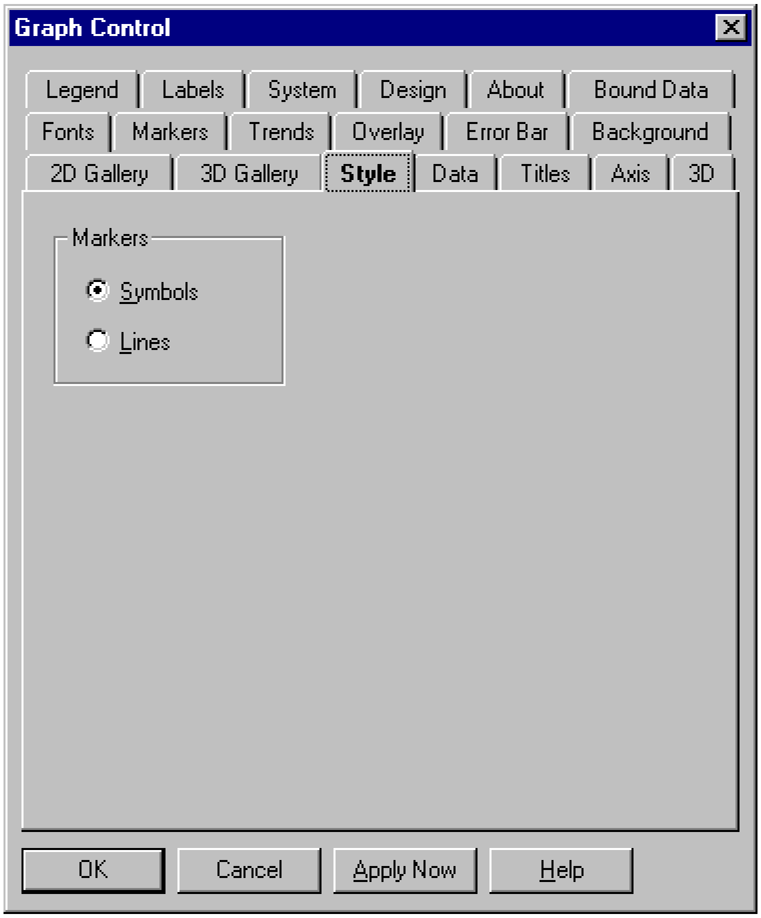
This box sets the color for the side walls, using the current color palette. To select a palette, go to the Background property page.

Related Property

SurfaceWallColor = ColorIndex

Style property page (Time series graph)

Click where you need help



Markers

● Symbols

○ Lines

▲▼

Topic

Style page (Time series graph)

Related

Style page (Area graph--2D)

Style page (Area graph--3D)

Style page (Bar graph--2D)

Style page (Bar graph--3D)

Style page (Box-whisker graph)

Style page (Gantt chart)

Style page (HLC graph)

Style page (Line graph)

Style page (Pie chart)

Style page (Polar graph)

Style page (Scatter graph--2D)

Style page (Scatter graph--3D)

Style page (Surface graph)

Symbols (default)

Select this option to draw a symbol at the position of each data point.

Related Property

GraphStyle = 0

Lines

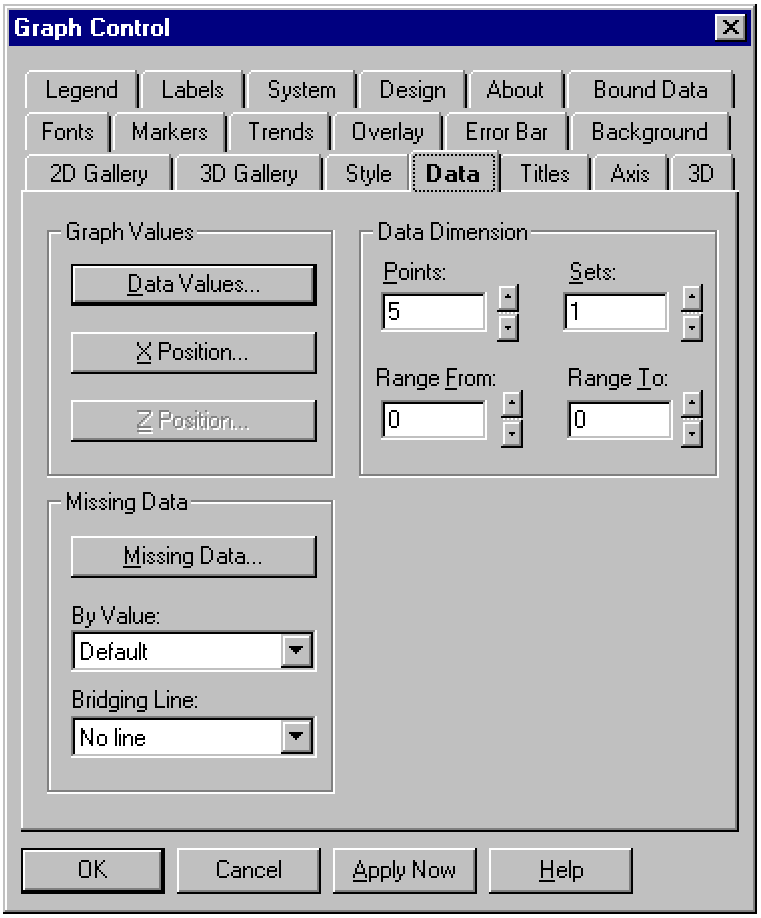
Select this option to connect data points with a continuous line. One line is drawn for each data set.

Related Property

GraphStyle = 1

Data property page

Click where you need help



Points: 5

Sets: 1

Range From: 0

Range To: 0

By Value: Default

Bridging Line: No line

Data Values (available for all graph types except polar and bubble)

In most cases, the Data Values grid holds the position of data points along the Y (vertical) axis. However, some graph types make different use of these values:

- In **pie charts,** each Data Values cell corresponds to one pie slice. There's only one row of cells because pie charts show a single data set.
- In **floating bar graphs,** the first row of Data Values graphs as the invisible bottom segment of each bar.
- In **Gantt charts,** each column of Data Values applies to a single bar. You need a minimum of two rows (sets) of cells: the first row contains the starting point for the first segment of each bar, the second row contains the end points of those first segments, and subsequent rows contain the end points of additional segments.
- In **high-low-close graphs,** each column of Data Values applies to a single symbol. You have three rows (sets) of cells: the first row contains the high values, the second the low values, and the third the closing values.
- In **open-high-low-close** and **candlestick graphs**, each column of Data Values applies to a single symbol. You have four rows (sets) of cells: the first row contains the opening values, the second the high values, the third the low values, and the fourth the closing values.
- In **box-whisker graphs,** each column of Data Values applies to a single symbol. If you're using parametric data, you need exactly seven rows of ascending values representing percentiles of 5, 10, 25, 50 (median), 75, 90, and 95; any additional rows are ignored. If you're using raw data, you need enough rows to hold all the values you want to process (never less than seven rows).
- In **surface graphs,** you need a minimum of two rows (sets) of cells: the first row contains the Y values for the points along the X axis at Z = 0 (the "back row" of points), and subsequent rows contain the Y values for additional rows of points along the X axis.

Related Property

Data()
GraphData

X Position (available for all graph types except pie, polar, bubble, Gantt, time series, and surface)

In most cases, X Position values are optional--you need them only if you want to set custom positions for data points along the X (horizontal) axis.   However, *lin/log and log/log graphs* always have X Position values, and *scatter graphs* usually do.

Most graphs have one X Position cell for each point, but high-low-close, open-high-low-close, candlestick, and box-whisker graphs have only one row of cells, each of which corresponds to a single symbol.

Related Property

XPosData

Z Position (available for 3D scatter graphs only)

In the Z Position dialog, you can enter points for the Z (depth) values of 3D scatter graphs.

Related Property

ZPosData

Missing Data (available for polar, 2D scatter, line, log/lin, lin/log, log/log, and tape graphs)

This dialog presents a grid in which you can mark certain data points as "missing."   Missing data points aren't displayed in the graph, whether or not you've set a value for the point using the Data Values grid.

The number of **rows** in the grid is equal to the number of **sets** in the graph. The number of **columns** is equal to the number of **points**. When you place the cursor in a particular cell, the lower right corner of the dialog displays a (set, point) indicator. If the cursor is in the second row, first column the indicator will display (2, 1), meaning (Set 2, Point 1).

To specify a data point as missing, enter a -1 in the appropriate cell.   To display a data point, leave its cell blank or enter 0.

To miss a particular point in all sets, you need only enter -1 in that point's column of the first row. To miss a point in one set but display it in all other sets, enter -1 in the cell for the (set, point) you want to miss, and then enter 0 in the remaining cells of the same column.

Related Property

ExtraData

By Value (line, tape, 2D scatter, and 3D scatter)

Rather than mark particular data *points* as missing, you can use this list to screen out certain data *values*. The settings in this list can be used in combination with the Missing Data grid.

Default. Do not screen out any values.

Zero data. Screen out zero values.

The remaining options screen out values in relation to limit lines. (See the Trends property page.) It is not necessary to display limit lines; however, you must have entered High and/or Low limit values for these options to work.

Below low limit. Screen out values less than the Low value.

Above high limit. Screen out values greater than the High value.

Outside limits. Screen out values less than the Low or greater than the High value.

Inside limits. Screen out values greater than the Low but less than the High value.

Related Property

MissingData

Bridging Line (default is no line)

Select the type of line used to bridge gaps caused by missing or excluded values. Enabled only for line graphs (including log variants).

Related Property

MissingLineMode

Points

Determines the number of columns in the Data Values grid and the number of points in each data set graphed. You normally need at least two points to create a meaningful graph.

Related Property

NumPoints

Sets

Determines the number of rows in the Data Values grid and the number of data sets graphed.

- In **pie charts,** Sets is always 1 because you can graph only one data set.
- In **bubble graphs,** Sets is always 2 because the X and Y Position dialog requires two rows of cells. Despite this setting, the Bubble Size dialog contains only one row of data.
- In **Gantt charts** and **surface graphs,** you need at least two rows of cells in the Data Values dialog. The default Sets value is 2, but you can enter a higher value.
- In **high-low-close graphs,** Sets is always 3 because you need three rows of cells in the Data Values dialog--high values, low values, and closing values.
- In **open-high-low-close** and **candlestick graphs,** Sets is always 4 because you need four rows of cells in the Data Values dialog--opening values, high values, low values, and closing values.
- In **box-whisker graphs,** you need enough rows in the Data Values dialog to include all of the values you want to graph.   The minimum (and default) Sets value is 7 because you need exactly seven rows for parametric data.   However, you can enter a higher Sets value if you're using raw data and have more than seven values making up your sample groups.

Related Property

NumSets

Range From

To graph only a portion of the data loaded, enter the number of the first point in the range.

For example, if you have data for 12 months but want to zoom in on just months 4 through 6, enter 4 in Range From.

Note: When both Range From and Range To are set to zero, all points are graphed.

Related Property

RangeMin

Range To

To graph only a subset of the data loaded, enter the last point in the range.

For example, if you have data for 12 months but want to zoom in on just months 4 through 6, enter 6 in Range To.

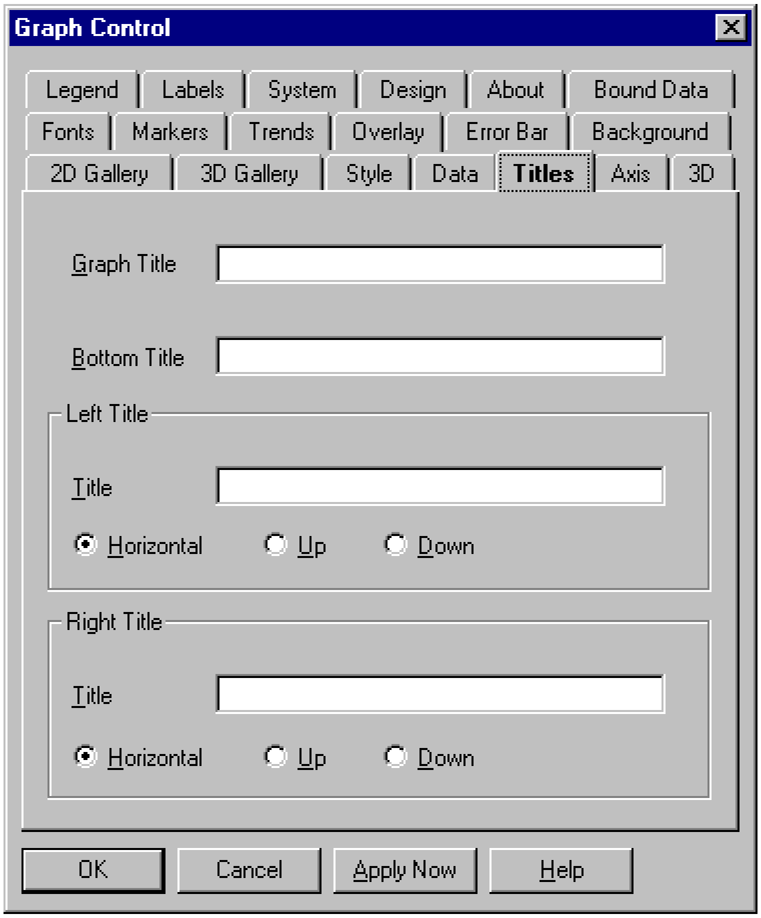
Note: When both Range From and Range To are set to zero, all points are graphed.

Related Property

RangeMax

Titles property page

Click where you need help



Graph Title

Bottom Title

Left Title

Title

● Horizontal ○ Up ○ Down

Right Title

Title

● Horizontal ○ Up ○ Down

▲▼

Topic

Titles page

Related

Background page

Fonts page

Graph Title

In this box, you can enter text for the graph title, which can be up to 80 characters in length. This title appears centered at the top of the graphing window.

Related Property

GraphTitle

Title text

When you enter text for a title, Graphics Server adjusts the rest of the graphing window to provide space--either redrawing the graph and associated objects at a smaller size or decreasing the space between objects.   When you clear the text box for a title, you disable it and provide more space for the rest of the graph.

If you enter a title that's too long to appear in a single line, Graphics Server automatically word-wraps it.   If a title doesn't display at all, it's because Graphics Server can't make the font small enough to fit all the text in the space provided.   Increase the size of the graphing window to make the title appear.

To select a font for the title text, go to the Fonts property page. To set the text color, go to the Background property page.

Related Property

NewLineChar

Bottom Title

In this box, you can enter text for the bottom title, which can be up to 80 characters in length. This title appears centered at the bottom of the graphing window. A bottom title frequently explains the X axis.

Related Property

BottomTitle

Left Title

This box lets you enter text for the left title, which can be up to 80 characters in length.

Related Property

LeftTitle

Horizontal (default)

Select this option if you want the title to print horizontally.

Related Property

LeftTitleStyle = 0

Up

Select this option if you want the title to print vertically, running in an upward direction.

Related Property

LeftTitleStyle = 1

Down

Select this option if you want the title to print vertically, running in a downward direction.

Related Property

LeftTitleStyle = 2

Right Title

This box lets you enter text for the right title, which can be up to 80 characters in length.   The right title frequently explains the right-hand Y axis when you have an overlay graph.

Related Property

RightTitle

Horizontal (default)

Select this option if you want the title to print horizontally.

Related Property

RightTitleStyle = 0

Up

Select this option if you want the title to print vertically, running in an upward direction.

Related Property

RightTitleStyle = 1

Down

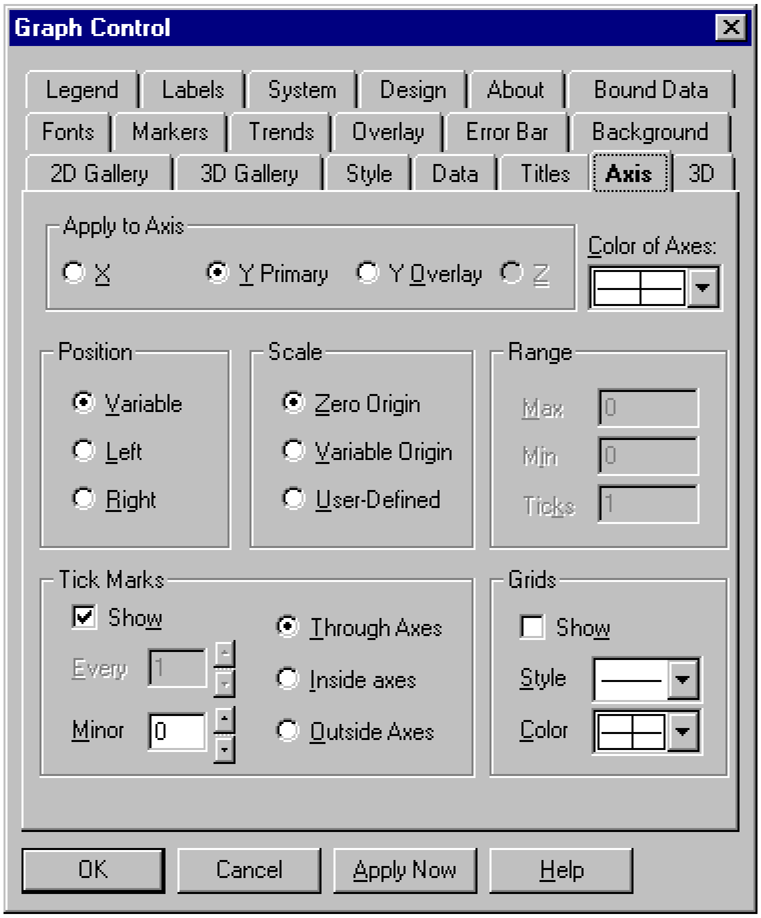
Select this option if you want the title to print vertically, running in a downward direction.

Related Property

RightTitleStyle = 2

Axis property page (Graph types with X-Y or X-Y-Z grids)

Click where you need help



Apply to Axis: ○ X ● Y Primary ○ Y Overlay ○ Z

Color of Axes:

Position: ● Variable ○ Left ○ Right

Scale: ● Zero Origin ○ Variable Origin ○ User-Defined

Range: Max 0 Min 0 Ticks 1

Tick Marks: ☑ Show Every 1 Minor 0 ● Through Axes ○ Inside axes ○ Outside Axes

Grids: □ Show Style Color

▲▼

Topic

Axis page (Graph types with X-Y grids)

Related

Axis page (Polar graphs)

Overlay page

X

Select this option to view and/or change the settings for the X axis.

Y Primary

Select this option to view and/or change the settings for the primary Y axis (which is the *only* Y axis when you don't have an overlay graph).

Related Property

YAxisUse = 0 (Primary Y axis)

Y Overlay

Select this option to view and/or change the settings for the Y axis of an overlay graph, which is always drawn to the right of a graph.   To enable an overlay graph, go to the Overlay property page.

Related Property

YAxisUse = 1 (Right-hand Y axis)

Z

Select this option to view and/or change the settings for the Z axis.

Color of Axes (default is automatic black or white)

In this list box, you can choose a color for axes from the current color palette. The same color is applied to all axes. The default color is automatic black or white, whichever provides more contrast.

To choose a color palette, go to the Background property page.

Related Property

ForegroundUse = 6
Foreground = ColorIndex

Variable (default)

- When X is selected in the Apply to Axis group, select this option to draw the X axis intersecting the Y origin, whether that's at the top, bottom, or middle of the graph.
- When Y Primary is selected in the Apply to Axis group, select this option to draw the primary Y axis intersecting the X origin, whether that's at the left, right, or middle of the graph.

Related Property

XAxisPos = 0 (Variable)
YAxisPos = 0 (Variable)

Left or Top

- When X is selected in the Apply to Axis group, select this option to draw the X axis at the top of the graph, regardless of the location of the Y origin.
- When Y Primary is selected in the Apply to Axis group, select this option to draw the primary Y axis at the left edge of the graph, regardless of the location of the X origin.

Related Property

XAxisPos = 1 (Top)
YAxisPos = 1 (Left)

Right or Bottom

- When X is selected in the Apply to Axis group, select this option to draw the X axis at the bottom of the graph, regardless of the location of the Y origin.
- When Y Primary is selected in the Apply to Axis group, select this option to draw the primary Y axis at the right edge of the graph, regardless of the location of the X origin.

Related Property

XAxisPos = 2 (Bottom)
YAxisPos = 2 (Right)

Zero Origin (default for all X-Y-Z graphs except time series)

Select this option to draw the axis including the origin (zero) and extending far enough in the positive and negative directions to include all of the graph's values.

Related Property

XAxisStyle = 0 (Zero origin)
YAxisUse
YAxisStyle = 0 (Zero origin)
ZAxisStyle = 0 (Zero origin)

Variable Origin

Select this option to have the axis "zoom in" on the range of the graph's actual values, whether or not that range includes zero.

For X and Z axes, this option differs from Auto only if you specify X or Z values for data points.

Related Property

XAxisStyle = 1 (Variable origin)
YAxisUse
YAxisStyle = 1 (Variable origin)
ZAxisStyle = 1 (Variable origin)

User-Defined (default for time series graphs)

Select this option to set your own values for the minimum, maximum, and number of ticks for the axis.   When you select User-Defined, the settings in the Range group are enabled.

Related Property

XAxisStyle = 2 (User-defined origin)
YAxisUse
YAxisStyle = 2 (User-defined origin)
ZAxisStyle = 2 (User-defined origin)

Max (enabled by selecting User-Defined in the Scale group)

The Max setting is generally the maximum point of the axis, with the following exceptions:

- If you have a positive Max with a negative Min, the Graph control may adjust the maximum or minimum point so that it lies on a tick.   See Ticks.
- In 3D graphs, if you specify a Max that's too low to show all your data values, the Graph control moves the maximum point to show all the values.   The Ticks setting still applies.
- For X and Z axes without specific X or Z values, the Max setting is ignored.   The maximum point of the axis is equal to the number of points or sets you're graphing.

Related Property

XAxisMax
YAxisUse
YAxisMax
ZAxisMax

Min (enabled by selecting User-Defined in the Scale group)

The Min setting is generally the minimum point of the axis, with the following exceptions:

- If you have a positive Max with a negative Min, the Graph control may adjust the maximum or minimum point so that it lies on a tick.   See Ticks.
- In 3D graphs, if you specify a Min that's too high to show all your data values, the Graph control moves the minimum point to show all the values.   The Ticks setting still applies.
- For X and Z axes without specific X or Z values, the Min setting is ignored.   The minimum point of the axis is always 0.

Related Property

XAxisMin
YAxisUse
YAxisMin
ZAxisMin

Ticks (enabled by selecting User-Defined in the Scale group)

The Ticks setting determines the number of ticks along the selected axis. (Note that ticks are distinct from tick marks--see the Tick Marks group.) The effect depends on the axis and the nature of your data:

- For **X axes without specific X values**, the Ticks setting must be greater than the Points setting--available in the Data property page--to have any effect. In that case, the X axis is extended to the Ticks value.
- For **Z axes without specific Z values** (3D bar, tape, area, and surface graphs), you can't specify Ticks. The option is disabled.
- For **Y axes, X axes with specific X values**, and **Z axes with specific Z values**, the Ticks setting specifies the number of ticks from the origin to the setting of either Max or Min, whichever has the higher magnitude (distance from 0). For example, if you set Min to -50 and Max to 20, Ticks applies to the axis segment between 0 and -50.
- To determine the Ticks value you want to set, divide the length of the axis (or axis segment) by the desired interval between ticks. For the axis segment 0 to -50 we've just described (whose length is 50 units), if you want to place ticks 25 units apart, set Ticks to 2.

  Both the maximum and minimum points of an axis must fall on a tick. If you have a negative Min with a positive Max, the Graph control may have to move the minimum or maximum point to make this happen. In our example axis, ticks would be placed at -50, -25, 0, and 25--overriding the Max setting of 20.

Related Property

XAxisTicks
YAxisUse
YAxisTicks
ZAxisTicks

Show (available for X and Y Primary axes only; default is on)

Select this option to draw tick marks along the axis. Deselect it for no tick marks.

If you have a Y Overlay axis, your Show setting for Y Primary also applies to Y Overlay. Either both Y axes have tick marks or both don't.

Related Property

Ticks

Minor (available for X, Y Primary, and Y Overlay axes; default is 0)

Specify the number of minor tick marks to be drawn between pairs of major ticks.

Related Property

XAxisMinorTicks
YAxisUse
YAxisMinorTicks

Every (available for X axis with zero or variable origin only; default is 1)

If you select Zero Origin or Variable Origin in the Scale group, you can use the Every setting to specify the frequency with which tick marks are displayed along the X axis. An Every setting of 1 places a mark at each tick, a setting of 2 places a mark at every other tick, and so on.

The X axis must end with a tick mark. If you set an Every value that doesn't include the last value on the axis, Graphics Server will extend the axis so that it ends on a tick mark.

This setting doesn't apply when you've set independent positional values using the X Position dialog (for horizontal bar graphs, the Y Position dialog) in the Data property page.

Related Property

TickEvery

Through Axes

Select this option if you want tick marks centered on the axis line.

Note: Your selection will apply to all axes for which tick marks have been enabled.

Related Property

TickStyle

Inside Axes

Select this option if you want tick marks to be drawn only on the graph side of the axes.

Note: Your selection will apply to all axes for which tick marks have been enabled.

Related Property

TickStyle

Outside Axes

Select this option if you want tick marks drawn outside the axes.

Note: Your selection will apply to all axes for which tick marks have been enabled.

Related Property

TickStyle

Show (available for X and Y Primary axes only; default is off)

Select this option to draw grid lines perpendicular to the axis, intersecting each tick mark.   Deselect it for no grid lines.

Related Property

GridStyle

Line (default is solid)

In this list box, you can choose a style for grid lines.   The same style is applied to both X and Y grids.

Related Property

GridLineStyle

Color (default is automatic black or white)

In this list box, you can choose a color for grids from the current palette. The same color is applied to both X and Y grids. The default color is automatic black or white, whichever provides more contrast.

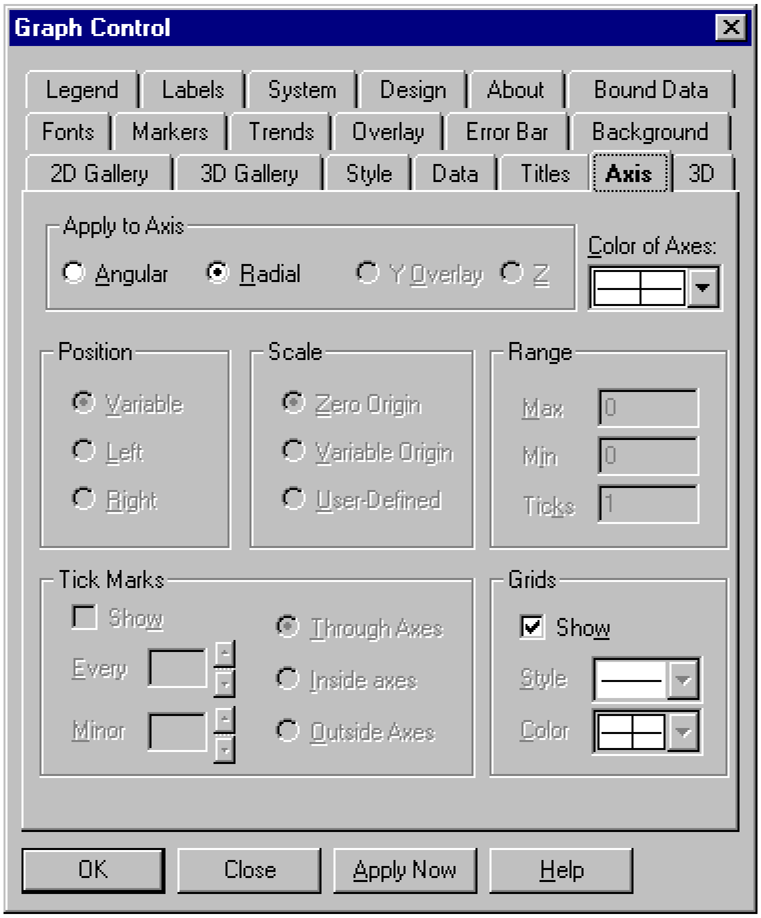
To choose a color palette, go to the Background property page.

Related Property

ForegroundUse = 7
Foreground = ColorIndex

Axis property page (Polar graphs)

Click where you need help



Apply to Axis

○ Angular ● Radial ○ Y Overlay ○ Z

Color of Axes:

Position

● Variable

○ Left

○ Right

Scale

● Zero Origin

○ Variable Origin

○ User-Defined

Range

Max 0

Min 0

Ticks 1

Tick Marks

□ Show

Every

Minor

● Through Axes

○ Inside axes

○ Outside Axes

Grids

☑ Show

Style

Color

▲▼

Topic

Axis page (Polar graphs)

Related

Axis page (Graph types with X-Y grids)

Angular (default)

Select this option to view and/or change the angular axis, which is marked by a series of radial lines ("spokes") at angular increments.   The origin is three o'clock.

Radial

Select this option to view and/or change the settings for the radial axis, which is marked by a series of concentric circles.

Color of Axes (default is automatic black or white)

In this list box, you can choose a color for axes from the current color palette.   The same color is applied to both the angular and radial axes.   The default color is automatic black or white, whichever provides more contrast.

To choose a color palette, go to the Background property page.

Related Property

ForegroundUse = 6
Foreground = ColorIndex

Show (default is off)

Select this option to enable grid lines for the axes.   Deselect it for no grid lines.

- For **angular axes**, the Graph control draws grid lines between the origin and outer wall of the polar graph at angles of 45, 135, 225, and 315 degrees.
- For **radial axes**, the Graph control draws four concentric circles at regular increments between the origin and the outer wall of the polar graph.

Related Property

GridStyle

Line (default is solid)

In this list box, you can choose a style for grid lines.   The same style is applied to both angular and radial grids.

Related Property

GridLineStyle

Color (default is automatic black or white)

In this list box, you can choose a color for grids from the current palette. The same color is applied to both angular and radial grids.   The default color is automatic black or white, whichever provides more contrast.

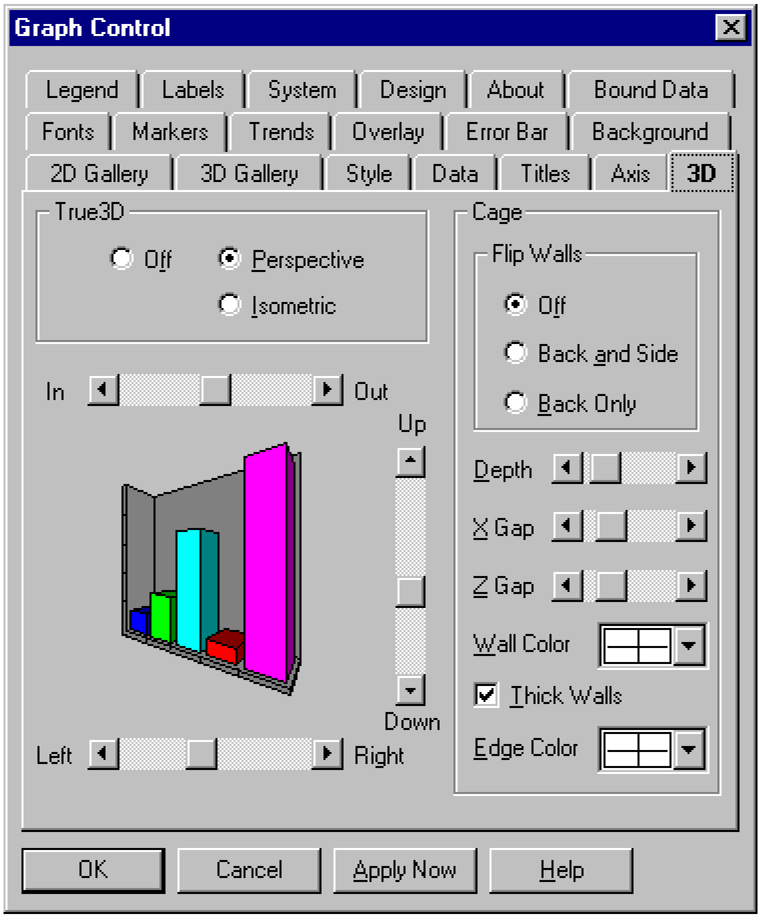
To choose a color palette, go to the Background property page.

Related Property

ForegroundUse = 7
Foreground = ColorIndex

3D property page

Click where you need help



True3D: ○ Off ● Perspective ○ Isometric

Flip Walls: ● Off ○ Back and Side ○ Back Only

☑ Thick Walls

▲▼

Topic

3D page

Related

Axis page

Off (available only for 3D bar, tape, and area graphs)

Select this option for standard 3D.   This option is included for compatibility with earlier editions of the Graph control.   Standard 3D doesn't support such True3D options as rotating cages and Z axis labels.   All you can do is set the color of the walls and edges of the graph's "cage."

Related Property

True3D = 0 (Off)

Perspective (default)

Select this option for True3D with perspective projection.   You can use the In-Out scroll bar to control the perspective at which graphs are drawn.

Related Property

True3D = 1 (On, perspective projection)

Isometric

Select this option for True3D with isometric projection.   In this mode, angles are rendered as if the graph is viewed from an infinite distance (so all parallels are preserved in the drawing), and the In-Out scroll bar is disabled.

Related Property

True3D = 2 (On, isometric projection)

In-Out

The In-Out scroll bar controls the degree of perspective foreshortening--or the perceived "distance" from which the graph is viewed--for a True3D graph with perspective projection.   This setting determines the acuteness of the angles in the graph.

The default In-Out bar setting places the viewing distance at about twice the width of the graph, which is defined as 50 units.   Each click "in" decreases the viewing distance by one unit (to a minimum of 0 units, a distance about equal to the graph's width), and each click "out" increases the viewing distance by one unit (to a maximum of 100, a distance about four times the graph's width).

The In-Out scroll bar is disabled for True3D graphs with isometric projection.   In those graphs, angles are drawn as if the viewing distance were infinite, so all parallels are preserved.

Related Property

Perspective

Up-Down (default is halfway up the graph)

The Up-Down bar sets the vertical viewing angle for a True3D graph.

The default Up-Down bar setting places the viewing angle halfway up the graph, which is defined as 0 degrees.   Each click up or down shifts the angle by one degree, to a maximum of 90 degrees (directly above the graph) and a minimum of -60 degrees (somewhat below the "floor" of the graph cage).

Related Property

Elevation

Left-Right (default is facing the front of the graph)

The Left-Right bar sets the horizontal viewing angle for a True3D graph.

The default Left-Right bar setting places the viewing angle directly facing the front of the graph. Each click to the left or right shifts the angle by one degree, to a maximum of 180 degrees (to the right) and a minimum of -180 (to the left).   The values of 180 and -180 provide the same view--directly facing the rear of the graph.

Related Property

Rotation

Cage flips

If you select a viewing angle that would normally cause part of the graph to be obscured, the back or side wall "flips" to the opposite edge of the cage, letting you see the whole graph.

Off (default)

Select this option to disable cage flips.

Related Property

CageFlip = 0 (Off)

Back and Side

Select this option to enable cage flips for both the back wall and side wall.

Related Property

CageFlip = 1 (On, flip back and side walls)

Back Only

Select this option to enable cage flips for the back wall only.   This option makes it easier for the viewer to stay oriented to the graph, because the side wall always represents the left edge of the cage.

Related Property

CageFlip = 2 (On, flip back walls only)

Depth

This scroll bar lets you vary the projected depth of True3D graphs.

The default Depth setting provides equal increments for units in the X and Z directions--a graph with an equal number of points and sets would be of equal width and depth.   The numeric value for this default is 100.   Each click to the left decreases the value by 5 (to a minimum of 10) and each click to the right increases it by 5 (to a maximum of 1000).

Related Property

True3DDepth

X Gap

This scroll bar lets you set the gap between the bars of a True3D bar graph.   This gap is in the X direction for vertical bar graphs and in the Y direction for horizontal bar graphs.

The default X Gap setting is 20 percent of the entire possible width of each bar--the remaining 80 percent is occupied by the bar itself.   Each click to the left decreases the gap percentage by 1 (to a minimum of 0) and each click to the right increases it by 1 (to a maximum of 95).   At the minimum X Gap setting of 0, there's no gap between bars.

Related Property

True3DXGap

Z Gap

This scroll bar lets you set the gap in the Z (depth) direction between data sets in three kinds of True3D graphs with multiple data sets: bar (z-clustered style), area (absolute style), and tape.

The default Z Gap setting is 20 percent of the entire possible width of each bar (or area plot or tape)--the remaining 80 percent is occupied by the bar itself.   Each click to the left decreases the gap percentage by 1 (to a minimum of 0) and each click to the right increases it by 1 (to a maximum of 95).   At the minimum Z Gap setting of 0, there's no gap between bars.

Related Property

True3DZGap

Wall Color (default is automatic color)

In this list box, you can choose a color for the walls and floor of a True3D graph cage from the current palette.   The same color is applied to the back wall, side wall, and floor of the cage.

To choose a color palette, go to the Background property page.

Related Property

CageWallColor

Thick Walls (default is on)

Select this option if you want the walls and floor of the True3D cage to appear "thick," with edges. Deselect it for "thin" walls, which have no discernible edges.

Related Property

CageStyle

Edge Color (default is automatic color)

In this list box, you can choose a color for the edges of a True3D graph cage from the current palette. This setting is disabled if you deselect the Thick Walls option.

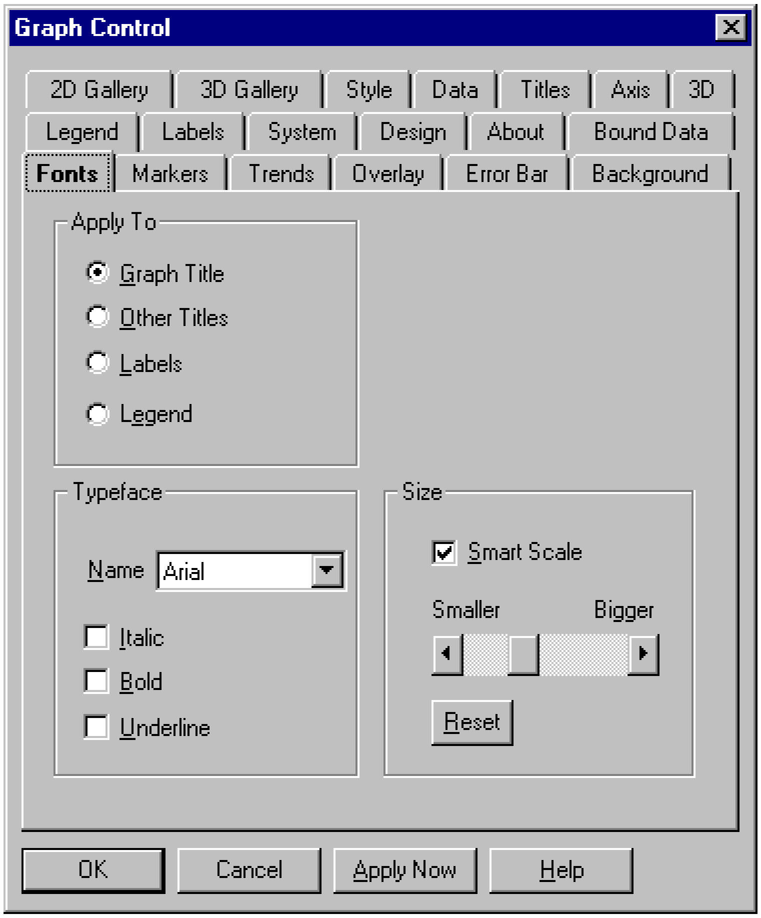
To choose a color palette, go to the Background property page.

Related Property

CageEdgeColor

Fonts property page

Click where you need help



Apply To

● Graph Title

○ Other Titles

○ Labels

○ Legend

Typeface

Name Arial

□ Italic

□ Bold

□ Underline

Size

☑ Smart Scale

▲▼

Topic

Fonts page

Related

Axis page

Background page

Data page

Titles page

Graph Title (default)

Select this option to apply font settings to the graph's title, which always appears centered at the top of the graphing window.

To set text color, go to the Background property page.

Related Property

FontUse = 0 (Graph title)

Other Titles

Select this option to apply font settings to the graph's left, right, and bottom titles.   The same settings apply to all three of these titles.

To set text color, go to the Background property page.

Related Property

FontUse = 1 (Other titles)

Labels

Select this option to apply font settings to the graph's labels--including axis labels (for graphs with X-Y-Z grids and polar graphs), pie chart labels, and data labels.   The same settings apply to all labels in use in the graph.

Axis labels are drawn in the same color as the axis to which they apply. To set the axis color, go to the Axis property page.

To select a color for data labels, go to the Data property page.

Related Property

FontUse = 2 (Labels)

Legend

Select this option to apply font settings to the graph's legend.

To set text color, go to the Background property page.

Related Property

FontUse = 3 (Legend)

Name (default is Arial)

In this list box, you can choose any installed Windows font for the selected text.

Related Property

FontName
FName()

Italic (default is off)

Select this option to have the Graph control italicize the text.

Related Property

FontStyle = 1 (Italic)

Bold (default is off)

Select this option to have the Graph control display the text in boldface.

Related Property

FontStyle = 2 (Bold)

Underline (default is off)

Select this option to have the Graph control underline the text.

Related Property

FontStyle = 4 (Underline)

Smart Scale (default is on)

Select this option to have the Graph control automatically use smaller type if the size you specify (using the Smaller-Bigger scroll bar) makes the text too large for the available space. If the Graph control can't make the type small enough to fit, the text won't display at all.

If you deselect Smart Scale, the Graph control won't attempt to use type smaller than you specify with the Smaller-Bigger scroll bar. If the text is too large for the space available, it simply won't display.

Related Property

FontSize **Note**: If size is positive, the effect is the same as checking Smart Scale. If the size is negative, the effect is the same as clearing the Smart Scale box.

Smaller-Bigger

This scroll bar lets you set the size of type.   If Smart Scale is selected, the Graph control may override your setting to make the text small enough to fit in the graphing window.

Each click on the Smaller end of the scroll bar decreases the text size by 5 arbitrary units (to a minimum of 50), and each click on the Bigger end increases it by 5 units (to a maximum of 500). The initial size depends on which type of text you're sizing.

Related Property

FontSize

Reset

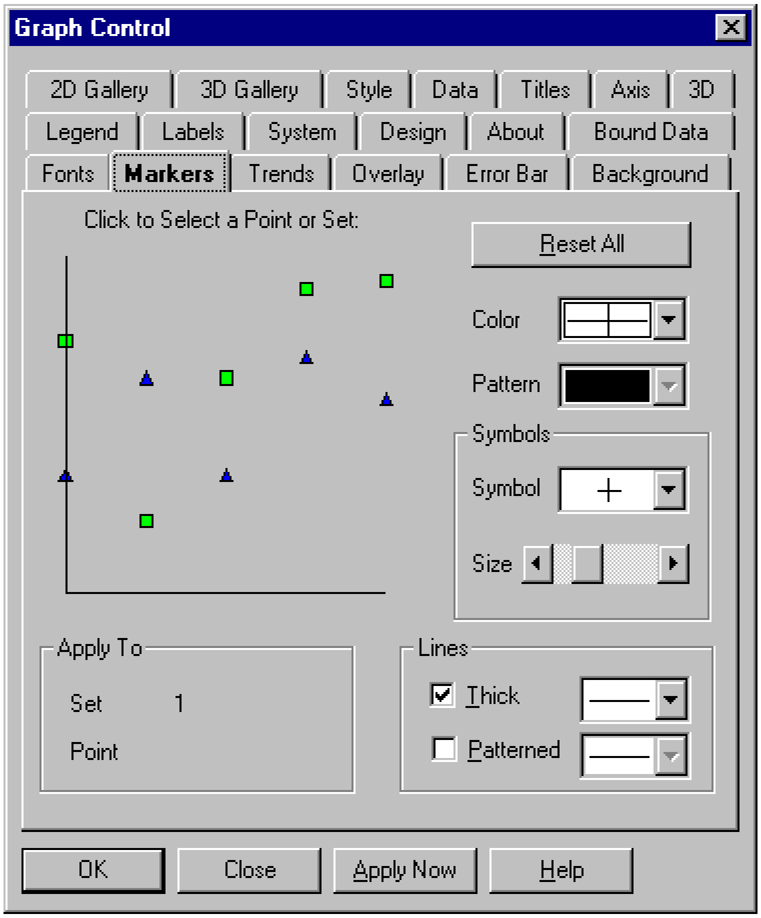
Click this button to reset the text size to the default.

