GLG Widget Reference Manual

This book provides information about the GLG Widgets. It contains the following chapters:

Table of Contents

Using the GLG Widgets
A general description of the GLG Widgets.

The GLG Widget Library
A list of Glg Widgets with description and information on how to use each widget.

GLG Widget Resource Sets
A list of GLG Widget resources.

Index

Please note that although the illustrations in this document represent the UNIX version of the GLG Graphics Builder, the information it contains is equally relevant to Microsoft Windows users. The two versions present the same functionality in equivalent user interfaces, with minimal, cosmetic differences caused by the different platforms.
# GLG Widget Reference Manual

## Chapter 1 Using the GLG Widgets .............................................................. 15

Using Widgets in the Graphics Builder................................................................. 15
Real-Time Charts and Graph Widgets................................................................. 17
  Real-Time Charts............................................................................................. 18
  2D and 3D Graphs ......................................................................................... 19
Components of a GLG Graphs.......................................................................... 19
Naming Conventions......................................................................................... 20
Performance Optimizations............................................................................. 21
Commonly Used Graph Resources ................................................................. 22
Data Samples and the Data Group................................................................. 23
Data Area.......................................................................................................... 27
Axes.................................................................................................................. 27
Major And Minor Ticks.................................................................................... 28
Grid Lines......................................................................................................... 29
Labels............................................................................................................... 29
Level Objects................................................................................................. 32
Status Object................................................................................................. 32
Titles............................................................................................................... 32
Legends........................................................................................................... 32
Multiple Axes............................................................................................... 33
3D viewing ..................................................................................................... 33
Input Widgets, Dials and Meters..................................................................... 33
Meters, Dials and Avionics gauges ............................................................... 34
Rotary Axis................................................................................................. 34
Slider............................................................................................................. 35
Knob.............................................................................................................. 38
Button.......................................................................................................... 40
Menu............................................................................................................. 40
Text............................................................................................................... 41
Chapter 2  The GLG Widget Library ................................................................. 47

Bar Graph
Signed Bar Graph
Histogram Graph ................................................................. 48
  Animation Data ................................................................. 48
  Files .................................................................................. 48
  Resource Sets ................................................................. 48
Horizontal Bar Graph .......................................................... 49
  Animation Data ................................................................. 49
  Files .................................................................................. 49
  Resource Sets ................................................................. 49
Symmetrical Bar Graph .......................................................... 49
  Animation Data ................................................................. 49
  Files .................................................................................. 50
  Resource Sets ................................................................. 50
Packed Bar Graph ................................................................. 50
  Animation Data ................................................................. 50
  Files .................................................................................. 51
  Resource Sets ................................................................. 51
Stacked Bar Graph ................................................................. 51
  Animation Data ................................................................. 51
  Files .................................................................................. 52
  Resource Sets ................................................................. 52
Step Graph ................................................................. 52
  Animation Data ................................................................. 52
  Files .................................................................................. 52
  Resource Sets ................................................................. 52
Filled Step Graph ................................................................. 53
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animation Data</td>
<td>53</td>
</tr>
<tr>
<td>Files</td>
<td>53</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>53</td>
</tr>
<tr>
<td>Horizontal Step Graph</td>
<td>54</td>
</tr>
<tr>
<td>Animation Data</td>
<td>54</td>
</tr>
<tr>
<td>Files</td>
<td>54</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>54</td>
</tr>
<tr>
<td>Horizontal Filled Step Graph</td>
<td>54</td>
</tr>
<tr>
<td>Animation Data</td>
<td>54</td>
</tr>
<tr>
<td>Files</td>
<td>55</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>55</td>
</tr>
<tr>
<td>Multiline Step Graph</td>
<td>55</td>
</tr>
<tr>
<td>Animation Data</td>
<td>55</td>
</tr>
<tr>
<td>Files</td>
<td>56</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>56</td>
</tr>
<tr>
<td>Filled Multiline Step Graph</td>
<td>56</td>
</tr>
<tr>
<td>Animation Data</td>
<td>56</td>
</tr>
<tr>
<td>Files</td>
<td>57</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>57</td>
</tr>
<tr>
<td>Multiline Graph</td>
<td>57</td>
</tr>
<tr>
<td>Line Graph</td>
<td></td>
</tr>
<tr>
<td>Filled Line Graph</td>
<td></td>
</tr>
<tr>
<td>Line Segment Graph</td>
<td></td>
</tr>
<tr>
<td>Point Graph</td>
<td></td>
</tr>
<tr>
<td>Animation Data</td>
<td>57</td>
</tr>
<tr>
<td>Files</td>
<td>58</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>58</td>
</tr>
<tr>
<td>Multiline Graph</td>
<td>58</td>
</tr>
<tr>
<td>Filled Multiline Graph</td>
<td></td>
</tr>
<tr>
<td>Multiset Line Segment Graph</td>
<td></td>
</tr>
<tr>
<td>Multiset Point Graph</td>
<td></td>
</tr>
<tr>
<td>Animation Data</td>
<td>58</td>
</tr>
<tr>
<td>Files</td>
<td>59</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>59</td>
</tr>
<tr>
<td>XY Line Graph</td>
<td></td>
</tr>
<tr>
<td>XY Point Graph</td>
<td></td>
</tr>
<tr>
<td>Animation Data</td>
<td>60</td>
</tr>
<tr>
<td>Files</td>
<td>60</td>
</tr>
</tbody>
</table>
Resource Sets ........................................................................................................................... 74
Multiset 3D Stacked Bar Graph................................................................................................. 74
  Animation Data .................................................................................................................... 74
  Files .................................................................................................................................... 75
  Resource Sets ..................................................................................................................... 75
3D Step Graph .......................................................................................................................... 75
  Animation Data .................................................................................................................... 75
  Files .................................................................................................................................... 76
  Resource Sets ..................................................................................................................... 76
Multiset 3D Step Graph............................................................................................................. 76
  Animation Data .................................................................................................................... 76
  Files .................................................................................................................................... 77
  Resource Sets ..................................................................................................................... 77
Ribbon Graph ............................................................................................................................ 77
  Animation Data .................................................................................................................... 77
  Files .................................................................................................................................... 77
  Resource Sets ..................................................................................................................... 77
Multiset Ribbon Graph ............................................................................................................ 78
  Animation Data .................................................................................................................... 78
  Files .................................................................................................................................... 78
  Resource Sets ..................................................................................................................... 78
Framed Surface Graph
Unframed Surface Graph ........................................................................................................ 79
  Animation Data .................................................................................................................... 79
  Files .................................................................................................................................... 79
  Resource Sets ..................................................................................................................... 79
Gantt Chart ................................................................................................................................ 80
  Animation Data .................................................................................................................... 80
  Files .................................................................................................................................... 80
  Resource Sets ..................................................................................................................... 80
Horizontal Slider Graph ......................................................................................................... 80
  Animation Data .................................................................................................................... 81
  Files .................................................................................................................................... 81
  Resource Sets ..................................................................................................................... 81
Vertical Slider Graph .............................................................................................................. 81
<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animation