Related Property

DataReset = 10 (Font properties)

Markers property page

Click where you need help



Graph Control

Set 1

Point

☑ Thick

□ Patterned

▲▼

Topic

Markers page

Related

Axis page

Background page

Data page

Overlay page

Click to Select a Point or Set

When you move the mouse pointer over the graph drawing at the upper left of the Markers page, the pointer becomes a large arrow.   Point and click on the marker you want to apply settings to.

Related Property

ThisPoint
ThisSet

Apply To group

This group shows you the point or set number of the graph object you've selected in the graph illustration.

Set. For all graph types *except* pie, bubble, and bar graphs having one data set, a Set number is shown. Your settings apply to a particular data set.

Point. For pie charts, bubble graphs, and bar graphs having one data set, a Point number is shown. Your settings apply to a particular data point.

Reset All button

Click this button to return all values in the Markers property page to their default.

Related Property

DataReset

Color (default is automatic color selection)

In this list box, you can choose a color for the selected marker from the current palette.   To choose a color palette, go to the Background property page.

By default, the Graph control assigns an automatic series of colors to markers, chosen for variety.   If you override this default by setting your own color for one marker, you have to set colors for the remaining markers as well--otherwise, they'll be shown in black.

Related Property

Color()
ColorData

Pattern (default is solid)

Choose a pattern for the selected marker.   If you don't choose one, the marker appears in a solid color.

Note: The Pattern list box is enabled only for pie, bubble, bar, area, Gantt, box-whisker, and tape graphs.

Related Property

Pattern()
PatternData

Symbols group

The options in this group are enabled for polar, scatter, line and logarithmic, time series, and (Size bar only) high-low-close and open-high-low-close graphs.

Symbol (default is automatic symbol)

In this list box, you can choose one of 14 symbol options.

By default, the Graph control assigns an automatic series of symbols to data sets, chosen for variety. If you override this default by setting your own symbol for one set, you have to set symbols for any remaining sets as well--otherwise, they'll all default to the first available symbol (+).

Related Property

Symbol()
SymbolData

Size

This scroll bar sets the size for symbols, based on a default of 100 arbitrary units. Each click to the left decreases the symbol size by 5 units (to a minimum of 10), and each click to the right increases it by 5 units (to a maximum of 1000).

Related Property

SymbolSize

Lines group

The Lines group is enabled for polar, line and logarithmic, and (Thick check box only) high-low-close and open-high-low-close graphs.

Thick (default is on)

Select this option to enable thick lines, which are three pixels thick by default. Deselect it for thin lines, which are one pixel thick.

For polar, line, and logarithmic graphs, you can choose a line thickness of one to five pixels (overriding the default three pixels) in the list box. This setting applies to all lines in the graph. You can't set your own thickness for high-low-close or open-high-low-close markers, which are always three pixels thick when Thick is on.

Related Property

Pattern()
PatternData
ThickLines

Patterned (default is off)

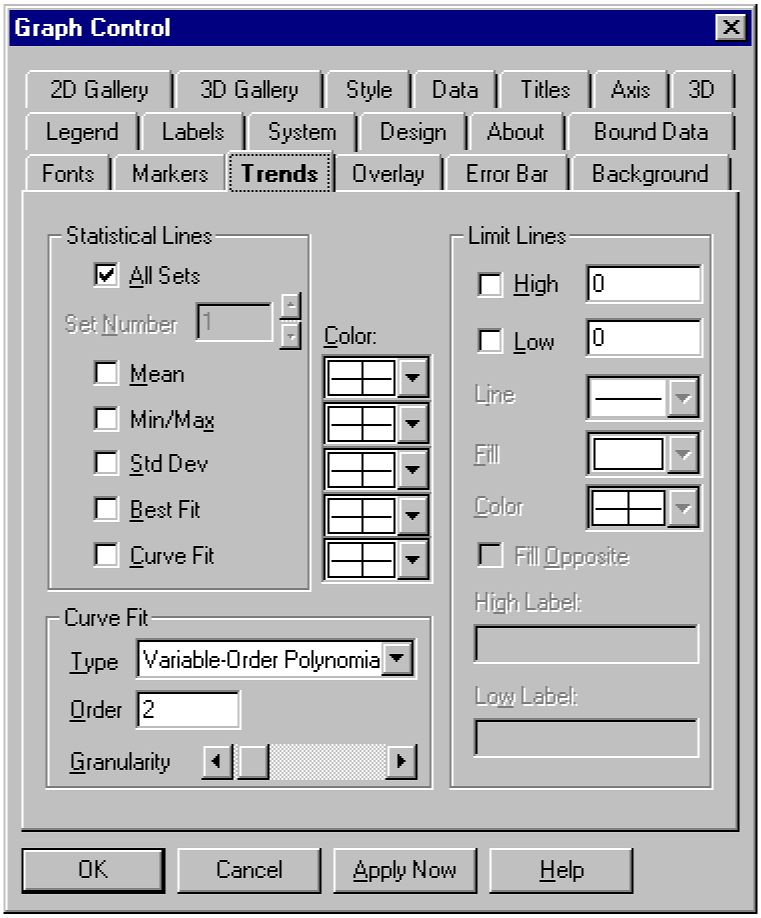
Select this option to enable patterned lines.   Then, in the list box, you can choose a pattern for each line.

Related Property

Pattern()
PatternData
PatternedLines

Trends property page

Click where you need help



Statistical Lines

☑ All Sets

Set Number 1

Color:

☐ Mean

☐ Min/Max

☐ Std Dev

☐ Best Fit

☐ Curve Fit

Curve Fit

Type Variable-Order Polynomia

Order 2

Granularity

Limit Lines

☐ High 0

☐ Low 0

Line

Fill

Color

☐ Fill Opposite

High Label:

Low Label:

▲▼

Topic

Trends page

Related

Data page

Error Bar page

All Sets

Select this option if you want your selection of statistical lines to apply to all data sets.

Related Property

LineStats

Set Number

This option is disabled when All Sets is checked. To select different statistical lines for each data set, clear the All Sets box. Then use the Set Number control to scroll through data sets one by one, selecting the statistical lines you want for each set.

If you've enabled an overlay graph, select statistical lines for that graphon the Overlay page.

Related Property

TrendSets

Mean (default is off)

Select this option to enable a mean line, which is drawn horizontally through the average value of all data points in a set. Clear the checkbox for no mean line.

Related Property

LineStats

TrendSets

Min/Max (default is off)

Select this option to enable min and max lines, which are drawn horizontally through the lowest and highest data points in a set. Clear the checkbox for no min and max lines.

Related Property

LineStats

TrendSets

Std Deviation (default is off)

Select this option to enable a pair of standard-deviation lines, which are drawn horizontally through the standard deviation from the mean (in both the positive and negative directions). Clear the checkbox for no standard-deviation lines.

Related Property

LineStats

TrendSets

Best Fit (default is off)

Select this option to enable a best-fit line, which is a straight line indicating the trend of data points (a first-order polynomial curve). Clear the checkbox for no best-fit line.

Related Property

LineStats

TrendSets

Curve Fit (default is off)

Select this option if you want to fit a curve through your data points. Deselect it for no curve.

Related Property

LineStats
TrendSets

Color (default is same color as data set)

Use these list boxes to choose colors for statistical lines from the current color palette. By default, statistical lines are drawn in the same color as the data sets they apply to. If you choose a color, it's applied to that type of line for all data sets.

To choose a color palette, go to the Background property page.

Related Property

ForegroundUse
Foreground

Curve Type (default is Variable-Order Polynomial)

Choose the type of curve you want to plot.

Related Property

CurveType

Order (default is 2)

This text box applies only to three curve types:

- For **variable-order polynomial curves,** Order is the order of the polynomial used in curve fitting. A setting of 1 produces a straight line (the same as a best-fit line); a setting one less than the number of points produces a curve that passes through every point.
- For **moving-average mid and moving-average end curves,** Order is the range of data points over which moving averages are averaged, beginning with the first point.

Related Property

CurveOrder

Granularity

This scroll bar sets the granularity of all curve types except moving-average.   The granularity is the number of "steps," or straight line segments, making up the curve.   Higher values create smoother curves, but require more drawing time.

The default Granularity setting is 50 curve steps, which generally creates a smooth-looking curve at a high drawing speed.   Each click to the left decreases the number of steps by 2 (to a minimum of 10), and each click to the right increases the number of steps by 2 (to a maximum of 1000).

With spline curves, you generally need higher granularities than normal--up to 10 times the number of points in the graph.

Related Property

CurveSteps

High

Select this option to enable a high limit line.

Related Property

LimitLines

High value

Enter a high limit value.

Note: You can enter a value without enabling a high limit line. You may want to specify a high value to be used in screening data before it is graphed. See the By Value list on the Data property page.

Related Property

LimitHighValue

Low

Select this option to enable a low limit line.

Related Property

LimitLines

Low value

Enter a low limit value.

Note: You can enter a value without enabling a low limit line. You may want to specify a low value to be used in screening data before it is graphed. See the Miss list on the Data property page.

Related Property

LimitLowValue

Line

Choose a pattern for the limit lines.

LimitLinePattern

Fill

Choose a pattern for filling the area outside of or between limit lines. The default is no pattern--the area will not be filled.

Related Property

LimitFillPattern

Color

Choose a color for limit lines and fills.

To choose a color palette, go to the Background property page.

Related Property

ForegroundUse
Foreground

Fill Opposite

Select this option if you want the area between limit lines filled with a pattern.

For this option to take effect, you must enable both high and low limit lines and select a fill pattern.

Related Property

LimitFillPattern
LimitLines

High Label

Type text for the high limit label. If this box is empty, the line will have no label.

Related Property

LimitHighLabel

Low Label

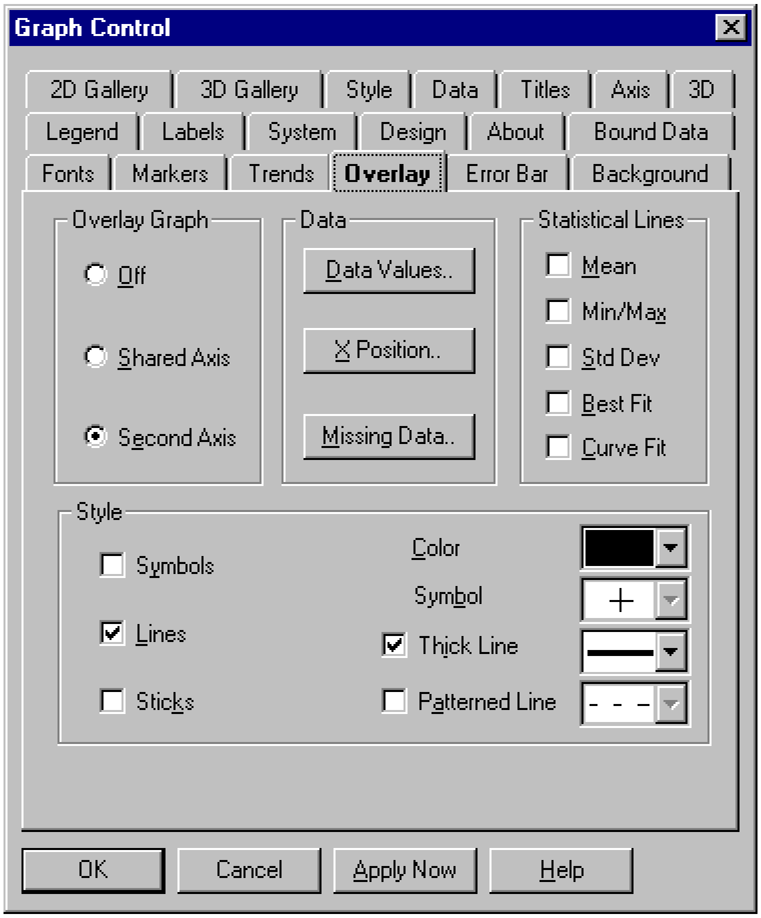
Type text for the low limit label. If this box is empty, the line will have no label.

Related Property

LimitLowLabel

Overlay property page

Click where you need help



Overlay Graph

○ Off

○ Shared Axis

● Second Axis

Statistical Lines

□ Mean

□ Min/Max

□ Std Dev

□ Best Fit

□ Curve Fit

Style

□ Symbols

☑ Lines

□ Sticks

☑ Thick Line

□ Patterned Line

▲▼

Topic

Overlay page

Related

Axis page

Markers page

Off (default)

Select this option for no overlay graph.

Related Property

OverlayGraph = 0 (Off)

Shared Axis

Select this option to draw an overlay graph using the same Y axis as the primary graph.

Related Property

OverlayGraph = 1 (On, shared axis)

Second Axis

Select this option to draw an overlay graph using a second Y axis, which is always drawn at the right edge of the graph.

Related Property

OverlayGraph = 2 (On, second axis)

Data Values

Click this button to enter the Data Values dialog, where you can enter values that will be plotted along the Y axis.

Related Property

OverlayData()
OverlayGraphData

X Position

X Position values are optional--you need them only if you want to set custom positions for data points along the X (horizontal) axis.

Related Property

OverlayXPosData

Missing Data

This dialog presents a grid in which you can mark certain data points as "missing." Missing data points aren't displayed in the graph, whether or not you've set a value for the point using the Data Values grid.

To specify a data point as missing, enter -1 in the cell corresponding to the point. To display a data point, leave its cell blank or enter 0.

Related Property

OverlayExtraData

Symbols (default is off)

Select this option to draw a symbol at the position of each data point.   Deselect it for no symbols.

Related Property

OverlayGraphStyle

Lines (default is off)

Select this option to draw lines between data points.   Deselect it for no lines.

Note that if *all* options in the Style group (Symbols, Lines, and Sticks) are deselected, lines are drawn in any case.

Related Property

OverlayGraphStyle

Sticks (default is off)

Select this option to draw a vertical "stick" between each data point and the Y origin.   Deselect it for no sticks.

Related Property

OverlayGraphStyle

Color (default is automatic black or white)

In this list box, you can choose a color for the overlay graph's markers from the selected color palette.   The default color is automatic black or white, whichever provides the most contrast.

To choose a color palette, go to the Background property page.

Related Property

OverlayColor

Symbol (default is +)

Choose one of 14 symbol options for the overlay graph.

Related Property

OverlaySymbol

Thick Line (default is on)

Select this option to enable thick lines, which are three pixels thick by default.   Deselect it for thin lines, which are one pixel thick.

In the list box, you can choose a line thickness of one to five pixels, overriding the default three pixels.

Related Property

OverlayThickLines

Thick Line list box

In this list box (available when the Thick Line option is enabled), you can choose a line thickness of one to five pixels, overriding the default three pixels.

Related Property

OverlayPattern

Patterned Line (default is off)

Select this option to enable patterned lines. Then, in the list box, you can choose a pattern for the line.

Related Property

OverlayPatternedLines

Patterned Line list box

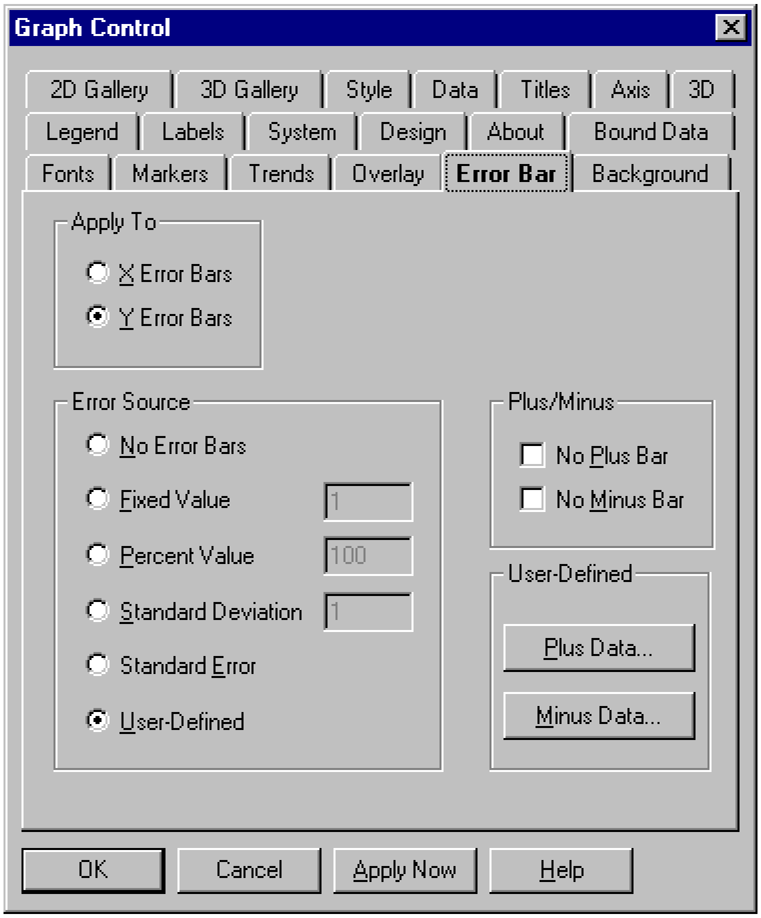
In this list box (available when the Patterned Line option is enabled), you can choose a pattern for the line.

Related Property

OverlayPattern

Error Bar property page

Click where you need help



Apply To

○ X Error Bars

● Y Error Bars

Error Source

○ No Error Bars

○ Fixed Value 1

○ Percent Value 100

○ Standard Deviation 1

○ Standard Error

● User-Defined

Plus/Minus

□ No Plus Bar

□ No Minus Bar

User-Defined

▲▼

Topic

Error Bar page

Related

Trends page

X Error Bars (available for scatter graphs only)

Select this option to apply settings to X error bars, which are available only in scatter graphs. These bars extend horizontally from plotted points.

Related Property

EBarUse = 1 (Horizontal bars)

Y Error Bars (default)

Select this option to apply settings to Y error bars.   These bars extend vertically from plotted points, except in horizontal bar graphs, in which they extend horizontally.

Related Property

EBarUse = 0 (Vertical bars)

No Error Bars (default)

Select this option for no error bars.

Related Property

EBarSource = 0 (No error bars)

Fixed Value

Select this option to set a fixed length for error bars.   You can enter this fixed length in the text box.

Related Property

EBarSource = 1 (Fixed)
EBarValue

Percent Value

Select this option to set a percentage length for error bars--the length of each bar is a percentage of the plotted value.   You can enter this percentage in the text box.

Related Property

EBarSource = 2 (Percentage)
EBarValue

Standard Deviation

Select this option to set the length of error bars as the standard deviation from the mean. In the text box, you can specify a multiplier for this standard deviation.

Related Property

EBarSource = 3 (Standard deviation)

Standard Error

Select this option to set the length of error bars as the standard error of the plotted values.

Related Property

EBarSource = 4 (Standard error)

User-Defined

Select this option to enter your own lengths for the error bars.

Related Property

EBarSource = 5 (User-defined)

No Plus Bar (default is off)

Select this option to disable error bars in the positive direction.   Deselect it to enable them.

Related Property

EBarStyle = 1 (Minus bars only)

No Minus Bar (default is off)

Select this option to disable error bars in the negative direction.   Deselect it to enable them.

Related Property

EBarStyle = 2 (Plus bars only)

Plus Data and Minus Data

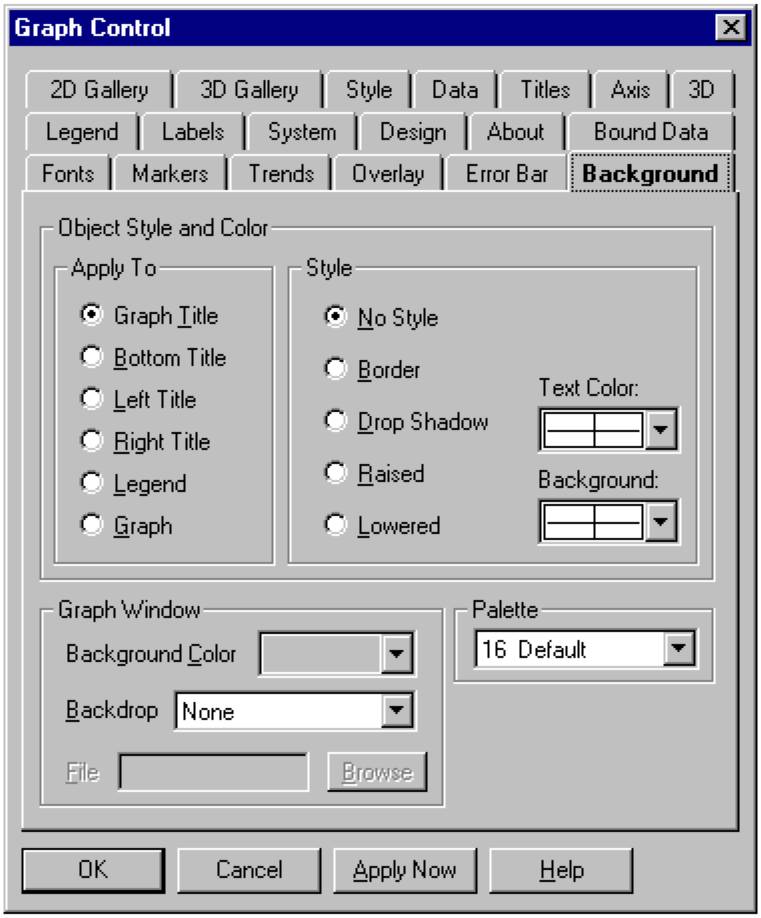
Click these buttons to enter the Plus Error Bars and Minus Error Bars dialogs, which let you define the lengths of error bars in the positive and negative directions.   In these dialogs, each cell corresponds to a data point and each row corresponds to a data set.

Related Property

EBarGraphDataMinus
EBarGraphDataPlus
EBarXPosDataMinus
EBarXPosDataPlus

Background property page

Click where you need help



Object Style and Color

Apply To
- ● Graph Title
- ○ Bottom Title
- ○ Left Title
- ○ Right Title
- ○ Legend
- ○ Graph

Style
- ● No Style
- ○ Border
- ○ Drop Shadow
- ○ Raised
- ○ Lowered

Text Color:

Background:

Graph Window

Background Color

Backdrop: None

File

Browse

Palette: 16 Default

▲▼

Topic

Background page

Related

Fonts page

Titles page

Apply To

Select the graph object to which you want to apply styles and colors--the **Graph Title (default), Bottom Title, Left Title, Right Title, Legend,** or **Graph** (including the graph itself and its axes).

Related Property

BackgroundUse
ForegroundUse

No Style (default)

Select this option for no styling effect.

Related Property

BackgroundStyle = 0 (No background styling)

Border

Select this option to draw a border around the object.

Related Property

BackgroundStyle = 1 (Frame)

Drop Shadow

Select this option to draw a black drop shadow behind the object.

Related Property

BackgroundStyle = 2 (Drop shadow)

Raised

Select this option to draw a border with a "raised" appearance around the object.

Related Property

BackgroundStyle = 3 (Raised)

Lowered

Select this option to draw a border with a "lowered" appearance around the object.

Related Property

BackgroundStyle = 4 (Lowered)

Text Color (default is automatic black or white)

Choose a color for the object's text from the current palette. The default color is automatic black or white, whichever provides the most contrast.

Related Property

ForegroundUse
Foreground
LegendStyle

Background (default is automatic black or white)

Choose a background color for the selected object.

Related Property

BackgroundUse
Background

Palette

Choose a 16-color or 128-color palette for your screen.   Whenever you have to set a color in the Graph control, this palette determines the choices in the color list box.

- **16-color palettes** consist of 16 differentiated colors--default, pastel, or grayscale.
- **128-color palettes** always consist of the 16 colors from the default palette, followed by 96 colors that vary according to the palette you select. The Graph control reserves 16 additional colors for special graphic needs, such as drawing the sides of 3D bars; you can't select these reserved colors yourself.

Related Property

Palette

Background Color (default is light gray)

Choose a background color for the graphing window from the current color palette.

Related Property

BackgroundUse = 0 (Entire graphing window)
Background = ColorIndex

Backdrop (default is None)

This list box lets you choose a type of graphic image (bitmap or metafile) to use for the backdrop of the graphing window.   You also choose how the image is displayed--centered, tiled, or stretched.

Although you can use any type of backdrop in any drawing mode, you'll get best performance if you choose a metafile backdrop when *drawing* and a bitmap backdrop when *blitting*.   At design time, you can set Draw or Blit in the Design property page, Draw Mode group.

Related Property

BackdropStyle

File

In this text box, you can enter the filename for the graph's backdrop image. If you don't include a path to the file as part of this string, the Graph control searches the current directory. The appropriate file extension (.BMP or .WMF) is added automatically, according to your selection in the Backdrop list box.

Related Property

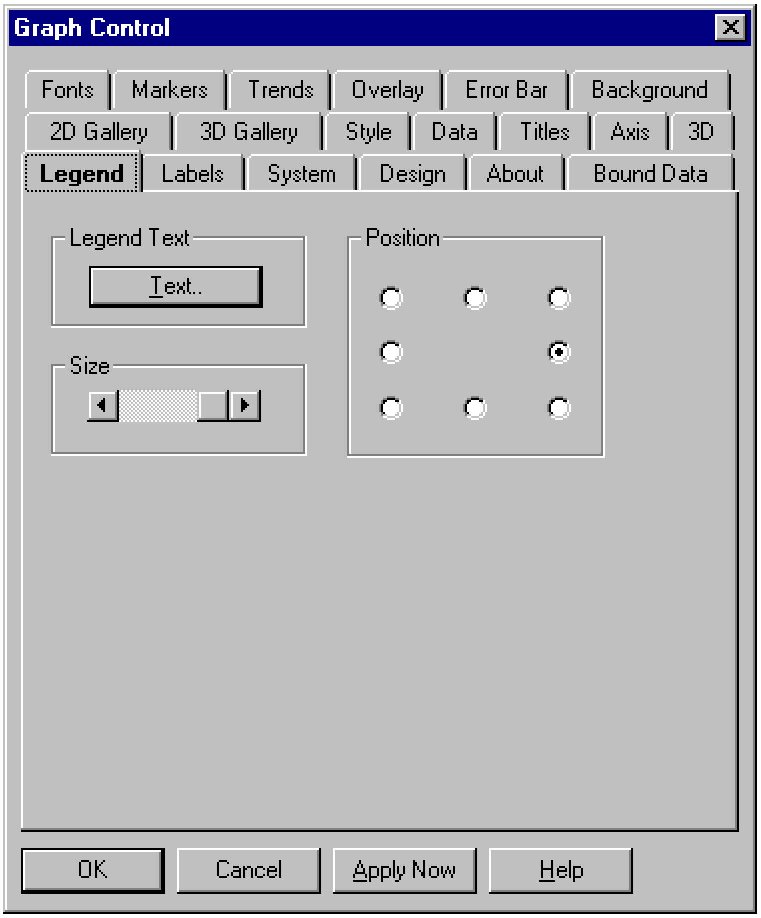
Backdrop

Browse

Click this button if you want to call up a standard Windows Open dialog to locate a backdrop file.

Legend property page

Click where you need help



▲▼

Topic

Legend page

Related

Background page

Bound data page

Data page

Markers page

Legend Text

Click this button to call up the Legend Text dialog, which lets you enter the individual text strings for a graph's legend. The number of legend strings depends on the graph type:

- For pie charts, bubble graphs, and bar graphs having only one data set, the number of legend strings is equal to the Points value on the Data page.
- For all other graph types, the number of legend strings is equal to the Sets value on the Data page.

Related Property

LegendText

Position (default is right-hand center)

If your graph has a legend, you can use these buttons to set the legend's position around the edges of the graphing window.

If you choose the top center or bottom center position, legend items are drawn horizontally in a single row. At all other positions, legend items are stacked vertically.

Related Property

LegendPos

Size (default is maximum)

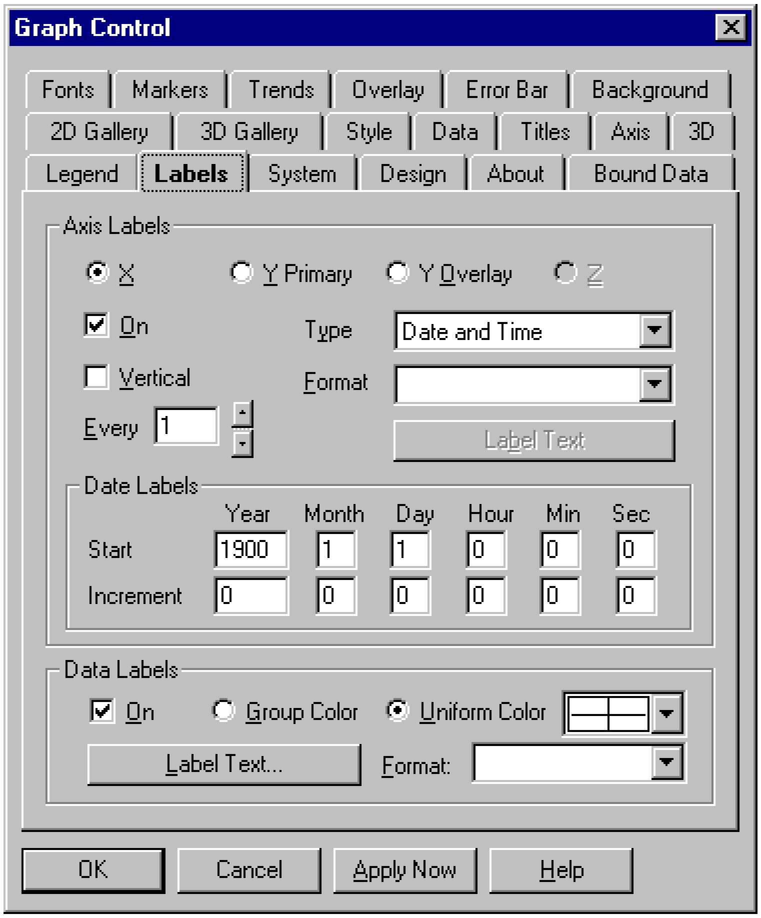
This scroll bar lets you set the size of a legend--including the text, marker, and gap between items. The Size setting is a percentage of the maximum legend size. The default setting is 100, and each click to the left or right decreases or increases the setting by 1 (to a minimum of 0 or maximum of 100).

Related Property

LegendSize

Labels property page

Click where you need help



Graph Control

Axis Labels

● X ○ Y Primary ○ Y Overlay ○ Z

☑ On

☐ Vertical

Every 1

Type: Date and Time

Format:

Date Labels

	Year	Month	Day	Hour	Min	Sec
Start	1900	1	1	0	0	0
Increment	0	0	0	0	0	0

Data Labels

☑ On ○ Group Color ● Uniform Color

Format:

▲▼

Topic

Labels page

Related

Axis page

Overlay page

Fonts page

Titles page

Label Text

This button is disabled until you select "Text Array" from the Type drop-down list. When enabled, clicking on the Label Text button presents a dialog with a grid in which you may enter text for labels along the selected axis.

Related Property

LabelText
YLabelText
ZLabelText

On (default is checked)

Select this option to show labels along the axis. Clear the checkbox for no labels.

Related Property

Labels

Type

Select a label type from the list.

If the current graph is pie or polar, the list has only one choice: numeric. For all other graph types, the options depend on which axis is selected.

Y and Z axis labels can be either numeric or text.

- If you select numeric, labels will be calculated automatically from the scale and range of the graph's data. You can format numeric labels by selecting a string from the Format listbox.
- If you select text, the Label Text button is enabled so that you can enter a list of labels. Formatting is not available for text labels.

X axis labels can be numeric, text, or one of several date/time types.

- If you select one of the date/time types, you must also set a starting date and an increment in the Date Labels group. You can format date/time labels by selecting a string from the Format listbox.

Related Property

LabelXType

Format

Applies formatting to numeric or date labels. Select a format from the list, or enter your own.

For information on how to construct a format string, see Numeric formats and Date/time formats.

Note: Label formatting is applied only when the Graph control automatically generates labels. If you supply your own label text, it's up to you to format the labels.

Related Property

LabelXFormat
LabelYFormat
LabelZFormat

Vertical (available for X and Z axes only; default is off)

Select this option to display X or Z labels vertically (rotated 90 degrees counterclockwise). Deselect it for horizontal labels.

Related Property

LabelStyle

Every (available for X axis only; default is 1)

The Every setting determines the frequency with which labels are displayed. A setting of 1 places a label at every tick along the X axis, a setting of 2 places a label at every other tick (beginning with the origin), and so on.

If you've defined text labels, they are displayed in the order you give them in the Label Text dialog regardless of the Every setting. Graphics Server doesn't "skip" any label text strings.

Related Property

LabelEvery

Start

Enter a starting date/time.

Note: Boxes for Year, Month and Day are enabled only when the label type is Date, Date and Time, or Date Skip Weekend. Boxes for Hour, Min and Sec are enabled only when the type is Time or Date and Time.

Related Property

LabelXDateStart

Increment

Enter the increment for each point label.

For example, if you want labels to increment by one year, enter 1 under Year and 0 in all other boxes. If you want them to increment by one month, enter 1 under Month and 0 in all other boxes. If you want to increment each label by one year and six months, enter 1 under Year, 6 under Month and 0 for Day (and, optionally, Hour, Min, Sec).

Related Property

LabelXDateInc

On (default is off)

Select this option to enable data labels, which are numeric or text strings associated with particular markers on a graph.

The Data Labels options are enabled for all 2D graph types except pie and time series. Data labels are not available for 3D graphs.

Related Property

DataLabels = 1 (On)

Format

Applies formatting to numeric labels. Select a format from the list, or enter your own.

For information on constructing a format string, see Numeric formats.

Note: Label formatting is applied only when the Graph control automatically generates labels. If you supply your own label text, it's up to you to format the labels.

Related Property

DataLabelFormat

Groups Color

Select this option to have each data label match the color of its associated marker.

Related Property

DataLabels = 2 (On, colored same as associated set/point)

Uniform Color (default is automatic black or white)

Select this option if you want to apply the same color (chosen from the list box) for all data labels. The default color is automatic black or white, whichever provides the most contrast.

Colors vary according to the color palette you've selected. To choose a palette, go to the Background property page.

Related Property

DataLabels = 1 (On)
ForegroundUse = 13 (Data Labels)
Foreground = ColorIndex

Label Text

Click this button to call up the Label Text dialog, which lets you specify text for each data label. If you don't define text labels, the Graph control derives numeric labels from the associated data.

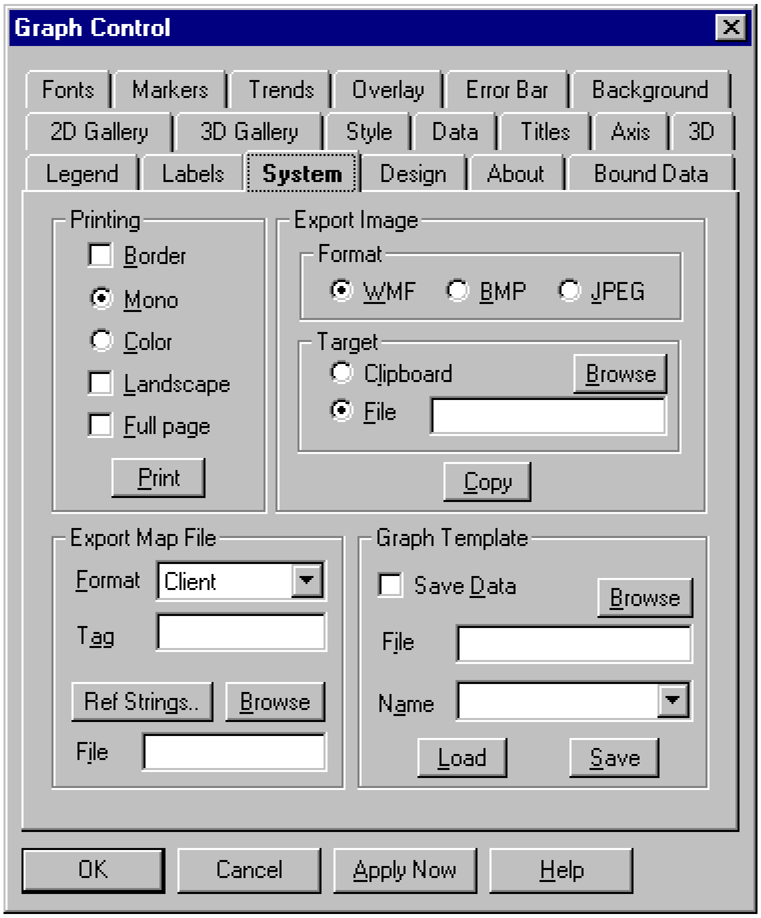
Note: Label formatting is applied only when the Graph control automatically generates labels. If you supply your own label text, it's up to you to format the labels.

Related Property

DataLabelText

System property page

Click where you need help



Printing

- □ Border
- ● Mono
- ○ Color
- □ Landscape
- □ Full page

Export Image

Format: ● WMF ○ BMP ○ JPEG

Target: ○ Clipboard ● File

Export Map File

Format: Client

Tag

File

Graph Template

□ Save Data

File

Name

▲▼

Topic

System page

Related

Background page

Design page

Border (default is off)

Select this option to include a border around the graphing window when you print it.   Deselect this option for no border.

Related Property

PrintStyle

Mono (default)

Select this option to print in monochrome, with a white background and black objects.   The Graph control supplies varying patterns, symbols, and line thicknesses to differentiate data sets and points from each other.

Related Property

PrintStyle
DrawStyle

Color

Select this option to print in color or grayscale.   The graphing window is printed as it appears on the screen.

Related Property

PrintStyle

Landscape

The default orientation of the graph on the page is **portrait**. Selecting this option changes the orientation to **landscape**.

Portrait Landscape

Related Property

PrintInfo(11)

Full page

Select this option to expand the graph to fit the dimensions of the paper.

Related Property

PrintInfo(11)

Print

Click this button to print the graphing window on the current printer.

Related Property

DrawMode = 5

WMF (default)

Select this option if you want to create a metafile of the graphing window.

Related Property

DrawMode

BMP

Select this option if you want to create a bitmap of the graphing window.

Related Property

DrawMode

JPEG

Select this option if you want to save an image of the graph in JPEG format.

Note: JPEG format requires GSJPG*16/32*.DLL to be in the home directory of GSW*16/32*.EXE. If the conversion DLL is not found, the JPEG option will be disabled.

Related Property

DrawMode

Clipboard

Select this option if you want to copy the image to the Windows clipboard.

Related Property

DrawMode

File

Select this option if you want to save the image to disk.

You can enter a filename for the image in the text box or click the Browse button to get a standard Windows Save As dialog. If you don't include a path, the Graph control places the file in the current directory.

Related Property

DrawMode
ImageFile

Browse

Click this button if you want to call up a standard Windows Save As dialog to set the filename and destination.

Copy

Click this button to save the image to disk or the clipboard according to your settings in the Format and Target subgroups.

Related Property

DrawMode

File

Enter the name of a graph template file. The default file extension is .GSP.

Related Property

GraphFile

Browse

Click this button to browse for a template file to open.

Name

What you enter here depends on whether you are saving or restoring a graph definition.

- To save a new graph definition, enter a name for the graph.
- To open a definition previously saved in this template, select a name from the drop-down list.

Related Property

GraphName

Load

Click this button to restore a previously saved graph definition.

Related Property

DrawMode

Save

Click this button to save the graph definition.

Related Property

DrawMode

Save Data

Select this option if you want to save graph data as well as the graph's appearance.

Related Property

DrawMode

Format

Select a format for an image map.

Related Property

MapFormat

Tag

Set the Name attribute for a client-side image map.

Related Property

MapName

Ref Strings

Enter an array of URLs for use in an image map.

Related Property

MapRefStrings

File

Name the file for an image map.

You can enter a path, filename and extension in the text box or click the Browse button to get a standard Windows Save As dialog. If you don't specify a path, the file is written to the current directory.

A client-side image map is stored within the HTML document that references it. The setting for MapFile should be the file name and extension of the HTML document.

A server-side image map is stored in a file external to the HTML document that references it. The convention is to use the same base file name for both the image and the map. Often the map file's extension is .MAP, though other extensions are usually acceptable.

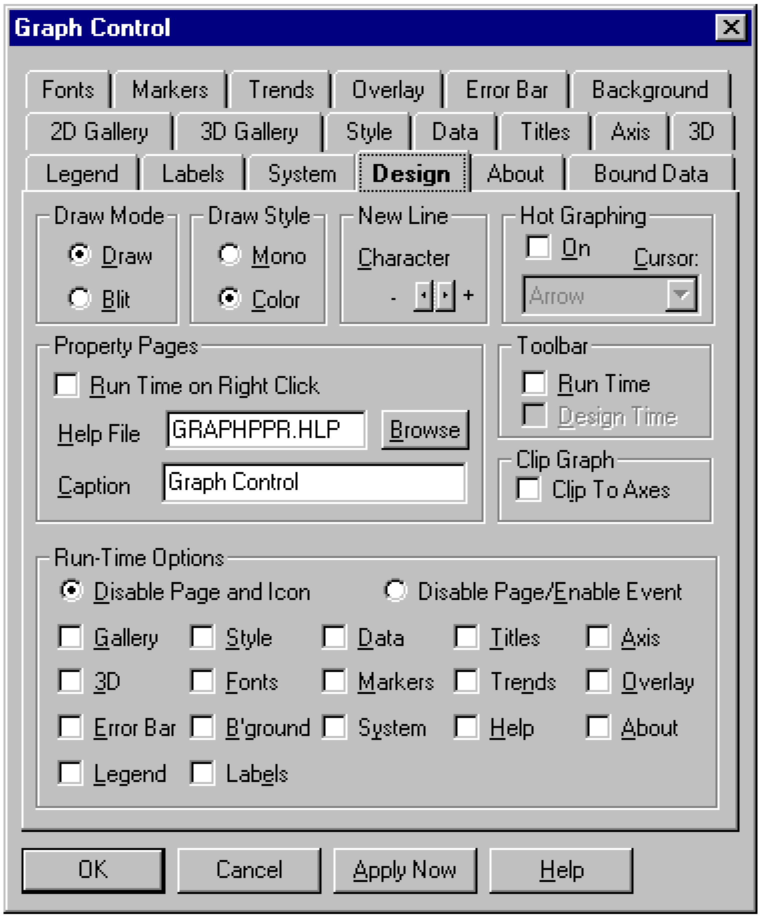
Note: Before an image map can be created, the Hot property must be set to 1 (On). Unlike what is involved in creating an image file, a map file cannot be created from the property page. It must be done in code by setting DrawMode to 12 (Create map) or 13 (Append map).

Related Property

MapFile

Design property page

Click where you need help



Draw Mode: ● Draw ○ Blit

Draw Style: ○ Mono ● Color

New Line Character: - +

Hot Graphing: □ On Cursor: Arrow

Property Pages: □ Run Time on Right Click

Help File: GRAPHPPR.HLP Browse

Caption: Graph Control

Toolbar: □ Run Time □ Design Time

Clip Graph: □ Clip To Axes

Run-Time Options: ● Disable Page and Icon ○ Disable Page/Enable Event

□ Gallery □ Style □ Data □ Titles □ Axis

□ 3D □ Fonts □ Markers □ Trends □ Overlay

□ Error Bar □ B'ground □ System □ Help □ About

□ Legend □ Labels

Draw Mode

In this group, you can set the way the graphing window is drawn when the graph first displays at run time.

Whenever you change a setting in code, the result of the change is not displayed until your code redraws the graph by again setting the DrawMode property. The setting you choose on the Design page affects only how the graph is drawn the first time it displays.

Draw (default) Select this option to display the graph as it is redrawn. With this option, the graph may flicker noticeably as it is drawn.

Blit Select this option to have the Graph control build a bitmap of the graph off-screen and then, when the drawing is complete, pop it into view.

Related Property

DrawMode

Draw Style

Mono. Select this option to draw graphs in monochrome. The background is set to white and all objects are drawn in black, with varying patterns and symbols supplied to differentiate data sets and points.

Color (default). Select this option to draw graphs in color.

Related Property

DrawStyle

On (default is off)

Select this option to enable the Graph control's hot-graphing system, which fires a HotHit event when the mouse is clicked on a particular set or point of a graph at run time.   Deselect this option for no hot graphing.

Related Property

Hot

Cursor (default is Arrow)

Choose a cursor shape.   If mouse events are enabled, the cursor takes this shape when it's within the graphing window at run time.

Related Property

MousePointer

Clip To Axes (default is off)

Select this option to mask areas of the graph that extend beyond the boundaries of the axes.

Related Property

ClipGraph

Character (default is space)

Use the spin buttons to select the character that you want to use to signal a line break in title and legend text.

Related Property

NewLineChar

Run Time on Right Click (default is off)

Select this option if you want the property pages to appear when the user right-clicks in the graphing window at run time.   Deselect it if you don't want to give the user access to the property pages.

Related Property

PropertyPages

Help File

In this text box, you can enter the filename for the Graph control's run-time Help file.   If you don't include a path to the file as part of this string, the Graph control searches the current directory.

Related Property

HelpFile

Browse

Click this button if you want to call up a standard Windows Open dialog to locate a Help file.

Caption (default is Graph control)

In this text box, you can enter the caption for the property pages window.

Related Property

PropertyCaption

Run Time (default is off)

Select this option to include the Graph control toolbar in the graphing window at run time.   Deselect it for no toolbar at run time.

Related Property

Toolbar

Design Time (default is on)

Select this option to include the Graph control toolbar in the graphing window at design time. Deselect it for no toolbar at design time.

Note: This option is available only for the VBX interface.

Related Property

Toolbar

Disable Page and Icon (default)

Select this option to disable various property pages (as well as the Graph control's run-time Help file). If you're using the toolbar at run time, the toolbar icons for the selected property pages are also disabled.

You can check any of the 15 boxes: **Gallery** (which applies to both 2D Gallery and 3D Gallery), **Style, Data, Titles, Axis, 3D, Fonts, Markers, Trends, Overlay, Error Bar, B'ground** (for Background), **System, Help** (file, not property page), and **About**.

Note: Disabling the Data property page also disables the Bound data page.

Related Property

Tool()
ToolStat

Disable Page/Enable Event

Select this option to disable various property pages (as well as the Graph control's run-time Help file) but leave their toolbar icons enabled if you're using the toolbar at run time.   When a user clicks on an icon with its property page disabled, a ToolHit event is fired.

This option lets you create your own dialog to replace the usual property page.   For details, refer to the entries for the ToolStat property and ToolHit event in the alphabetical property reference.

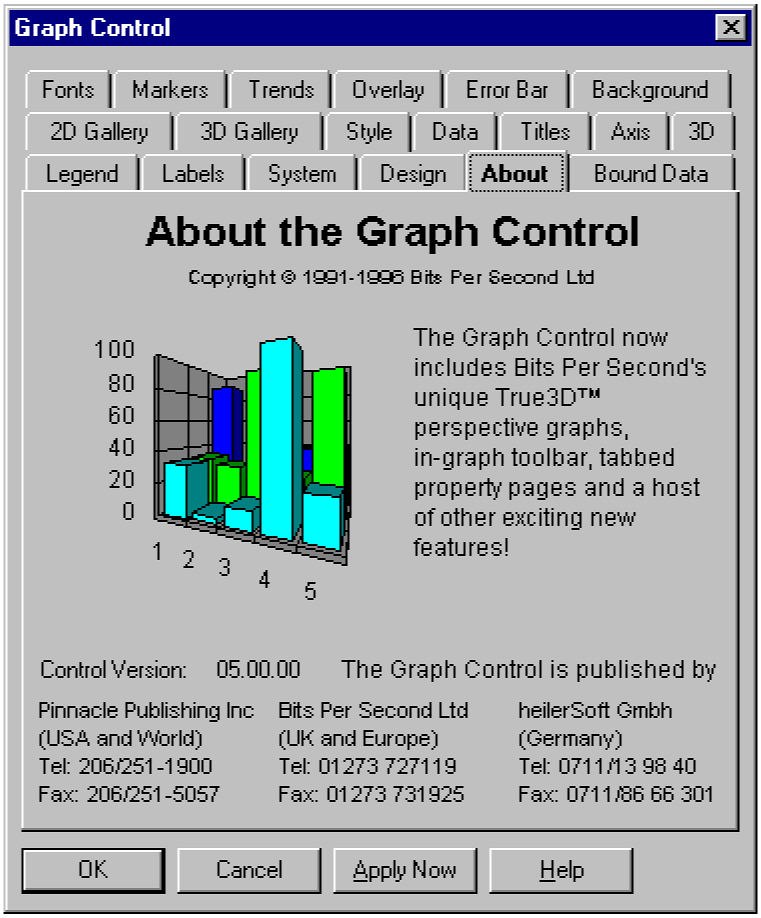
You can check any of the 15 boxes: **Gallery** (which applies to both 2D Gallery and 3D Gallery), **Style, Data, Titles, Axis, 3D, Fonts, Markers, Trends, Overlay, Error Bar, B'ground** (for Background), **System, Help** (file, not property page), and **About**.

Related Property

ToolStat

About the Graph control property page

Click where you need help



About the Graph Control

Copyright © 1991-1996 Bits Per Second Ltd

The Graph Control now includes Bits Per Second's unique True3D™ perspective graphs, in-graph toolbar, tabbed property pages and a host of other exciting new features!

Control Version: 05.00.00

The Graph Control is published by

Pinnacle Publishing Inc
(USA and World)
Tel: 206/251-1900
Fax: 206/251-5057

Bits Per Second Ltd
(UK and Europe)
Tel: 01273 727119
Fax: 01273 731925

heilerSoft Gmbh
(Germany)
Tel: 0711/13 98 40
Fax: 0711/86 66 301

About

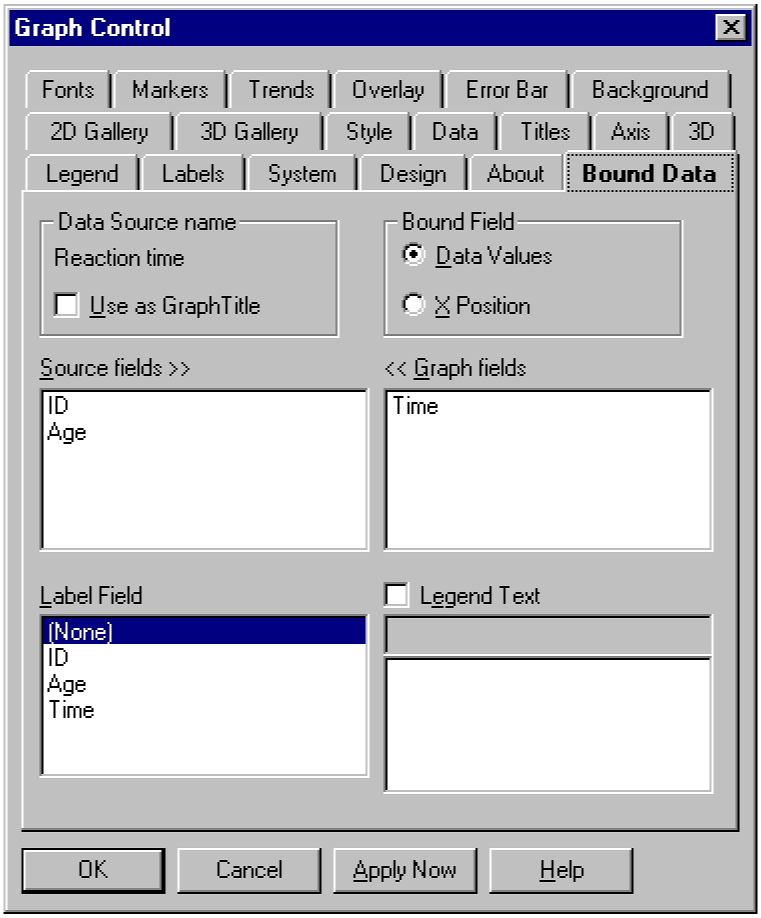
The About the Graph control property page contains the Graph control's version number and related information.   There are no settable options in this property page.

Related Property

CtlVersion

Bound data property page

Click where you need help



Data Source name

Reaction time

☐ Use as GraphTitle

Bound Field

● Data Values

○ X Position

Source fields >>

ID
Age

<< Graph fields

Time

Label Field

(None)
ID
Age
Time

☐ Legend Text

Data Values

Select this option to set fields used in graphing Y values.

X Position

Select this option to set fields used in graphing X values.

Data Source name

The name of the database or table that is the source of the graph's data. You cannot change the name. It is provided only for your information.

Use as Graph Title

Select this option if you want to use the Data Source name as the title for the graph.

Related Property

GraphTitle

Source data fields

This is a list of database fields not yet selected for graphing. To add a field to the graph, double-click its name in the list. The name will then move to the list of Graph data fields.

Graph data fields

This is a list of database fields selected for graphing. To remove a field from the list, double-click its name. It will then move to the list of Source data fields.

Note: Data fields are graphed in the order in which they are listed.

Related Property

DataField()
DataFields
XPosField()
XPosFields

Label Field

If you want one of the fields in the database to provide text for X axis labels, select the field by highlighting its name in the list.

Related Property

LabelField

Legend Text

Select this option if you want to enter text strings for the graph's legend. (You can also set legend text from the Data property page.)

Related Property

LegendField
LegendText

Edit box

By default, text for the legend is taken from the names of the fields selected for graphing. To edit an item, select it by highlighting the item in the list below the edit box. Then type replacement text in the edit box.

Selection box

Select the text you want to edit by highlighting it in this list.

▲▼

Topic

Bound data page

Related

Data page

Angular axis

In a polar graph, a scale represented by angular displacements from the origin of three o'clock.   Like the X axis of a graph with standard perpendicular axes, the angular axis usually shows an independent variable or cause.

Bitmap

An image file format with the extension .BMP. The pixels that make up an image on the screen are stored, or mapped, to the file as a collection of bits. When the image is in color, each pixel maps to more than one bit.

Error bars

Vertical or horizontal lines indicating a range of error or uncertainty in the plotted values.

Color index number

0	Black	8	Dark gray
1	Blue	9	Light blue
2	Green	10	Light green
3	Cyan	11	Light cyan
4	Red	12	Light red
5	Magenta	13	Light magenta
6	Brown	14	Yellow
7	Light gray	15	White

16	Automatic black or white, whichever provides higher contrast
17-127	Colors from 128-color palettes (see the Palette property)

Graph Template

Graph templates provide a mechanism for saving and later restoring changes made to a graph. You can create as many template files as you need. Each one is capable of storing definitions for multiple graphs. A graph definition automatically includes settings for all properties affecting a graph's appearance. It can optionally include graph data as well.

Image map

A tool for mapping areas of an image to hypertext links. The Graph control can map each data point or point marker to a hypertext link that you supply, and output the result in a map file. The map can then be processed by either a web browser (client-side) or web server (server-side).

The format for client-side image maps is defined by the HTML 3.0 specification.

The format for server-side image maps is defined by the server software. Two server-side formats are in common use. One was developed by the National Center for Supercomputing Applications (NCSA). The other was developed at the birthplace of the World Wide Web, the European Laboratory for Particle Physics (CERN).

ISV Data Binding

ISV data binding is the result of an initiative by some Independent Software Vendors (ISVs) who supply custom controls for Visual Basic to define an independent standard for passing data between their controls.   The new data binding scheme uses a custom interface called ISVCursor, designed by Gary Hill of Apex Software Corporation, the creators of TrueDBGrid.

JPEG

A graphics file format developed by the Joint Photographic Experts Group. JPEG image files normally take the file extension JPG.

The JPEG graphics file format uses compression techniques to reduce the size of the image file. Unlike some compression techniques, those used in producing JPEG images are *lossy*, meaning that the original image is not reconstructed bit by bit. Instead, some image information is left out, or lost, during compression.

Some forms of JPEG compression allow the user to adjust the compression ratio. The ratio used by the Graph control is fixed at 90. This ratio has been chosen as an optimal compromise between image quality and compression.

Labels

Strings of text used to identify a graph's reference values, categories, and specific measurements. Labels are placed along graph axes or around the perimeter of a pie chart.

Legend

A list associating the colors or patterns used in the elements of a graph with the variables they represent.

Limit lines

Lines drawn on a graph to highlight data that falls outside prescribed limits.   Limit lines are not drawn on pie, polar, time-series, or any 3D graphs.

Metafile

A Windows image file format with the extension .WFM.   The file stores the graphical objects (lines, circles, polygons) that compose an image rather than the pixels that make it up.   A metafile preserves an image better than a bitmapped image when the image is resized.

Overlay graph

The Graph control lets you draw a second graph, called an *overlay graph,* using the same X axis as your primary graph.   Overlay graphs are always line graphs and show only one data set.   They can be overlaid on seven 2D graph types: scatter, line, bar (vertical), area, high-low-close, open-high-low-close, and candlestick.

Radial axis

In a polar graph, a scale represented by the radial distance from the center point of the graph. Like the Y axis of a graph with standard perpendicular axes, the radial axis usually shows a dependent variable or effect.

X axis

A graph's horizontal scale.   Usually, it shows an independent variable.

Y axis

A graph's vertical scale.   Usually, it shows a dependent variable.

Z axis

In 3D graphs, a third axis indicating a depth scale.

Graph control properties and events

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z

A

(About) design-time property	Displays Graph control's About box
AutoInc property	Automatically increments ThisPoint and ThisSet counters

B

Backdrop property	Selects a backdrop image for the graphing window
BackdropStyle property	Sets type and presentation style of backdrop image
Background property	Sets background color for graphic object specified by BackgroundUse
BackgroundStyle property	Sets background style for graphic object specified by BackgroundUse
BackgroundUse property	Selects graphic object to which background colors and styles are applied
Bar2DGap property	Sets the gap between bars in 2D bar graphs
BottomTitle property	Specifies text string to place at bottom of graphing window

C

CageEdgeColor property	Sets color of edges of True3D graph cage
CageFlip property	Allows back and side walls of True3D graph cage to switch to opposite edge for better viewing
CageStyle property	Sets style of True3D graph cage walls (thick or thin)
CageWallColor property	Sets color of True3D graph cage walls

ClipGraph property	Enables clipping of a graph within its axes
Color() run-time property	Alternative way to set colors of graph elements at run time
ColorData property	Sets graph colors
CtlVersion property	Returns version number of current copy of Graph control
CurveOrder property	Sets order of polynomial used in curve fitting or range of points for moving averages
CurveSteps property	Sets curve resolution
CurveType property	Selects mathematical formula for curve

D

Data() run-time property	Alternative way to set graph data values at run time
DataLabel() run-time property	Alternative way to set text of data labels at run time
DataField() run-time property	Selects the fields of a bound database to be graphed
DataFields property	Binds a graph to one or more fields in a database
DataLabelFormat property	Sets the text formatting for numeric data labels
DataLabels property	Enables and disables data labels
DataLabelText property	Sets text of data labels
DataReset property	Resets any or all array-based properties for graph
DataSource design-time property	Binds the graph to a Data control
DrawMode property	Sets drawing mode for Graph control
DrawStyle property	Sets drawing style to color or monochrome

E

EBarDataMinus() run-time property	Alternative way to set error bar minus values at run time
EBarDataPlus() run-time property	Alternative way to set error bar plus values at run time
EBarGraphDataMinus property	Sets user-defined minus values for error bars, based on GraphData
EBarGraphDataPlus property	Sets user-defined plus values for error bars, based on GraphData
EBarSource property	Enables error bars and sets method for determining bar length
EBarStyle property	Sets style for error bars--plus, minus, or both
EBarUse property	Specifies whether error bar options in scatter graphs apply to vertical or horizontal bars
EBarValue property	Sets value used to calculate length of error bars
EBarXPosDataMinus property	Sets user-defined minus values for error bars, based on XPosData
EBarXPosDataPlus property	Sets user-defined plus values for error bars, based on XPosData
EBarXPosMinus() run-time property	Alternative way to set EBarXPosDataMinus property values at run time
EBarXPosPlus() run-time property	Alternative way to set EBarXPosDataPlus property values at run time
Elevation property	Sets vertical viewing angle for True3D graphs
Extra() run-time property	Alternative way to set ExtraData property values at run time
ExtraData property	Holds additional data for various graph types

F

FName() run-time property	Sets name of font for text at run time
FontFamily run-time property	Applies font name to text
FontName design-time property	Sets name of font for text
FontSize property	Sets size for text
FontStyle property	Sets treatment (bold, italic, or underlined) for text
FontUse property	Selects text component to which other font properties apply

Foreground property	Sets color of titles, labels, legends, axes, grids, statistical lines, curves, and data labels
ForegroundUse property	Selects graphic object to which color setting applies
FSize() run-time property	Alternative way to set font size at run time.
FStyle() run-time property	Alternative way to set font style at run time.

G

(Graph Properties) design-time property	Opens Graph control's property pages
GraphCaption property	Sets caption text for empty graphing window
GraphData property	Sets data to be graphed
GraphFile property	Selects a graph template file
GraphName property	Selects a graph definition within a template file
GraphStyle property	Sets characteristics for all graph types
GraphTitle property	Places text string at top of graphing window
GraphType property	Selects type of graph
GridLineStyle property	Sets style for grid lines
GridStyle property	Places reference grids on graph

H

HCtlWin run-time property	Returns handle of Graph control window
(Help) design-time property	Displays Windows Help
HelpFile property	Sets name of help file callable at run time
Hot property	Enables and disables hot graphing
HotButton run-time property	Returns number of mouse button pressed on hot graph
HotHit event	Fired when hot graph receives mouse click at run time
HWin run-time property	Returns handle of graphing window

I

ImageFile property	Sets filename for writing bitmap or metafile to disk
IndexStyle property	Sets indexing method for array-based properties
ISVDataSource property	Binds the graph to another control

L

Label() run-time property	Alternative way to set text of graph labels at run time
LabelEvery property	Sets frequency of X axis labels
LabelField property	Selects a text field in a bound database to provide labels for the X axis
Labels property	Enables or disables graph labels
LabelStyle property	Sets style (horizontal or vertical) for axis labels
LabelText property	Sets text for graph labels
LabelXDateInc property	Sets a date/time increment for labels on the X axis
LabelXDateStart property	Sets a starting date/time for labels on the X axis
LabelXFormat property	Sets format for numeric or date/time X labels
LabelXType property	Sets the type of labels to be used on the X axis
LabelYFormat property	Sets format for numeric Y labels
LabelZFormat property	Sets format for numeric Z labels
LeftTitle property	Specifies text string to place at left of graphing window
LeftTitleStyle property	Sets style for left graph title
Legend() run-time property	Alternative way to set legend text at run time
LegendField property	Selects a text field in a bound database to provide text for the graph legend

LegendPos property	Sets position of graph legend
LegendSize property	Sets overall legend size
LegendStyle property	Sets legend text to color or monochrome
LegendText property	Sets text for graph legend
LimitFillPattern property	Sets the fill pattern of the shaded area above and below limit lines
LimitHighLabel property	Places label text next to the high limit line
LimitHighValue property	Sets the position of the high limit line
LimitLinePattern property	Selects a pattern for high and low limit lines
LimitLines property	Enables limit lines for the graph
LimitLowLabel property	Places label text next to the low limit line
LimitLowValue property	Sets the position of the low limit line
LineStats property	Enables statistical lines and curves

M

MapFile property	Names the file for an image map
MapFormat property	Selects the format for an image map
MapName property	Sets the Name attribute for a client image map
MapRefString() run-time property	Alternative way to store URLs for an image map at run time
MapRefStrings property	Stores an array of URLs for use in an image map
MissingData property	Sets the mode for flagging data points as missing
MissingLineMode property	Selects option for bridging gaps caused by missing data
MousePointer property	Sets appearance of mouse pointer displayed in graphing window

N

NewLineChar property	Selects a character for imposing line-breaks
NumPoints property	Sets number of points per data set
NumSets property	Sets number of data sets to be graphed
NView run-time property	Specifies the Graphics Server view in which to display the graph
NWin run-time property	Specifies the Graphics Server window in which to display the graph

O

OverlayColor property	Sets color of overlay graph
OverlayData() run-time property	Alternative way to set data to be graphed in overlay graph at run time
OverlayExtra() run-time property	Alternative way to flag an overlay's missing points at run time
OverlayExtraData property	Stores flags to mark missing points for an overlay graph
OverlayGraph property	Allows second graph to be overlaid on primary graph
OverlayGraphData property	Sets data to be graphed in overlay graph
OverlayGraphStyle property	Sets graph style for overlay graph
OverlayGraphType property	Sets graph type for overlay graph
OverlayPattern property	Sets line pattern or thickness for overlay graph
OverlayPatternedLines property	Sets style of lines in overlay graph
OverlaySymbol property	Sets symbol design for overlay graph
OverlayThickLines property	Enables thick lines for overlay graph
OverlayTrendSet() run-time property	Alternative way to enable trend lines on an overlay graph at run time
OverlayTrendSets property	Enables trend lines for an overlay graph
OverlayXPos() run-time property	Alternative way to set X values for an overlay at run time
OverlayXPosData property	Sets X values for an overlay graph

P

Palette property	Sets color palette for graph
Pattern() run-time property	Alternative way to set fill pattern, line pattern, or line thickness at run time
PatternData property	Sets fill pattern, line pattern, or line thickness
PatternedLines property	Sets style of lines
Perspective property	Sets degree of perspective foreshortening for True3D graphs
Picture run-time property	Passes graph image directly to Picture control
PrintInfo() run-time property	Sets information needed to print graph at high resolution
PrintStyle property	Sets printing style
PropertiesSet event	Fired when a user changes property settings at run time
PropertyCaption property	Sets caption for property pages window
PropertyPages property	Makes property pages available at run time

Q

QuickData run-time property	Sets or returns all graph data in one operation

R

RandomData property	Enables generation of random data for graph
RangeMax property	Defines the upper range of data to be graphed
RangeMin property	Defines the lower range of data to be graphed
RightTitle property	Specifies text string to place at right of graphing window
RightTitleStyle property	Sets style for right graph title
Rotation property	Sets horizontal viewing angle for True3D graphs

S

SDKHit event	Fired when hot region receives mouse click at run time
SDKInfo() run-time property	Provides internal Graphics Server information about last graph displayed
SDKMouse property	Enables SDKHit and SDKPress events
SDKPaint event	Fired each time graphing window is redrawn at run time
SDKPaint property	Enables SDKPaint events
SDKPress event	Fired when mouse button is pressed or released in graphing window at run time
SDKTrack event	Fired when mouse cursor is moved in graphing window at run time
SeeThru run-time property	Lets graphing windows be drawn without clearing background
SmartLabels property	Enables auto-arranging of pie labels to avoid overlap
StackedTabs property	Selects either stacked or scrolling tabs for the property pages
SurfaceColorMax property	Sets color index used at highest point of True3D surface graph
SurfaceColorMin property	Sets color index used at lowest point of True3D surface graph
SurfaceWallColor	Sets color of end walls of True3D surface graph
Symbol() run-time property	Alternative way to set symbol design at run time
SymbolData property	Sets symbol design
SymbolSize property	Sets size of symbols

T

ThickLines property	Enables thick lines
ThisPoint property	Indexes array-based properties, usually by data point number
ThisSet property	Indexes array-based properties by data set number
TickEvery property	Sets frequency of ticks on X axis

Ticks property	Enables or disables tick marks for axis
TickStyle property	Selects a style option for drawing tick marks
Tool() run-time property	Alternative to ToolStat property for enabling toolbar icons and property pages at run time
Toolbar property	Enables display of Graph control toolbar
ToolHit event	Fired when mouse is clicked on toolbar icon enabled for events
ToolStat property	Enables or disables toolbar icons and associated property pages
TrendSet() run-time property	Alternative way to enable trend lines at run time
TrendSets property	Enables trend lines for individual data sets
True3D property	Enables True3D graphing and sets projection method
True3DDepth property	Sets projected depth of True3D graph
True3DXGap property	Sets gap between bars in True3D bar graph
True3DZGap property	Sets gap in Z direction between data sets in True3D area, bar, or tape graph

X

XAxisMax property	Sets maximum value for X axis
XAxisMin property	Sets minimum value for X axis
XAxisMinorTicks property	Enables minor tick marks on X axis
XAxisPos property	Sets position of X axis in 2D graph
XAxisStyle property	Sets method for determining scale and range of X axis
XAxisTicks property	Sets number of ticks on X axis
XPos() run-time property	Alternative way to set independent X values at run time
XPosData property	Sets independent X values
XPosField() property	Alternative way to select fields for X values at run time
XPosFields property	Selects database fields to be graphed as X values

Y

YAxisMax property	Sets maximum value for Y axis
YAxisMin property	Sets minimum value for Y axis
YAxisMinorTicks property	Enables minor tick marks on Y axis
YAxisPos property	Sets position of Y axis in 2D graph with one Y axis
YAxisStyle property	Sets method for determining scale and range of Y axis
YAxisTicks property	Sets number of tick marks on Y axis
YAxisUse property	Determines Y axis to which other axis properties apply
YLabel() run-time property	Alternative way to set label text for Y axis at run time
YLabelText property	Sets label text for Y axis

Z

ZAxisMax property	Sets maximum value for Z axis of True3D scatter graph
ZAxisMin property	Sets minimum value for Z axis of True3D scatter graph
ZAxisStyle property	Sets method for determining scale and range of Z axis in True3D scatter graph
ZAxisTicks property	Sets number of tick marks on Z axis of True3D scatter graph
ZLabel() run-time property	Alternative way to set label text for Z axis of True3D graph at run time
ZLabelText property	Sets label text for Z axis of True3D graph
ZPos() run-time property	Alternative way to set independent Z values for True3D scatter graph at run time
ZPosData property	Sets independent Z values for True3D scatter graph

(About) design-time property

Displays Graph control's About box

Value

Read-only string

Description

The (About) design-time property displays the Graph control's About box, which provides contact information for Bits Per Second Ltd. and Pinnacle Publishing, Inc.

▲▼

Topic

(About)

Related

(Help)

AutoInc property

Automatically increments ThisPoint and ThisSet counters

Syntax

```
[form.]graph.AutoInc[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Off
1	On (default)

Description

The AutoInc property lets you set values for the Graph control's array-based properties consecutively, without manually incrementing the ThisPoint and ThisSet counters. The array-based properties include:

ColorData
DataFields
DataLabelText
EBarGraphDataMinus
EBarGraphDataPlus
EBarXPosDataMinus
EBarXPosDataPlus
ExtraData
GraphData
LabelText
LegendText
MapRefStrings
OverlayExtraData
OverlayGraphData
OverlayTrendSets
OverlayXPosData
PatternData
SymbolData
TrendSets
XPosData
XPosFields
YLabelText
ZLabelText
ZPosData

When you have only one data set (the NumSets property is set to 1), AutoInc increments the ThisPoint property (from 1 to the value of the NumPoints property) each time you set a value. With more data sets, AutoInc also updates the ThisSet property (from 1 to the value of the NumSets property) when you fill all the values for a set.

ThisPoint and ThisSet are reset to 1 whenever you turn AutoInc on or off. Also, when AutoInc is on, ThisPoint and ThisSet are reset to 1 when you switch from setting one array-based property to setting another.

Note that AutoInc updates ThisPoint and ThisSet only when you *set* values. When you *get* or *use* values, ThisPoint and ThisSet remain unchanged.

Example

Without AutoInc, you'd have to write for-next loops like these to update the ThisSet and ThisPoint values:

```
Graph1.ThisSet = 1
For I% = 1 to Graph1.NumSets
   Graph1.ThisPoint = 1
   For J% = 1 to Graph1.NumPoints
      Graph1.GraphData = J%*I%
      If Graph1.ThisPoint < Graph1.NumPoints Then
         Graph1.ThisPoint = Graph1.ThisPoint + 1
      End If
   Next J%
   If Graph1.ThisSet < Graph1.NumSets Then
      Graph1.ThisSet = Graph1.ThisSet + 1
   End If
NEXT I%
```

With AutoInc, this process is reduced to:

```
Graph1.DrawMode = 2
Graph1.AutoInc = 1
For I% = 1 To (Graph1.NumSets * Graph1.NumPoints)
   Graph1.GraphData = Graph1.ThisPoint * Graph1.ThisSet
Next I%
Graph1.DrawMode = 2
```

Keep in mind that Visual Basic won't let you use property names like ThisPoint or ThisSet as counters in For statements.

Topic

AutoInc

Related

ThisPoint
ThisSet
NumPoints
NumSets
IndexStyle

Backdrop property

Selects filename for graphing window's backdrop image

Syntax

```
[form.]graph.Backdrop[ = "filename"]
```

Data type

String

Description

The Backdrop property sets the filename for a graphic image (either bitmap or metafile, as specified by the BackdropStyle property) that will be read and displayed as the backdrop for an entire graphing window.

When you give the filename string, you can include a path to the file. If you don't specify a path, the Graph control searches the current directory. The appropriate file extension (.BMP or .WMF) is appended automatically according to the setting in BackdropStyle.

If the filename is invalid or the file doesn't exist, the Graph control ignores the request without reporting an error.

Backdrop files aren't stored as part of the Visual Basic form. Instead, the Graph control rereads the file whenever the form is opened.

The Graph control displays 256-color bitmaps in their full colors if your hardware allows.

If you're drawing or printing a graph in monochrome mode (with the DrawStyle or PrintStyle property set to monochrome), backdrops won't display.

Tip You can use the Graph control as a bitmap viewer to preview images. Set the GraphType property to 0.

▲▼

Topic

Backdrop

Related

BackdropStyle

Background

Background property page

DrawStyle

PrintStyle

BackdropStyle property

Sets type and presentation style of backdrop image

Syntax

```
[form.]graph.BackdropStyle[ = setting]
```

Data type

Integer (0-4, enumerated)

Settings

0	None (default)
1	Centered bitmap
2	Tiled bitmap
3	Stretched bitmap
4	Stretched metafile

Description

The BackdropStyle property sets the type of graphic image (bitmap or metafile) used for a graphing window's backdrop. It also determines the way the backdrop is presented--centered, tiled, or stretched.

Although you can choose a backdrop type without regard to the setting of a graph's DrawMode property, you'll get best performance if you use a metafile backdrop when *drawing* (DrawMode set to 2) and a bitmap backdrop when *blitting* (DrawMode set to 3).

▲▼

Topic

BackdropStyle

Related

Backdrop

Background

Background property page

DrawMode

Background property

Sets background color for graphic object specified by BackgroundUse

Syntax

[*form*.]*graph*.**Background**[= *setting*]

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The Background property sets the background color for the graphic object currently specified by the BackgroundUse property. This color is drawn as a rectangle behind the object.

If you use the default BackgroundUse setting (0), Background sets the background color of the entire graphing window.

▲▼

Topic

Background

Related

Background property page

BackgroundStyle

BackgroundUse

Palette

Foreground

ForegroundUse

BackgroundStyle property

Sets background style for graphic object specified by BackgroundUse

Syntax

[*form.*]*graph*.**BackgroundStyle**[= *setting*]

Data type

Integer (0-4, enumerated)

Settings

0	No background (default)
1	Frame
2	Drop shadow
3	Sculptured raised
4	Sculptured lowered

Description

The BackgroundStyle property lets you apply a background style to the graphic object specified by the BackgroundUse property. This style is applied to a rectangular area surrounding the object.

Styling options include frames, drop shadows, and "sculptured" looks (the object appears raised or lowered from the plane of the graph). If you use one of the sculptured styles (setting 3 or 4), set the background color to light gray (7) for best effect.

Topic

BackgroundStyle

Related

Background

Background property page

BackgroundUse

BackgroundUse property

Selects graphic object to which background colors and styles are applied

Syntax

```
[form.]graph.BackgroundUse[ = setting]
```

Data type

Integer (0-6, enumerated)

Settings

0	Entire graphing window (default)
1	Graph and axes
2	Graph title
3	Left title
4	Right title
5	Bottom title
6	Legend

Description

The BackgroundUse property selects the graphic object (graph, title, legend, etc.) to which the BackgroundStyle and Background properties are applied.

When you set BackgroundUse, the Graph control creates a rectangle around the selected object, then fills the rectangle with a background color or applies the style to it.

The default BackgroundUse setting (0) lets you use the Background property to set the background color of the entire graphing window. Note that this setting doesn't work with BackgroundStyle--if you set BackgroundStyle when BackgroundUse is set to 0, nothing happens.

Background colors and styles are actually arrays of seven values each (one for each object). The BackgroundUse setting indexes those arrays.

▲▼

Topic

BackgroundUse

Related

Backdrop
BackdropStyle
Background
Background property page
BackgroundStyle
Foreground
ForegroundUse

Bar2DGap property

Sets the gap between bars in 2D bar graphs

Syntax

```
[form.]graph.Bar2DGap[ = setting]
```

Data type

Integer (0 to 100)

Settings

Settings specify a percentage of the maximum possible space between adjacent bars in a 2D bar graph. By default a gap of 20% is placed between bars.

Description

Bar2DGap defines the space between adjacent bars in a 2D bar graph. The size of the space is expressed as a percentage of the maximum possible gap, which varies according to the size of the graphing window and the number of bars in the graph. A setting of 0 eliminates the gap altogether, placing the bars directly next to each other. A setting of 100 eliminates the bars altogether, making them infinitely thin.

▲▼

Topic

Bar2DGap

Related

GraphStyle
XAxisTicks

BottomTitle property

Specifies text string to place at bottom of graphing window

Syntax

[*form*.]*graph*.**BottomTitle**[= "*title*"]

Data type

String (up to 80 characters)

Description

The BottomTitle property specifies a text string of up to 80 characters that appears underneath a graph. Bottom titles are often used to explain or clarify a graph's X axis.

If you specify a title and it doesn't appear, it may be too long to display at the specified size. In that case, increase the size of the graphing window or use a shorter title.

You can introduce a line break by inserting the newline character -- CHR$(10) -- in the string. This forces a new line to start at the point where the newline character appears, overriding automatic word-wrap. Forcing a new line is likely to increase the area required for the text and may lead to text being omitted if there is not enough room for it. Since the newline character is unprintable, it cannot be inserted when typing text; it can only be inserted with code.

Note BottomTitle text always appears at the bottom of the graphing window. If, for example, you set BottomTitle for a bar graph with vertical bars and then change to horizontal bars (switching the X and Y axes), the text for BottomTitle will remain at the bottom.

▲▼

Topic

BottomTitle

Related

Titles property page
GraphTitle
LeftTitle
RightTitle
BackgroundStyle
FontName
FontStyle
FontSize
NewLineChar

CageEdgeColor property

Sets color of edges of True3D graph cage

Syntax

```
[form.]graph.CageEdgeColor[ = setting]
```

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The CageEdgeColor property sets the color of the edges of the cage for any True3D graph.

You can use any valid color index as the CageEdgeColor setting. By default, the Graph control automatically selects an edge color calculated for maximum contrast with the background color of the graphing window.

▲▼

Topic

CageEdgeColor

Related

3d property page

CageWallColor

CageStyle

CageFlip

Palette

True3D

CageFlip property

Allows back and side walls of True3D graph cage to switch to opposite edge for better viewing

Syntax

```
[form.]graph.CageFlip[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Off (default)
1	On--flip back and side walls
2	On--flip back wall only

Description

The CageFlip property enables cage "flips" for True3D graphs. If the user selects a viewing angle that would normally cause part of the graph to be obscured, the back or side wall "flips" to the opposite edge of the cage, letting you see the whole graph.

If you set CageFlip to 2, cage flips are enabled only for the back wall of the cage. The side wall always represents the cage's left edge, helping to orient the viewer.

▲▼

Topic

CageFlip

Related

3d property page
CageEdgeColor
CageWallColor
CageStyle
True3D
True3DDepth

CageStyle property

Sets style of True3D graph cage walls (thick or thin)

Syntax

```
[form.]graph.CageStyle[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Thick cage walls (default)
1	Thin cage walls

Description

The CageStyle property specifies whether a True3D graph cage is drawn with thick or thin walls.

▲▼

Topic

CageStyle

Related

3d property page

CageEdgeColor

CageWallColor

CageFlip

True3D

CageWallColor property

Sets color of True3D graph cage walls

Syntax

```
[form.]graph.CageWallColor[ = setting]
```

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The CageWallColor property sets the color of the back wall, side wall, and floor of a True3D graph cage.

You can use any valid color index as the CageWallColor setting. By default, the Graph control automatically selects a wall color calculated for maximum contrast with the background color of the graphing window.

▲▼

Topic

CageWallColor

Related

3d property page
CageEdgeColor
CageStyle
CageFlip
Palette
True3D

ClipGraph property

Enables clipping of a graph within its axes

Syntax

```
[form.]graph.ClipGraph[ = setting ]
```

Data type

Integer (0-1 enumerated)

Settings

0	Off
1	On

Description

When the axes in a 2D graph have user-defined scale and range, the axes are drawn independently of the data values. As a result, the graph may extend beyond the boundaries of the axes. You can mask areas beyond the boundaries of the axes by setting ClipGraph = 1 (On).

This property does not apply to 3D graphs.

▲▼

Topic

ClipGraph

Related

Design property page
XAxisStyle
YAxisStyle

Color() run-time property

Alternative way to set colors of graph elements at run time

Syntax

`[form.][graph.]Color(index%)[ = setting]`

Parameter

index% Element number of array normally set by ColorData property

Data type

Integer (0-127, enumerated)

Settings

Same as for ColorData property

Description

The Color() run-time property specifies the colors used in displaying the elements (lines, bars, pie slices, and so on) of a graph. Color() is a run-time alternative to the ColorData property.

Color values are stored as a one-dimensional array. Unlike ColorData, Color() doesn't require you to use the ThisPoint or ThisSet property to index the color array. Instead, you provide the element (point or set) number as your Color() parameter.

Example

- If you have a bar graph with five data points and you want to set the third bar (point) to yellow, you can set up your code in two ways:
- If you use the ColorData property, you set the ThisPoint property first:

```
Graph1.ThisPoint = 3
Graph1.ColorData = 14
```

If you use the Color() property, you don't have to set ThisPoint:

```
Graph1.Color(3) = 14
```

▲▼

Topic

Color()

Related

ColorData
AutoInc
ThisPoint
ThisSet
Palette
DataReset
IndexStyle

ColorData property

Sets graph colors

Syntax

```
[form.]graph.ColorData[ = setting]
```

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The ColorData property specifies the colors used in displaying the elements (lines, bars, pie slices, and so on) of a graph.

For most graph types, ColorData sets a color for each data *set*. However, for pie charts, bubble graphs, and bar graphs having only one data set, ColorData sets a color for each data *point*.

Once you've chosen a color for one data set or point, you should select colors for the remaining sets or points. Otherwise, those that haven't been assigned a color will display in black.

ColorData is a one-dimensional array-based property indexed by either the ThisPoint or ThisSet property (see the IndexStyle property for details on the Graph control's indexing of array-based properties). You can use the AutoInc property to automatically increment the ThisPoint or ThisSet counter each time you set a ColorData element.

▲▼

Topic

ColorData

Related

AutoInc

Color()

DataReset

IndexStyle

Markers property page

Palette

ThisPoint

ThisSet

CtlVersion property

Gives version number of current copy of Graph control

Syntax

variable$ = *graph*.**CtlVersion**

Data type

String

Description

CtlVersion is a read-only property that gives you the version number of the current copy of the Graph control.

CurveOrder property

Sets order of polynomial used in curve fitting or range of points for moving averages

Syntax

[*form*.]*graph*.**CurveOrder**[= *setting*]

Data type

Integer

Settings

Polynomials

Valid settings are integers from 1 to 9 (default is 2).

- A setting of 1 produces a straight line.
- A setting 1 less than the NumPoints property setting results in a curve that passes through every point.

Moving averages

Valid settings are all values greater than 0 and less than the total number of points in the graph (default is 2).

Description

The CurveOrder property is used with three CurveType values:

CurveType		**What CurveOrder does**
0	Variable-order polynomial	Sets order of polynomial used for curve fitting
11 *or* 12	Moving average mid *or* Moving average end	Sets range of points over which moving averages are averaged, beginning with first point

CurveOrder applies only to five graph types: line, 2D scatter, high-low-close, open-high-low-close, and box-whisker (parametric data only).

▲▼

Topic

CurveOrder

Related

CurveSteps

CurveType

Trends property page

CurveSteps property

Sets curve resolution

Syntax

[*form*.]*graph*.**CurveSteps**[= *setting*]

Data type

Integer (1-1000)

Settings

Valid settings are integers from 1 to 1000 (default is 50).

Description

The CurveSteps property sets the resolution, or granularity, of the curve displayed in curve fitting. The "curve" that's drawn is actually a series of straight lines joining points on the mathematical curve selected with the CurveType property.

The higher the CurveSteps value, the smoother the curve appears. Note, however, that smoother curves are drawn more slowly than rougher ones.

On a VGA display, the default CurveSteps setting of 50 generally creates a smooth-looking curve at a high drawing speed. However, if you're using a very high-resolution output device, you may want to increase the CurveSteps setting for better smoothness.

Note that spline curves (CurveType setting 10) may require a CurveSteps value up to 10 times the number of points in the graph.

CurveSteps applies only to five graph types: line, 2D scatter, high-low-close, open-high-low-close, and box-whisker (parametric data only).

▲▼

Topic

CurveSteps

Related

CurveOrder

CurveType

Trends property page

CurveType property

Selects mathematical formula for curve

Syntax

```
[form.]graph.CurveType[ = setting]
```

Data type

Integer (0-12, enumerated)

Settings

0	Variable-order polynomial (default)	
1	Logarithmic	y = a + b * log(x)
2	Exponential	y = a * exp(b * x)
3	Exponential	y = a * x * exp(-b * x)
4	Power y = a * (x ^ b)	
5	Inverse	y = a + b / x
6	Inverse	y = a / (b + x)
7	Inverse	y = 1 / (a + b * x)
8	Inverse	y = x / (a * x + b)
9	Inverse	y = 1 / (a + b * x) ^ 2
10	Spline fit through all points	
11	Moving average plotted at midpoint of averaged group	
12	Moving average plotted at end point of averaged group	

Description

The CurveType sets the mathematical formula used in curve fitting. You can enable curve fitting using the LineStats property or, for scatter graphs, the GraphStyle property.

Graphics Server draws curves in three ways:

- Curve types 0 through 9 use least-squares regression to minimize the error between the curve and the fitted points.
- The spline fit (10) uses cubic splines to produce a smooth fit through all points.
- The moving-average types (11 and 12) take an unweighted average over a group of points and plot that average at the midpoint or end point of the group.

Some of the mathematical models are invalid for x = 0. If the curve is set to pass through the X origin (the Y axis), no curve will be drawn. If you want to experiment with each of the possible curves, enter some non-zero values using the XPosData property.

The curve's thickness and pattern are controlled by the ThickLines, PatternedLines, and PatternData properties. The CurveSteps property determines the granularity of the curve.

Three curve types--variable-order polynomial (0), moving-average mid (11), and moving-average end (12)--also require you to set the CurveOrder property.

CurveType applies only to five graph types: line, 2D scatter, high-low-close, open-high-low-close, and box-whisker (parametric data only).

▲▼

Topic

CurveType

Related

CurveOrder
CurveSteps
GraphStyle
LineStats
PatternData
PatternedLines
ThickLines
Trends property page
XPosData

Data() run-time property

Alternative way to set graph data values at run time

Syntax

```
[form.][graph.]Data(index%)[ = value]
```

Parameter

index% Element (point) number of array normally set by GraphData property

Data type

Numeric (integer, long, single, double)

Description

The Data() run-time property sets the actual data values you want to graph. It's a run-time alternative to the GraphData property.

Graph data values are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But Data(), unlike GraphData, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your Data() parameter.

Example

If you have a graph with multiple data sets and you want to set the value of the third data point in the second data set to 10, you can set up your code in two ways:

- If you use the GraphData property, you set the ThisSet and ThisPoint properties first:

  ```
  Graph1.ThisSet = 2
  Graph1.ThisPoint = 3
  Graph1.GraphData = 10
  ```

- If you use the Data() property, you don't have to set ThisPoint:

  ```
  Graph1.ThisSet = 2
  Graph1.Data(3) = 10
  ```

Speeding up data transfer

The VBX and OCX use dynamic arrays that grow as values are assigned to individual elements. When loading data into large arrays, a considerable time-saving can be achieved by assigning a value to the highest element in the array before entering a loop. This has the effect of reserving memory for the whole array. Hence there is no overhead in memory reorganization during the loading process.

For example with a 1000-element array, assign a dummy value to the 1000th element:

```
Graph1.Data(1000) = 0
```

before going into the For ... Next loop. This technique will save time when loading any array property.

▲▼

Topic

Data()

Related

GraphData
AutoInc
ThisPoint
ThisSet
XPosData
DataReset

DataField() run-time property

Selects fields from a bound database

Syntax

[*form.*]*graph*.**DataField(*index*%)**[= *data*]

Parameter

index% Element number of array normally set by the DataFields property.

Data type

String

Description

The DataField() property is a run-time alternative to the DataFields property.

Two lines of code are needed to read or set the DataFields property. You must first point to an element of the DataFields array by setting ThisPoint. Then you set or get DataFields.

The DataField() property lets you skip the first step. Instead of pointing to an element with ThisPoint, you provide the element (set) number as your DataField() parameter.

▲▼

Topic

DataField()

Related

DataFields

DataSource

DrawMode

DataFields property

Binds a graph to one or more fields in a database

Syntax

```
[form.]graph.DataFields[ = "field name"]
```

Data type

String

Description

DataFields is a one-dimensional array-based property indexed by ThisPoint. DataFields identifies which fields of a bound database should be graphed.

For information on binding a graph to a database, see the DataSource property.

When the AutoInc property is set to 1 (On), the NumSets property automatically adjusts to the number of field names loaded into the DataFields property. You can increase the number of data sets by setting more elements of the DataFields array. NumSets increments each time a new field name is added.

To decrease the number of data sets, you must first clear the DataFields array by setting DataReset = 20 before setting the DataFields array. This is because the DataFields array works just as other array properties do. Setting an element to "" is not the same as removing it. To clear any array property, you must set DataReset.

▲▼

Topic

DataFields

Related

Bound data property page

DataField()

DataReset

DataSource

DrawMode

LabelField

LegendField

DataLabelFormat property

Defines the text formatting of numeric data labels

Syntax

```
[form.]graph.DataLabelFormat[ = data]
```

Data type

String

Setting

A valid numeric format expression.

Description

The DataLabelFormat property defines a template for formatting numeric labels. It is not applied to user-defined text labels.

Just as is the case with the Visual Basic function Format$(), the characters within a DataLabelFormat format string determine how the number is formatted. For details, see numeric formats.

▲▼

Topic

DataLabelFormat

Related

Labels property page
DataLabels
LabelXFormat
LabelYFormat
LabelZFormat
Numeric formats

DataLabel() run-time property

Alternative way to set text of data labels at run time

Syntax

```
[form.][graph.]DataLabel(index%)[ = "text"]
```

Parameter

index% Element number of array normally set by DataLabelText property

Data type

String (up to 80 characters)

Description

The DataLabel() run-time property sets the text of data labels. It's a run-time alternative to the DataLabelText property.

Data label text is stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But DataLabel(), unlike DataLabelText, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your DataLabel() parameter.

Example

If you have a graph with multiple data sets and you want to place a data label ("DLabel") on the third data point in the second data set, you can set up your code in two ways:

- If you use the DataLabelText property, you set the ThisSet and ThisPoint properties first:

```
Graph1.DataLabels = 1     'Enables data labels
.
.
Graph1.ThisSet = 2
Graph1.ThisPoint = 3
Graph1.DataLabelText = "DLabel"
```

- If you use the DataLabel() property, you don't have to set ThisPoint:

```
Graph1.DataLabels = 1     'Enables data labels
.
.
Graph1.ThisSet = 2
Graph1.DataLabel(3) = "DLabel"
```

▲▼

Topic

DataLabel()

Related

DataLabelText
DataLabels
DataReset

DataLabels property

Enables and disables data labels

Syntax

```
[form.]graph.DataLabels[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Labels off (default)
1	Labels on, colored according to the setting of the Foreground property
2	Labels on, colored as the associated data group

Description

The DataLabels property enables data labels, which are text or numeric labels attached to each point of a graph.

You can add data labels to all 2D graph types except pie charts (which have their own labeling scheme) and time series graphs. Data labels aren't available for 3D graphs.

Text labels

To set the text for graph labels, pass an array to the DataLabelText property.

Numeric labels

If you haven't defined your own labels using the DataLabelText property, the Graph control derives numeric labels from the associated data values. To switch from text labels to numeric ones, you have to reset the DataLabelText array first (set the DataReset property to 18).

Numeric data labels are normally the values of the GraphData property for each point. The exceptions are as follows:

- In **high-low-close**, **open-high-low-close**, and **candlestick** graphs, data labels show the "close" values.
- In **box-whisker graphs,** data labels show the "median" values.

▲▼

Topic

DataLabels

Related

Labels property page
DataLabelFormat
DataLabelText
DataReset
Foreground

DataLabelText property

Sets text of data labels

Syntax

```
[form.]graph.DataLabelText[ = "data label text"]
```

Data type

String

Description

The DataLabelText property sets the text for data labels. It's a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. The number of labels varies by graph type:

- **Box-whisker, candlestick, high-low-close,** and **open-high-low-close graphs** have one label for each compound symbol. The total number of labels is equal to the setting of the NumPoints property.
- **Other graph types supporting data labels** have one label for each data point in each data set. The total number of labels is equal to the product of the NumSets and NumPoints property settings.

DataLabelText is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint values as you enter text for each data label.

▲▼

Topic

DataLabelText

Related

Labels property page

DataLabels

DataLabel()

DataReset

DataReset property

Resets any or all array-based properties for graph

Syntax

[*form*.]*graph*.**DataReset**[= *setting*]

Data type

Integer (0-26, enumerated)

Settings

0	No properties (default)
1	GraphData
2	ColorData
3	ExtraData
4	LabelText
5	LegendText
6	PatternData
7	SymbolData
8	XPosData
9	All array-based properties subject to DataReset
10	Font properties (FontFamily, FontName, FontSize, FontStyle)
11	YLabelText
12	ZLabelText
13	ZPosData
14	EBarGraphDataMinus
15	EBarGraphDataPlus
16	EBarXPosDataMinus
17	EBarXPosDataPlus
18	DataLabelText
19	OverlayGraphData
20	DataFields

Note: Settings listed below are not available with the VBX interface.

21	MapRefStrings
22	XPosFields
23	OverlayXPosData
24	OverlayExtraData
25	TrendSets
26	OverlayTrendSets

Description

The DataReset property resets, or clears, the values of the specified array-based property. When you reset a property, it returns to its original empty state--each element is set to the default value.

Remember to set DataReset to 9 to reset all of the array-based properties subject to the DataReset property.

DataSource design-time property

Binds a graph to a Data control

Value

Read/write string at design time; unavailable at run time.

Description

The DataSource property specifies a Data control as the source of data for a graph. This property can only be set at design time and only by using the Properties window. It is not available on the Graph control's property pages.

To bind a graph to a database at run time, you must prepare the connection at design time. First hook up the data side of the pipe by placing a Data control on the form and setting its DatabaseName and RecordSource properties. Although you can change these properties at run time, no connection will be made unless you begin with valid settings at design time.

The next step is to hook up the graph side of the pipe. Do this by specifying the Name property of the Data control in the DataSource property of Graph control. This must be done at design time and cannot be changed at run time.

Note: If you later change the Name property of the Data control without changing the DataSource property of the Graph control, you will break the connection. If you change one, take care to change the other as well.

To complete the graph's connection to the database, set the DataFields property so that it specifies one or more valid field names in the Data control's record set. This step can be done either at design time or at run time.

Once the graph is bound to a database, each field specified by the DataFields property is graphed as a data *set*. The graph's NumSets property is set equal to the number of fields selected in the DataFields property. Each record in the database is graphed as a data *point*. The graph's NumPoints property is set equal to the number of records in the Recordset managed by the Data control.

The graph's connection to the database is "live" in the sense that whenever the Recordset managed by the Data control is updated or refreshed, the graph is redraw automatically. For additional information, see "Special considerations for bound graphs" under the entry for DrawMode .

Data from the DataSource is graphed only when the GraphData array is empty. If you assign values to the GraphData array, the Graph control will assume you want to graph those values rather than what it receives from the Data control. To return to graphing data from the Data control, clear the GraphData array by setting DataReset to 1.

▲▼

Topic

DataSource

Related

Bound data property page

DataField()

DataFields

DataReset

DrawMode

ISVDataSource

DrawMode property

Sets drawing mode for Graph control

Syntax

[*form.*]*graph*.**DrawMode**[= *setting*]

Data type

Integer (0-11, enumerated)

Settings

0	No action	The control stays blank and the graph doesn't appear. When you want to show the graph, reset DrawMode to 2.
1	Clear	No graph appears, but the background of the control is set to the color specified by the Background property. Also, any GraphCaption text appears in the center of the control.
2	Draw (default)	This setting operates in different ways at design time and run time: At design time, the graph is redrawn every time you change a property. At run time, the graph is redrawn every time you reset DrawMode to 2.
3	Blit	After a brief pause, the graph appears all at once. Graphics Server builds a hidden bitmap of the graph, then displays it using the Windows API function BitBlt. This mode is useful if you want to draw a graph and then update it with changing data--the graph appears to change instantaneously.
4	Copy	The image of the graph is copied to the Clipboard. If DrawMode was previously set to 3 (blit), the image is in bitmap format. Otherwise, it's in metafile format.
5	Print	This setting operates in different ways at design time and run time: At design time, the graph itself (though not the rest of the form) is printed. At run time, the graph is printed along with text and other graphics according to the settings of the PrintInfo() run-time property. If you don't use PrintInfo(), only the graph is printed. The PrintStyle property lets you choose whether to print in monochrome or color, with a border or without.
6	Write	An image of the graph is written to disk. You must first provide a name for the file by setting the ImageFile property, and you must prepare the

		image by setting DrawMode to either 2 or 3 immediately before setting DrawMode to 6. To save an image in Windows metafile format, set ImageFile = "..\filename.wmf". Set DrawMode = 2 and then DrawMode = 6. To save an image as BMP, JPG or GIF, set ImageFile with a path, file name and appropriate file extension. Set DrawMode = 3 (blit), followed by DrawMode = 6.
7	Rotate (True3D graphs only)	A portion of a True3D graph--including the graph itself and the cage--is redrawn. This setting lets Graphics Server update the graph quickly, without the overhead of recalculating scales and redrawing titles and legends.
8	Update time series graph	Every time you add one or more data points to a time series graph, you have to set DrawMode to 8 to redraw the graph with the new point(s).
9	Load properties	Restores properties of a previously saved graph. Before setting DrawMode = 9, you must first specify the name of the graph template file by setting the GraphFile property, and the name of the graph within the template file by setting the GraphName property. After loading a graph it must be redrawn by setting DrawMode = 2 or 3.
10	Save properties excluding data	Saves the current property settings to a graph template file. Only the properties defining the appearance of the graph, including any text, are saved. Before setting DrawMode = 10, you must first specify the name of the file by setting the GraphFile property and the name of the graph within the file by setting the GraphName property.
11	Save properties including data	Saves the current property settings, including data (ErrorBar, ExtraData, GraphData, OverlayData, OverlayExtraData, OverlayTrendSets, OverlayXPos Data, TrendSets, XPosData and ZPosData). You must first specify the name of the file by setting the GraphFile property and the name of the graph within the file by setting the GraphName property.

Note: Settings listed below are not available with the VBX interface.

12	Write map file	Writes an image map to the file specified by MapFile in the format specified by MapFormat . The Hot property must be set to 1 (On) prior to setting DrawMode to 12.
13	Append map file	Appends an image map to the file specified by MapFile in the format specified by MapFormat. The Hot property must be set to 1 (On) prior to setting DrawMode to 13.

Description

The DrawMode property sets the drawing mode for the Graph control. It determines the output destination--screen, clipboard, printer, image file, image map, template file. In the case of template files, DrawMode is also the means by which graph definitions are loaded.

Redrawing graphs

At **design time**, the DrawMode value is automatically reset to 2 whenever you change a property value. This causes the graph to be redrawn so you can see the effect of the change.

At **run time**, graphs are redrawn only when you explicitly set DrawMode to 2 or 3. This way, you can change as many other properties as you want before redrawing.

Permanent vs. transient settings

DrawMode settings 0 through 3 are **permanent**. They are the only settings retained when a project is saved at design time. When you switch from design to run mode, these settings will determine how the graph is first displayed. They are also the only settings retained in the property at run time; any attempt to read the value of DrawMode will always return 0, 1, 2, or 3, even if the last value set was greater than 3.

DrawMode settings 4 through 13 are **transient**. After the actions these settings trigger (copying to the Clipboard, printing, or writing to disk) are completed, DrawMode returns to the last permanent setting it held. For example, if you change the DrawMode setting from 2 to 6, the graph is written to a file once, then the setting automatically returns to 2. If your code reads the value of DrawMode, it will return 2, not 6.

Special considerations for bound graphs

The distinction between transient and permanent settings has special implications for bound graphs. For a bound graph, if the permanent setting of DrawMode is 2 (Draw), the graph is automatically redrawn whenever you use the Data control's Refresh or UpdateRecord methods. The graph is also redrawn whenever you assign a new Recordset to the Data control or use the Update method to change records in the current Recordset.

If for some reason you do not want the graph redrawn, then set DrawMode to 0 (No action) or 1 (Clear) prior to calling Refresh. When you are ready to show the graph, set DrawMode back to 2.

Time series graphs

Time series graphs are built progressively over time by adding data points. Every time DrawMode is set to 8, another point is added to the graph at whatever Y value is specified in the first (and, for this graph type, the only) element of GraphData. New points are added at the X origin, on the right end of the axis. Previous points scroll to the left. When the number of points on display exceeds NumPoints, the oldest point is discarded.

The code below draws a time series graph with two data sets, one for the sine of an angle, another for the cosine. New data is added in a timer event.

```
Const Frequency = 6  'Sets sine frequency
Const Amplitude = 100 'Sets wave amplitude
Const XLen = 50       'Sets number of points visible
Dim Degrees As Double

Private Sub Form_Load()
With Graph1
    .GraphType = 22  'Time series
    .GraphStyle = 1  'Continuous lines
    .NumSets = 2
    .NumPoints = XLen 'Defines length of X axis
    .AutoInc = 0     'Off

    'Axes
    .TickEvery = XLen / 10 'X axis-show every 5th tick
    .Labels = 3            'Label Y axis only
    .YAxisStyle = 2        'Y axis-user defined scale
    .YAxisMax = Amplitude
    .YAxisMin = -Amplitude
    'Set number of ticks on positive and negative arms
    .YAxisTicks = Amplitude / 25

    'Legend
    .LegendPos = 7  'Bottom right
    .LegendSize = 23
    .Legend(1) = "Sine"
    .Legend(2) = "Cosine"

    'Graph fills entire form
    .Top = Me.ScaleTop
    .Left = Me.ScaleLeft
    .Width = Me.ScaleWidth
    .Height = Me.ScaleHeight
End With
Degrees = 0         'Initialize global variable
Timer1.Interval = 1 'Start timer
End Sub

Private Sub Timer1_Timer()
'Always load newest value into ThisPoint = 1, for each
'data set being modified, and then redisplay
Degrees = (Degrees + Frequency) Mod 360
With Graph1
    .ThisSet = 1
    .Data(1) = GSGetSin(Degrees) * Amplitude
    .ThisSet = 2
    .Data(1) = GSGetCos(Degrees) * Amplitude
    'Special DrawMode only for time series
    .DrawMode = 8
End With
End Sub
```

▲▼

Topic

DrawMode

Related

Background
Design property page
GraphFile
GraphName
ImageFile
MapFile
MapFormat
PrintInfo
PrintStyle
System property page
True3D

DrawStyle property

Sets drawing style to color or monochrome

Syntax

```
[form.]graph.DrawStyle[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Monochrome
1	Color (default)

Description

The DrawStyle property determines whether graphs are drawn in color or monochrome.

When you set DrawStyle to monochrome (0), the background goes white and all colors go black, with varying patterns, symbols, and line thicknesses supplied to differentiate data sets and points from each other. If you haven't already set the PatternData and SymbolData properties, the default settings are used.

▲▼

Topic

DrawStyle

Related

Design property page
PrintStyle

EBarDataMinus() run-time property

Alternative way to set error bar minus values at run time

Syntax

[*form*.][*graph*.]**EBarDataMinus(*index*%)**[= *value*]

Parameter

index% Element number of array normally set by EBarGraphDataMinus property

Data type

Numeric (integer, long, single, double)

Description

The EBarDataMinus() run-time property sets the minus values for error bars. It's a run-time alternative to the EBarGraphDataMinus property.

Error bar minus values are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But EBarDataMinus(), unlike EBarGraphDataMinus, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your EBarDataMinus() parameter.

Example

If you have a graph with multiple data sets and you want to set the minus error bar value for the third data point in the second data set to -10, you can set up your code in two ways:

- If you use the EBarGraphDataMinus property, you set the ThisSet and ThisPoint properties first:

  ```
  Graph1.ThisSet = 2
  Graph1.ThisPoint = 3
  Graph1.EBarGraphDataMinus = 10
  ```

- If you use the EBarDataMinus() property, you don't have to set ThisPoint:

  ```
  Graph1.ThisSet = 2
  Graph1.EBarDataMinus(3) = 10
  ```

▲▼

Topic

EBarDataMinus()

Related

DataReset
EBarGraphDataMinus
EBarDataPlus()
Error Bar property page

EBarDataPlus() run-time property

Alternative way to set error bar plus values at run time

Syntax

`[form.][graph.]`**`EBarDataPlus(index%)`**`[ = value]`

Parameter

index% Element number of array normally set by EBarGraphDataPlus property

Data type

Numeric (integer, long, single, double)

Description

The EBarDataPlus() run-time property sets the plus values for error bars. It's a run-time alternative to the EBarGraphDataPlus property.

Error bar plus values are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But EBarDataPlus(), unlike EBarGraphDataPlus, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your EBarDataPlus() parameter.

For an illustration of the different ways to set error bar properties at design time and run time, see the Example under the entry for the EBarDataMinus() run-time property.

▲▼

Topic

EBarDataPlus()

Related

DataReset
EBarGraphDataPlus
EBarDataMinus()
Error Bar property page

EBarGraphDataMinus property

Sets user-defined minus values for error bars, based on GraphData

Syntax

```
[form.]graph.EBarGraphDataMinus[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The EBarGraphDataMinus property sets user-defined values for the lengths of minus (negative) error bars, based on the values of the GraphData property. It's used only when the EBarSource property is set to 5 (user-defined).

In horizontal bar graphs, EBarGraphDataMinus bars are drawn horizontally from the plotted points; in all other graph types allowing error bars, they're drawn vertically.

You can set EBarGraphDataMinus to any numeric value. Negative values are treated as positives. The default setting is 0, meaning that no error bar is drawn in the minus (negative) direction unless you specifically set a value for that point.

EBarGraphDataMinus is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint counters as you enter values.

▲▼

Topic

EBarGraphDataMinus

Related

DataReset
EBarDataMinus()
EBarGraphDataPlus
EBarSource
EBarStyle
EBarUse
EBarValue
EBarXPosDataMinus
EBarXPosDataPlus
Error Bar property page

EBarGraphDataPlus property

Sets user-defined plus values for error bars, based on GraphData

Syntax

`[form.]graph.EBarGraphDataPlus[ = data]`

Data type

Numeric (integer, long, single, double)

Description

The EBarGraphDataPlus property sets user-defined values for the lengths of plus (positive) error bars, based on the values of the GraphData property. It's used only when the EBarSource property is set to 5 (user-defined).

In horizontal bar graphs, EBarGraphDataPlus bars are drawn horizontally from the plotted points; in all other graph types allowing error bars, they're drawn vertically.

You can set EBarGraphDataPlus to any numeric value. Negative values are treated as positives. The default setting is 0, meaning that no error bar is drawn in the plus (positive) direction unless you specifically set a value for that point.

EBarGraphDataPlus is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint counters as you enter values.

▲▼

Topic

EBarGraphDataPlus

Related

DataReset
EBarDataPlus()
EBarGraphDataMinus
EBarSource
EBarStyle
EBarUse
EBarValue
EBarXPosDataMinus
EBarXPosDataPlus
Error Bar property page

EBarSource property

Enables error bars and sets method for determining bar length

Syntax

[*form*.]*graph*.**EBarSource**[= *setting*]

Data type

Integer (0-5, enumerated)

Settings

0 No error bars (default)
1 Fixed--error bar length is equal to the setting of the EBarValue property
2 Percentage--error bar length is a percentage (set by the EBarValue property) of the data value
3 Standard deviation--error bar length is the standard deviation of all data points multiplied by the EBarValue setting
4 Standard error--error bar length is the standard error of all data points
5 User-defined--error bar length is set for each data point by paired EBarGraphDataMinus/EBarGraphDataPlus or EBarXPosDataMinus/EBarXPosDataPlus properties

Description

The EBarSource property enables error bars and lets you set the method by which the Graph control determines error bar length.

Error bars--which are available for 2D bar, 2D line, and 2D scatter graphs--extend from plotted points (either vertically or horizontally) and indicate a range of error or uncertainty in the values.

In scatter graphs, error bars can be based on the values of the GraphData property, the XPosData property, or both. In bar and line graphs, error bars are always based on GraphData.

Error bars extend in the positive direction, the negative direction, or both, according to the setting of the EBarStyle property.

Note If you're accustomed to the graphing conventions of Microsoft Excel, keep in mind that Graphics Server's standard-deviation error bars (EBarSource setting 3) are drawn from each data point, not from a mean line as in Excel.

▲▼

Topic

EBarSource

Related

EBarGraphDataMinus
EBarGraphDataPlus
EBarStyle
EBarUse
EBarValue
EBarXPosDataMinus
EBarXPosDataPlus
Error Bar property page

EBarStyle property

Sets style for error bars--plus, minus, or both

Syntax

```
[form.]graph.EBarStyle[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Both plus and minus error bars (default)
1	Minus error bars only
2	Plus error bars only

Description

The EBarStyle property sets the style for error bars--plus (positive direction), minus (negative direction), or both. Error bars are enabled by the EBarSource property.

▲▼

Topic

EBarStyle

Related

EBarGraphDataMinus
EBarGraphDataPlus
EBarSource
EBarUse
EBarValue
EBarXPosDataMinus
EBarXPosDataPlus
Error Bar property page

EBarUse property

Specifies whether error bar options in scatter graphs apply to vertical or horizontal bars

Syntax

```
[form.]graph.EBarUse[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Settings apply to vertical bars based on GraphData property (default)
1	Settings apply to horizontal bars based on XPosData property (allowed for 2D scatter graphs only)

Description

The EBarUse property selects the type of error bars--GraphData or XPosData--to which other error bar properties (EBarSource, EBarStyle, and EBarValue) are applied.

The only graph type that allows XPosData bars (EBarUse setting 1) is the 2D scatter graph. If you choose, you can have both XPosData and GraphData bars in a scatter graph, in which case you'll have to toggle the EBarUse setting to set options for the two kinds of bars. For all other graph types that allow error bars, EBarUse should always be set for GraphData bars (0).

▲▼

Topic

EBarUse

Related

EBarGraphDataMinus

EBarGraphDataPlus

EBarSource

EBarStyle

EBarValue

EBarXPosDataMinus

EBarXPosDataPlus

Error Bar property page

EBarValue property

Sets value used to calculate length of error bars

Syntax

[*form*.]*graph*.**EBarSource**[= *setting*]

Data type

Numeric (positive integer, long, single, double)

Settings

Valid settings vary according to the setting of the EBarSource property:

EBarSource setting		**Values**
0	No error bars	n/a
1	Fixed	Positive values greater than or equal to 0 (default is 0), representing the actual length of each bar
2	Percentage	Positive values greater than or equal to 0 (default is 100), representing a percentage of the data value
3	Standard deviation	Positive values greater than or equal to 0 (default is 1.0), representing a multiplier for the standard deviation
4	Standard error	n/a
5	User-defined	n/a

Description

The EBarValue property sets the value that the Graph control uses to calculate the lengths of error bars. The way the EBarValue setting is used depends on the setting of the EBarSource property.

▲▼

Topic

EBarValue

Related

EBarGraphDataMinus

EBarGraphDataPlus

EBarSource

EBarStyle

EBarUse

EBarXPosDataMinus

EBarXPosDataPlus

Error Bar property page

EBarXPosDataMinus property

Sets user-defined minus values for error bars, based on XPosData

Syntax

`[`*form*`.]`*graph*`.`**EBarXPosDataMinus**`[ = `*data*`]`

Data type

Numeric (integer, long, single, double)

Description

The EBarXPosDataMinus property sets user-defined values for the lengths of minus (negative) error bars, based on the values of the XPosData property. It's used only with 2D scatter graphs and only when the EBarSource property is set to 5 (user-defined). EBarXPosDataMinus error bars are drawn horizontally.

You can set EBarXPosDataMinus to any numeric value. Negative values are treated as positives. The default setting is 0, meaning that no error bar is drawn in the minus (negative) direction unless you specifically set a value for that point.

EBarXPosDataMinus is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint counters as you enter values.

▲▼

Topic

EBarXPosDataMinus

Related

DataReset
EBarUse
EBarSource
EBarStyle
EBarValue
EBarXPosDataPlus
EBarGraphDataMinus
EBarGraphDataPlus
EBarXPosMinus()
Error Bar property page

EBarXPosDataPlus property

Sets user-defined plus values for error bars, based on XPosData

Syntax

```
[form.]graph.EBarXPosDataPlus[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The EBarXPosDataPlus property sets user-defined values for the lengths of plus (positive) error bars, based on the values of the XPosData property. It's used only with 2D scatter graphs and only when the EBarSource property is set to 5 (user-defined). EBarXPosDataPlus error bars are drawn horizontally.

You can set EBarXPosDataPlus to any numeric value. Negative values are treated as positives. The default setting is 0, meaning that no error bar is drawn in the plus (positive) direction unless you specifically set a value for that point.

EBarXPosDataPlus is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint counters as you enter values.

▲▼

Topic

EBarXPosDataPlus

Related

DataReset
EBarGraphDataMinus
EBarGraphDataPlus
EBarSource
EBarStyle
EBarUse
EBarValue
EBarXPosDataMinus
EBarXPosPlus()
Error Bar property page

EBarXPosMinus() run-time property

Alternative way to set EBarXPosDataMinus property values at run time

Syntax

[*form*.][*graph*.]**EBarXPosMinus(*index*%)**[= *value*]

Parameter

index% Element number of array normally set by EBarXPosDataMinus property

Data type

Numeric (integer, long, single, double)

Description

The EBarXPosMinus() run-time property sets user-defined values for the lengths of minus (negative) error bars, based on the values of the XPosData property. It's a run-time alternative to the EBarXPosDataMinus property.

The minus values for XPosData error bars are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But EBarXPosMinus(), unlike EBarXPosDataMinus, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your EBarXPosMinus() parameter.

For an illustration of the different ways to set error bar properties at design time and run time, see the example under the entry for the EBarDataMinus() run-time property.

▲▼

Topic

EBarXPosMinus()

Related

DataReset

EBarXPosDataMinus

EBarXPosPlus()

EBarXPosPlus() run-time property

Alternative way to set EBarXPosDataPlus property values at run time

Syntax

[*form*.][*graph*.]**EBarXPosPlus(*index*%)**[= *value*]

Parameter

index% Element number of array normally set by EBarXPosDataPlus property

Data type

Numeric (integer, long, single, double)

Description

The EBarXPosPlus() run-time property sets user-defined values for the lengths of plus (positive) error bars, based on the values of the XPosData property. It's a run-time alternative to the EBarXPosDataPlus property.

The plus values for XPosData error bars are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But EBarXPosPlus(), unlike EBarXPosDataPlus, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your EBarXPosPlus() parameter.

For an illustration of the different ways to set error bar properties at design time and run time, see the example under the entry for the EBarDataMinus() run-time property.

▲▼

Topic

EBarXPosPlus()

Related

DataReset

EBarXPosDataPlus

EBarXPosMinus()

Elevation property

Sets vertical viewing angle for True3D graphs

Syntax

```
[form.]graph.Elevation[ = setting]
```

Data type

Integer

Settings

Valid settings for 3D pie graphs are integers from -30 to 60.

- A setting of 0 tilts the pie 30 degrees. This is the default setting.
- A setting of -30 tilts the pie 0 degrees, resulting in a 2D view from directly above the pie.
- A setting of 60 tilts the pie 90 degrees, resulting in a 2D view of the pie's edge.

Valid settings for all other 3D graphs are integers from -60 to 90.

- A setting of 0 sets the viewing angle halfway up the graph.
- A setting of 90 produces a "plan" view from directly above the graph.

Description

The Elevation property sets the vertical viewing angle (in degrees) for 3D Pie graphs and True3D graphs. The setting represents an angle above or below a point halfway up the graph.

▲▼

Topic

Elevation

Related

3d property page
CageFlip
Perspective
Rotation
True3D

Extra() run-time property

Alternative way to set ExtraData property values at run time

Syntax

```
[form.][graph.]Extra(index%)[ = setting]
```

Parameter

index% Element number of array normally set by ExtraData property

Data type

Integer

Settings

Same as for ExtraData property

Description

The Extra() run-time property gives you an alternative method for setting the values normally set by the ExtraData property, which are stored as a one-dimensional array. Unlike ExtraData, Extra() doesn't require you to use the ThisPoint or ThisSet property to index the array. Instead, you provide the element (point or set) number as your Extra() parameter.

Example

If you want to explode the third data point in a pie chart, you can set up your code in two ways:

- If you use the ExtraData property, you set the ThisPoint property first:

  ```
  Graph1.ThisPoint = 3
  Graph1.ExtraData = 3
  ```

- If you use the Extra() property, you don't have to set ThisPoint:

  ```
  Graph1.Extra(3) = 1
  ```

▲▼

Topic

Extra()

Related

ExtraData
DataReset

ExtraData property

Holds additional data for various graph types

Syntax

[*form*.]*graph*.**ExtraData**[= *setting*]

Data type

Integer

Settings

Pie charts

0	Not exploded (default)
non-zero	Exploded

Line, lin/log, log/lin, log/log, polar, scatter (2D), and tape graphs

0	Point is shown (default)
non-zero	Point isn't shown

Time series graphs

0	No statistical lines (default)
1	Mean line
2	Standard deviation line
3	Mean and standard-deviation lines

Description

The ExtraData property has three uses:

- To explode the slices of a pie chart (2D or 3D).
- To set a data point to null in a line, lin/log, log/lin, log/log, polar, scatter or tape graph.
- To add statistical lines to a time series graph.

ExtraData is a two-dimensional array-based property indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint counters as you enter values.

Examples

These lines explodes segments 2 and 4 of a pie chart:

```
For I% = 1 to 4
   Graph1.GraphData = I%
Next I%
ThisPoint = 2 Graph1.ExtraData = 1
ThisPoint = 4 Graph1.ExtraData = 1
Graph1.DrawMode = 2
```

These lines omit points 2 and 6 from a line graph:

```
For I% = 0 to 9
   Graph1.GraphData = I%
Next I%
ThisPoint = 2 Graph1.ExtraData = 1
ThisPoint = 6 Graph1.ExtraData = 1
Graph1.DrawMode = 2
```

▲▼

Topic

ExtraData

Related

Data property page
AutoInc
ThisPoint
ThisSet
DataReset
Extra()

FName() run-time property

Sets name of font for text at run time

Syntax

`[form.][graph.]FName(index%)[ = font]`

Parameter

index%	Element number normally set by FontUse property	
	0	Graph title
	1	Other titles
	2	Labels
	3	Legend

Data type

String (default is Arial)

Description

The FName() property sets the font name for a particular text class (graph title, other titles, labels, and legend). It's a run-time equivalent to the FontName property.

Unlike FontName, FName() doesn't require you to use the FontUse property to provide a pointer to the text you want to set. Instead, you use the normal FontUse setting as your FName() parameter.

▲▼

Topic

FName()

Related

FontName

FontUse

FontFamily run-time property

Applies font name to text

Syntax

```
[form.]graph.FontFamily[ = setting]
```

Data type

Integer (0-3, enumerated)

Settings

Value	**FontName applies to**
0	Graph title
1	Other titles
2	Labels
3	Legend

Description

The FontFamily property applies the font name specified by the FontName property to the text specified by the FontUse property. FontFamily is included for compatibility with previous versions of the Graph control, which were called ChartBuilder. It works only at run time and isn't available in the properties window.

In ChartBuilder, the FontFamily property was used to attach a font family (Roman, Swiss, or Modern) to each class of text object (graph title, other titles, labels, or legend). The FontName property attached a particular font name to each FontFamily setting. If that font wasn't available at run time, ChartBuilder substituted a typeface matching the family description of Roman, Swiss, or Modern.

In the current Graph control, you can assign font names to text objects directly, using only the FontUse and FontName properties. This works because the FontFamily property, while still present, is now essentially a second "layer" of FontUse--unless you set FontFamily yourself in code (not recommended), its setting is always the same as the FontUse setting. If you specify a FontName that isn't available, Windows automatically looks for a similar typeface to replace it.

▲▼

Topic

FontFamily

Related

FontUse
FontName
FontSize
FontStyle

FontName design-time property

Sets name of font for text

Syntax

```
[form.]graph.FontName[ = setting]
```

Data type

String (default is Arial)

Description

The FontName property sets the font name for the text specified by the FontUse property. The default font is Arial for all text classes (graph title, other titles, labels, and legend).

If you specify a font name that isn't available on that system, Windows will choose a substitute font for the text.

The FontName property works only at design time. To change fonts at run time, use the FName() property.

Note Font names aren't case sensitive. You can use uppercase and lowercase letters interchangeably.

Topic

FontName

Related

FName()

Fonts property page

FontUse

FontSize

FontStyle

FontFamily

FontSize property

Sets size for text

Syntax

```
[form.]graph.FontSize[ = setting]
```

Data type

Integer (50-500 and -50--500)

Settings

Valid settings are integers from 50 to 500 or -50 to -500. The default FontSize setting depends on the setting of the FontUse property.

FontUse setting		FontSize default
0	Graph title	200
1	Other titles	150
2	Labels	100
3	Legend	100

Description

The FontSize property sets the size of the text specified by the FontUse property, using the System font as the basis for sizing. The actual size of type at any FontSize setting will vary from one typeface to another.

FontSize settings can be either positive or negative:

- Positive FontSize values are actually *maximum* sizes for the associated text. If there isn't enough space for the text at that size, the Graph control reduces the type size to fit, without changing the FontSize setting.

  Under Windows 3.0, where font sizes are fixed, there's a limit on how small type can become. Later versions of Windows and Windows NT have no such limitation, so it's possible for the Graph control to make type so small that it's hard to read. If this happens, increase the size of the graphing window or use less text.

- If you set a negative FontSize value, the Graph control won't attempt to reduce the type size to fit the available space. If there isn't enough room for the text at the size you specify, it won't display at all.

▲▼

Topic

FontSize

Related

Fonts property page

FontUse

FontName

FontStyle

FSize()

FontStyle property

Sets treatment (bold, italic, or underlined) for text

Syntax

[*form*.]*graph*.**FontStyle**[= *setting*]

Data type

Integer (0-7, enumerated)

Settings

0	No treatment (default)
1	*Italic*
2	**Bold**
3	***Bold italic***
4	Underlined
5	*Underlined italic*
6	**Underlined bold**
7	***Underlined bold italic***

Description

The FontStyle property applies a treatment--bold, italic, underlined, or any combination of these--to the text specified by the FontUse property. The default is no treatment ("regular" or "normal" type).

▲▼

Topic

FontStyle

Related

Fonts property page

FontUse

FontName

FontSize

FStyle()

FontUse property

Selects text component to which other font properties apply

Syntax

[*form*.]*graph*.**FontUse**[= *setting*]

Data type

Integer (0-4, enumerated)

Settings

0	Graph title (default)
1	Other titles
2	Labels
3	Legend
4	All text

Description

The FontUse property sets the text component to which the FontName, FontSize, and FontStyle properties apply. Those three properties are actually arrays holding four values, one for each of the graph's text components. FontUse indexes the arrays.

If you select all text (setting 4), you can apply a font family, style, or size to all text components of the graph at once.

Note At design time, when you set FontUse to 4 (all text), the properties window shows the FontName, FontStyle, and FontSize values only for the graph title. The other components may have different settings, but you can't see them unless you change FontUse to show those components individually. However, any changes you make to FontName, FontStyle, or FontSize apply to all components.

▲▼

Topic

FontUse

Related

Fonts property page

FontName

FontSize

FontStyle

Foreground property

Sets color of titles, labels, legends, axes, grids, statistical lines, curves, and data labels

Syntax

```
[form.]graph.Foreground[ = setting]
```

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The Foreground property sets the color of the foreground graphics (titles, labels, legend, axes, grids, statistical lines, curves, and data labels) specified by the ForegroundUse property.

By default, the Graph control uses either black or white--whichever offers the most contrast with the background color--for these objects.

To set the colors of actual graph elements (bars, pie slices, lines, and so on), use the ColorData property.

Topic

Foreground

Related

Background
BackgroundUse
Backdrop
BackdropStyle
ForegroundUse
Palette

ForegroundUse property

Selects graphic object to which color setting applies

Syntax

[*form*.]*graph*.**ForegroundUse**[= *setting*]

Data type

Integer (0-13, enumerated)

Settings

0	Graph title (default)
1	Left title
2	Right title
3	Bottom title
4	Labels
5	Legend
6	Axes
7	Grids
8	Mean line
9	Min and max lines
10	Standard deviation lines
11	Best-fit line
12	Fitted curve
13	Data labels
14	Limit lines and fills

Description

The ForegroundUse property selects the foreground object (title, labels, legend, etc.) to which a color is applied by the Foreground property.

▲▼

Topic

ForegroundUse

Related

Background
BackgroundUse
Labels property page
Foreground

FSize() run-time property

Alternative way to set font size at run time.

Syntax

```
[form.]graph.FSize(index%)[ = setting]
```

Parameter

index% Element number of array normally set by FontSize property

Index%	**Applies to**	**Default**
0	Graph title	200
1	Other titles	150
2	Labels	100
3	Legend	100

Data type

Integer (50-500 and -50--500)

Settings

Valid settings are integers from 50 to 500 or -50 to -500.

Description

The FSize() property sets the size of the text specified by the *index%* parameter, using the System font as the basis for sizing. The actual size of type at any FSize() setting will vary from one typeface to another.

FontSize settings can be either positive or negative:

- Positive FSize() values are actually *maximum* sizes for the associated text. If there isn't enough space for the text at that size, the Graph control reduces the type size to fit, without changing the FSize() setting.

  Under Windows 3.0, where font sizes are fixed, there's a limit on how small type can become. Later versions of Windows and Windows NT have no such limitation, so it's possible for the Graph control to make type so small that it's hard to read. If this happens, increase the size of the graphing window or use less text.

- If you set a negative FSize() value, the Graph control won't attempt to reduce the type size to fit the available space. If there isn't enough room for the text at the size you specify, it won't display at all.

▲▼

Topic

FSize()

Related

Fonts property page

FontUse

FontName

FontSize

FontStyle

FStyle()

FStyle() run-time property

Alternative way to set font style at run time.

Syntax

```
[form.]graph.FStyle(index%)[ = setting]
```

Parameter

index% Element number of array normally set by FontStyle property

Index%	**Applies to**
0	Graph title
1	Other titles
2	Labels
3	Legend

Data type

Integer (0-7, enumerated)

Settings

0	No treatment (default)
1	*Italic*
2	**Bold**
3	***Bold italic***
4	Underlined
5	*Underlined italic*
6	**Underlined bold**
7	***Underlined bold italic***

Description

The FStyle() property applies a treatment--bold, italic, underlined, or any combination of these--to the text specified by *index%* parameter. The default is no treatment ("regular" or "normal" type).

▲▼

Topic

FStyle()

Related

Fonts property page

FontUse

FontName

FontSize

FontStyle

(Graph Properties) design-time property

Opens Graph control's property pages

Value

Read-only string

Description

The (Graph Properties) design-time property opens the Graph control's property pages at design time. You can also open these pages at design time by right-clicking in a graphing window.

GraphCaption property

Sets caption text for empty graphing window

Syntax

```
[form.]graph.GraphCaption[ = "caption"]
```

Data type

String

Description

The GraphCaption property lets you specify a single line of text that will display when the DrawMode property is set to 1 (clear). In this case, instead of displaying the graph, the Graph control displays the caption in the center of the graphing window.

You can use the Foreground and Background properties to set the colors of the caption text and background.

Example

The following lines use the GraphCaption and DrawMode properties to display a caption instead of an actual graph:

```
Graph1.GraphCaption = "Graphics Server"
Graph1.DrawMode = 1
```

▲▼

Topic

GraphCaption

Related

DrawMode

Background

Foreground

GraphData property

Sets data to be graphed

Syntax

```
[form.]graph.GraphData[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The GraphData property holds the actual data values you want to graph. The values are stored as a two-dimensional array indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint values as you enter data.

▲▼

Topic

GraphData

Related

Data property page

AutoInc

ThisPoint

ThisSet

XPosData

DataReset

Data()

GraphFile property

Defines the file containing one or more graph templates

Syntax

[*form*.]*graph*.**GraphFile**[= *data*]

Data type

String

Setting

The name of a graph template file.

Template files always take the extension .GSP. This extension will be added to the filename if you do not include it. Any other extension you specify will be ignored. You may specify a path to the file. If you do not, the current working directory will be assumed.

Description

A graph's current property settings can be saved to a template file from which they may later be restored. The GraphFile property specifies the name of the file. The name of a particular graph within a template file is specified by the GraphName property.

Saving and restoring graph templates

The definition of a graph can be saved and restored at either design time or run time. This feature is particularly useful in those environments which have no means for serializing a graph object on a form. It may also be used to save and restore changes made by users at run time.

Graph definitions are saved in a template file. The name and location of the template file is specified in the setting for the GraphFile property. A template file can contain the definition for more than one graph. A particular graph within a template file is referenced by the GraphName property.

Graph template files are in ASCII text format and are structured like an INI file. Each graph definition begins with the GraphName in brackets and includes a list of saved properties with their settings.

To save a graph definition, first specify the name of the template file by setting the GraphFile property and, optionally, the name of the graph by setting the GraphName property. If the GraphName property is empty, Graphics Server will use the default name Graph. Save the graph by setting DrawMode = 10 or 11.

If DrawMode = 10, only the properties defining the appearance of the graph, including any text, are saved.

If DrawMode = 11, all properties not set to default values, including data, are saved.

Note The size of the file will grow in proportion to the number of points in the graph. A template file can store no more than 500 total graph points (the product of NumPoints and NumSets). When the limit is reached, no further graph information will be saved in the file.

To restore a graph definition, specify the name of the template file by setting the GraphFile property, and the name of the graph within the file by setting the GraphName property. Then load the graph definition by setting DrawMode = 9. Once the graph definition is loaded, redraw the graph by setting DrawMode = 2 or 3.

▲▼

Topic

GraphFile

Related

DrawMode

GraphName

System property page

GraphName property

Defines the name of a graph within a template file

Syntax

```
[form.]graph.GraphName[ = data]
```

Data type

String

Setting

Name of a saved graph within a template file.

Description

A graph's current property settings can be saved to a template file from which they may later be restored. The name of the template file is specified by the GraphFile property. The name of a saved graph within a template file is specified by the GraphName property.

A graph template file is an ASCII text file that can be inspected either manually or programmatically. The entry for each saved graph begins with the GraphName in brackets. Below the GraphName is a list of properties and their settings.

If the GraphName property is empty when a graph is saved, it is given the default name Graph.

▲▼

Topic

GraphName

Related

DrawMode

GraphFile

System property page

GraphStyle property

Sets characteristics for all graph types

Syntax

```
[form.]graph.GraphStyle[ = setting]
```

Data type

Integer (enumerated)

Settings

Area (2D)

0	Stacked (default)
1	Absolute
2	Stacked percentage
3	Stacked semi-logarithmic
4	Absolute semi-logarithmic

Area (3D)

0	Stacked (default)
1	Absolute
2	Stacked percentage

Bar (2D)

0	Vertical bars, clustered if multiple data sets (default)
1	Horizontal
2	Stacked
3	Horizontal stacked
4	Stacked percentage
5	Horizontal stacked percentage
8	Vertical stacked floating
9	Horizontal stacked floating
	Note: Floating bars require more than one data set. The first data set in a floating bar graph is invisible. The missing bottom element of the stacked bar makes the bar appear to be suspended above the X axis, hence the name "floating" bar graph.
10	Vertical Pareto
11	Horizontal Pareto
	Note: With a Pareto graph, bars are sorted in descending order; any attached user-defined X axis labels are sorted with the data. If there is more than one data set, bars are sorted in groups such that the first data set appears in descending order.

Bar (3D)

As for 2D bar, plus:

6	
7	Z-clustered

Horizontal Z-clustered

Box-whisker

0	Raw data, samples drawn (default)
1	Raw data, no samples drawn
2	Parametric data

The following values may be added to settings 0-2 and may be used in combination. For example, to produce a graph with parametric data, no notch and a black border, the setting would be:

```
Graph1.GraphStyle = 2 + 4 + 32
```

Note that additive settings cannot be made through the property window. Set these combinations either in code or through the property pages.

+4	No notch
+8	No whisker
+16	No median line **Note:** Median lines are visible with the black border option and when the marker's fill pattern is non-solid.
+32	Black border

Bubble

No style options

Candlestick

No style options

Gantt

0	Adjacent bars (default)
1	Spaced bars

High-low-close and open-high-low-close

0	Open (if used), high, low, and close bars (default)
1	No open or close bars
2	No high or low bars
3	No bars of any kind

Line, log/lin, lin/log, and log/log

0	Lines only (default)
1	Symbols
2	Sticks between points and Y origin
3	Sticks and symbols
4	Lines
5	Lines and symbols
6	Lines and sticks
7	Lines, sticks, and symbols

Pie (2D/3D)

0	Lines join labels to pie (default)
1	No label lines
2	Colored labels
3	Colored labels without lines
4	Percentage labels
5	Percentage labels without lines
6	Colored percentage labels
7	Colored percentage labels without lines

Polar

0	Lines only (default)
1	Symbols
2	Sticks between points and origin (center)
3	Sticks and symbols
4	Lines
5	Lines and symbols
6	Lines and sticks

7	Lines, sticks, and symbols

Scatter (2D)

0	Symbols only (default)
1	Curve only
2	Symbols only
3	Curve and symbols

Scatter (3D)

0	Symbols only (default)
1	Vertical sticks and symbols
2	Symbols and lines between points
3	Symbols, vertical sticks, and lines between points

Surface

0	Filled panels and boundary lines (default)
1	Boundary lines only
2	Filled panels only
3	Filled panels and boundary lines with side wall
4	Boundary lines only with side wall
5	Filled panels only with side wall

Tape

No style options

Time series

0	Symbols only (default)
1	Continuous line

Description

The GraphStyle property sets the styling option for the graph type.

▲▼

Topic

GraphStyle

Related

Style property page
DrawMode
GraphType
OverlayGraphStyle

GraphTitle property

Places text string at top of graphing window

Syntax

```
[form.]graph.GraphTitle[ = "title"]
```

Data type

String (up to 80 characters)

Description

The GraphTitle property places a text string at the top of a graphing window.

If you specify a title and it doesn't display, it may be too long for the space available. In that case, increase the width of the graphing window.

You can introduce a line break by inserting the newline character -- CHR$(10) -- in the string. This forces a new line to start at the point where the newline character appears, overriding automatic word-wrap. Forcing a new line is likely to increase the area required for the text and may lead to text being omitted if there is not enough room for it. Since the newline character is unprintable, it cannot be inserted when typing text; it can only be inserted with code.

▲▼

Topic

GraphTitle

Related

Titles property page
BottomTitle
LeftTitle
RightTitle
BackgroundStyle
FontName
FontStyle
FontSize
NewLineChar

GraphType property

Selects type of graph

Syntax

[*form*.]*graph*.**GraphType**[= *setting*]

Data type

Integer (0-22, enumerated)

Settings

0	No graph	12	Bubble
1	2D pie	13	Tape
2	3D pie	14	3D area
3	2D bar	15	Log/log
4	3D bar (default)	16	Lin/log
5	Gantt	17	Box-whisker
6	Line	18	Open-high-low-close
7	Log/lin	19	Candlestick
8	2D area	20	True3D surface
9	2D scatter	21	True3D scatter
10	Polar	22	Time series
11	High-low-close		

Description

Most graph types offer a variety of style options. Refer to the entry for the GraphStyle property.

▲▼

Topic

GraphType

Related

2D Gallery property page
3D Gallery property page
GraphStyle
True3D

GridLineStyle property

Sets style for grid lines

Syntax

```
[form.]graph.GridLineStyle[ = setting]
```

Data type

Integer (0-5, enumerated)

Settings

0	Solid (default)
1	Dashed
2	Dotted
3	Dash-dot
4	Dash-dot-dot
5	Null

Description

The GridLineStyle property sets the style of the grid lines specified by the GridStyle property. Grids are available for all graph types except pie charts.

To set the color for grid lines, use the ForegroundUse and Foreground properties.

▲▼

Topic

GridLineStyle

Related

Axis property page
GridStyle
Foreground
ForegroundUse

GridStyle property

Places reference grids on graph

Syntax

```
[form.]graph.GridStyle[ = setting]
```

Data type

Integer (0-3, enumerated)

Settings

0	No grids (default)
1	Y (horizontal) grids--radial (concentric circles) in polar graphs
2	X (vertical) grids--angular ("spokes") in polar graphs
3	Both horizontal and vertical grids--both radial and angular in polar graphs

Description

The GridStyle property places reference grids on a graph. Grids take two forms:

- **For graph types having regular X and Y axes (area, bar, line, scatter, etc.),** Y grids are drawn as a horizontal lines and X grids are drawn as vertical lines.
- **For polar graphs,** Y grids are concentric circles and X grids are radial lines ("spokes").

If you set GridStyle to 3 (both Y and X grids) for a True3D graph, you'll also get Z (depth) grids.

Grids don't apply to pie charts.

▲▼

Topic

GridStyle

Related

Axis property page
GridLineStyle
True3D

HCtlWin run-time property

Returns handle of Graph control window

Syntax

*[graph.]***HCtlWin**

Data type (returned)

Integer

Description

The HCtlWin run-time property returns the handle of a Graph control window. This window is the parent of any graphing windows within it. To get the handle of a graphing window, you can use the HWin run-time property.

The HCtlWin property is read-only and works only at run time.

▲▼

Topic

HCtlWin

Related

HWin

(Help) design-time property

Displays Windows Help

Value

Read-only string

Description

The (Help) design-time property lets you call up a Windows Help file for the Graph control.

HelpFile property

Sets name of help file callable at run time

Syntax

[*form*.]*graph*.**HelpFile**[= "*title*"]

Data type

String

Description

The HelpFile property lets you specify the name of the help file that can be called at run time through the property pages or toolbar. If you don't specify a help file, the Graph control uses the same one at run time that you use at design time.

▲▼

Topic

HelpFile

Related

Design property page

Toolbar

ToolStat

PropertyPages

PropertyCaption

Hot property

Enables and disables hot graphing

Syntax

```
[form.]graph.Hot[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The Hot property lets you enable and disable Graphics Server's "hot graphing" system. With Hot on, a mouse click at run time on a graph marker (such as a symbol, pie slice or bar) fires a HotHit event, returning the data set and/or point number of that marker.

Note When the hot area is a symbol, the size of the hot area changes with the size of the symbol. That is, if you set the SymbolSize property to a larger value, both the symbol and the hot area will become larger. With line graphs, the point marker is not an entire line segment; it's the intersection of X and Y coordinates. To make it easier to find this spot, select the "lines and symbols" style.

When you set Hot to on, the HotHit capability begins when the graph is next redrawn. If you want to enable HotHit events immediately, reset the DrawMode property to force a redraw right after you set Hot to on.

The hot graphing system doesn't distinguish between single and double clicks. However, you can determine which button was clicked using the HotButton run-time property.

Turning on hot graphing disables the standard Visual Basic mouse events (Click and DblClick) in the graphing window. It also disables right mouse click access to the property pages at run time. However, you can still access the property pages through the toolbar if it is enabled.

The Hot property can also be used in combination with DrawMode and MapFile to create an image map for use with graphs displayed as inline images in an HTML document. For details, see MapFile.

▲▼

Topic

Hot

Related

Design property page

HotHit (event)

HotButton

DrawMode

MapFile

PropertyPages

SymbolSize

HotButton run-time property

Returns number of mouse button pressed on hot graph

Syntax

[*graph*.]**HotButton**[= *setting*]

Data type (returned)

Integer (1-3, enumerated)

Settings

1	Left button
2	Middle button
3	Right button

Description

The HotButton run-time property returns a number corresponding to the mouse button pressed on a hot graph--left, middle, or right. HotButton is a read-only property and works only at run time.

▲▼

Topic

HotButton

Related

Hot

HotHit event

Fired when hot graph receives mouse click at run time

Syntax

```
Sub Graph_HotHit (HitSet As Integer, HitPoint As Integer)
```

Parameters

HitSet   Data set receiving a mouse click
HitPoint Data point receiving a mouse click

Description

The HotHit event is fired when a hot graph's marker--bar, pie slice, line, or point--receives a mouse click at run time. To enable the hot-graph system, you must first set the Hot property to on.

The values returned by HotHit depend on the graph type:

Graph type	Marker	Information returned
Area (2D & 3D)	Filled area	ThisSet value only
Bar (2D & 3D)	Bar	Both ThisSet and ThisPoint values
Box-whisker	Box-whisker symbol	ThisPoint value only
Bubble	Bubble	ThisPoint value only
Gantt	Bar	Both ThisSet and ThisPoint values
High-low-close, open-high-low-close, and candlestick	Open, high, low, or close point	Both ThisSet and ThisPoint values
Line, log/lin, lin/log, and log/log	Point on line	Both ThisSet and ThisPoint values
Pie (2D & 3D)	Slice	ThisPoint value only
Polar	Point on line	ThisPoint value only (works with one data set only)
Scatter (2D & 3D)	Symbol	Both ThisSet and ThisPoint values
Surface	Filled panel	Both ThisSet and ThisPoint values
Tape	Tape face	Both ThisSet and ThisPoint values
Time series	(Not implemented)	

▲▼

Topic

HotHit

Related

Replacing property pages

Hot

HWin run-time property

Returns handle of graphing window

Syntax

```
[graph.]HWin
```

Data type (returned)

Integer

Description

The HWin run-time property returns the handle of a graphing window (the window Graphics Server creates to display a graph). This window is a child of an overall Graph control window, whose handle you can get via the HCtlWin run-time property.

The HWin property is read-only and works only at run time.

▲▼

Topic
HWin

Related
HCtlWin

ImageFile property

Sets filename for writing bitmap or metafile to disk

Syntax

[*form*.]*graph*.**ImageFile**[= "*filename*"]

Data type

String

Setting

Path (optional), file name and file extension. If you don't specify a path, the file is written to the current directory.

Description

The ImageFile property specifies a filename for writing a bitmap or metafile to disk when the DrawMode property is set to 6.

Four graphics file formats are supported: Windows Metafile (WMF), Windows Bitmap (BMP), Joint Photographic Experts Group (JPG) and Graphics Interchange Format (GIF).

BMP files are 16-color. You can't use the ImageFile property to create a 256-color bitmap.

JPG and GIF require an appropriate conversion library, GSJPG16/32.DLL or GSGIF16/32.DLL. The library must reside in the home directory of GSW16/32.EXE or on the path. If the library cannot be found, the image file will not be created.

Note: The JPEG conversion DLLs are on the installation CD. Because of licensing restrictions, libraries for converting to GIF are available only from our web site. Before using them, you must obtain a license from Unisys, who hold the patent for LZW compression.

Saving an image to disk

Saving an image of a graph is a three-step operation. First, specify the output file's name and type by setting the ImageFile property. The file extension helps determine the graphics file format, so don't vary from the standards. If you want the output file to be a metafile, make the file extension WMF. If you want a bitmap, make it BMP. For JPEG, the file extension must be JPG. For Graphics Interchange Format, it must be GIF.

Next, prepare the image in memory by setting DrawMode either to 2 or 3. Which of these two settings you should choose depends on which graphics file format you want for the output file. For WMF, set DrawMode to 2. For BMP, JPG or GIF, set DrawMode to 3.

Finally, write the image to disk by setting DrawMode to 6.

Example

```
'Gets a file name from the user and then
'saves the graph to a file
Private Sub cmdSaveFile_Click()
Dim Filename As String
Dim Filetype As String

'Set Common Dialog options
With CMDialog1
    .Filter = "WMF - Windows Metafile|*.wmf|BMP - Windows Bitmap|*.bmp |JPG -
JPEG|*.jpg|GIF - Graphics Interchange Format|*.gif"
```

```
    .FilterIndex = 2
    .Flags = cdlOFNHideReadOnly + cdlOFNOverwritePrompt
    .Filename = ""
End With

'Get a file name from the user
CMDialog1.ShowSave
Filename = CMDialog1.Filename

'Check for a valid file type
Filetype = UCase(Mid(Filename, InStr(Filename, ".") + 1, 3))
If Filetype = "WMF" Or Filetype = "BMP" Or Filetype = "JPG" Or Filetype =
"GIF" Then

    'Set the Graph control's ImageFile property
    Graph1.ImageFile = Filename

    'Set the DrawMode appropriate for the file type
    Select Case Filetype
        Case "WMF"
            Graph1.DrawMode = 2
        Case Else
            Graph1.DrawMode = 3
    End Select

    'Write the image to disk
    Graph1.DrawMode = 6

End If
End Sub
```

▲▼

Topic

ImageFile

Related

DrawMode

MapFile

System property page

IndexStyle property

Sets indexing method for array-based properties

Syntax

`[form.]graph.IndexStyle[ = setting]`

Data type

Integer (0-1, enumerated)

Settings

0	Standard method (default)
1	Enhanced method

Description

When you graph data using Graphics Server--and when you deal with such issues as colors and labels--you work with the concepts of *data sets* and *data points*. In general, we define a data point as the intersection of two or more variables and a data set as a collection of data points, but these definitions don't hold true for every graph type. The two properties you use to set values for your graph (ThisSet and ThisPoint) frequently don't correspond to the definitions, either.

To ease the potential confusion that may arise when you work with data sets and points, the Graph control lets you choose from two methods for indexing (that is, "pointing" to an element of) the array-based properties that hold graph data, color selections, label and legend text, and other important graph values.

The IndexStyle property lets you select the indexing method you want to use. This setting has no effect on the eventual appearance of the graph--only on the way you set and get values.

Standard method

When you use the standard method (IndexStyle setting 0), one-dimensional array-based properties are always indexed by ThisPoint, and two-dimensional array-based properties are always indexed by both ThisSet and ThisPoint.

Property	**Accessed by**
ColorData	ThisPoint
DataLabelText	ThisSet and ThisPoint (2D array)
EBarGraphDataMinus	ThisSet and ThisPoint (2D array)
EBarGraphDataPlus	ThisSet and ThisPoint (2D array)
EBarXPosDataMinus	ThisSet and ThisPoint (2D array)
EBarXPosDataPlus	ThisSet and ThisPoint (2D array)
ExtraData	ThisPoint
GraphData	ThisSet and ThisPoint (2D array)
LabelText	ThisPoint
LegendText	ThisPoint
OverlayGraphData	ThisPoint
PatternData	ThisPoint

SymbolData	ThisPoint
XPosData	ThisSet and ThisPoint (2D array)
YLabelText	ThisPoint
ZLabelText	ThisPoint
ZPosData	ThisSet and ThisPoint (2D array)

The standard method is set up simply, but it isn't always as straightforward as it might seem. The LabelText property works well with this method because it actually applies to data *points* (each point gets its own label), so the ThisPoint index is appropriate. On the other hand, the property SymbolData, which by definition applies to data *sets* (each set gets its own type of symbol), is also indexed by ThisPoint--so each "point" actually refers to a set.

Other properties vary according to context. ColorData, for example, applies to points when a bar graph has only one set of data and to sets when it has more than one. But under the standard indexing method, the Graph control indexes by ThisPoint in both cases.

These differences may not concern you, particularly if you set the AutoInc property on--AutoInc increments ThisSet and ThisPoint correctly no matter what the context. But in cases where you don't want to use AutoInc, you may prefer the enhanced indexing method.

Enhanced method

When you use the enhanced method (IndexStyle setting 1), one-dimensional array-based properties are indexed in two ways--by ThisSet when they apply to data sets and by ThisPoint when they apply to data points.

Some of the one-dimensional array-based properties (ColorData, ExtraData, LegendText, and PatternData) are context-sensitive and use either ThisSet or ThisPoint according to the graph type. For graphs that display legend text for each data point (pie charts, bubble graphs, and bar graphs having only one data set), these properties are indexed by ThisPoint. For all other graph types, they're indexed by ThisSet.

The enhanced indexing method has no effect on the two-dimensional array-based properties. They're indexed by both ThisSet and ThisPoint, just as with the standard method.

Property	**Accessed by**
ColorData	*Pie charts, bubble graphs, and bar graphs having only one data set:* ThisPoint *All other graph types:* ThisSet
DataLabelText	ThisSet and ThisPoint (2D array)
EBarGraphDataMinus	ThisSet and ThisPoint (2D array)
EBarGraphDataPlus	ThisSet and ThisPoint (2D array)
EBarXPosDataMinus	ThisSet and ThisPoint (2D array)
EBarXPosDataPlus	ThisSet and ThisPoint (2D array)
ExtraData	*Pie charts, bubble graphs, and bar graphs having only one data set:* ThisPoint *All other graph types:* ThisSet
GraphData	ThisSet and ThisPoint (2D array)
LabelText	ThisPoint
LegendText	*Pie charts, bubble graphs, and bar graphs*

	having only one data set: ThisPoint
	All other graph types: ThisSet
OverlayGraphData	ThisPoint
PatternData	*Pie charts, bubble graphs, and bar graphs having only one data set:* ThisPoint
	All other graph types: ThisSet
SymbolData	ThisSet
XPosData	ThisSet and ThisPoint (2D array)
YLabelText	ThisPoint
ZLabelText	ThisSet
ZPosData	ThisSet and ThisPoint (2D array)

Examples

The following code segments compare the different approaches taken by the standard and enhanced indexing methods. In these examples, values are assigned to a line graph having an unknown number of data sets and points.

Standard method

```
Graph1.GraphType = 6         ' line graph
Graph1.IndexStyle = 0        ' standard index style
For I% = 1 To Graph1.NumSets
   Graph1.ThisSet = I%
   For J% = 1 To Graph1.NumPoints
      Graph1.ThisPoint = J%
      Graph1.GraphData = <your data value>
      Graph1.XPosData = <your data value>
   Next
Next
For I% = 1 To Graph1.NumSets
   Graph1.ThisPoint = I%     ' use ThisPoint as index
   Graph1.LegendText = "Legend " + Str$(I%)
   Graph1.ExtraData = <your data value>
   Graph1.ColorData = <your data value>
   Graph1.PatternData = <your data value>
   Graph1.SymbolData = <your data value>
Next
For I% = 1 To Graph1.NumPoints
   Graph1.ThisPoint = I%
   Graph1.LabelText = "Label " + Str$(I%)
Next
Graph1.DrawMode = 2
```

Enhanced method

```
Graph1.GraphType = 6         ' line graph
Graph1.IndexStyle = 1        ' enhanced index style
For I% = 1 To Graph1.NumSets
```

```
   Graph1.ThisSet = I%
   For J% = 1 To Graph1.NumPoints
      Graph1.ThisPoint = J%
      Graph1.GraphData = <your data value>
      Graph1.XPosData = <your data value>
   Next
Next
For I% = 1 To Graph1.NumSets
   Graph1.ThisSet = I%        ' use ThisSet as index
   Graph1.LegendText = "Data set " + Str$(I%)
   Graph1.ExtraData = <your data value>
   Graph1.ColorData = <your data value>
   Graph1.PatternData = <your data value>
   Graph1.SymbolData = <your data value>
Next
For I% = 1 To Graph1.NumPoints
   Graph1.ThisPoint = I%
   Graph1.LabelText = "Data point " + Str$(I%)
Next
Graph1.DrawMode = 2
```

▲▼

Topic

IndexStyle

Related

ThisPoint

ThisSet

AutoInc

ISVDataSource property

Binds a graph to another control

Syntax

[*form*.][*graph*.]**ISVDataSource**[= "*objectname*"]

Data type

String

Settings

The name of an instance of another control that will act as a source of data for the graph. The control must support ISV data binding .

Description

The Graph control can participate as a client in ISV data binding, linking to and receiving data from an ISV *source* control such as TrueDBGrid.

In the Graph control, ISV data binding is initiated by setting the ISVDataSource property with the name of a suitable ISV data source. This may be done at design time through the property browser, or at run time; for example

```
Graph1.ISVDataSource = "TDBGrid1"
```

Setting ISVDataSource with a control name causes the graph to search the container and check that the specified control exists and supports ISV data binding. If it succeeds, the graph creates a binding to the source control.

Alternatively, at run time, you can bypass the search by setting ISVDataSource to the value of a special property called ISVServices exported by ISV data sources; for example

```
Graph1.ISVDataSource = TDBGrid1.ISVServices
```

Once an ISV data binding is operational, the Graph control uses it in the same way as it does a binding with a standard VB data control. The fields that go in the graph are specified through the DataFields, LabelField and LegendField properties. Again, this may be done at design time or run time. At design time the easiest way to select fields is through the Bound data property page.

The way the Graph control uses an ISV data binding is closely tied to the standard data binding facility that manifests itself through the DataSource property. In fact, any graph can only have one source of data at any time and DataSource and ISVDataSource are effectively different ways of specifying the same thing. Setting a new data source through ISVDataSource automatically cancels any existing binding set through DataSource, and vice versa.

Unfortunately the relationship between DataSource and ISVDataSource is not entirely symmetric. This is because ISVDataSource is a custom property of the Graph control itself, whereas DataSource is effectively a property of the container. These are some of the points to watch:

- Setting DataSource is only possible at design time. ISVDataSource may be set at design time and run time.
- If it is essential to switch the kind of data binding at design time, make sure that the existing binding is clear before setting the new one. For example, to switch from standard to ISV data binding, clear DataSource by setting it blank in the property browser first, before selecting a new source name in ISVDataSource.
- If you do have a graph in which both kinds of data binding are specified, the standard

binding in DataSource takes precedence on loading and switching modes.

- At run time, you can change the kind of data binding from standard to ISV, but not vice versa. You can also change to different ISV data sources during the running of your application.

The binding between a graph and an ISV data source may be ended by clearing the control name in ISVDataSource; for example

```
Graph1.ISVDataSource = ""
```

▲▼

Topic

ISVDataSource

Related

Bound data property page
DataFields
DataSource
LabelField
LegendField

Label() run-time property

Alternative way to set text of graph labels at run time

Syntax

`[form.][graph.]Label(index%)[ = "text"]`

Parameter

index% Element number of array normally set by LabelText property

Data type

String (up to 80 characters)

Description

The Label() run-time property sets the text of graph labels. It's a run-time alternative to the LabelText property.

The text for labels is stored as a one-dimensional array. Unlike LabelText, Label() doesn't require you to use the ThisPoint property to index the label array. Instead, you provide the element (point) number as your Label() parameter.

Example

If you want to label the fifth slice of a pie chart "May," you can set up your code in two ways:

- If you use the LabelText property, you set the ThisPoint property first:

  ```
  Graph1.ThisPoint = 5
  Graph1.LabelText = "May"
  ```

- If you use the Label() property, you don't have to set ThisPoint:

  ```
  Graph1.Label(5) = "May"
  ```

▲▼

Topic

Label()

Related

LabelText

LabelEvery property

Sets frequency of X axis labels

Syntax

`[form.]graph.LabelEvery[ = setting]`

Data type

Integer (1-1000)

Settings

Valid settings are integers from 1 to 1000 (default is 1).

Description

The LabelEvery property sets the frequency with which labels are displayed on the X axis. Labels are displayed at every *n*th data point, starting with the first point, where *n* is the setting for LabelEvery.

You can't use LabelEvery if you're also using the XPosData property. This means that LabelEvery never applies to bubble, lin/log, or log/log graphs, which always have XPosData.

▲▼

Topic

LabelEvery

Related

Labels property page
Labels
LabelStyle
LabelText
XAxisTicks

LabelField property

Specifies which text field in a bound database should provide X axis labels

Syntax

[*form*.][*graph*.]**LabelField**[= "*field name*"]

Data type

String

Setting

Valid field name in a table of a bound database.

Description

The LabelField property allows you optionally to select a text field to provide labels along the X axis.

Note At design time, labels along the X axis show the numbers of the data points. At run time, the table is accessed and the content of the label field appears in labels for points on the axis.

▲▼

Topic

LabelField

Related

Bound data property page
DataFields
DataSource
LegendField

Labels property

Enables or disables graph labels

Syntax

```
[form.]graph.Labels[ = setting]
```

Data type

Integer (0-3, enumerated)

Settings

0	No labels
1	Both X and Y labels (default)
2	X axis labels only
3	Y axis labels only

Description

The Labels property enables or disables graph labels. It operates differently for pie charts than for other graph types:

Pie charts

For pie charts, the Labels property determines if labels are displayed at all--labels are displayed only if Labels is set to 1.

Other graph types

For graph types with X and Y axes, the Labels property determines not only if labels are displayed, but also their position--along the X axis, the Y axis, or both.

The Labels property operates independently of the Ticks property.

▲▼

Topic

Labels

Related

Labels property page
LabelEvery
LabelStyle
LabelText
LabelXFormat
LabelYFormat
YLabelText
Ticks

LabelStyle property

Sets style (horizontal or vertical) for axis labels

Syntax

[*form*.]*graph*.**LabelStyle**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Horizontal (default)
1	Vertical

Description

The LabelStyle property determines the arrangement of labels for X and Z axes--horizontal or vertical (rotated 90 degrees clockwise).

In horizontal bar graphs and Gantt charts, LabelStyle applies to the Y axis rather than the X axis. In other graph types, Y axis labels are always horizontal.

▲▼

Topic

LabelStyle

Related

Labels property page

LabelEvery

Labels

LabelText

ZLabelText

LabelText property

Sets text for graph labels

Syntax

```
[form.]graph.LabelText[ = "label"]
```

Data type

String (up to 80 characters)

Description

The LabelText property sets the text for graph labels. For pie charts, it labels each pie slice; for graph types having X-Y grids, it labels points along the independent axis (normally the X axis).

LabelText is a one-dimensional array-based property indexed by the ThisPoint property. You can use the AutoInc property to automatically increment the ThisPoint counter each time you set a LabelText element.

Each label can contain no more than 80 characters. Also, if any of the labels you define are too long to fit in the graphing window at the specified size, no labels are drawn. If this happens, you may be able to make the labels appear by increasing the size of the graphing window.

Pie charts

By default, pie chart labels show the value for each data point (each point is a "slice" of the pie). With the LabelText property, you can specify text for each point.

Other graph types

For graphs drawn on X-Y grids, the LabelText property specifies the text for labels on the independent axis, which is normally the X axis. (For labels on the other axes, see the YLabelText and ZLabelText properties.)

By default, these labels show the numbers of the data points (the ThisPoint values). With the LabelText property, you can specify text for each point.

▲▼

Topic

LabelText

Related

Labels property page

LabelEvery

Labels

LabelStyle

YLabelText

ZLabelText

Label()

LabelXDateInc property

Sets a date/time increment for labels on the X axis

Syntax

[*form*.]*graph*.**LabelXDateInc**[= *setting*]

Data type

String

Settings

Fixed format string defining the increment for date/time labels. The format depends on the setting for LabelXType.

LabelXType	**Format for LabelXDateInc**

1 (Date)	"yyyy:mm:dd" where
	
	yyyy is the year increment (0000-0100)
	mm is the month increment (00-99)
	dd is the day increment (00-99)
	
	Example
	`'Labels increment by 100 years`
	`LabelXDateInc = "0100:00:00"`

2 (Time)	"hh:mm:ss" where
	
	hh is the hour increment (00-99)
	mm is the minute increment (00-99)
	ss is the second increment (00-99)
	
	Example
	`'Increment by 90 seconds`
	`LabelXDateInc = "00:00:90"`

3 (Date and Time)	"yyyy:mm:dd:hh:mm:ss" where
	
	yyyy is the year increment (0000-0100)
	mm is the month increment (00-99)
	dd is the day increment (00-99)
	hh is the hour increment (00-99)
	mm is the minute increment (00-99)
	ss is the second increment (00-99)
	
	Example
	`'Increment 1 year and 6 seconds`
	`LabelXDateInc = "0001:00:00:00:00:06"`

4 (Date, skip weekends)	"yyyy:mm:dd" where *yyyy* is the year increment (0000-0100) *mm* is the month increment (00-99) *dd* is the day increment (00-99) **Example** `'Increment by 3 months`

```
                     LabelXDateInc = "0000:03:00"
```

Description

The LabelXDateInc property defines the increment for date/time labels along the X axis. The starting date is set by the LabelXDateStart property. The type of labels (date, time, date and time, or weekday) is determined by the setting for the LabelXType property.

Example

The code below generates a series of date and time labels on the X axis. The starting date/time is 1 Jan 97 at 00:00. Labels increment by 56 hours.

```
With Graph1
    'Date and time labels
    .LabelXType = 3
    'Start on 1 Jan 97 at 00:00
    .LabelXDateStart = "1997:01:01:00:00:00"
    'Increment every 56 hours
    .LabelXDateInc = "0000:00:00:56:00:00"
    .LabelXFormat = "d mmm hh:nn"

    'Redraw the graph
    .DrawMode = 3
End With
```

▲▼

Topic

LabelXDateInc

Related

Labels property page

LabelXDateStart

LabelXType

LabelXDateStart property

Sets a starting date/time for labels on the X axis

Syntax

[*form.*]*graph*.**LabelXDateStart**[= *setting*]

Data type

String

Settings

Fixed format string with the starting date and/or time. The format depends on the setting for LabelXType.

LabelXType	**Format for LabelXDateStart**

1 (Date)	"yyyy:mm:dd" where
	
	yyyy is the year (1900-2036)
	mm is the month (01-12)
	dd is the day (01-31)
	Example
	`'Start at 31 Jan 97`
	`LabelXDateStart = "1997:01:31"`

2 (Time)	"hh:mm:ss" where
	
	hh is the hour (00-24)
	mm is the minute (00-59)
	ss is the second (00-59)
	Example
	`'Start at 1:30 pm`
	`LabelXDateStart = "13:30:00"`

3 (Date and Time)	"yyyy:mm:dd:hh:mm:ss" where
	
	yyyy is the year (1900-2036)
	mm is the month (01-12)
	dd is the day (01-31)
	hh is the hour (00-24)
	mm is the minute (00-59)
	ss is the second (00-59)
	Example
	`'Start at 9:15 pm on 15 Nov 93`
	`LabelXDateStart = "1993:11:15:21:15:00"`

4 (Date, skip weekends)	"yyyy:mm:dd" where *yyyy* is the year (1900-2036) *mm* is the month (01-12) *dd* is the day (01-31) **Example** `'Start at 1 Jan 00`

```
LabelXDateStart = "2000:01:01"
```

Description

The LabelXDateStart property sets the starting date and/or time for a series of automatically generated X axis labels. The type of labels (date, time, date and time, or weekday) is determined by the setting for the LabelXType property. The increment for intervals along the axis is set by the LabelXDateInc property.

Example

The code below generates a series of "date, skip weekend" labels for the X axis. The labels start at 30 Dec 99 and increment one day for each point.

```
With Graph1
    .LabelXType = 4 'weekday
    .LabelXDateStart = "1999:12:30" '30 Dec 99
    .LabelXDateInc = "0000:00:01"   '1 day intervals
    .LabelXFormat = "d mmm yy"

    'Redraw the graph
    .DrawMode = 3
End With
```

▲▼

Topic

LabelXDateStart

Related

Labels property page
LabelXDateInc
LabelXType
LabelXFormat

LabelXFormat property

Sets format for numeric or date/time X labels

Syntax

[*form*.]*graph*.**LabelXFormat**[= *data*]

Data type

String

Setting

A valid numeric format expression.

Description

The LabelXFormat property defines a template for formatting numeric, date and time labels. It is not applied to user-defined text labels.

Just as is the case with the Visual Basic function Format$(), the characters within a LabelXFormat format string determine how the number or serial date/time is formatted. See Date/time formats and Numeric formats for a full description.

Example

The code below labels the X axis with a date series in one-year increments. The labels are formatted to display as FY93, FY94, FY95. . . .

```
With Graph1
    .LabelXType = 1                 'date
    .LabelXDateStart = "1993:12:31" 'last day of 1993
    .LabelXDateInc = "0001:00:00"   'one year

    .LabelXFormat = "\F\Y yy"

    'Redraw the graph
    .DrawMode = 3
End With
```

▲▼

Topic

LabelXFormat

Related

Labels property page
DataLabelFormat
LabelXType
LabelYFormat
LabelZFormat
Date/time formats
Numeric formats

LabelXType property

Sets the type of labels to be used on the X axis

Syntax

[*form*.]*graph*.**LabelXType**[= *setting*]

Data type

Integer (0-4, enumerated)

Settings

0	Numeric or Text (default)
1	Date
2	Time
3	Date and time
4	Date, skip weekends

Description

The LabelXType property selects the type of labeling for the X axis.

With settings 1-4, LabelXType can be used in combination with LabelXDateStart and LabelXDateInc to generate a series of automatically incremented date or time labels.

Example

The code below generates a series of time labels for the X axis. Time starts at 6:00 am and points increment by 15 minutes.

```
With Graph1
    .LabelXType = 2 'time labels
    .LabelXDateStart = "06:00:00"  '6 am
    .LabelXDateInc = "00:15:00"    '15 min intervals
    .LabelXFormat = "hh:nn"

    'Redraw the graph
    .DrawMode = 3
End With
```

▲▼

Topic

LabelXType

Related

Labels property page
LabelXFormat
LabelXDateInc
LabelXDateStart

LabelYFormat property

Sets format for numeric Y labels

Syntax

`[form.]graph.LabelYFormat[ = data]`

Data type

String

Setting

A valid numeric format expression.

Description

The LabelYFormat property defines a template for formatting numeric labels. It is not applied to user-defined text labels.

Just as is the case with the Visual Basic function Format$(), the characters within a LabelYFormat format string determine how the number is formatted. See numeric formats for a full description.

▲▼

Topic

LabelYFormat

Related

Labels property page

DataLabelFormat

LabelXFormat

LabelZFormat

Numeric formats

LabelZFormat property

Sets format for numeric Z labels

Syntax

[*form.*]*graph*.**LabelZFormat**[= *data*]

Data type

String

Setting

A valid numeric format expression.

Description

The LabelZFormat property defines a template for formatting numeric labels. It is not applied to user-defined text labels.

Just as is the case with the Visual Basic function Format$(), the characters within a LabelZFormat format string determine how the number is formatted. See numeric formats for a full description.

▲▼

Topic

LabelZFormat

Related

Labels property page

DataLabelFormat

LabelXFormat

LabelYFormat

Numeric formats

LeftTitle property

Specifies text string to place at left of graphing window

Syntax

```
[form.]graph.LeftTitle[ = "title"]
```

Data type

String (up to 80 characters)

Description

The LeftTitle property specifies a text string of up to 80 characters that appears at the left edge of a graphing window.

If you specify a title and it doesn't appear, it's probably too long to display at the specified size. In that case, increase the size of the graphing window or use a shorter title.

You can introduce a line break by inserting the newline character -- CHR$(10) -- in the string. This forces a new line to start at the point where the newline character appears, overriding automatic word-wrap. Forcing a new line is likely to increase the area required for the text and may lead to text being omitted if there is not enough room for it. Since the newline character is unprintable, it cannot be inserted when typing text; it can only be inserted with code.

Note LeftTitle text always appears at the left edge of the graphing window. If, for example, you set LeftTitle for a bar graph with vertical bars and then change to horizontal bars (switching the X and Y axes), the text for LeftTitle will remain at the left.

▲▼

Topic

LeftTitle

Related

Titles property page

LeftTitleStyle

GraphTitle

BottomTitle

RightTitle

BackgroundStyle

FontName

FontStyle

FontSize

Foreground

ForegroundUse

NewLineChar

LeftTitleStyle property

Sets style for left graph title

Syntax

[*form*.]*graph*.**LeftTitleStyle**[= *setting*]

Data type

Integer (0-2, enumerated)

Settings

0	Horizontal (default)
1	Up
2	Down

Description

The LeftTitleStyle property sets the style for the left title of a graph. You can display the text horizontally, up (text rotated counterclockwise 90 degrees), or down (text rotated clockwise 90 degrees).

The text of the left title is specified by the LeftTitle property.

▲▼

Topic

LeftTitleStyle

Related

BackgroundStyle
LeftTitle
Titles property page

Legend() run-time property

Alternative way to set legend text at run time

Syntax

[*form*.][*graph*.]**Legend(*index*%)**[= "*text*"]

Parameter

index% Element number of array normally set by LegendText property

Data type

String (up to 80 characters)

Description

The Legend() run-time property sets the text for a graph legend. It's a run-time alternative to the LegendText property.

Legend strings are stored as a one-dimensional array, which is normally indexed by either the ThisPoint or ThisSet property (see the IndexStyle property for details on the Graph control's indexing of array-based properties). However, Legend()--unlike LegendText--doesn't require you to use ThisPoint or ThisSet. Instead, you provide the element (point or set) number as your Legend() parameter.

Example

If you want to set the fourth legend string for a pie chart to "Western Region," you can set up your code in two ways:

- If you use the LegendText property, you set the ThisPoint property firs

  ```
  Graph1.ThisPoint = 4
  Graph1.LegendText = "Western Region"
  ```

- If you use the Legend() property, you don't have to set ThisPoint:

  ```
  Graph1.Legend(4) = "Western Region"
  ```

▲▼

Topic

Legend()

Related

Legend property page

LegendText

LegendField property

Specifies which text field in a bound table should provide text for the graph legend

Syntax

[*form*.][*graph*.]**LegendField**[= "*field name*"]

Data type

String

Setting

Valid field name in a table of a bound database.

Description

When you are graphing data from a single field in a bound table, you may want text for the legend to be taken from the content of a field. If so, set the LegendField property with the name of a field.

If you are graphing several fields from a bound table, use the LegendText property to specify text for the legend.

▲▼

Topic

LegendField

Related

Bound data property page

DataFields

DataSource

LabelField

LegendText

LegendPos property

Sets position of graph legend

Syntax

```
[form.]graph.LegendPos[ = setting]
```

Data type

Integer (0-7, enumerated)

Settings

0	Right (default)
1	Top right
2	Top
3	Top left
4	Left
5	Bottom left
6	Bottom
7	Bottom right

Description

The LegendPos property sets the position of a legend in the graphing window.

▲▼

Topic

LegendPos

Related

Legend property page
LegendSize
LegendStyle
LegendText

LegendSize property

Sets overall legend size

Syntax

```
[form.]graph.LegendSize[ = setting]
```

Data type

Integer (0-100)

Settings

Settings specify a percentage of the maximum possible gap between lines in the space allotted for the legend. The default setting is 100.

Description

The LegendSize property sets the approximate overall size of a graph legend, including the text and the gaps between items. The size is expressed as a percentage of the maximum possible gap, which varies according to the size of the graphing window and the number of items in the legend. A setting of 0 produces the smallest size.

▲▼

Topic

LegendSize

Related

Legend property page
LegendPos
LegendStyle
LegendText
FontUse
FontName
FontSize
FontStyle

LegendStyle property

Sets legend text to color or monochrome

Syntax

[*form*.]*graph*.**LegendStyle**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Legend text in color set by Foreground property (default)
1	Legend text in same colors as legend markers

Description

The LegendStyle property gives you the option of displaying the text of graph legends in color. By default, legend text takes the color specified by the Foreground property (which has a default of automatic black or white, whichever provides the most contrast).

If you set LegendStyle to 1, the legend text for each data set is displayed in the same color as the legend marker (block, line, or symbol) for that set.

▲▼

Topic

LegendStyle

Related

LegendPos

LegendSize

LegendText

LegendText property

Sets text for graph legend

Syntax

```
[form.]graph.LegendText[ = "legend text"]
```

Data type

String (up to 80 characters)

Description

The LegendText property sets the text for a graph legend. The text strings you provide display beside a graph (the legend's position is specified by the LegendPos property) along with a sample of the color, pattern, or symbol for the corresponding data set or data point. The Graph control automatically determines the spacing between items on a legend.

Pie charts, bubble graphs, and bar graphs based on a single data set need one LegendText string for each data *point*. All other graph types need one LegendText string for each data *set*.

LegendText is a one-dimensional array-based property indexed by either the ThisPoint or ThisSet property (see the IndexStyle property for details on the Graph control's indexing of array-based properties). You can use the AutoInc property to automatically increment the ThisPoint or ThisSet counter each time you set a LegendText element.

Each LegendText string can contain no more than 80 characters. Also, if the legend doesn't display, it may mean that you've entered LegendText strings too long for the space available. Increase the width of the graphing window to make the legend appear.

You can introduce a line break by inserting the newline character -- CHR$(10) -- in the string. This forces a new line to start at the point where the newline character appears, overriding automatic word-wrap. Forcing a new line is likely to increase the area required for the text and may lead to text being omitted if there is not enough room for it. Since the newline character is unprintable, it cannot be inserted when typing text; it can only be inserted with code.

If you want to remove a legend, deleting each LegendText string won't do the job--that simply replaces the previous strings with null strings. To actually remove the legend, you have to set the DataReset property to 5 (LegendText).

▲▼

Topic

LegendText

Related

Legend property page
LegendField
LegendPos
LegendSize
LegendStyle
BackgroundUse
BackgroundStyle
Background
FontUse
FontName
FontStyle
FontSize
Legend()
NewLineChar

LimitFillPattern property

Sets the fill pattern of the shaded area above and below limit lines

Syntax

```
[form.]graph.LimitFillPattern[ = setting]
```

Data type

Integer (0-7 and 16-31 enumerated)

Settings

Twenty four fill patterns are available. See PatternData for a list.

Description

The LimitFillPattern property sets the pattern for filling areas above and below limit lines. The default value is 1, which is a null pattern that results in no shading.

Note: Patterns 16 to 31 are bitmapped and may lose definition on high resolution printers.

▲▼

Topic

LimitFillPattern

Related

LimitLinePattern

LimitLines

PatternData

Trends property page

LimitHighLabel property

Places label text next to the high limit line

Syntax

```
[form.]graph.LimitHighLabel[ = data]
```

Data type

String

Description

The LimitHighLabel property places label text next to the high limit line. If the area above or below the limit line is shaded, it places text on the opposite side.

▲▼

Topic

LimitHighLabel

Related

LimitLines

LimitHighValue

LimitLowLabel

Trends property page

LimitHighValue property

Sets the position of the high limit line

Syntax

```
[form.]graph.LimitHighValue[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The LimitHighValue property defines the position on the graph where the high limit line is drawn, as enabled by the LimitLines property. It is expressed in the same units as the data in the graph.

▲▼

Topic

LimitHighValue

Related

LimitHighLabel
LimitLowValue
LimitLines
MissingData
Trends property page

LimitLinePattern property

Defines the line pattern of the high and low limit lines

Syntax

[*form.*]*graph*.**LimitLinePattern**[= *setting*]

Data type

Integer (0-5 enumerated)

Settings

Six line patterns are available. See PatternData for a list.

Description

The LimitLinePattern property defines the line style for limit lines as enabled by the LimitLines property.

▲▼

Topic

LimitLinePattern

Related

ForegroundUse
LimitFillPattern
LimitHighValue
LimitLines
LimitLowValue
PatternData
Trends property page

LimitLines property

Enables limit lines to be superimposed on graphs

Syntax

[*form.*]*graph*.**LimitLines**[= *settings*]

Data type

Integer (0-4 enumerated)

Settings

0	Off
1	High line only
2	Low line only
3	Both high and low lines
4	Both high and low lines with shading between the lines

Description

The LimitLines property superimposes lines on graphs to highlight data that falls outside prescribed limits. Both a high and low limit line can be shown. Text can be attached to the lines, and the areas either inside or outside the limits can be shaded with a pattern. Limit lines are not drawn on pie, polar, time-series, or any 3D graphs.

If a LimitFillPattern is set to 1 (default null value), the graph contains no shading. If it is set to any other pattern index, the areas above the high limit line and below the low limit line are filled to the extremes of the axes. If the LimitLines property is set to 4, the area between the lines is shaded.

Text is printed next to the line if a string is passed in LimitHighLabel or LimitLowLabel . No text is shown if these properties are null. When there is no shading, text is printed immediately above the high line and immediately below the low line. When there is shading, text is placed on the side of the line opposite the shading.

The color of the lines and shading is set by the Foreground property with ForegroundUse set to 14.

▲▼

Topic

LimitLines

Related

ForegroundUse
LimitFillPattern
LimitHighLabel
LimitHighValue
LimitLinePattern
LimitLowLabel
LimitLowValue
MissingData
Trends property page

LimitLowLabel property

Places label text next to the low limit line

Syntax

```
[form.]graph.LimitLowLabel[ = data]
```

Data type

String

Description

The LimitLowLabel property places label text next to the low limit line, as enabled by LimitLines . If the area above or below the limit line is shaded, text is placed on the opposite side.

▲▼

Topic

LimitLowLabel

Related

LimitLines

LimitHighLabel

LimitLowValue

Trends property page

LimitLowValue property

Sets the position of the low limit line

Syntax

```
[form.]graph.LimitLowValue[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The LimitLowValue property defines the position on the graph where the low limit line is drawn, as enabled by LimitLines . It is expressed in the same units as the data in the graph.

▲▼

Topic

LimitLowValue

Related

LimitHighValue
LimitLines
LimitLowLabel
MissingData
Trends property page

LineStats property

Enables statistical lines and curves

Syntax

```
[form.]graph.LineStats[ = setting]
```

Data type

Integer (0-31, enumerated)

Settings

0	No statistical lines (default)
1	Mean line
2	Min/max line
3	Mean and min/max lines
4	Standard-deviation line
5	Standard-deviation and mean lines
6	Standard-deviation and min/max lines
7	Standard-deviation, min/max, and mean lines
8	Best-fit line
9	Best-fit and mean lines
10	Best-fit and min/max lines
11	Best-fit, min/max, and mean lines
12	Best-fit and standard-deviation lines
13	Best-fit, standard-deviation, and mean lines
14	Best-fit, standard-deviation, and min/max lines
15	All lines--best-fit, standard-deviation, min/max, and mean
16-31	Same as above, with curve fitting--add 16 to each of the above settings to enable a curve (curves aren't available for candlestick graphs, for which settings 16-31 duplicate settings 0-15)

Description

The LineStats property lets you superimpose statistical lines and curves on several graph types. Statistical lines are available for line (including lin/log, log/lin, log/log, and overlay line graphs), 2D scatter, high-low-close, open-high-low-close, candlestick, and box-whisker (parametric data only) graphs. Curves are available for all of these graph types except candlestick.

The LineStats settings offer five options:

- **Mean** plots a line parallel to the X axis through the mean (average) value of the data set.
- **Min/max** plots a pair of lines parallel to the X axis through the maximum and minimum values of the data set.
- **Standard deviation** plots a pair of lines parallel to the X axis through the standard deviation of the data set above and below the mean.
- **Best fit** plots a first-order regression fit--a straight line fitted to the trend of data points. It's identical to the polynomial curve of order 1 (CurveType setting 0, CurveOrder setting 1).
- **Curve** plots a curve defined by the CurveType, CurveSteps, and CurveOrder properties.

Number of lines or curves

The number of statistical lines or curves drawn depends on the graph type:

- **For line and scatter graphs,** you get one line (or pair of lines for min/max and standard deviation) per data set.
- **For high-low-close, open-high-low-close, and candlestick graphs,** you get one line (or pair of lines) for the entire graph, based on the close values.
- **For box-whisker graphs,** you get one line (or pair of lines) for the entire graph, based on the median values.

Overlay graphs

If you enable statistical lines or curves when an overlay graph is present, those lines or curves are applied to both the overlay graph *and* the primary graph if the primary graph type supports statistical lines or curves. This means you'll get two sets of lines or curves within the same axes except when the primary graph is bar or area, in which case only the overlay graph has the lines or curves.

Color of lines

By default, statistical lines and curves are drawn in the same color as their associated data sets. You can set your own colors for lines and curves using the ForegroundUse and Foreground properties. Note that any colors you set are applied to that type of line or curve for *all* data sets.

▲▼

Topic

LineStats

Related

CurveOrder
CurveType
CurveSteps
TrendSets
OverlayTrendSets
Trends property page

MapFile property

Names the file for an image map

Syntax

`[form.]graph.MapFile[ = filename]`

Note: This property is not available with the VBX interface.

Data type

String

Setting

Path (optional), file name and file extension. If you don't specify a path, the file is written to the current directory.

Server maps. A server-side image map is stored in a special text file external to the HTML document that references it. The convention is to use the same base file name for both the image and the map. Often the map file's extension is .MAP, though other extensions are usually acceptable.

Client maps. A client-side image map is stored within the HTML document that references it. The setting for MapFile should be the file name and extension of the HTML document.

Description

An image of a graph can be saved in a format recognized by web browsers by setting the ImageFile and DrawMode properties. (See ImageFile for details.) Data points on the graph can be mapped on the image so that when the image is viewed as part of a web page mouse clicks will trigger jumps to another web page. The MapFile property specifies the name for the image map file.

Creating an image map

Image maps are useless without an image, so the first step is to write an image of the graph to a file. For detailed instructions, see the entry for ImageFile.

To map the image, select a map format by setting the MapFormat property. If you select setting 0 (Client map), you'll need to set the MapName property too. Next specify a file name by setting MapFile. If you want hot link addresses to be written to the file, set MapRefStrings.

Hot spot coordinates are generated by the Graph Control's hot-graph feature, so set the Hot property to 1 (On).

Finally, create the map by setting DrawMode to 12 (Create map) or 13 (Append map).

Note: If you are creating a client map, you'll probably want to set DrawMode to 13 (Append map). Setting DrawMode to 12 (Create map) overwrites the file if it already exists.

Linking an image to a map

Mapped images must be displayed inline using the HTML tag IMG. The image is linked to its map by one of the tag's attributes, either ISMAP or USEMAP. Which attribute you use, as well as how you point to the map's location, depends on whether you have a server or a client map.

Server maps are referenced by an image tag nested in an anchor tag:

```
<A HREF="/cgi-bin/imagemap/image.map">
```

```
<IMG SRC="image.jpg" ISMAP>
</A>
```

The ISMAP attribute of the IMG tag identifies the picture as having a server-side image map. Clicking the picture causes the browser to send the server the coordinates of the click and a reference to the map. The server looks up the map, matches the coordinates to one of its regions, and then serves up the corresponding page.

Note that the anchor's HREF attribute points to the name and location of the map file. Use the file name you set in the MapFile property.

Client maps are referenced by the IMG tag alone:

```
<IMG SRC="image.jpg" USEMAP="#mapname">
```

The USEMAP attribute identifies the picture as having a client-side image map. Clicking the picture causes the browser to process the named map and request the appropriate page from the server.

Note that the USEMAP attribute points to the map's name. Use the name you set in the MapName property.

Client vs server maps

The best reason for using server maps is that any browser supporting HTML 2.0 or higher can handle them.

Client maps, on the other hand, have the great advantage that the browser does all the processing, reducing the load on the server and speeding response time. The trade off is that not all browsers support client maps. If the browser supports the HTML 3.0 specification, it will process a client map. If not, the browser will ignore it.

Fortunately, HTML 3.0 provides a way to use both:

```
<A HREF="/cgi-bin/imagemap/image.map">
<IMG SRC="image.jpg" ISMAP USEMAP="#maptag"></A>
```

The USEMAP attribute has precedence over ISMAP, so if the browser supports client maps, it will use the client map. Otherwise, it will ignore USEMAP but see ISMAP and use the server map.

Example

The code listed below writes a JPEG image of a pie graph, creates an HTML document to display the image, and inserts an image map with links to external documents. The linked documents are not created, though a complete procedure would most likely create them as well.

```
Private Sub cmdMap_Click()
Dim i%, temp%
Dim Filename As String

'Set the working directory
ChDrive "F:"
ChDir "F:\Web\GServer"
With Graph1
    .GraphTitle = "Sales by Category"
    Filename = "Categories"
    'Save the graph as a JPEG image
    .ImageFile = Filename & ".jpg"
    .DrawMode = 3   'Blit
    .DrawMode = 6   'Write image file

    '
    'Create a client-side image map
    '
    .MapFile = Filename & ".htm"
    .MapFormat = 0  'Client map
    .MapName = Filename
    'Store a URL for each point
    For i% = 1 To .NumPoints
        .MapRefString(i%) = "Point-" & i% & ".htm"
    Next i%
    'Prepare the header for the HTML file
    Open .MapFile For Output As #1
    Print #1, "<HTML>"
    Print #1, "<HEAD>"
    Print #1, "<TITLE>" & .GraphTitle & "</TITLE>"
    Print #1, "</HEAD>"
    Print #1, "<BODY>"
    Print #1, "<IMG SRC=" &_
        Chr(34) & .ImageFile & Chr(34) & " USEMAP=" & Chr(34) & "#"
& .MapName & Chr(34) & ">"
    Close #1
    'Append image coordinate map
    temp% = .Hot
    .Hot = 1        'On
    .DrawMode = 13  'Append map file
    .Hot = temp%
    'Append footer to the HTML file
    Open .MapFile For Append As #1
    Print #1, "</BODY>"
    Print #1, "</HTML>"
    Close #1
End With
End Sub
```

The resulting HTML is listed below. (For purposes of this illustration, we've omitted each hot spot's coordinate list.)

```
<HTML>
<HEAD>
<TITLE>Sales by Category</TITLE>
</HEAD>
<BODY>
<IMG SRC="Categories.jpg" USEMAP="#Categories">
<MAP NAME="Categories">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-8.htm">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-7.htm">
```

```
<AREA SHAPE="POLYGON" COORDS="…"_HREF="Point-6.htm">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-5.htm">
<AREA SHAPE="POLYGON" COORDS="…"_HREF="Point-4.htm">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-3.htm">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-2.htm">
<AREA SHAPE="POLYGON" COORDS="…" HREF="Point-1.htm">
</MAP>
</BODY>
</HTML>
```

▲▼

Topic

MapFile

Related

DrawMode
Hot
ImageFile
MapFormat
MapName
MapRefStrings

MapFormat property

Selects the format for an image map

Syntax

```
[form.]graph.MapFormat[ = setting]
```

Note: This property is not available with the VBX interface.

Data type

Integer (0-2, enumerated)

Settings

0	Client map
1	NCSA (National Center for Supercomputing Applications) server map
2	CERN (European Laboratory for Particle Physics) server map

Description

The MapFormat property selects a format for image maps created by the control. For details on creating image maps, see the entry for MapFile.

Hot spot definitions in the image map will take one of the following forms, depending on the setting for MapFormat:

```
<! Client map >
<AREA SHAPE=shape COORDS=x1,y1 x2,y2. . . HREF=url>

# NCSA server map
shape url x1,y1 x2,y2. . .

# CERN server map
shape (x1,y1) (x2,y2). . . url
```

One line is created for each data point in the graph. Shape and coordinate information are generated automatically from the displayed graph. The address link (*url*) for each hot spot is taken from a corresponding element of the MapRefStrings property.

With client maps, hot spot definitions are framed by MAP tags:

```
<MAP NAME=mapname>
<AREA SHAPE=shape COORDS=x1,y1 x2,y2. . . HREF=url>
. . .
</MAP>
```

The NAME attribute is taken from the setting of the MapName property.

▲▼

Topic

MapFormat

Related

DrawMode

Hot

ImageFile

MapFile

MapName

MapRefStrings

MapName property

Sets the Name attribute for a client image map

Syntax

`[`*`form`*`.]`*`graph`*`.`**`MapName`**`[ = `*`setting`*`]`

Note: This property is not available with the VBX interface.

Data type

String

Setting

The name for a client image map.

Description

The MapName property names a client-side image map.

Each map within an HTML document must be uniquely identified using the NAME attribute of the MAP tag.

```
<MAP NAME="mapname">
    <AREA … >
    <AREA … >
</MAP>
```

When the Graph Control creates a client map, it inserts the current setting of the MapName property after NAME= in the MAP tag. If the MapName property has not been set, no map name is inserted.

▲▼

Topic

MapName

Related

DrawMode
Hot
ImageFile
MapFile
MapFormat
MapRefStrings

MapRefString() run-time property

Alternative way to store URLs for an image map at run time

Syntax

[*form*.]*graph*.**MapRefString(*index*%)**[= *setting*]

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the MapRefStrings property.

Data type

String

Settings

Same as for the MapRefStrings property.

Description

The MapRefString() property gives you an alternative method for setting the values normally set by the MapRefStrings property. MapRefString() is available only at run time.

Both MapRefStrings and MapRefString() store values in a two-dimensional array. With MapRefStrings you must first point to an element of the array by setting ThisSet and ThisPoint, and then store a value by setting MapRefStrings. You can avoid having to set ThisPoint if you use MapRefString(). Rather than setting ThisPoint, simply pass the number of an array element as a parameter of MapRefString().

▲▼

Topic

MapRefString()

Related

MapRefStrings

MapRefStrings property

Stores an array of URLs for use in an image map

Syntax

```
[form.]graph.MapRefStrings[ = setting]
```

Note: This property is not available with the VBX interface.

Data type

String

Settings

A destination address, in URL form, for a hot spot corresponding to each data point in the graph.

Client maps can use either partial URLs (bar.htm) or fully qualified URLs (http://www.foo.com/bar.htm). Server maps often require that URLs be fully qualified. Consult your server documentation.

Description

MapRefStrings is a two-dimensional array property indexed by ThisSet and ThisPoint. It is designed to store a hypertext link for each hot spot defined in an image map.

When an image map is created, hot spots are defined for every data point on the graph. For each data point, a string from the corresponding element of MapRefStrings is written to the map file, specifying the destination to which the hot spot is linked.

It is not necessary to set MapRefStrings. If the property is empty, no substitution will be made in the map file, and it is left for your program to perform its own substitutions.

To clear the MapRefStrings array, set DataReset to 21.

▲▼

Topic

MapRefStrings

Related

DrawMode
Hot
ImageFile
MapFile
MapFormat
MapName
MapRefString()

MissingData property

Sets the mode for flagging data points as missing

Syntax

`[form.]graph.MissingData[ = setting]`

Data type

Integer (0-5 enumerated)

Settings

0	Use ExtraData array only (Default)
1	Exclude zero data
2	Exclude data below LimitLowValue
3	Exclude data above LimitHighValue
4	Exclude data below LimitLowValue or above LimitHighValue
5	Exclude data above LimitLowValue and below LimitHighValue

Description

The MissingData property applies to the graph types line, scatter and tape. It sets the mode for flagging which data values should be excluded from the graph. Missing data points are not displayed and are ignored in the calculation of the scale of the axis and any superimposed statistical or trends lines.

With the default setting (0), missing data may be individually flagged using the ExtraData array. Also excluded are any log data below the value 1E-50, as these values are too small to be shown on a graph.

The default operation applies with all settings. Any points flagged as a result of comparison with the LimitLowValue and LimitHighValue are in addition to those individually flagged in the ExtraData array or omitted as being too small to render in a log graph. While values must be assigned to LimitLowValue and LimitHighValue, it is not necessary for limit lines actually to be displayed.

▲▼

Topic

MissingData

Related

Data property page
ExtraData
LimitLines
LimitHighValue
LimitLowValue

MissingLineMode property

Selects option for bridging gaps caused by missing data

Syntax

[*form*.]*graph*.**MissingLineMode**[= *setting*]

Data type

Integer (0-4, enumerated)

Settings

0	No bridging line (default)
1	Bridge with same line style as the graph
2	Bridge with a thin line
3	Bridge with a broken line
4	Bridge with a thick line

Description

The MissingLineMode property applies to the graph type line, including log variants. It sets the mode for bridging gaps in the line left by missing or excluded data values. If there is an overlay graph, the setting for MissingLineMode is applied to the overlay line as well.

▲▼

Topic

MissingLineMode

Related

ExtraData
MissingData

MousePointer property

Sets appearance of mouse pointer displayed in graphing window

Syntax

[*form*.]*graph*.**MousePointer**[= *setting*]

Data type

Integer (0-6, enumerated)

Settings

0	Arrow (default)
1	I-beam
2	Hourglass
3	Crosshairs
4	Up arrow
5	Size
6	Icon

Description

The MousePointer property sets the appearance of the mouse pointer, effective when the pointer is within the graphing window.

The MousePointer setting takes effect only at run time and only when the graph's window has been activated by the Hot or SDKMouse property.

▲▼

Topic

MousePointer

Related

Design property page

Hot

SDKMouse

NewLineChar property

Defines the character used for imposing line-breaks

Syntax

[*form.*]*graph*.**NewLineChar**[= *data*]

Data type

Integer (32 - 127)

Setting

ASCII value of a new line character.

Description

The NewLineChar property defines the special character that--when encountered in a text string--forces a line break in a title or legend, overruling automatic word-wrap. Forcing new lines is likely to increase the area required for the text and may lead to the text being omitted if there isn't enough room.

By default the NewLineChar is a space. This character is unique in that is does not force a line break. You can force a line break by setting this property to the ASCII value of any other printable character. If you set...

```
NewLineChar = 126    ' (~ character)
```

...the string "Hello~World" will be written on two lines:

Hello
World

▲▼

Topic

NewLineChar

Related

BottomTitle
Design property page
GraphTitle
LeftTitle
LegendText

NumPoints property

Sets number of points per data set

Syntax

```
[form.]graph.NumPoints[ = setting]
```

Data type

Integer (1-6000)

Settings

The default is 5 points. Valid settings depend on whether you are using 16- or 32-bit Graphics Server.

16 bit. The product of NumPoints times NumSets must be no greater than 8000.

32 bit. The product of NumPoints times NumSets must be no greater than 128,000.

Description

The NumPoints property sets the number of points in the data sets making up a graph.

You can change the NumPoints setting at any time. If you set a NumPoints value that's less than the number of points carried by the GraphData property, the excess GraphData values are discarded. If you set a NumPoints value greater than the number of points in GraphData, the Graph control creates additional points with values of 0.

When the Graph control is bound to a Data control, the graph's NumPoints property is automatically set to equal the number of records in the Data control's Recordset.

You can graph a subset of the data by telling Graphics Server to look at a range of points. For more information, see RangeMax and RangeMin .

▲▼

Topic

NumPoints

Related

NumSets
RangeMax
RangeMin
AutoInc

NumSets property

Sets number of data sets to be graphed

Syntax

```
[form.]graph.NumSets[ = setting]
```

Data type

Integer (1-6000)

Settings

Valid settings are integers from 1 to 6000 (default is 1).

Note In addition to the maximum NumSets setting of 6000, the total number of points in any graph (the product of the NumPoints and NumSets property settings) is also limited to 6000.

Description

The NumSets property sets the number of data sets making up a graph, dimensioning the array of values carried by GraphData, XPosData, and certain other array-based properties. For most graph types, NumSets can be any value between 1 and 6000. However, there are several exceptions:

- **Pie charts** use only one data set, so set NumSets to 1.
- **Bubble graphs** need two sets of GraphData property values, so set NumSets to 2.
- **Gantt charts and surface graphs** need at least two sets of GraphData values, so set NumSets to 2 or higher.
- **High-low-close graphs** need three sets of GraphData values--high, low, and close--so set NumSets to 3.
- **Open-high-low-close and candlestick graphs** need four sets of GraphData values--open, high, low, and close--so set NumSets to 4.
- **Box-whisker graphs** need enough sets of GraphData values to hold all of the values you want to graph. With parametric data, this is always seven data sets, so set NumSets to 7. With raw data, you may need a higher NumSets value if you have more than seven values making up your sample groups.

Changing the NumSets value

You can change the NumSets value at any time. In general, the Graph control minimizes the effect of a change in NumSets on your graph:

- If you set a new NumSets value *greater* than the number of sets currently carried by GraphData or another array-based property, the Graph control creates additional sets, with all points set initially to 0 or null.

  One exception is the XPosData (X positions) property. If you've defined only one set of XPosData, the Graph control applies those values to all sets rather than assigning 0 to each point.
- If you set a new NumSets value *lower* than the number of sets carried by GraphData or another array-based property, the excess values aren't discarded; you can access them later by setting a higher NumSets. To reset an array-based property, use the DataReset property.

▲▼

Topic

NumSets

Related

NumPoints
AutoInc

NView run-time property

Specifies the Graphics Server view in which to display the graph

Syntax

[*form*.]*graph*.**NView**[= *setting*]

Data type

Integer

Settings

A view number. The default is 0 (full graphing window).

Description

NView is a read/write property that gets or sets the current Graphics Server view number within the NWin graphing window.

You can open a new view within NWin using the Graphics Server function GSOpenView, then set NView to the new view number. Subsequently, the Graph control will draw into this *view* but copy, print, and write a bitmap or metafile at the *window* level. This distinction lets you use DLL functions for drawing objects and text while still using the Graph control's window-level printing and copying techniques.

Example

The following code draws four graphs in the window of a single instance of the Graph control:

```
Private Sub Graph1_Click()
Dim w!, h!
Dim r%, x%, y%, v%

w! = GSGetVXExt() / 2000
h! = GSGetVYExt() / 2000
Graph1.GraphData = Array(1, 2, 3, 4, 5)
Graph1.LabelText = Array("One", "Two", "Three", "Four", "Five")
For y% = 0 To 1
    For x% = 0 To 1
        v% = GSOpenView(Graph1.nWin, x% * w!, y% * h!, w!, h!, 1000)
        r% = GSClearView(1)
        Graph1.nView = v%
        Graph1.GraphType = v%
        Graph1.DrawMode = 2
    Next x%
Next y%

End Sub
```

▲▼

Topic

NView

Related

NWin

NWin run-time property

Specifies the Graphics Server window in which to display the graph

Syntax

```
r% = GSOpenView([form.]graph.NWin, xOrg%, yOrg, Width!, Height!, yExtent!)
```

Data type

Integer

Description

NWin is a read-only property that returns the number of the Graphics Server window for an instance of the Graph control. You need the window number if you want to use standard Graphics Server function calls to draw into the same window as the Graph control.

For additional information, see the "Interfacing guide" in *The Graph Library* manual.

▲▼

Topic

NWin

Related

NView

OverlayColor property

Sets color of overlay graph

Syntax

[*form*.]*graph*.**OverlayColor**[= *setting*]

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The OverlayColor property sets the color of the elements (lines, symbols, and/or sticks) of an overlay graph. Because this edition of the Graph control allows only one data set for an overlay graph, OverlayColor sets a single value.

▲▼

Topic

OverlayColor

Related

Overlay property page
OverlayGraph
OverlayGraphType
OverlayGraphStyle
OverlayGraphData
ColorData
Palette

OverlayData() run-time property

Alternative way to set data to be graphed in overlay graph at run time

Syntax

`[form.][graph.]OverlayData(index%)[ = data]`

Parameter

index% Element (point) number of array normally set by OverlayGraphData property

Data type

Numeric (integer, long, single, double)

Description

The OverlayData() run-time property sets the data values for an overlay graph. It's a run-time alternative to the OverlayGraphData property.

Overlay graphs are stored as a one-dimensional array (this edition of the Graph control allows only one data set for overlay graphs). Unlike OverlayGraphData, OverlayData() doesn't require you to use the ThisPoint property to index the array. Instead, you provide the element (point) number as your OverlayData() parameter.

Example

If you want to assign a value of 10 to the third data point in an overlay graph, you can set up your code in two ways:

- If you use the OverlayGraphData property, you set the ThisPoint property first:

  ```
  Graph1.ThisPoint = 3
  Graph1.OverlayGraphData = 10
  ```

- If you use the OverlayData() property, you don't have to set ThisPoint:

  ```
  Graph1.OverlayData(3) = 10
  ```

▲▼

Topic

OverlayData()

Related

DataReset
Overlay property page
OverlayGraphData

OverlayExtra() run-time property

Alternative way to flag an overlay's missing points at run time

Syntax

[*form*.]*graph*.**OverlayExtra(*index*%)**[= *setting*]

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the OverlayExtraData property.

Data type

Integer

Settings

0	Point is shown (default)
non-zero	Point is not shown

Description

The OverlayExtra() property gives you an alternative method for setting the values normally set by the OverlayExtraData property. OverlayExtra() is available only at run time.

Both OverlayExtraData and OverlayExtra() store values in a one-dimensional array. With OverlayExtraData you must first point to an element of the array by setting ThisPoint, and then store a value by setting OverlayExtraData. You can skip the first step by using OverlayExtra(). Rather than setting ThisPoint, you can simply pass the number of an array element as a parameter of OverlayExtra().

▲▼

Topic

OverlayExtra()

Related

OverlayExtraData

OverlayExtraData property

Stores flags to mark missing points for an overlay graph

Syntax

[*form*.]*graph*.**OverlayExtra**[= *setting*]

Note: This property is not available with the VBX interface.

Data type

Integer

Settings

0	Point is shown (default)
non-zero	Point is not shown

Description

OverlayExtraData is a one-dimensional array property indexed by ThisPoint. It can be used to flag individual data points so that they are treated as missing.

If OverlayExtraData has never been set, the setting for ExtraData applies. Points flagged as missing for the primary graph by settings in ExtraData will also be treated as missing for the overlay.

If you do not want points flagged in ExtraData to be treated as missing for the overlay, set all elements of OverlayExtraData to zero. If you want to mark different points for each graph, flag the primary graph's missing points by setting ExtraData and the overlay graph's missing points by setting OverlayExtraData.

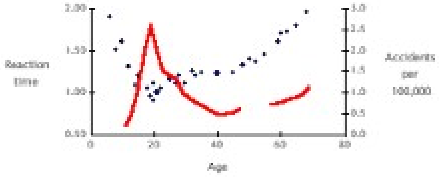
To clear OverlayExtraData array, set DataReset to 24.

Example

The code listed below flags certain points of an overlay graph as missing by setting corresponding elements of OverlayExtraData to 1. When the graph is redrawn, the missing points create a gap in the overlay line graph.

Note that before explicitly setting ThisPoint we turn off the AutoInc property. This is always good practice.

```
Private Sub Command1_Click()
Dim i%
With Graph1
    .AutoInc = 0    'off
    'Make sure the array is emptied of any
    'previously loaded values
    .DataReset = 24
    'Then put non-zeroes into elements
    'we want to treat as missing
    For i% = 1 To .NumPoints
        .ThisPoint = i%
        If .OverlayXPosData > 47_
        And .OverlayXPosData < 56 Then
            .OverlayExtraData = 1
        End If
    Next i%
    'Redraw the graph
    .DrawMode = 3
End With
End Sub
```



▲▼

Topic

OverlayExtraData

Related

OverlayExtra()
OverlayGraph
OverlayGraphData
OverlayXPosData

OverlayGraph property

Allows second graph to be overlaid on primary graph

Syntax

[*form*.]*graph*.**OverlayGraph**[= *setting*]

Data type

Integer (0-2, enumerated)

Settings

0	Off (default)
1	On, share the Y axis with the primary graph
2	On, draw with a separate Y axis

Description

The OverlayGraph property, when set to on, lets you overlay a second graph on the primary graph. You can draw an overlay graph on the same Y scale as the primary graph or on its own scale, with a separate Y axis on the right side. The X scale for an overlay graph is always the same as for the primary graph.

Overlay graphs are available for the following primary graph types, all in 2D form only:

Area
Bar (vertical)
Candlestick
High-low-close
Line
Open-high-low-close
Scatter

▲▼

Topic

OverlayGraph

Related

Overlay property page
OverlayGraphType
OverlayGraphStyle
OverlayExtraData
OverlayGraphData
OverlayXPosData
OverlayColor
OverlayPattern
OverlayPatternedLines
OverlaySymbol
OverlayThickLines
OverlayTrendSets
YAxisUse
YAxisStyle
YAxisMax
YAxisMin
YAxisTicks

OverlayGraphData property

Sets data to be graphed in overlay graph

Syntax

[*form*.]*graph*.**OverlayGraphData**[= *data*]

Data type

Numeric (integer, long, single, double)

Description

The OverlayGraphData property sets the data values for an overlay graph. The values are stored as a one-dimensional array (this edition of the Graph control allows only one data set for overlay graphs). The array is indexed by the ThisPoint property. You can use the AutoInc property to automatically increment the ThisPoint value as you enter data.

Topic

OverlayGraphData

Related

DataReset
Overlay property page
OverlayGraph
OverlayGraphType
OverlayGraphStyle
OverlayData()

OverlayGraphStyle property

Sets graph style for overlay graph

Syntax

```
[form.]graph.OverlayGraphStyle[ = setting]
```

Data type

Integer (0-7, enumerated)

Settings

0	Lines only (default)
1	Symbols
2	Sticks between points and Y origin
3	Sticks and symbols
4	Lines
5	Lines and symbols
6	Lines and sticks
7	Lines, sticks, and symbols

Description

The OverlayGraphStyle property sets the graph style for an overlay graph. Since the only overlay graph type now available is the line graph, OverlayGraphStyle offers only line graph styling options--line only, symbols only, sticks only, or sticks and symbols.

To draw the overlay graph with thick or patterned lines, use the ThickLines or PatternedLines property.

Topic

OverlayGraphStyle

Related

Overlay property page
OverlayGraph
OverlayGraphType

OverlayGraphType property

Sets graph type for overlay graph

Syntax

```
[form.]graph.OverlayGraphType[ = setting]
```

Data type

Integer (enumerated)

Setting

0	Line (only setting now available)

Description

The OverlayGraphType() setting sets the graph type of an overlay graph. The only type now available is the line graph (the default setting of 0), and any other setting is ignored. Other types may be added in a future edition.

Topic

OverlayGraphType

Related

OverlayGraph
OverlayGraphStyle

OverlayPattern property

Sets line pattern or thickness for overlay graph

Syntax

```
[form.]graph.OverlayPattern[ = setting]
```

Data type

Integer (0-5, enumerated)

Settings

Settings vary according to context:

Line pattern

Valid settings are integers from 0 to 5. The setting selects one of six line patterns available, including an "invisible" line (setting 5).

Line thickness

Valid settings are integers from 1 to 5. The setting is the line width in pixels.

Description

The OverlayPattern property has two uses in overlay graphs:

- To set a line pattern for patterned lines when OverlayPatternedLines is set to on.
- To set a line width (in pixels) for thick lines when OverlayThickLines is set to on.

Note The OverlayPatternedLines and OverlayThickLines properties are mutually exclusive. When one is on, the other is always off.

Because this edition of the Graph control allows only one data set for an overlay graph, OverlayPattern sets a single value.

▲▼

Topic

OverlayPattern

Related

Overlay property page
OverlayGraph
OverlayGraphData
OverlayGraphStyle
OverlayGraphType

OverlayPatternedLines property

Sets style of lines in overlay graph

Syntax

[*form*.]*graph*.**OverlayPatternedLines**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The OverlayPatternedLines property sets the style of the lines connecting data points in an overlay graph.

When OverlayPatternedLines is on, the graph is plotted with a dotted line--the line you get when you set the OverlayPattern property to 1--unless you've set a different OverlayPattern value.

The PatternedLines property is mutually exclusive with the OverlayThickLines property. When you set one of these properties to on (1), the other is automatically set to off (0).

▲▼

Topic

OverlayPatternedLines

Related

Overlay property page
OverlayGraph
OverlayGraphStyle

OverlaySymbol property

Sets symbol design for overlay graph

Syntax

[*form*.]*graph*.**OverlaySymbol**[= *setting*]

Data type

Integer (0-13, enumerated)

Description

The OverlaySymbol property sets the symbol design for overlay graphs. Fourteen symbol options are available.

Because this edition of the Graph control allows only one data set for an overlay graph, OverlaySymbol sets a single value.

▲▼

Topic

OverlaySymbol

Related

Overlay property page
OverlayGraph
OverlayGraphType
OverlayGraphStyle
OverlayGraphData

OverlayThickLines property

Enables thick lines for overlay graph

Syntax

`[form.]graph.OverlayThickLines[ = setting]`

Data type

Integer (0-1, enumerated)

Settings

0	Off
1	On (default)

Description

The OverlayThickLines property lets you enable a thick line for an overlay graph. By default, the thick line is three pixels thick, although you can set your own thickness using the OverlayPattern property. A thin line (OverlayThickLines set to off) is one pixel thick.

The OverlayThickLines property is mutually exclusive with the OverlayPatternedLines property. When you set one of these properties to on (1), the other is automatically set to off (0).

▲▼

Topic

OverlayThickLines

Related

Overlay property page
OverlayGraph
OverlayGraphStyle

OverlayTrendSet() run-time property

Alternative way to enable trend lines on an overlay graph at run time

Syntax

```
[form.]graph.OverlayTrendSet(index%)[ = setting]
```

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the OverlayTrendSets property. Currently the array has only one element, so *index%* should always be 1.

Data type

Integer (0-31, enumerated)

Settings

Same as for the OverlayTrendSets property.

Description

The OverlayTrendSet() property gives you an alternative method for setting the values normally set by the OverlayTrendSets property. OverlayTrendSet() is available only at run time.

Both OverlayTrendSets and OverlayTrendSet() store values in a one-dimensional array. Currently the array has only one element, so OverlayTrendSet() offers no real advantage over OverlayTrendSets.

▲▼

Topic

OverlayTrendSet()

Related

OverlayTrendSets

OverlayTrendSets property

Enables trend lines for an overlay graph

Syntax

[*form*.]*graph*.**OverlayTrendSets**[= *setting*]

Note: This property is not available with the VBX interface.

Data type

Integer (0-31, enumerated)

Settings

0	No statistical lines (default)
1	Mean line
2	Min/max line
3	Mean and min/max lines
4	Standard-deviation line
5	Standard-deviation and mean lines
6	Standard-deviation and min/max lines
7	Standard-deviation, min/max, and mean lines
8	Best-fit line
9	Best-fit and mean lines
10	Best-fit and min/max lines
11	Best-fit, min/max, and mean lines
12	Best-fit and standard-deviation lines
13	Best-fit, standard-deviation, and mean lines
14	Best-fit, standard-deviation, and min/max lines
15	All lines--best-fit, standard-deviation, min/max, and mean
16-31	Same as above, with curve fitting. Add 16 to each of the above settings to enable a curve. To select which type of curve is drawn, set the CurveType property.

Description

OverlayTrendSets is a one-dimensional array property. Currently it has only one element, so there is no need for an index property as there is with other array properties.

Like the LineStats property, OverlayTrendSets lets you superimpose statistical lines and curves on a graph. With LineStats trend lines are applied to both the primary graph and the overlay. OverlayTrendSets offers greater flexibility. It can be used to enable trend lines for just the overlay graph, to turn off overlay trend lines while displaying trend lines for the primary graph, or to enable a different set of lines for the overlay from those used for the primary graph.

If OverlayTrendSets has never been set, the setting for LineStats applies. Once you have set OverlayTrendSets, the setting for LineStats will be ignored. Before the LineStats setting can take effect again, you must clear the OverlayTrendSets array by setting DataReset to 26.

By default, trend lines are drawn in the same color as the overlay. You can select another color using the ForegroundUse and Foreground properties.

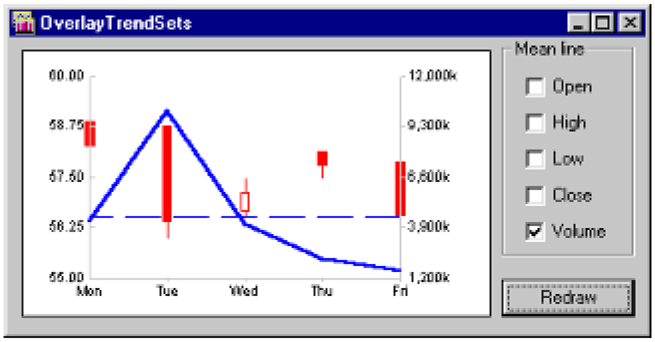
Example

This example assumes an application with a candlestick graph depicting a stock's daily open, high, low and close prices. A line graph showing volume has been overlaid.

In order to enable users to see a mean line for selected data sets, we've set up an array of checkboxes. The first four turn on mean lines for each of the four data sets. The fifth turns on a mean line for the overlay graph.

The form also has a command button for redrawing the graph with a new set of trend lines once a user has made new selections. Code for the command button's click event is listed below.

```
Private Sub cmdRedraw_Click()
Dim i%

Graph1.DataReset = 25 'Clear TrendSets array
Graph1.DataReset = 26 'Clear OverlayTrendSets array

'Loop through the first four checkboxes,
'turning on a mean line for any data set
'that is checked.
For i% = 0 To 3
    If chkSet(i%).Value = Checked Then
        Graph1.ThisSet = i% + 1
        Graph1.TrendSets = 1
    End If
Next i%

'If the volume checkbox is checked...
If chkSet(4).Value = Checked Then
    'turn on a mean line for the overlay
    Graph1.OverlayTrendSets = 1
End If

Graph1.DrawMode = 2   'Redraw the graph
End Sub
```



Mean line

☐ Open
☐ High
☐ Low
☐ Close
☑ Volume

▲▼

Topic

OverlayTrendSets

Related

OverlayTrendSet()

CurveType

LineStats

TrendSets

OverlayXPos() run-time property

Alternative way to set X values for an overlay at run time

Syntax

`[`*form*`.]`*graph*`.`**`OverlayXPos(`*`index%`*`)`**`[ = `*setting*`]`

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the OverlayXPosData property.

Data type

Numeric (integer, long, single, double)

Description

The OverlayXPos() property gives you an alternative method for setting the values normally set by the OverlayXPosData property. OverlayXPos() is available only at run time.

Both OverlayXPosData and OverlayXPos() store values in a one-dimensional array. With OverlayXPosData you must first point to an element of the array by setting ThisPoint, and then store a value by setting OverlayXPosData. You can skip the first step by using OverlayXPos(). Rather than setting ThisPoint, you can simply pass the number of an array element as a parameter of OverlayXPos().

▲▼

Topic

OverlayXPos()

Related

OverlayXPosData

OverlayXPosData property

Sets X values for an overlay graph

Syntax

[*form*.]*graph*.**OverlayXPosData**[= *setting*]

Note: This property is not available with the VBX interface.

Data type

Numeric (integer, long, single, double)

Description

The OverlayXPosData property sets X values for an overlay graph. The values are stored as a one-dimensional array. (This version of the Graph Control allows only one data set for overlay graphs.) The array is indexed by the ThisPoint property.

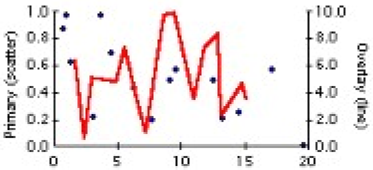
If you load X values for the overlay graph by setting OverlayXPosData, you should also load X values for the primary graph by setting XPosData.

To clear the OverlayXPosData array, set DrawMode to 23.

Example

The code below draws a primary scatter graph with an overlay line graph. Primary and overlay each have a single set of X and Y values. For simplicity's sake, we've used random data for all values.

```
Private Sub Form_Load()
Dim i%

With Graph1
    'Set up primary graph
    .GraphType = 9   'scatter
    .NumPoints = 15
    .NumSets = 1

    'Set up overlay graph
    .OverlayGraph = 2 'second Y axis

    'Label Y axes
    .LeftTitle = "Primary (scatter)"
    .LeftTitleStyle = 1  'up
    .RightTitle = "Overlay (line)"
    .RightTitleStyle = 2 'down

    'Load data
    .AutoInc = 0       'off
    Randomize
    For i% = 1 To .NumPoints
        .ThisPoint = i%
        .GraphData = Rnd
        .XPosData = i% * Rnd * 1.5
        .OverlayGraphData = 10 * Rnd
        .OverlayXPosData = i% + Rnd
    Next i%

    'Draw the graph
    .DrawMode = 2
End With

End Sub
```



▲▼

Topic

OverlayXPosData

Related

OverlayXPos()
OverlayExtraData
OverlayGraphData
XPosData

Palette property

Sets color palette for graph

Syntax

[*form.*]*graph*.**Palette**[= *setting*]

Data type

Integer (0-12, enumerated)

Settings

<table>
<tr><th>Palette setting</th><th>Number of hues</th><th>Description</th></tr>
<tr><td>0</td><td>16</td><td>Solid colors (default)
<table>
<tr><td>0</td><td>Black</td><td>8</td><td>Dark gray</td></tr>
<tr><td>1</td><td>Blue</td><td>9</td><td>Light blue</td></tr>
<tr><td>2</td><td>Green</td><td>10</td><td>Light green</td></tr>
<tr><td>3</td><td>Cyan</td><td>11</td><td>Light cyan</td></tr>
<tr><td>4</td><td>Red</td><td>12</td><td>Light red</td></tr>
<tr><td>5</td><td>Magenta</td><td>13</td><td>Light magenta</td></tr>
<tr><td>6</td><td>Brown</td><td>14</td><td>Yellow</td></tr>
<tr><td>7</td><td>Light gray</td><td>15</td><td>White</td></tr>
</table>
<table>
<tr><td>16</td><td>Automatic black or white, whichever provides higher contrast</td></tr>
</table>
</td></tr>
<tr><td>1</td><td>16</td><td>Pastel
0-15 are the same as setting 1, dithered with white for a pastel effect
16 is automatically black or white, whichever provides higher contrast</td></tr>
<tr><td>2</td><td>16</td><td>Grayscale--0-15 ascend from black to white
0-15 ascend from black to white
16 is automatically black or white, whichever provides higher contrast</td></tr>
<tr><td>3</td><td>128</td><td>Grayscale
0-16 are the same as setting 1
17-31 are half colors of 1-15
32-127 ascend from black to white</td></tr>
<tr><td>4</td><td>128</td><td>Pastel
0-16 are the same as setting 1
17-31 are half colors of 1-15
32-127 are six groups of 16 pastel colors ascending from fully saturated to white:
<table>
<tr><td>32-47</td><td>Red</td></tr>
<tr><td>48-63</td><td>Green</td></tr>
<tr><td>64-79</td><td>Blue</td></tr>
<tr><td>80-95</td><td>Cyan</td></tr>
<tr><td>96-111</td><td>Magenta</td></tr>
<tr><td>112-127</td><td>Yellow</td></tr>
</table>
</td></tr>
</table>

5	128	Solid 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 are six groups of 16 undithered colors in ascending intensity: 32-47 Red 48-63 Green 64-79 Blue 80-95 Cyan 96-111 Magenta 112-127 Yellow
6	128	Graded 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 are two groups of 48 graded hues at two intensities: 32-79 Blend from green to red to blue to red again at full intensity 80-127 Blend from green to red to blue to red again at half intensity
7	128	Black to red 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 ascend from black to red
8	128	Black to green 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 ascend from black to green
9	128	Black to blue 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 ascend from black to blue
10	128	Black to cyan 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 ascend from black to cyan
11	128	Black to magenta 0-16 are the same as setting 1 17--31 are half colors of 1-15 32-127 ascend from black to magenta
12	128	Black to yellow 0-16 are the same as setting 1 17-31 are half colors of 1-15 32-127 ascend from black to yellow

Description

The Palette property sets the Graph control's color palette. The palette determines the choice of colors available through Foreground and other properties that let you assign colors. You can select a palette with either 16 hues (the default) or 128 hues.

16-color palettes

The Graph control's 16-color palettes work with any VGA driver. The default Palette setting (0) gives you a choice of 16 solid colors, and the two other 16-color options give you pastels and grayscales. An additional setting (16) applies either black or white to the selection, whichever provides the most contrast with the rest of the graph.

Note that the names of the 16 "colors" remain the same on the Graph control's menus even if you've chosen pastel or grayscale. For example, with Palette set to grayscale (2), setting an object's color to light blue (9) actually selects a shade of gray about midway between black and white.

128-color palettes

The Graph control's 128-color palettes require a SuperVGA driver. In contrast to the 16-color palettes, the 128-color palettes are *realized*--they can display up to 128 different undithered colors (or grayscales) at once on systems capable of showing them.

Each 128-entry palette contains the standard 16 colors in settings 0-15, an automatic black/white setting at 16, and the half colors of settings 1-15 from 17-31. (Half colors are standard colors dithered with black; the Graph control uses them as the default colors for the sides of 3D bars, among other things.)   Settings 32-127 are created according to the Palette setting you choose.

▲▼

Topic

Palette

Related

Background

Background property page

Pattern() run-time property

Alternative way to set fill pattern, line pattern, or line thickness at run time

Syntax

`[form.]graph.Pattern(index%)[ = setting]`

Parameter

index% Element number of array normally set by PatternData property

Data type

Integer (0-7 and 16-31, enumerated)

Settings

Same as for PatternData property

Description

The Pattern() run-time property gives you an alternative method for setting the values normally set by the PatternData property.

Pattern values are stored as a one-dimensional array, which is normally indexed by either the ThisPoint or ThisSet property. Unlike PatternData, Pattern() doesn't require you to use ThisPoint or ThisSet. Instead, you provide the element (point or set) number as your Pattern() parameter.

Example

If you want to assign fill pattern 7 (crosshatch) to the second data point on a bar graph, you can set up your code in two ways:

- If you use the PatternData property, you set the ThisPoint property first:

  ```
  Graph1.ThisPoint = 2
  Graph1.PatternData = 7
  ```

- If you use the Pattern() property, you don't have to set ThisPoint:

  ```
  Graph1.Pattern(2) = 7
  ```

▲▼

Topic

Pattern()

Related

PatternData

PatternData property

Sets fill pattern, line pattern, or line thickness

Syntax

```
[form.]graph.PatternData[ = setting]
```

Data type

Integer (0-7 and 16-31, enumerated)

Settings

Settings vary according to context:

Fill pattern

Valid settings are integers from 0 to 7 and from 16 to 31. The setting selects one of 24 fill patterns available.

Line pattern

Valid settings are integers from 0 to 5. The setting selects one of six line patterns available, including an "invisible" line (setting 5).

Line thickness

Valid settings are integers from 1 to 5. The setting is the line width in pixels.

Description

The PatternData property has three uses:

- **To set a pattern for solid fills.** PatternData serves this purpose when you're working with pie, area, bar, and bubble graphs.
- **To set a line pattern for patterned lines.** PatternData serves this purpose when the PatternedLines property is set to on and you're working with line, lin/log, log/lin, log/log, and polar graphs, or with scatter graphs with curve fitting.
- **To set a line width (in pixels) for thick lines.** PatternData serves this purpose when the ThickLines property is set to on and you're working with line, lin/log, log/lin, log/log, and polar graphs, or with scatter graphs with curve fitting.

Note The PatternedLines and ThickLines properties are mutually exclusive. When one is on, the other is always off.

Pie charts, bubble graphs, based on a single data set need one PatternData value for each data *point*. All other graph types need one PatternData value for each data *set*.

PatternData is a one-dimensional array-based property indexed by either the ThisPoint or ThisSet property (see the IndexStyle property for details on the Graph control's indexing of array-based properties). You can use the AutoInc property to automatically increment the ThisPoint or ThisSet counter each time you set a PatternData element.

▲▼

Topic

PatternData

Related

AutoInc
DataReset
Markers property page
ThisPoint
ThisSet

PatternedLines property

Sets style of lines

Syntax

[*form*.]*graph*.**PatternedLines**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The PatternedLines property sets the style of the lines connecting data points.

When PatternedLines is on, the graph is plotted with a dotted line--the line you get when you set the PatternData property to 1--unless you've set a different PatternData value.

The PatternedLines property is mutually exclusive with the ThickLines property. When you set one of these properties to on (1), the other is automatically set to off (0).

▲▼

Topic

PatternedLines

Related

Markers property page
PatternData
ThickLines

Perspective property

Sets degree of perspective foreshortening for True3D graphs

Syntax

`[form.]graph.Perspective[ = setting]`

Data type

Integer (0-100)

Settings

Valid settings are integers from 0 to 100 (default is 50).

- A setting of 0 sets the viewing distance at about four times the graph's width.
- A setting of 50 sets the viewing distance at about twice the graph's width.
- A setting of 100 sets the viewing distance about equal to the graph's width.

Description

The Perspective property sets the degree of perspective foreshortening for True3D graphs. The setting represents, in arbitrary units, the perceived "distance" from which the graph is viewed.

If the True3D property is set to 2 (isometric projection), the Perspective property has no effect. In isometric projections, all parallels are preserved.

Topic

Perspective

Related

3d property page
Elevation
Rotation
True3D

Picture run-time property

Passes graph image directly to Picture control

Syntax

[*form*.]*picture*.**Picture** =[*form*.]*graph*.**Picture**

Data type

Integer

Description

The Picture run-time property passes the image of a graph directly to a picture control. If the picture control has a different aspect ratio from the Graph control, the image is stretched or compressed.

The Picture property is read-only and works only at run time.

PrintInfo() run-time property

Sets information needed to print graph at high resolution

Syntax

[*form.*]*graph*.**PrintInfo(*index%*)**[= *setting*]

Data type

Single

Parameter

index% Element number of the PrintInfo() array.

1 Visual Basic printer device context:
`PrintInfo(1) = Printer.hDC`

2 X coordinate of top left corner of graph as it appears on the page, expressed in scale units of the printer

3 Y coordinate of top left corner of graph as it appears on the page, expressed in scale units of the printer

4 Graph width in scale units

5 Graph height in scale units

6 X coordinate of the top left corner of the page:
`PrintInfo(6) = Printer.ScaleLeft`

7 Y coordinate of the top left corner of the page:
`PrintInfo(7) = Printer.ScaleTop`

8 Width of the page in scale units:
`PrintInfo(8) = Printer.ScaleWidth`

9 Height of the page in scale units:
`PrintInfo(9) = Printer.ScaleHeight`

10 Select cloned window or metafile printing:

```
PrintInfo(10) = 0 'Cloned window
PrintInfo(10) = 1 'Metafile
```

11 Select portrait or landscape orientation:

```
PrintInfo(11) = 0 'Portrait
PrintInfo(11) = 1 'Landscape
```

12 Select print size:

```
PrintInfo(12) = 0 'Size on screen
PrintInfo(12) = 1 'Fit to page
```

13 Low-order word pointer to an MFC device context object. This element is used only in AFX applications.

14 High-order word pointer to an MFC device context object. This element is used only in AFX applications.

Description

The PrintInfo() run-time property lets you set the information needed to print a graph into the Printer object's device context. PrintInfo() lets you print graphs at the full resolution of your printer, on the same page as text and other graphics.

The values for PrintInfo() are stored as a one-dimensional array with twelve elements, as listed above.

PrintInfo() works when the DrawMode property is set to 5 (print). If you set DrawMode to 5 without setting PrintInfo() values, only the graph itself is printed, centered at the top of the page.

PrintInfo() works only at run time. At design time, you can print the graph itself by setting DrawMode to 5, but you can't print text or other graphics on that page.

Positioning and sizing the graph

PrintInfo() lets you position and size a graph using the coordinates and units defined by the Printer object's Scale method and ScaleMode property. You start by setting the first element of the PrintInfo() array to the Printer object's device context:

```
Graph1.PrintInfo(1) = Printer.hDC
```

Next, you set elements 6-9 of PrintInfo() to use the Printer object's scale factors:

```
Graph1.PrintInfo(6) = Printer.ScaleLeft
Graph1.PrintInfo(7) = Printer.ScaleTop
Graph1.PrintInfo(8) = Printer.ScaleWidth
Graph1.PrintInfo(9) = Printer.ScaleHeight
```

Then, you set the position of the upper left corner of the graph--using the coordinate system defined by Visual Basic's Scale method--with PrintInfo() elements 2 and 3.

```
Graph1.PrintInfo(2) = <X value of origin>
Graph1.PrintInfo(3) = <Y value of origin>
```

Finally, you set the width and height of the graph--using the units defined by Visual Basic's ScaleMode property--with PrintInfo() elements 4 and 5:

```
Graph1.PrintInfo(2) = <width>
Graph1.PrintInfo(3) = <height>
```

Aspect ratios of printed graphs

The width and height of your printed graph depends on your settings for PrintInfo(4) and PrintInfo(5):

- If you specify neither a width nor a height--leaving PrintInfo(4) and PrintInfo(5) set to the default zero--the graph prints at the same size it appears on the screen.
- If you set a width using PrintInfo(4) but leave PrintInfo(5) set to the default zero, the Graph control calculates a height that preserves the graph's aspect ratio. Conversely, you can set the height and let the Graph control compute the width.
- If you set *both* PrintInfo(4) and PrintInfo(5) to an exact width and height, the Graph control stretches or compresses the graph to fit.

Printing full-page graphs

If you want to print a graph so that it fills the paper, you needn't fool with either PrintInfo(4) or PrintInfo(5). Just set PrintInfo(12) to 1. This mode will expand the graph to fit the dimensions of the paper, while preserving the aspect ratio of the image on screen.

PrintInfo(12) applies only when you print using DrawMode = 5, and only when you have not set PrintInfo(1) to a printer device context.

Selecting Landscape Orientation

While Visual Basic's printer device context permits changing orientation of printed output from portrait to landscape, other development platforms are not so flexible. However, no matter what platform you are using, you can switch to landscape orientation simply by setting PrintInfo(11) to 1.

PrintInfo(11) applies only when you print using DrawMode = 5, and only when you have not set PrintInfo(1) to a printer device context.

Cloned window vs metafile printing

Prior to Graphics Server version 4.03, printing was performed by playing the metafile of the image in the graph window into the printer device context. If the shape of the graph on-screen differed significantly from the target area on the printer, the metafile image became distorted, leading to anomalies with font size and position.

In version 4.03, Graphics Server changed how it printed graphs, with substantially improved results. The new method printed by first constructing a *cloned* window with the same shape as the target area on the printer. Unfortunately, this new method caused problems for users who had overlaid free-form graphics on the image. The positioning of free-form graphics no longer registered properly.

It is now possible to select which of the two methods you want to use by setting PrintInfo(10) to either 0 (selecting the cloned window method) or 1 (selecting the metafile method). The default is to use the cloned window method. Unless you have a legacy system that requires the old method, we recommend that you use the default method.

SDKInfo() and cloned window printing

The SDKInfo() property has been adapted to return information about the graph drawn in a cloned window. With cloned printing, calls to SDKInfo() in a paint event return coordinates for the image in the cloned window when the paint event is fired during printing.

Example

```
Sub Graph1_Click ()
   Printer.Print ""       'Create a device context
   Printer.ScaleMode = 5  'Set scale mode to inches
   Graph1.PrintInfo(1) = Printer.hDC
   Graph1.PrintInfo(6) = Printer.ScaleLeft
   Graph1.PrintInfo(7) = Printer.ScaleTop
   Graph1.PrintInfo(8) = Printer.ScaleWidth
   Graph1.PrintInfo(9) = Printer.ScaleHeight
   Graph1.PrintInfo(2) = 1 '1 inch from left edge
   Graph1.PrintInfo(3) = 1 '1 inch from top edge
   Graph1.PrintInfo(4) = 3 '3 inches wide
   Graph1.PrintInfo(5) = 4 '4 inches high
   Graph1.DrawMode = 5
   Printer.EndDoc         'End the job and eject page
End Sub
```

▲▼

Topic

PrintInfo()

Related

DrawMode

System property page

PrintStyle property

Sets printing style

Syntax

```
[form.]graph.PrintStyle[ = setting]
```

Data type

Integer (0-3, enumerated)

Settings

0	Monochrome (default)
1	Color
2	Monochrome with border
3	Color with border

Description

The PrintStyle property sets the printing style for a graph--monochrome or color, with or without a border. Graphs are printed when the DrawMode property is set to 5.

The monochrome settings (0 and 2) temporarily change the DrawStyle property to monochrome before printing. It returns to its previous style afterward.

If you're using a color printer, or have a printer capable of printing grayscales, use one of the color settings (1 or 3).

Whatever PrintStyle setting you use, the graph is printed at full printer resolution.

Note If you've assigned a bitmap or metafile backdrop to your graph using the Backdrop property, it prints only if you use a color setting (1 or 3) for PrintStyle.

▲▼

Topic

PrintStyle

Related

Backdrop

DrawMode

DrawStyle

System property page

PropertiesSet event

Fired when a user changes property settings at run time

Syntax

Sub *Graph*_**PropertiesSet()**

Description

The PropertiesSet event is fired whenever a user clicks the Ok or Close buttons on the tabbed property pages at run time. The event is fired only if a property setting has been changed.

The PropertiesSet event is useful if you have enabled run-time access to the property pages and want to save a user's editing changes to a template file so that you can restore them on the next run of your program.

▲▼

Topic

PropertiesSet event

Related

GraphFile
PropertyPages

PropertyCaption property

Sets caption for property pages window

Syntax

[*form*.]*graph*.**PropertyCaption**[= *setting*]

Data type

String

Description

The PropertyCaption property sets the caption for the title bar of the property pages window at run time. At design time, the caption is always "Graph control."

▲▼

Topic

PropertyCaption

Related

Design property page
PropertyPages
StackedTabs
Toolbar

PropertyPages property

Makes property pages available at run time

Syntax

[*form.*]*graph*.**PropertyPages**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Disabled at run time (default)
1	Enabled at run time
2	Display immediately at run time

Description

The PropertyPages property lets you make the Graph control's property pages available at run time. If you set PropertyPages on, the property pages will appear when the user clicks the right mouse button in a graphing window.

Note Property pages are always available at design time. You can access them through the property list window or the toolbar or by clicking the right mouse button in a graphing window.

To enable or disable individual property pages, use the ToolStat property.

▲▼

Topic

PropertyPages

Related

Design property page
PropertiesSet event
Tool()
ToolStat

QuickData run-time property

Sets or returns all graph data in one operation

Syntax

[*form*.]*graph*.**QuickData**[= *"data delimited by tabs"*]

Data type

String

Description

The QuickData run-time property lets you set or get all the data values for a graph (the array normally set by the GraphData property) in a single operation.

The QuickData setting is a string consisting of the characters for each data point. You separate each data *point* with a tab character, CHR$(9); you separate each data *set* with a carriage return and a line feed, CHR$(13) + CHR$(10).

Here's how this might look in code:

```
Dim T As String
Dim CRLF As String
Dim MyData As String
T = Chr$(9)
CRLF = Chr$(13) + Chr$(10)
MyData = "11" + T + "12" + T + "13" + CRLF
MyData = MyData + "21" + T + "22" + T + "23" + CRLF
MyData = MyData + "31" + T + "32" + T + "33" + CRLF
Graph1.QuickData = MyData
```

When you call QuickData, the NumPoints and NumSets properties are set automatically according to the number of points and sets you include in the QuickData string.

Note The QuickData property must contain at least two data points for each data set.

If you set up the QuickData string incorrectly--for example, the data sets contain different numbers of points--you won't get an error, but the data values won't be set.

The QuickData property is useful when you want to exchange data between the Graph control and the grid control's Clip property. To assign a grid's data to your graph's array of data values, all you need is one line of code:

```
Graph1.QuickData = Grid1.Clip
```

Conversely, you can pass data from the QuickData property directly to the grid for modification:

```
Grid1.Clip = Graph1.QuickData
```

▲▼

Topic

QuickData

Related

GraphData

RandomData property

Enables generation of random data for graph

Syntax

[*form*.]*graph*.**RandomData**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Off
1	On (default)

Description

The RandomData property enables or disables the generation of random data for a graph. When you set RandomData on, the Graph control generates random data for each data point every time the graph is redrawn. This property is primarily useful at design time, when you want to get an idea of what a graph will look like at run time.

Random numbers are always equal to or greater than 0. To see the effect of negative values, you have to enter your own data using the GraphData property.

Note The RandomData property is automatically off if GraphData values are present. You can, however, set RandomData on and display randomly generated data without erasing GraphData. Setting it off again redisplays the GraphData values. Also, if you reset GraphData using the DataReset property, RandomData is set back on.

▲▼

Topic

RandomData

Related

GraphData
DataReset

RangeMax property

Defines the upper range of data to be graphed

Syntax

```
[form.]graph.RangeMax[ = data]
```

Data type

Integer

Description

The RangeMax and RangeMin properties define a subset of the data to be graphed. Without changing the composition of the data array assigned to Data() or GraphData, a section of the complete data can be viewed, with the graph axes adjusting appropriately.

The lower and upper bounds of the data subset are defined by RangeMin and RangeMax. By default both RangeMin and RangeMax are set to 0. They have no effect until they are set to a non-zero number, either through the property pages or in code.

▲▼

Topic

RangeMax

Related

Data property page
RangeMin
NumPoints

RangeMin property

Defines the lower range of data to be graphed

Syntax

[*form*.]*graph*.**RangeMin**[= *data*]

Data type

Integer

Description

The RangeMax and RangeMin properties define a subset of the data to be graphed. Without changing the composition of the data array assigned to Data() or GraphData , a section of the complete data can be viewed, with the graph axes adjusting appropriately.

The lower and upper bounds of the data subset are defined by RangeMin and RangeMax. By default both RangeMin and RangeMax are set to 0. They have no effect until they are set to a non-zero number, either through the property pages or in code.

▲▼

Topic

RangeMin

Related

Data property page
RangeMax
NumPoints

RightTitle property

Specifies text string to place at right of graphing window

Syntax

```
[form.]graph.RightTitle[ = "title"]
```

Data type

String (up to 80 characters)

Description

The RightTitle property specifies a text string of up to 80 characters that appears at the right edge of a graphing window.

If you specify a title and it doesn't appear, it's probably too long to display at the specified size. In that case, increase the size of the graphing window or use a shorter title.

You can introduce a line break by inserting the newline character -- CHR$(10) -- in the string. This forces a new line to start at the point where the newline character appears, overriding automatic word-wrap. Forcing a new line is likely to increase the area required for the text and may lead to text being omitted if there is not enough room for it. Since the newline character is unprintable, it cannot be inserted when typing text; it can only be inserted with code.

▲▼

Topic

RightTitle

Related

BottomTitle
BackgroundStyle
FontName
FontStyle
FontSize
Foreground
ForegroundUse
GraphTitle
LeftTitle
RightTitleStyle
Titles property page

RightTitleStyle property

Sets style for right graph title

Syntax

```
[form.]graph.RightTitleStyle[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Horizontal (default)
1	Up
2	Down

Description

The RightTitleStyle property sets the style for the right title of a graph. You can display the text horizontally, up (text rotated counterclockwise 90 degrees), or down (text rotated clockwise 90 degrees).

The text of the right title is specified by the RightTitle property.

▲▼

Topic

RightTitleStyle

Related

BackgroundStyle

RightTitle

Titles property page

Rotation property

Sets horizontal viewing angle for True3D graphs

Syntax

```
[form.]graph.Rotation[ = setting]
```

Data type

Integer   (-180-180)

Settings

Valid settings are integers from -180 to 180 (default is 0).

- A setting of 0 produces a face-on view.
- A setting of -90 produces a side view from the left.
- A setting of 90 produces a side view from the right.

Description

The Rotation property sets the horizontal viewing angle (in degrees) for True3D graphs. The setting represents an angle around the base of the graph.

At Rotation settings less than -90 and greater than 90, the graph itself can't be seen--only the back of the cage is visible.

Note: 3D pie charts can be tilted but not rotated. The Rotation property does not apply to that graph type.

▲▼

Topic

Rotation

Related

3d property page
CageFlip
Elevation
Perspective
True3D

SDKHit event

Fired when hot region receives mouse click at run time

Syntax

Sub *Graph*_**SDKHit (***HitRegion* **As Integer)**

Parameter

HitRegion The hot region receiving a mouse click.

Description

The SDKHit event is fired when a hot region within a graph receives a mouse click at run time. You can cf ate hot regions using Graphics Server DLL functions.

To enable SDKHit events, you must first set the SDKMouse property to on.

▲▼

Topic

SDKHit

Related

SDKPress (event)

SDKInfo() run-time property

Provides internal Graphics Server information about last graph displayed

Syntax

graph.**SDKInfo(*index*%)**

Data type

Integer

Parameter

index% Element number of the SDKInfo() array.

1	X axis maximum (your data units)
2	X axis minimum (your data units)
3	Y axis maximum (your data units)
4	Y axis minimum (your data units)
5	X axis length (GS units)
6	Y axis length (GS units)
7	X origin (GS units)
8	Y origin (GS units)
9	Label font size (percentage of system font height)
10	Right Y axis maximum (your data units)
11	Right Y axis minimum (your data units)
12	Z axis maximum (your data units)
13	Z axis minimum (your data units)
14	Number of ticks on +ve X axis
15	Number of ticks on -ve X axis
16	Number of ticks on +ve Y axis
17	Number of ticks on -ve Y axis
18	Number of ticks on +ve right Y axis
19	Number of ticks on -ve right Y axis
20	Number of ticks on +ve Z axis
21	Number of ticks on -ve Z axis

Description

SDKInfo() is a run-time, read-only property that returns internal Graphics Server information about the last graph you've displayed. It's intended primarily for users of Graphics Server's DLL functions, but you can use it without the DLL to relate mouse clicks to the data on which a graph is based.

Typically, the SDKInfo() array is read in the SDKPaint event in order to superimpose drawing on the graph when it is refreshed. When drawing to the screen, SDKInfo returns coordinates relating to the screen image. When printing, it returns coordinates relating to the image drawn in the hidden window prior to printing.

The values for SDKInfo() are stored as a one-dimensional array with 21 elements, as listed above.

Example

The following code adds a subtitle to a graph, using the SDKInfo() property to determine the placement and the Graphics Server DLL function GSRText to display the text.

```
Sub Graph1_SDKPaint ()
   xLen = Graph1.SDKInfo(5)
   yLen = Graph1.SDKInfo(6)
   xOrg = Graph1.SDKInfo(7)
   yOrg = Graph1.SDKInfo(8)
   x = xOrg + xLen / 2  ' Horizontal center
   y = yOrg + yLen      ' Top of graph
   r = GSRText(x, y, CSRASTER, TXMID, BLACK, "West")
End Sub
```

SDKMouse property

Enables SDKHit and SDKPress events

Syntax

`[`*`form.`*`]`*`graph`*`.`**`SDKMouse`**`[ = `*`setting`*`]`

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The SDKMouse property enables or disables SDKHit and SDKPress events, which are fired by mouse clicks in a graph window. These events are intended primarily for users of Graphics Server's DLL functions, but you can use them without the DLL to relate mouse clicks to the data on which a graph is based.

When you set SDKMouse to on, the standard Visual Basic mouse events (Click, DblClick, and so on) are disabled. However, the Graph control's HotHit events take priority over SDKHit and SDKPress events. You can't use both HotHit and SDKHit/SDKPress events at the same time.

▲▼

Topic

SDKMouse

Related

SDKHit (event)
SDKPress (event)

SDKPaint event

Fired each time graphing window is redrawn at run time

Syntax

Sub *Graph*_**SDKPaint ()**

Description

The SDKPaint event is fired every time a graphing window is redrawn at run time.

SDKPaint is useful when you're using Graphics Server DLL functions to draw additional objects (besides the graph and its associated text and graphics) in a graphing window. The SDKPaint event alerts your program when the graphing window is redrawn, letting you update the additional objects.

Topic

SDKPaint (event)

Related

SDKPaint (property)

SDKPaint property

Enables SDKPaint events

Syntax

`[form.]graph.SDKPaint[ = setting]`

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The SDKPaint property enables or disables SDKPaint events. SDKPaint events occur after each redraw, in a similar fashion as the picture control's Paint event.

The SDKPaint property and events are intended primarily for users of Graphics Server's DLL functions.

▲▼

Topic

SDKPaint (property)

Related

SDKPaint (event)

SDKPress event

Fired when mouse button is pressed or released in graphing window at run time

Syntax

```
Sub Graph_SDKPress (PressStatus As Integer, PressX As Double, PressY As
Double, PressDataX As Double, PressDataY As Double)
```

Parameters

PressStatus Button press information:

0	Button released
1	Left button pressed
2	Middle button pressed
3	Both left and middle buttons pressed
4	Right button pressed
5	Both left and right buttons pressed
6	Both middle and right buttons pressed
7	Left, middle, and right buttons pressed at once

PressX	X coordinate (Graphics Server units)
PressY	Y coordinate (Graphics Server units)
PressDataX	X coordinate (your data units)
PressDataY	Y coordinate (your data units)

Description

The SDKPress event is fired when a mouse button is pressed or released in a graphing window at run time. To enable SDKPress events, you must first set the SDKMouse property to on.

The PressDataX and PressDataY parameters return valid information only for 2D graph types drawn on X-Y grids. For pie charts, polar graphs, and all 3D graphs, these parameters return 0.

In this version of the Graph control, the height of a graphing window is always 1000 units, with the width determined by the aspect ratio of the window. If you want to obtain the actual height and width of the window, you can use the Graphics Server DLL functions GSGetWYExt and GSWXExt.

▲▼

Topic

SDKPress

Related

SDKHit (event)

SDKTrack event

Fired when mouse cursor is moved in graphing window at run time

Syntax

Sub *Graph*_**SDKTrack (***TrackX* **As Double,** *TrackY* **As Double,** *TrackDataX* **As Double,** *TrackDataY* **As Double)**

Parameters

TrackX	X coordinate (Graphics Server units)
TrackY	Y coordinate (Graphics Server units)
TrackDataX	X coordinate (your data units)
TrackDataY	Y coordinate (your data units)

Description

The SDKTrack event is fired when a mouse cursor is moved in a graphing window at run time. To enable SDKTrack events, you must first set the SDKMouse property to on.

The TrackDataX and TrackDataY parameters return valid information only for 2D graph types drawn on X-Y grids. For pie charts, polar graphs, and all 3D graphs, these parameters return 0.

In this version of the Graph control, the height of a graphing window is always 1000 units, with the width determined by the aspect ratio of the window. If you want to obtain the actual height and width of the window, you can use the Graphics Server DLL functions GSGetWYExt and GSWXExt.

SeeThru run-time property

Lets graphing windows be drawn without clearing background

Syntax

[*form*.]*graph*.**SeeThru**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Off (default)
1	On

Description

The SeeThru run-time property lets you draw a graphing window without clearing the background.

When SeeThru is set to on, the graph's background doesn't clear when you set the DrawMode property to draw a graph (the exception being DrawMode setting 3, blit, which doesn't work with SeeThru). Instead, whatever was there before--be it another graphing window, the previous contents of the same graphing window, or any other graphics--remains visible underneath the graph.

You can use SeeThru to create custom combination graphs, placing one graphing window on top of another.

Example

The following example demonstrates a potential use of the SeeThru property.

First, create a picture (Picture1) and then create a graph (Graph1)--not as a child of the picture, but directly on the form. Move the graph over the top of the picture, leaving the picture's edges visible. Then, set the graph's BorderStyle property (in the Form property window) to no border.

When the picture receives a paint message, it refreshes both itself and the graph, ensuring the graph is still on top of the picture with the picture showing through. The flag is necessary to prevent entering the loop again (the Paint event is triggered again by the Refresh/ZOrder method).

```
Dim flag As Integer
Sub Form_Load ()
   flag = 0
   picture1.ZOrder 0  'Bring the picture box to the top
                      '(VB2 only)
   graph1.SeeThru = 1
End Sub

Sub Picture1_Paint ()
   If flag Then
      flag = 0
      picture1.Refresh   'Refresh picture (VB1)
      picture1.ZOrder 0  'Bring the picture box to the
                         'top (VB2)
      graph1.Refresh
   Else
      flag = 1
   End If
End Sub
```

This code is almost the same for VB2 as for VB1, but instead of using the picture box's Refresh method, you must use the VB2 ZOrder method to bring the picture box to the top before the graph is displayed. Otherwise, the graph's background remains totally transparent rather than being drawn over the contents of the picture box.

SmartLabels property

Enables auto-arranging of pie labels to avoid overlap

Syntax

```
[form.]graph.SmartLabels[ = setting]
```

Data Type

Integer (0-1 enumerated)

Settings

0	Off
1	On

Description

The SmartLabels property auto-arranges pie chart labels to avoid overlapping. The algorithm is not infallible, and may cause labels to extend outside the visible area of the graph if too many are concentrated in a small sector.

▲▼

Topic

SmartLabels

Related

GraphStyle

StackedTabs property

Selects either stacked or scrolling tabs for the property pages

Syntax

```
[form.]graph.StackedTabs[ = setting]
```

Data Type

Boolean

Settings

True	(Default) Tabs in property pages are "stacked" in rows.
False	Tabs in property pages appear as a single, scrollable row.

Description

The StackedTabs property selects either stacked or unstacked tabs in the property pages. By default, tabs are stacked.

▲▼

Topic

StackedTabs

Related

PropertyCaption

PropertyPages

SurfaceColorMax property

Sets color index used at highest point of True3D surface graph

Syntax

[*form*.]*graph*.**SurfaceColorMax**[= *setting*]

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The SurfaceColorMax property sets the color index used at the highest point of a True3D surface graph. SurfaceColorMax must be greater than or equal to the value of the SurfaceColorMin property.

In a surface graph, each point is colored according to its position above the graph origin. At the low point of the axis, the graph is colored according to the color index specified by SurfaceColorMin; at the top of the axis, it's colored according to the color index specified by SurfaceColorMax. Points in between are colored with values interpolated from the palette between SurfaceColorMin and SurfaceColorMax.

You can use the Palette property to select a 128-entry extended palette with a variety of graded hues suitable for surface graphs.

Example

To select a palette and set the minimum and maximum colors for a surface graph, you could use these lines:

```
Graph1.Palette = 8            ' select 128-entry palette
Graph1.SurfaceColorMin = 32   ' index of lowest entry
Graph1.SurfaceColorMax = 127  ' index of highest entry
```

▲▼

Topic

SurfaceColorMax

Related

Palette
Style property page
SurfaceColorMin
SurfaceWallColor

SurfaceColorMin property

Sets color index used at lowest point of True3D surface graph

Syntax

```
[form.]graph.SurfaceColorMin[ = setting]
```

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The SurfaceColorMin property sets the color index used at the lowest point of a True3D surface graph. SurfaceColorMin must be less than or equal to the value of the SurfaceColorMax property.

In a surface graph, each point is colored according to its position above the graph origin. At the low point of the axis, the graph is colored according to the color index specified by SurfaceColorMin; at the top of the axis, it's colored according to the color index specified by SurfaceColorMax. Points in between are colored with values interpolated from the palette between SurfaceColorMin and SurfaceColorMax.

You can use the Palette property to select a 128-entry extended palette with a variety of graded hues suitable for surface graphs.

▲▼

Topic

SurfaceColorMin

Related

Palette

Style property page

SurfaceColorMax

SurfaceWallColor

SurfaceWallColor property

Sets color of end walls of True3D surface graph

Syntax

[*form*.]*graph*.**SurfaceWallColor**[= *setting*]

Data type

Integer (0-127, enumerated)

Settings

Color index number

Description

The SurfaceWallColor property sets the color of the end walls of True3D surface graphs. To enable end walls on surface graphs, set the GraphStyle property to 3, 4, or 5.

Topic

SurfaceWallColor

Related

GraphStyle
Palette
Style property page
SurfaceColorMin
SurfaceColorMax

Symbol() run-time property

Alternative way to set symbol design at run time

Syntax

```
[form.][graph.]Symbol(index%)[ = setting]
```

Parameter

index% Element number of array normally set by SymbolData property

Data type

Integer (0-9, enumerated)

Settings

Same as for SymbolData property

Description

The Symbol() run-time property sets the symbol design for graph types set up to use symbols--line, logarithmic, polar, and scatter graphs. Symbol() is a run-time alternative to the SymbolData property.

Symbol values are stored as a one-dimensional array. Unlike SymbolData, Symbol() doesn't require you to use the ThisPoint or ThisSet property to index the symbol array. Instead, you provide the element (point or set) number as your Symbol() parameter.

Example

If you have graph with two or more data sets and you want to set the symbol for the second set to a solid square (symbol pattern 7), you can set up your code in two ways:

- If you use the SymbolData property, you set the ThisSet property first:

  ```
  Graph1.ThisSet = 2
  Graph1.SymbolData = 7
  ```

- If you use the Symbol() property, you don't have to set ThisSet:

  ```
  Graph1.Symbol(2) = 7
  ```

▲▼

Topic

Symbol()

Related

SymbolData

SymbolData property

Sets symbol design

Syntax

```
[form.]graph.SymbolData[ = setting]
```

Data type

Integer (0-13, enumerated)

Description

The SymbolData property sets the symbol design for graph types set up to use symbols--line, logarithmic, polar, and scatter graphs. Fourteen symbol options are available; each data set needs its own SymbolData value.

Each data set needs its own SymbolData value. SymbolData is a one-dimensional array-based property indexed by either the ThisPoint or ThisSet property (see the IndexStyle property for details on the Graph control's indexing of array-based properties). You can use the AutoInc property to automatically increment the ThisPoint or ThisSet counter each time you set a SymbolData element.

▲▼

Topic

SymbolData

Related

AutoInc

DataReset

Markers property page

Symbol()

SymbolSize

ThisPoint

ThisSet

SymbolSize property

Sets size of symbols

Syntax

```
[form.]graph.SymbolSize[ = setting]
```

Data type

Integer (10-1000)

Settings

Valid settings are integers from 10 to 1000 (default is 100).

Description

The SymbolSize property sets the size of the symbols for graph types set up to use symbols--line, logarithmic, polar, and scatter graphs. It also determines the length of crossbars (the open, high, low, and close markers) in high-low-close and open-high-low-close graphs. The SymbolSize setting is a percentage of the default symbol size or crossbar length.

In you're using the HotGraph feature with line, scatter, high-low-close, and open-high-low-close graphs, SymbolSize determines the size of hot regions--whether or not symbols are actually displayed. You can use SymbolSize to set the "resolution" of the click area. Note that large SymbolSize values may cause hot regions to overlap.

▲▼

Topic

SymbolSize

Related

Markers property page
SymbolData

ThickLines property

Enables thick lines

Syntax

```
[form.]graph.ThickLines[ = setting]
```

Data type

Integer (0-1, enumerated)

Settings

0	Off
1	On (default)

Description

The ThickLines property lets you enable thick lines for line, lin/log, log/lin, log/log, polar, high-low-close, and open-high-low-close graphs. By default, thick lines are three pixels thick. Thin lines (ThickLines set to off) are one pixel thick.

For line, logarithmic, and polar graphs, you can use the PatternData property to override the default of three pixels and set your own line thickness. For high-low-close and open-high-low-close graphs, you can't change the thickness of three pixels (unless you set ThickLines to off, in which case the thickness is one pixel).

The ThickLines property is mutually exclusive with the PatternedLines property. When you set one of these properties to on (1), the other is automatically set to off (0).

▲▼

Topic

ThickLines

Related

Markers property page

PatternedLines

PatternData

ThisPoint property

Indexes array-based properties, usually by data point number

Syntax

[*form*.]*graph*.**ThisPoint**[= *number*]

Data type

Integer

Settings

Valid settings are integers from 1 to the setting of the NumPoints property.

Description

The ThisPoint property, along with the ThisSet property, indexes (or "points" to an element of) most of the Graph control's array-based properties. ThisPoint can apply to any of the following properties:

ColorData
DataLabelText
EBarGraphDataMinus
EBarGraphDataPlus
EBarXPosDataMinus
EBarXPosDataPlus
ExtraData
GraphData
LabelText
LegendText
OverlayGraphData
PatternData
SymbolData
XPosData
YLabelText
ZLabelText
ZPosData

In general, ThisPoint specifies the number of a data point, but it sometimes actually refers to a data set if you're using the Graph control's standard indexing method (IndexStyle setting 0). See the IndexStyle property for details.

When you set a value for ThisPoint, the AutoInc property is automatically set to off. This lets you address a single point rather than running through all points in order.

Note Unlike most properties, which retain their design-time settings at run time, ThisPoint automatically resets to 1 when you run a project. Keep this in mind as you write your code.

Example

The following lines create five data points using the AutoInc property, then use the ThisPoint property to override the initial setting for the third point:

```
Graph1.NumPoints = 5
Graph1.NumSets = 1
Graph1.AutoInc = 1
For I% = 1 To 5
```

```
   Graph1.GraphData = I%
Next I%
Graph1.ThisPoint = 3
Graph1.GraphData = 10
Graph1.GraphType = 4
Graph1.DrawMode = 2
```

▲▼

Topic

ThisPoint

Related

AutoInc

Data property page

IndexStyle

Markers page

NumPoints

NumSets

ThisSet

ThisSet property

Indexes array-based properties by data set number

Syntax

[*form*.]*graph*.**ThisSet**[= *number*]

Data type

Integer

Settings

Valid settings are integers from 1 to the setting of the NumSets property.

Description

The ThisSet property, along with the ThisPoint property, indexes (or "points" to an element of) most of the Graph control's array-based properties. ThisSet can apply to any of the following properties:

ColorData
DataLabelText
EBarGraphDataMinus
EBarGraphDataPlus
EBarXPosDataMinus
EBarXPosDataPlus
ExtraData
GraphData
LegendText
PatternData
SymbolData
XPosData
ZLabelText
ZPosData

Unlike ThisPoint, which can index by either data points or data sets, ThisSet always indexes by data set. However, if you're using the Graph control's standard indexing method (IndexStyle setting 0) on a one-dimensional array-based property, the number of the data set is specified by ThisPoint instead of ThisSet. See the IndexStyle property for details.

When you set a value for ThisSet, the AutoInc property is automatically set to off. This lets you address a single set rather than running through all sets in order.

Note Unlike most properties, which retain their design-time settings at run time, ThisSet automatically resets to 1 when you run a project. Keep this in mind as you write your code.

Example

The following lines create three data sets (of five data points each) using the AutoInc property, then use the ThisSet and ThisPoint properties to override the initial setting for the third point of the second set:

```
Graph1.NumPoints = 5
Graph1.NumSets = 3
Graph1.AutoInc = 1
For I% = 1 To Graph2.NumPoints * Graph2.NumSets
   Graph1.GraphData = 5
Next I%
```

```
Graph1.ThisSet = 2
Graph1.ThisPoint = 3
Graph1.GraphData = 10
Graph1.GraphType = 4
Graph1.DrawMode = 2
```

▲▼

Topic

ThisSet

Related

AutoInc

Data property page

IndexStyle

Markers property page

NumPoints

NumSets

ThisPoint

TickEvery property

Sets frequency of ticks on X axis

Syntax

[*form*.]*graph*.**TickEvery**[= *number*]

Data type

Integer (1-1000)

Settings

Valid settings are integers from 1 to 1000 (default is 1).

Description

The TickEvery property sets the frequency of ticks on a graph's X axis (or the Y axis in horizontal bar graphs and Gantt charts). A tick is drawn for every *n*th data point, where *n* is the TickEvery setting.

TickEvery has no effect when you assign independent X values to a graph using the XPosData property. However, you can use the XAxisStyle and XAxisTicks properties to customize the X axis for these graph types.

The X axis always has to end with a tick. If you set a TickEvery value that isn't a multiple of the number of points (the NumPoints property setting), the axis will be extended to the first tick at which all points are shown.

▲▼

Topic

TickEvery

Related

Axis property page

XAxisStyle

XAxisTicks

Ticks property

Enables or disables tick marks for axis

Syntax

[*form*.]*graph*.**Ticks**[= *setting*]

Data type

Integer (0-3, enumerated)

Settings

0	Off
1	On--ticks on both X and Y axes (default)
2	On--ticks on X axis only
3	On--ticks on Y axis only

Description

The Ticks property enables and disables axis ticks for the X axis, Y axis, or both. It has no effect on grids or labels.

The Ticks setting doesn't apply to pie charts or polar graphs.

▲▼

Topic

Ticks

Related

Axis property page

TickEvery

TickStyle property

Selects a style option for drawing tick marks

Syntax

```
[form.]graph.TickStyle[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Through (default)
1	Inside axes
2	Outside axes

Description

The TickStyle property sets how tick marks are drawn. The default is to center them horizontally on the axis. You can choose to have them drawn just inside or just outside the axes. The setting for TickStyle controls both major and minor tick marks.

Note: Turn the drawing of major tick marks on or off by setting the Ticks property. Turn minor ticks on or off by setting XAxisMinorTicks and YAxisMinorTicks.

TickStyle applies to all 2D graphs except pie and polar graphs. It doesn't apply to 3D graphs.

▲▼

Topic

TickStyle

Related

Axis property page
Ticks
XAxisMinorTicks
YAxisMinorTicks

Tool() run-time property

Alternative to ToolStat property for enabling toolbar icons and property pages at run time

Syntax

`[form.]graph.Tool(index%)[ = setting]`

Parameter

index% Element number corresponding to a property page.

Index%	Association
1	2D Gallery property page
2	3D Gallery property page
3	Data property page
4	Style property page
5	Titles property page
6	Axis property page
7	3D property page
8	Fonts property page
9	Markers property page
10	Trends property page
11	Overlay property page
12	Error Bar property page
13	Background property page
14	System property page
15	(Omitted--placeholder for Design property page)
16	Help file
17	About property page
18	Legend property page
19	Labels property page

Data type

Integer (0-2, enumerated)

Settings

0	Toolbar icon enabled, property page enabled (default)
1	Toolbar icon disabled, property page disabled
2	Toolbar icon enabled, property page disabled--ToolHit event fired when mouse clicked on icon

Description

The Tool() property lets you enable and disable individual icons on the Graph Control toolbar as well as their associated property pages.

The Tool() property is the run-time alternative to ToolStat. Both are one-dimensional array properties, but Tool() is easier to set in code. Before setting ToolStat, you must first point to an element of the array by setting the ThisPoint property. You can skip that step by using the Tool() property. Just pass the number of an array element as a parameter when you set Tool().

Disabling individual property pages

You can enable toolbar access to the property pages at run time by setting the Toolbar property to 2. You can enable right-click access at run time by setting the PropertyPages property to 1. In both cases, all but the Design page become available to the user.

To disable a property page, pass the number of the page as a parameter of the Tool() property and set the property to 1. For example, this code disables the System page:

```
Graph1.Tool(14) = 1
```

When this code executes, the System page will be omitted from the tabbed property sheets. If the toolbar is enabled, the user will see the disabled icon on the toolbar, but it will be drawn in gray and will not respond to mouse clicks.

Replacing property pages with your own dialogs

If you set a Tool() element to 2 (icon enabled, property page disabled), a ToolHit event is fired when a user clicks on that toolbar icon. This lets you create your own dialog in place of the usual property page.

If you have disabled the toolbar at run time by setting the Toolbar property off, you can still display your own dialogs. Just set the PropertyPages property to 1 (Enabled at run time). Then set the first element of the Tool() array to 2. This disables the first property page and enables an event. Since clicking a right mouse button within the graphing window normally displays the first property page, it will in this case fire the ToolHit event associated with the the first page. When the event is fired, you can display your own dialog.

Interaction with the Hot property

Turning on hot graphing disables the standard Visual Basic mouse events (Click and DblClick) in the graphing window. It also disables right mouse click access to the property pages at run time. However, you can still access the property pages through the toolbar if it is enabled.

▲▼

Topic

Tool()

Related

Design property page
PropertyPages
PropertiesSet event
Toolbar
ToolHit (event)
ToolStat
Hot

Toolbar property

Enables display of Graph control toolbar

Syntax

[*form*.]*graph*.**Toolbar**[= *setting*]

Data type

Integer (0-3, enumerated)

Settings

0	Off (default)
1	On--design time only
2	On--run time only
3	On--both design time and run time

Note: The toolbar cannot be displayed at design time by the OCX. It can be made available at design time only when you are using the VBX.

Description

The Toolbar property enables the display of the Graph control toolbar in a graphing window. The toolbar lets you call up property pages and set graph attributes using convenient dialogs.

See the Tool() property for a list of the property pages available through the toolbar.

▲▼

Topic

Toolbar

Related

Design property page
PropertyPages
PropertiesSet event
Tool()
ToolStat

ToolHit event

Fired when mouse is clicked on toolbar icon enabled for events

Syntax

Sub *Graph*_**ToolHit (***ToolNum* **As Integer)**

Parameter

ToolNum	Association
0	2D Gallery property page
1	3D Gallery property page
2	Data property page
3	Style property page
4	Titles property page
5	Axis property page
6	3D property page
7	Fonts property page
8	Markers property page
9	Trends property page
10	Overlay property page
11	Error Bar property page
12	Background property page
13	System property page
14	(Omitted--placeholder for Design property page)
15	Help file
16	About property page
17	Legend property page
18	Labels property page

Description

The ToolHit event is fired when a user clicks on a toolbar icon that's been enabled for events. With this event available, you can design a dialog to replace the property page or Help file normally associated with that icon.

To enable an icon for events, set the Tool() property for that icon to 2 (icon enabled, property page disabled). If you enable events for several icons, your ToolHit event procedure should check the value of the ToolNum parameter to determine how to respond.

Example

```
Private Sub Form_Load()
Dim i%
  'Turn on the toolbar
  Graph1.Toolbar = 2
  'Disable property page, enable ToolHit event
  'Note: Page numbers are 1-based
  For i% = 1 to 17
    Graph1.Tool(i%) = 2
  Next i%
End Sub

Sub Graph1_ToolHit (ToolNum As Integer)
  'Note: Tool numbers are 0-based
  If ToolNum = 0 Then
```

```
    MsgBox "Show your own 2D Gallery dialog here"
  End If
  If ToolNum = 3 Then
    MsgBox "Show your own Graph Style dialog here"
  End If
End Sub
```

▲▼

Topic

ToolHit

Related

Design property page
PropertiesSet event
Tool()
Toolbar
ToolStat

ToolStat property

Enables or disables toolbar icons and associated property pages

Syntax

```
[form.]graph.ToolStat[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Toolbar icon enabled, property page enabled (default)
1	Toolbar icon disabled, property page disabled
2	Toolbar icon enabled, property page disabled--ToolHit event fired when mouse clicked on icon

Description

The ToolStat property lets you enable and disable individual property pages at run time.

ToolStat is a one-dimensional array property indexed by ThisPoint. It has 19 elements corresponding to various property pages.

ThisPoint	Association
1	2D Gallery property page
2	3D Gallery property page
3	Data property page
4	Style property page
5	Titles property page
6	Axis property page
7	3D property page
8	Fonts property page
9	Markers property page
10	Trends property page
11	Overlay property page
12	Error Bar property page
13	Background property page
14	System property page
15	(Omitted--placeholder for Design property page)
16	Help file
17	About property page
18	Legend property page
19	Labels property page

Note that there's no ToolStat element for the Design property page, which is never accessible from the toolbar.

Disabling individual property pages

You can enable toolbar access to the property pages at run time by setting the Toolbar property to 2. You can enable right-click access at run time by setting the PropertyPages property to 1. In both cases, all pages except the Design page become available to the user.

To disable a property page, point to the corresponding array element by setting ThisPoint, then set ToolStat. For example, this code disables the System page:

```
Graph1.ThisPoint = 14
```

```
Graph1.ToolStat = 1
```

When this code executes, the System page will be omitted from the tabbed property sheets. If the toolbar is enabled, the user will see the disabled icon on the toolbar, but it will be drawn in gray and will not respond to mouse clicks.

Replacing property pages with your own dialogs

If you set a ToolStat element to 2 (icon enabled, property page disabled), a ToolHit event is fired when a user clicks on that icon. This lets you create your own dialog in place of the usual property page.

If you have disabled the toolbar at run time by setting the Toolbar property off, you can still display your own dialogs. Just set the PropertyPages property to 1 (Enabled at run time). Then set the first element of the ToolStat array to 2. This disables the first property page and enables an event. Since clicking a right mouse button within the graphing window normally displays the first property page, it will in this case fire the ToolHit event associated with the the first page. When the event is fired, you can display your own dialog.

Interaction with the Hot property

Turning on hot graphing disables the standard Visual Basic mouse events (Click and DblClick) in the graphing window. It also disables right mouse click access to the property pages at run time. However, you can still access the property pages through the toolbar if it is enabled.

▲▼

Topic

ToolStat

Related

Design property page
PropertyPages
PropertiesSet event
Tool()
Toolbar
ToolHit (event)
ThisPoint
Hot

TrendSet() run-time property

Alternative way to enable trend lines at run time

Syntax

[*form*.]*graph*.**TrendSet(*index*%)**[= *setting*]

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the TrendSets property.

Data type

Integer (0-31, enumerated)

Settings

Same as for the TrendSets property.

Description

The TrendSet() property gives you an alternative method for setting the values normally set by the TrendSets property. TrendSet() is available only at run time.

Both TrendSets and TrendSet() store values in a one-dimensional array. With TrendSets you must first point to an element of the array by setting ThisSet, and then store a value by setting TrendSets. You can skip the first step by using TrendSet(). Rather than setting ThisSet, you can simply pass the number of an array element as a parameter of TrendSet().

▲▼

Topic

TrendSet()

Related

TrendSets

TrendSets property

Enables trend lines for individual data sets

Syntax

```
[form.]graph.TrendSets[ = setting]
```

Note: This property is not available with the VBX interface.

Data type

Integer (0-31, enumerated)

Settings

0	No statistical lines (default)
1	Mean line
2	Min/max line
3	Mean and min/max lines
4	Standard-deviation line
5	Standard-deviation and mean lines
6	Standard-deviation and min/max lines
7	Standard-deviation, min/max, and mean lines
8	Best-fit line
9	Best-fit and mean lines
10	Best-fit and min/max lines
11	Best-fit, min/max, and mean lines
12	Best-fit and standard-deviation lines
13	Best-fit, standard-deviation, and mean lines
14	Best-fit, standard-deviation, and min/max lines
15	All lines--best-fit, standard-deviation, min/max, and mean
16-31	Same as above, with curve fitting. Add 16 to each of the above settings to enable a curve. To select which type of curve is drawn, set the CurveType property.

Description

Like the LineStats property, TrendSets lets you superimpose statistical lines and curves on a graph. However, TrendSets offers greater flexibility than LineStats. With LineStats, trend lines are applied to all data sets for line and scatter graphs, to median values for box-whisker graphs, and to close values for HLC, OHLC and candlestick graphs. With TrendSets, you can apply trend lines to any data set, and you can select different trend lines for each set.

TrendSets is a one-dimensional array property indexed by ThisSet. First select the number of a data set using ThisSet. Then apply trend lines to the selected data set by setting TrendSets.

Note: To apply trend lines to an overlay graph, use the OverlayTrendSets property.

If no element of TrendSets has been set, the setting for LineStats applies. Once you have set an element of the TrendSets array, the setting for LineStats will be ignored. Before the LineStats setting can take effect, you must clear the TrendSets array by setting DataReset to 25.

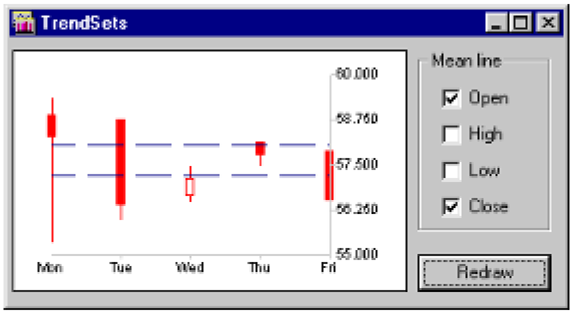
By default, trend lines are drawn in the same color as the associated data set. You can select another color using the ForegroundUse and Foreground properties.

Example

Let's suppose you have an application with a candlestick graph depicting a stock's daily open, high, low and close prices. You want to enable a user to see a mean line for selected data sets, so you've set up an array of checkboxes for that purpose.

The form also has a command button for redrawing the graph with a new set of trend lines once a user has made new selections. Code for the command button's click event is listed below.

```
Private Sub cmdRedraw_Click()
Dim i%

Graph1.DataReset = 25 'Clear TrendSets array

'Loop through the checkbox control array,
'turning on a mean line for any data set
'that is checked.
For i% = 0 To 3
    If chkSet(i%).Value = Checked Then
        Graph1.ThisSet = i% + 1
        Graph1.TrendSets = 1
    End If
Next i%

Graph1.DrawMode = 2   'Redraw the graph

End Sub
```



Mean line

☑ Open
□ High
□ Low
☑ Close

Mean lines for the first and fourth data sets

▲▼

Topic

TrendSets

Related

TrendSet()
CurveType
LineStats
OverlayTrendSets

True3D property

Enables True3D graphing and sets projection method

Syntax

`[form.]graph.True3D[ = setting]`

Data type

Integer (0-2, enumerated)

Settings

0	Off (default if True3D is unavailable for the graph type; not available for 3D scatter or surface graphs, which are always True3D)
1	On--perspective projection (default if True3D is available for the graph type)
2	On--isometric projection

Description

The True3D property enables True3D graphing and sets the projection method to *perspective* or *isometric*. When True3D is on, you can rotate the graph's 3D "cage" and view the graph from any angle. True3D graphing is available for five graph types: area, bar, scatter, surface, and tape.

You can choose from two methods of True3D projection:

- With a **perspective projection** (True3D = 1), the view of the graph is foreshortened. This means that lines that would appear parallel on a 2D graph appear to diverge toward the front of the graph and converge toward the back.
- With an **isometric projection** (True3D = 2), there's no foreshortening. Lines that would appear parallel on a 2D graph also appear parallel on the True3D graph.

The Rotation, Elevation, and Perspective properties control the angle and perspective from which the graph is viewed.

True3D graphs are always drawn in a 3D "cage" instead of the conventional X and Y axes. You can customize the cage using the CageStyle, CageEdgeColor, CageWallColor, XAxisStyle, YAxisStyle, and ZAxisStyle properties. Note that most of the conventional axis properties don't apply.

By nature, True3D graphs are easy to "animate"--you can redraw them repeatedly at differently angles and create the effect of movement. For fast animation, you can set the DrawMode property to 7 (rotate). Every time you set a new value for the Rotation, Elevation, or Perspective property, the Graph control redraws only the part of the graph that rotates and leaves scales, titles, and legends unchanged. This mode cuts down on processing time, speeding up True3D redrawing.

When you rotate True3D graphs, graph markers and axis labels can become obscured behind the back or side walls of the cage. You can use the CageFlip property to cause walls to "flip" (switch to the opposite side of the cage) when objects are obscured.

True3D graphs don't support overlaid statistics or curve fitting.

▲▼

Topic

True3D

Related

3d property page
Elevation
CageFlip
CageStyle
CageEdgeColor
CageWallColor
DrawMode
Perspective
Rotation
True3DDepth
True3DXGap
True3DZGap
XAxisStyle
YAxisStyle
ZAxisStyle

True3DDepth property

Sets projected depth of True3D graph

Syntax

```
[form.]graph.True3DDepth[ = setting]
```

Data type

Integer (positive)

Settings

For 3D pie charts, valid settings are positive integers from 10 to 200. The default is 100.

For all other 3D graph types, valid settings are all positive integers. The default is 100.

Description

For 3D pie charts, the True3DDepth property adjusts the pie's thickness. Settings are expressed as a percentage of the default thickness, which is half the radius. The default setting is 100--100% of the default thickness. A setting of 200 doubles the default thickness and a setting of 50 halves it.

For all other 3D graph types, the True3DDepth property sets the projected depth (in the Z direction). The depth is expressed as a percentage of the default (100), which is calculated to provide equal increments for units in the X and Z directions.

Since the X width is determined by the setting of the NumPoints property and the Z depth by the NumSets property, a graph with 10 points and 10 sets would be of equal width and depth at a True3DDepth setting of 100. A True3DDepth value of 50 would halve the depth and a value of 200 would double it.

▲▼

Topic

True3DDepth

Related

3D property page
True3D
True3DXGap
True3DZGap

True3DXGap property

Sets gap between bars in True3D bar graph

Syntax

[*form*.]*graph*.**True3DXGap**[= *setting*]

Data type

Integer (0-95)

Settings

Valid settings are integers from 0 to 95 (default is 20).

Description

The True3DXGap property sets the gap between bars in a True3D bar graph. This value is expressed as a percentage of the distance from the center of one bar to the center of the next (in other words, a percentage of the entire possible width of each bar). The remaining percentage is occupied by the bar itself.

The default True3DXGap setting of 20 creates a gap one-fourth as wide as each bar--the gap occupies 20 percent of the available space, the bar 80 percent. A setting of 50 produces gaps and bars of equal width, and a setting of 0 produces no gap (the bars are adjacent).

▲▼

Topic

True3DXGap

Related

3D property page
True3D
True3DZGap

True3DZGap property

Sets gap in Z direction between data sets in True3D area, bar, or tape graph

Syntax

```
[form.]graph.True3DZGap[ = setting]
```

Data type

Integer (0-95)

Settings

Valid settings are integers from 0 to 95 (default is 20).

Description

The True3DZGap property sets the gap in the Z direction between data sets in three kinds of True3D graphs with multiple data sets: bar (z-clustered style), area (absolute style), and tape.

The True3DZGap value is expressed as a percentage of the entire possible width of each bar (or area plot or tape). The remaining percentage is occupied by the bar itself.

The default True3DZGap setting of 20 creates a gap one-fourth as wide as each bar--the gap occupies 20 percent of the available space, the bar 80 percent. A setting of 50 produces gaps and bars of equal width, and a setting of 0 produces no gap (the bars are adjacent).

▲▼

Topic

True3DZGap

Related

3D property page
GraphStyle
True3D
True3DXGap

XAxisMax property

Sets maximum value for X axis

Syntax

[*form*.]*graph*.**XAxisMax**[= *number*]

Data type

Numeric (integer, long, single, double)

Description

The XAxisMax property sets the maximum value for the X axis. It's used only when the XAxisStyle property is set to 2 (user-defined) and works in combination with the XAxisMin and XAxisTicks properties. See the entry for XAxisStyle for details.

▲▼

Topic

XAxisMax

Related

Axis property page
XAxisMin
XAxisPos
XAxisStyle
XAxisTicks

XAxisMin property

Sets minimum value for X axis

Syntax

[*form*.]*graph*.**XAxisMin**[= *number*]

Data type

Numeric (integer, long, single, double)

Description

The XAxisMin property sets the minimum value for the X axis. It's used only when the XAxisStyle property is set to 2 (user-defined) and works in combination with the XAxisMax and XAxisTicks properties. See the entry for XAxisStyle for details.

▲▼

Topic

XAxisMin

Related

Axis property page
XAxisMax
XAxisPos
XAxisStyle
XAxisTicks

XAxisMinorTicks property

Enables minor tick marks on X axis

Syntax

[*form*.]*graph*.**XAxisMinorTicks**[= *setting*]

Data type

Integer (enumerated)

Settings

0	Off (default)
1	On--five minor ticks between pairs of major ticks
-1 to -100	On--number of minor ticks is defined by setting, expressed as a negative integer

Description

The XAxisMinorTicks property enables minor tick marks on the X axis. Minor tick marks further subdivide the X axis between pairs of major (standard) tick marks.

When XAxisMinorTicks is set to 1, five minor ticks are drawn between each pair of major ticks. You can control exactly how many minor ticks are drawn by expressing the desired number as a negative integer. A setting of -1 results in 1 minor tick. A setting of -100 results in 100 minor ticks.

Note: The ability to specify the number of minor tick marks was added to the Graph Control in version 5.0. In order to avoid breaking code written for earlier versions of the control, the affect of setting the property to 1 had to be retained. Thus a user-defined number of ticks must be expressed as a negative integer.

XAxisMinorTicks applies to all 2D graphs except pie charts and polar graphs. It doesn't apply to 3D graphs.

▲▼

Topic

XAxisMinorTicks

Related

Axis property page
TickStyle
XAxisMax
XAxisMin
XAxisPos
XAxisStyle
XAxisTicks

XAxisPos property

Sets position of X axis in 2D graph

Syntax

```
[form.]graph.XAxisPos[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Bottom, top, or middle, depending on GraphData values (default)
1	Top
2	Bottom

Description

The XAxisPos property sets the position of a 2D graph's X axis--at the bottom, at the top, or in the middle of the graph.

By default, the Graph control positions the X axis automatically according to the data values carried by the GraphData property. If all values are positive, the X axis is at the bottom; if all are negative, the axis is at the top; if there's a mix of positive and negative values, the axis is in the middle (intersecting the Y origin).

By setting XAxisPos to 1 (top) or 2 (bottom), you can override the default placement of the X axis regardless of the GraphData values.

In 3D graphs, X axes are always drawn at the bottom.

▲▼

Topic

XAxisPos

Related

Axis property page
XAxisMax
XAxisMin
XAxisStyle
XAxisTicks

XAxisStyle property

Sets method for determining scale and range of X axis

Syntax

[*form*.]*graph*.**XAxisStyle**[= *setting*]

Data type

Integer (0-2, enumerated)

Settings

0	Zero origin (default)
1	Variable origin
2	User-defined

Description

The XAxisStyle property sets the method for determining the scale and range of a graph's X axis (or the Y axis of horizontal bar graphs and Gantt charts). This property doesn't apply to pie charts or polar graphs.

The effect of XAxisStyle depends on whether or not you've set values for the XPosData property. Bubble, lin/log, and log/log graphs always have XPosData values, scatter graphs generally do, and other graph types (except pie and Gantt charts) can have them if you choose.

Without XPosData

If you don't provide XPosData values, each data point in each data set has an explicit Y value (carried by the GraphData property) and an implicit X value.

0 Zero origin The X axis always begins at 0 (1 for bar graphs) and has one tick mark per point. You can't set a minimum value, maximum value, or number of ticks for the axis.

1 Variable origin This setting has no effect--it works just like setting 0 (zero origin).

2 User-defined You can specify the number of tick marks for the X axis using the XAxisTicks property. However, you can't set the minimum and maximum values for the X axis--any settings for XAxisMax and XAxisMin are ignored. For details, see the entry for XAxisTicks.

With XPosData

Graphs with XPosData values are true X-Y graphs--each point has both an explicit X value (carried by the XPosData property) and an explicit Y value (carried by the GraphData property).

In this type of graph, the XAxisStyle property provides the same control over the X axis as the YAxisStyle property provides over the Y axis.

0 Zero origin The Graph control automatically calculates the X axis range based on your XPosData values. The range includes the X origin (zero) and extends at least far enough in the positive and negative directions to include all XPosData. You can't set a minimum value, maximum value, or number of ticks for the axis.

1 Variable origin The Graph control attempts to zoom in on your XPosData values when it calculates the X axis range. The range is made at least wide enough to include all XPosData values, but it doesn't necessarily include the X

		origin (zero). You can't set a minimum value, maximum value, or number of ticks for the axis. This setting is useful when all your XPosData values are clustered in a narrow range that doesn't include zero.
2	User-defined	The X axis is drawn according to your settings for the minimum, maximum, and number of ticks. Refer to the XAxisTicks, XAxisMax, and XAxisMin properties.

▲▼

Topic

XAxisStyle

Related

Axis property page
ClipGraph
XAxisMax
XAxisMin
XAxisPos
XAxisTicks

XAxisTicks property

Sets number of ticks on X axis

Syntax

[*form*.]*graph*.**XAxisTicks**[= *number*]

Data type

Integer (1-100)

Settings

Valid settings are integers from 1 to 100 (default is 1).

Description

The XAxisTicks property sets the number of ticks on a graph's X axis (or the Y axis of horizontal bar graphs and Gantt charts). This property doesn't apply to pie charts or polar graphs.

XAxisTicks is used only when the XAxisStyle property is set to 2 (user-defined) and works in combination with the XAxisMin and XAxisMax properties.

The effect of XAxisTicks depends on whether or not the graph uses the XPosData property to set independent X values.

Without XPosData

- If XAxisTicks is less than or equal to the value of the NumPoints property, setting the property has no effect.
- If XAxisTicks is greater than the value of the NumPoints property, the X axis is extended to the value of XAxisTicks. You can also label the additional ticks using the LabelText property.

This feature helps when you want to create graphs for which you have incomplete data. For example, you might want to label your X axis for 12 months (January through December) when you have data only for five months (January through May). Although the NumPoints property is set to 5, you can set XAxisTicks to 12 and create labels for all 12 points.

With XPosData

If you have XPosData, XAxisTicks specifies the number of ticks from the origin to the setting of *either* the XAxisMax or XAxisMin property--whichever has the higher magnitude (distance from zero). Because both the maximum and minimum points of an axis must fall evenly on a tick, the Graph control will override the lower-magnitude range setting (XAxisMax or XAxisMin) if necessary.

▲▼

Topic

XAxisTicks

Related

Axis property page

XAxisMax

XAxisMin

XAxisPos

XAxisStyle

XPos() run-time property

Alternative way to set independent X values at run time

Syntax

[*form*.]*graph*.**XPos(*index%*)**[= *setting*]

Data type

Numeric (integer, long, single, double)

Parameter

index% Element number of array normally set by XPosData property

Description

The XPos() run-time property sets independent X values for a graph. It's a run-time alternative to the XPosData property.

X values are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But XPos(), unlike XPosData, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your XPos() parameter.

Note In earlier versions of the Graph control (called ChartBuilder), XPosData/XPos() values were stored as a one-dimensional array. Values stored in the old format are loaded by the current Graph control as the first set of XPosData/XPos().

Example

If you want to assign an X value of 12.5 to the fifth data point in a graph's second data set, you can set up your code in two ways:

- If you use the XPosData property, you set the ThisSet and ThisPoint properties first:

```
Graph1.ThisSet = 2
Graph1.ThisPoint = 5
Graph1.XPosData = 12.5
```

- If you use the XPos() property, you don't have to set ThisPoint:

```
Graph1.ThisSet = 2
Graph1.XPos(5) = 12.5
```

▲▼

Topic

XPos()

Related

XPosData

XPosData property

Sets independent X values

Syntax

[*form*.]*graph*.**XPosData**[= *data*]

Data type

Numeric (integer, long, single, double)

Description

The XPosData property sets independent X values for a graph. It's normally used with scatter graphs, which generally require XPosData to be meaningful, and with bubble, lin/log, and log/log graphs, which *always* require XPosData. However, you can also apply XPosData to all other graph types except pie and Gantt charts.

XPosData values are stored as a two-dimensional array indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint values as you enter data.

Note In earlier versions of the Graph control (called ChartBuilder), XPosData values were stored as a one-dimensional array. XPosData stored in the old format are loaded by the current Graph control as the first set of XPosData.

If you're graphing more than one data set (multiple sets stored in the GraphData property) but store only one set of XPosData, the Graph control applies those XPosData values to all GraphData sets.

Example

The following lines use the AutoInc property to assign XPosData values to all data points of all data sets in order:

```
Graph1.AutoInc = 1
For I% = 1 To Graph1.NumSets
    For J% = 1 To Graph1.NumPoints
        Graph1.XPosData = <your data value>
    Next
Next
Graph1.DrawMode = 2
```

▲▼

Topic

XPosData

Related

AutoInc

Data property page

DataReset

ThisPoint

ThisSet

XPos()

XPosField() property

Alternative way to select fields for X values at run time

Syntax

[*form*.]*graph*.**XPosField(*index*%)**[= *fieldname*]

Note: This property is not available with the VBX interface.

Parameter

index% Element number of array normally set by the XPosFields property.

Data type

String

Settings

One or more valid field names in a table of a bound database.

Description

The XPosField() property gives you an alternative method for setting the values normally set by the XPosFields property. XPosField() is available only at run time.

Both XPosFields and XPosField() store values in a one-dimensional array. With XPosFields you must first point to an element of the array by setting ThisPoint, and then store a value by setting XPosFields. You can skip the first step by using XPosField(). Rather than setting ThisPoint, you can simply pass the number of an array element as a parameter of XPosField().

▲▼

Topic

XPosField()

Related

XPosFields

XPosFields property

Selects database fields to be graphed as X values

Syntax

[*form*.]*graph*.**XPosFields**[= *fieldname*]

Note: This property is not available with the VBX interface.

Data type

String

Settings

One or more valid field names in a table of a bound database.

Description

XPosFields identifies which fields of a bound database should be graphed as independent X values. It is a one-dimensional array property indexed by ThisPoint. The array is one-based, it has NumSets elements, and each element corresponds to a data set. The setting for each element should be the name of a field containing that data set's X values.

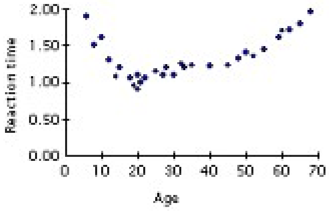
If no element of XPosFields is set, the Graph Control graphs only Y data. In this case the X positions of data points are determined by sequence rather than value.

If you are graphing multiple sets of Y values but set only the first element of XPosFields, that field's X values will be applied to all data sets.

If you have set XPosFields and want to return to graphing only Y data, you must clear the XPosFields array. This cannot be done by setting XPosFields elements to null strings. To clear the array, you must set DataReset to 22.

Example

```
'Load a new table
Data1.RecordSource = "Reaction time"

With Graph1
    'Start with a fresh set of field names
    .DataReset = 20 'Clears DataFields
    .DataReset = 22 'Clears XPosFields

    'Note: This graph has only one data set

    'Select fields to graph
    .DataFields = "Time"    'Y values
    .XPosFields = "Age"     'X values

    'Label the axes
    .RightTitle = "Reaction time"
    .BottomTitle = "Age"

    'Redraw the graph
    Data1.Refresh

End With
```


2.00
1.50
1.00
0.50
0.00
Reaction time
0
10
20
30
40
50
60
70
Age

▲▼

Topic

XPosFields

Related

XPosField()

DataFields

XPosData

YAxisMax property

Sets maximum value for Y axis

Syntax

```
[form.]graph.YAxisMax[ = setting]
```

Data type

Numeric (integer, long, single, double)

Description

The YAxisMax property sets the maximum value for the Y axis specified by the YAxisUse property. It's used only when the YAxisStyle property is set to 2 (user-defined) and works in combination with the YAxisMin and YAxisTicks properties. See the entry for YAxisStyle for details.

▲▼

Topic

YAxisMax

Related

Axis property page

YAxisUse

YAxisMin

YAxisMin property

Sets minimum value for Y axis

Syntax

`[form.]graph.YAxisMin[ = setting]`

Data type

Numeric (integer, long, single, double)

Description

The YAxisMin property sets the minimum value for the Y axis specified by the YAxisUse property. It's used only when the YAxisStyle property is set to 2 (user-defined) and works in combination with the YAxisMax and YAxisTicks properties. See the entry for YAxisStyle for details.

▲▼

Topic

YAxisMin

Related

Axis property page

YAxisUse

YAxisMax

YAxisMinorTicks property

Enables minor tick marks on Y axis

Syntax

```
[form.]graph.YAxisMinorTicks[ = setting]
```

Data type

Integer (enumerated)

Settings

0	Off (default)
1	On--five minor ticks between pairs of major ticks
-1 to -100	On--number of minor ticks is defined by setting, expressed as a negative integer

Description

The YAxisMinorTicks property enables minor tick marks on the Y axis specified by the YAxisUse property (or the X axis of horizontal bar graphs and Gantt charts). Minor tick marks further subdivide the axis between pairs of major (standard) tick marks.

When YAxisMinorTicks is set to 1, five minor ticks are drawn between each pair of major ticks. You can control exactly how many minor ticks are drawn by expressing the desired number as a negative integer. A setting of -1 results in 1 minor tick. A setting of -100 results in 100 minor ticks.

Note: The ability to specify the number of minor tick marks was added to the Graph Control in version 5.0. In order to avoid breaking code written for earlier versions of the control, the affect of setting the property to 1 had to be retained. Thus a user-defined number of ticks must be expressed as a negative integer.

YAxisMinorTicks applies to all 2D graphs except pie charts and polar graphs. It doesn't apply to 3D graphs.

▲▼

Topic

YAxisMinorTicks

Related

Axis property page
TickStyle
YAxisUse
YAxisMax
YAxisMin
YAxisPos
YAxisStyle
YAxisTicks

YAxisPos property

Sets position of Y axis in 2D graph with one Y axis

Syntax

```
[form.]graph.YAxisPos[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Left, right, or middle, depending on X values (default)
1	Left
2	Right

Description

The YAxisPos property sets the position of a 2D graph's Y axis--to the left, to the right, or in the middle of the graph--when you're using only one Y axis.

By default, the Graph control positions the Y axis automatically according to the graph's X values. If all values are positive, the Y axis is at the left; if all are negative, the axis is at the right; if there's a mix of positive and negative values, the axis is in the middle (intersecting the X origin).

By setting YAxisPos to 1 (left) or 2 (right), you can override the default placement of the Y axis regardless of the X values.

In combination graphs with two Y axes, the YAxisPos setting is ignored.

In 3D graphs, Y axes are always drawn at the left side of the graph, except when you use the CageFlip property to move the side wall to the opposite side of the cage.

▲▼

Topic

YAxisPos

Related

Axis property page

YAxisUse

YAxisMax

YAxisMin

YAxisStyle

YAxisTicks

YAxisStyle property

Sets method for determining scale and range of Y axis

Syntax

```
[form.]graph.YAxisStyle[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Zero origin (default for all graph types except time series)
1	Variable origin
2	User-defined (default for time series graphs)

Description

The YAxisStyle property sets the method for determining the scale and range of the Y axis specified by the YAxisUse property.

- When YAxisStyle is set to 0 (zero origin), the Graph control automatically calculates the Y axis range based on the values carried by the GraphData property. The range includes the Y origin (zero) and extends at least far enough in the positive and negative directions to include all GraphData values. You can't set a minimum value, maximum value, or number of ticks for the axis.
- When YAxisStyle is set to 1 (variable origin), the Graph control attempts to zoom in on your data values (as carried by the GraphData property) when it calculates the Y axis range. The range is made at least wide enough to include all GraphData values, but it doesn't necessarily include the Y origin (zero). You can't set a minimum value, maximum value, or number of ticks for the axis.

  This style is useful when all your GraphData values are clustered in a narrow range that doesn't include zero.
- When YAxisStyle is set to 2 (user-defined), the Y axis is drawn according to your settings for the minimum, maximum, and number of ticks. Refer to the entries for the YAxisTicks, YAxisMax, and YAxisMin properties.

  This style lets you set the scale and range of the Y axis without regard to the actual values you're graphing.

  Note If any of the values carried by the GraphData property fall outside the limits of the Y axis range you specify, those values will still be drawn. However, they'll appear outside the bounds of your axes.

▲▼

Topic

YAxisStyle

Related

Axis property page
ClipGraph
YAxisUse
YAxisMax
YAxisMin
YAxisPos
YAxisTicks

YAxisTicks property

Sets number of tick marks on Y axis

Syntax

```
[form.]graph.YAxisTicks[ = number]
```

Data type

Integer (1-100)

Settings

Valid settings are integers from 1 to 100 (default is 1).

Description

The YAxisTicks property sets the number of ticks on the Y axis specified by the YAxisUse property. It's used only when the YAxisStyle property is set to 2 (user-defined) and works in combination with the YAxisMin and YAxisMax properties.

YAxisTicks specifies the number of ticks from the origin to the setting of either the YAxisMax or YAxisMin property--whichever has the higher magnitude (distance from zero). Because both the maximum and minimum points of an axis must fall evenly on a tick, the Graph control will override the lower-magnitude range setting (YAxisMax or YAxisMin) if necessary.

Topic

YAxisTicks

Related

Axis property page
YAxisUse
YAxisMax
YAxisMin
YAxisPos

YAxisUse property

Determines Y axis to which other axis properties apply

Syntax

[*form*.]*graph*.**YAxisUse**[= *setting*]

Data type

Integer (0-1, enumerated)

Settings

0	Primary Y axis (default)
1	Right-hand axis in combination graphs

Description

The YAxisUse property determines whether the other Y axis properties--YAxisMax, YAxisMin, YAxisMinorTicks , YAxisStyle, YAxisTicks, and YLabelText--apply to the primary Y axis or to the right-hand Y axis available in combination graphs.

▲▼

Topic

YAxisUse

Related

Axis property page
YAxisMax
YAxisMin
YAxisMinorTicks
YAxisPos
YAxisStyle
YAxisTicks

YLabel() run-time property

Alternative way to set label text for Y axis at run time

Syntax

```
[form.]graph.YLabel(index%)[ = "label"]
```

Data type

String (up to 80 characters)

Parameter

index% Element number of array normally set by YLabelText property

Description

The YLabel() run-time property sets the label text for the Y axis specified by the YAxisUse property. YLabel() is a run-time equivalent to the YLabelText property.

The text for Y labels is stored as a two-dimensional array, which is normally indexed by the YAxisUse and ThisPoint properties. However, unlike YLabelText, YLabel() doesn't require you to use the ThisPoint property to index the label array. Instead, you provide the element (point) number as your YLabel() parameter.

Example

If you want to label the first tick on the primary Y axis "January," you can set up your code in two ways:

- If you use the YLabelText property, you set the YAxisUse and ThisPoint properties first:

  ```
  Graph1.YAxisUse = 0
  Graph1.ThisPoint = 1
  Graph1.YLabelText = "January"
  ```

- If you use the YLabel() property, you don't have to set ThisPoint:

  ```
  Graph1.YAxisUse = 0
  Graph1.YLabel(1) = "January"
  ```

▲▼

Topic

YLabel()

Related

Labels property page

YLabelText

YLabelText property

Sets label text for Y axis

Syntax

```
[form.]graph.YLabelText[ = "label"]
```

Data type

String (up to 80 characters)

Description

The YLabelText property sets the label text for the Y axis specified by the YAxisUse property.

If you don't set YLabelText values, Y axis labels show the numeric scaling of the axis. With the YLabelText property, you can specify text labels instead.

YLabelText is a one-dimensional array-based property indexed by the ThisPoint property. You can use the AutoInc property to automatically increment the ThisPoint counter each time you set a YLabelText element.

Each label can contain no more than 80 characters. Also, if any of the labels you define are too long to fit in the graphing window at the specified size, no labels are drawn. If this happens, you may be able to make the labels appear by increasing the size of the graphing window.

▲▼

Topic

YLabelText

Related

Labels property page
YAxisUse
YLabel()
LabelText
ZLabelText

ZAxisMax property

Sets maximum value for Z axis of True3D scatter graph

Syntax

[*form*.]*graph*.**ZAxisMax**[= *number*]

Data type

Numeric (integer, long, single, double)

Description

The ZAxisMax property sets the maximum value for the Z axis of a True3D scatter graph. It's used only when the ZAxisStyle property is set to 2 (user-defined) and works in combination with the ZAxisMin and ZAxisTicks properties. See the entry for ZAxisStyle for details.

▲▼

Topic

ZAxisMax

Related

Axis property page

ZAxisMin

ZAxisStyle

ZAxisTicks

ZAxisMin property

Sets minimum value for Z axis of True3D scatter graph

Syntax

[*form*.]*graph*.**ZAxisMin**[= *number*]

Data type

Numeric (integer, long, single, double)

Description

The ZAxisMin property sets the minimum value for the Z axis of a True3D scatter graph. It's used only when the ZAxisStyle property is set to 2 (user-defined) and works in combination with the ZAxisMax and ZAxisTicks properties. See the entry for ZAxisStyle for details.

▲▼

Topic

ZAxisMin

Related

Axis property page

ZAxisMax

ZAxisStyle

ZAxisTicks

ZAxisStyle property

Sets method for determining scale and range of Z axis in True3D scatter graph

Syntax

```
[form.]graph.ZAxisStyle[ = setting]
```

Data type

Integer (0-2, enumerated)

Settings

0	Zero origin (default)
1	Variable origin
2	User-defined

Description

The ZAxisStyle property sets the method for determining the scale and range of the Z axis in a True3D scatter graph. It provides the same control over the Z axis as the XAxisStyle and YAxisStyle properties provide over the X and Y axes.

0 Zero origin — The Graph control automatically calculates the Z axis range based on your ZPosData values. The range includes the Z origin (zero) and extends at least far enough in the positive and negative directions to include all ZPosData. You can't set a minimum value, maximum value, or number of ticks for the axis.

1 Variable origin — The Graph control attempts to zoom in on your ZPosData values when it calculates the Z axis range. The range is made at least wide enough to include all ZPosData values, but it doesn't necessarily include the Z origin (zero). You can't set a minimum value, maximum value, or number of ticks for the axis.

This setting is useful when all your ZPosData values are clustered in a narrow range that doesn't include zero.

2 User-defined — The Z axis is drawn according to your settings for the minimum, maximum, and number of ticks. Refer to the ZAxisTicks, ZAxisMax, and ZAxisMin properties.

▲▼

Topic

ZAxisStyle

Related

Axis property page

ZAxisMax

ZAxisMin

ZAxisTicks

ZLabelText

ZAxisTicks property

Sets number of tick marks on Z axis of True3D scatter graph

Syntax

[*form*.]*graph*.**ZAxisTicks**[= *number*]

Data type

Integer (1-100)

Settings

Valid settings are integers from 1 to 100 (default is 1).

Description

The ZAxisTicks property sets the number of ticks on the Z axis of a True3D scatter graph. It's used only when the ZAxisStyle property is set to 2 (user-defined) and works in combination with the ZAxisMin and ZAxisMax properties.

In a True3D scatter graph, ZAxisTicks specifies the number of ticks from the origin to the setting of *either* the ZAxisMax or ZAxisMin property--whichever has the higher magnitude (distance from zero). Because both the maximum and minimum points of an axis must fall evenly on a tick, the Graph control will override the lower-magnitude range setting (ZAxisMax or ZAxisMin) if necessary.

▲▼

Topic

ZAxisTicks

Related

Axis property page

ZAxisStyle

ZAxisMax

ZAxisMin

ZLabel() run-time property

Alternative way to set label text for Z axis of True3D graph at run time

Syntax

`[form.][graph.]ZLabel(index%)[ = "label"]`

Parameter

index% Element number of array normally set by ZLabelText property

Data type

String (up to 80 characters)

Description

The ZLabel() run-time property sets the label text for the Z axis of a True3D graph. It's a run-time alternative to the ZLabelText property.

The text for Z labels is stored as a one-dimensional array. Unlike ZLabelText, ZLabel() doesn't require you to use the ThisPoint property to index this array. Instead, you provide the element (point) number as your ZLabel() parameter.

For an illustration of the different ways to set label text at design time and run time, see the example under the entry for the Label() run-time property.

▲▼

Topic

ZLabel()

Related

Labels property page

ZLabelText

ZLabelText property

Sets label text for Z axis of True3D graph

Syntax

`[form.]graph.ZLabelText[ = "label"]`

Data type

String (up to 80 characters)

Description

The ZLabelText property sets the label text for the Z axis of a True3D graph. For True3D scatter graphs, if you don't set ZLabelText values, Z axis labels show the numeric scaling of the axis. For True3D bar, tape, area, and surface graphs, numeric labels aren't provided; if you don't set ZLabelText, no labels appear.

ZLabelText is a one-dimensional array-based property indexed by the ThisPoint property. You can use the AutoInc property to automatically increment the ThisPoint counter each time you set a ZLabelText element.

Each label can contain no more than 80 characters. Also, if any of the labels you define are too long to fit in the graphing window at the specified size, no labels are drawn. If this happens, you may be able to make the labels appear by increasing the size of the graphing window.

▲▼

Topic

ZLabelText

Related

Labels property page

LabelText

LabelStyle

ZPos() run-time property

Alternative way to set independent Z values for True3D scatter graph at run time

Syntax

```
[form.]graph.ZPos(index%)[ = setting]
```

Data type

Numeric (integer, long, single, double)

Parameter

index% Element number of array normally set by ZPosData property

Description

The ZPos() run-time property sets independent Z values for a True3D scatter graph. It's a run-time alternative to the ZPosData property.

Z values are stored as a two-dimensional array, which is normally indexed by the ThisSet and ThisPoint properties. But ZPos(), unlike ZPosData, requires only a ThisSet value. Instead of setting ThisPoint, you provide the element (point) number as your ZPos() parameter.

Example

If you want to assign a Z value of 10 to the fifth data point in a graph's second data set, you can set up your code in two ways:

- If you use the ZPosData property, you set the ThisSet and ThisPoint properties first:

  ```
  Graph1.ThisSet = 2
  Graph1.ThisPoint = 5
  Graph1.ZPosData = 10
  ```

- If you use the ZPos() property, you don't have to set ThisPoint:

  ```
  Graph1.ThisSet = 2
  Graph1.ZPos(5) = 10
  ```

▲▼

Topic

ZPos()

Related

ZPosData

ZPosData property

Sets independent Z values for True3D scatter graph

Syntax

```
[form.]graph.ZPosData[ = data]
```

Data type

Numeric (integer, long, single, double)

Description

The ZPosData property sets independent Z values for a 3D scatter graph.

ZPosData values are stored as a two-dimensional array indexed by the ThisSet and ThisPoint properties. You can use the AutoInc property to automatically increment the ThisSet and ThisPoint values as you enter data.

If you're graphing more than one data set (multiple sets stored in the GraphData property) but store only one set of ZPosData, the Graph control applies those ZPosData values to all GraphData sets.

▲▼

Topic

ZPosData

Related

Data property page
DataReset
XPosData
ZPos()

Label Formats

Numeric format strings
Date/time format strings

Numeric formats

Axes and data points can be labeled either with text or with numbers. When the labels are numeric (set, point, or data values) the numbers can be formatted for display by setting the properties DataLabelFormat, LabelXFormat, LabelYFormat and LabelZFormat. These properties accept a format string similar to those for user-defined numeric formats in Visual Basic. A sequence of placeholder and control characters defines how numbers are formatted for display.

Numeric format characters

Character **Meaning**

0 *Digit placeholder*

Display a digit or a zero. If the number being formatted has a digit in the position where the 0 appears, display it. Otherwise display a zero.

If the number has fewer digits than there are zeros in the format expression, display leading or trailing zeros.

*Digit placeholder*

Display a digit or nothing. If the number being formatted has a digit in the position where the # appears, display it. Otherwise display nothing.

This symbol works like the 0 except that leading and trailing zeros aren't displayed.

. *Decimal placeholder*

The decimal placeholder determines how many digits are displayed to the left and right of the decimal separator. If the format expression has nothing to the left of this symbol, numbers smaller than 1 begin with a decimal separator. If you want a leading zero always to be displayed with fractional numbers, use 0 as the first digit placeholder to the left of the decimal separator.

The actual character used as a decimal separator in the formatted output depends on your system settings.

, *Thousand separator*

The thousand separator separates thousands from hundreds within a number that has four or more places to the left of the decimal separator. Standard use of the thousand separator is specified if the format contains a comma surrounded by digit placeholders (0 or #). Two adjacent commas or a comma immediately to the left of the decimal separator (whether or not a decimal is specified) means "scale the number by dividing by a thousand, rounding as needed". You can scale large numbers using this technique. For example, you can use the format string "##0,," to represent 100 million as 100.

The actual character used as the thousand separator in the formatted output depends on your system settings.

% *Percentage placeholder*

The number is multiplied by 100. The percent character (%) is inserted in the position where it appears in the format string.

E-E+e-e+ *Scientific format*

If the format expression contains at least one digit placeholder (0 or #) to the right of E-, E+, e- or e+ the number is displayed in scientific format. The character "E" or "e" is inserted between the number and its exponent. The number of digits in the exponent is determined by the number of digit placeholders to the right.

Use E- or e- to place a minus sign next to negative exponents. Use E+ or e+ to place a minus sign next to negative exponents and a plus sign next to positive exponents.

\ *Literal character*

Display the character immediately following "\" in the format string.

Literal characters can be inserted before or after the formatted number. For example "\$##0.0" will insert "$" before the formatted number. "##0.0\D\M" will insert "DM" after the formatted number.

Examples

Number	**Format string**	**Displayed**
1234	0	1234
1234	00000	01234
1234	#,##0	1,234
1234	#,##0.00	1,234.00
1234	\$#,##0.00	$1,234.00
123456	\$#,##0,\k	$123k
12345678	\$#,##0,,\m	$12m
12345678	0.0E+00	1.2E+07
0.1234	0.00	0.12
0.1234	0.00000	0.12340
0.1234	0.0E-00	1.2E-01
0.1234	0.00%	12.34%
0.1234	0%	12%

▲▼

Topic

Numeric formats

Related

Date/time formats

DataLabelFormat

LabelXType

LabelXFormat

LabelYFormat

LabelZFormat

Date/time formats

The X axis can be labeled with text, numbers or an automatically generated series of date and time labels. To label the axis with a series of dates or times, use the LabelXType property to select the type of labels (date, time, date and time, or weekday). Set a starting date using the LabelXDateStart property and an increment using LabelXDateInc.

To apply a format to dates or times, set LabelXFormat.This property accepts a label format string similar to those for user-defined date and time formats in Visual Basic. A sequence of placeholder and control characters defines how dates and times are formatted for display.

Date format characters

Character	**Meaning**
/ - : . ,	Delimiters. Any of these characters, as well as a space character, may be used to separate elements of the displayed date.
c	Display the date as ddddd and the time as ttttt in that order.
d	Display the day as a number without a leading zero (1-31).
dd	Display the day as a number with a leading zero (01-31).
ddd	Display the day as an abbreviation (Sun/Sat).
dddd	Display the day as a full name (Sunday/Saturday).
ddddd	Display the date as a short date (including day, month, and year), formatted according to your system's short date format setting. The default short date format is m/d/yy.
dddddd	Display the date as a long date (including day, month, and year) formatted according to the long date setting recognized by your system. The default long date format is mmmm dd, yyyy.
w	Display the day of the week as a number (1 for Sunday through 7 for Saturday).
ww	Display the week of the year as a number (1-53).
m	Display the month as a number without a leading zero (1-12).
mm	Display the month as a number with a leading zero (01-12).
mmm	Display the month as an abbreviation (Jan-Dec).
mmmm	Display the month as a full month name (January-December).
q	Display the quarter of the year as a number (1-4).
y	Display the day of the year as a number (1-366).
yy	Display the year as a 2-digit number (00-99).
yyyy	Display the year as a 4-digit number (1900-2037).

Examples

yyyy-mm-dd	**Format string**	**Displayed**
1997-03-01	d mmm yy	1 Mar 97
1997-03-01	dddd	Saturday
1997-03-01	mmmm	March

1997-03-01	dd mmmm yyyy	01 March 1997
1997-03-01	dddd, mmmm d, yyyy	Saturday, March 1, 1997
1997-03-01	w	7
1997-03-01	q	1
1997-03-01	y	60

Time format characters

Character	Meaning
/ - : . ,	Delimiters. Any of these characters, as well as a space character, may be used to separate elements of the displayed time.
h	Display the hour as a number without leading zeros (1-23).
hh	Display the hour as a number with leading zeros (01-23).
n	Display the minute as a number without leading zeros (0-59).
nn	Display the minute as a number with leading zeros (00-59).
s	Display the second as a number without leading zeros (0-59).
ss	Display the second as a number with leading zeros (00-59).
t t t t t	Display a time as a complete time (including hour, minute, and second), formatted using the time separator defined by the time format recognized by your system. The default time format is h:mm:ss.
AM/PM	Use the 12-hour clock and display an uppercase AM with any hour before noon; display an uppercase PM with any hour between noon and 11:59 P.M.
am/pm	Use the 12-hour clock and display a lowercase AM with any hour before noon; display a lowercase PM with any hour between noon and 11:59 P.M.
A/P	Use the 12-hour clock and display an uppercase A with any hour before noon; display an uppercase P with any hour between noon and 11:59 P.M.
a/p	Use the 12-hour clock and display a lowercase A with any hour before noon; display a lowercase P with any hour between noon and 11:59 P.M.

Examples

hh-mm-ss	Format string	Displayed
13-30-25	ttttt	13:30:25
13-30-25	h:nn	13:30
13-30-25	nn:ss	30:25
13-30-25	h:nn AM/PM	1:30 PM
13-30-25	hh:nn:ss a/p	01:30:25 p

▲▼

Topic

Date/time formats

Related

Numeric formats
LabelXType
LabelXDateStart
LabelXDateInc
LabelXFormat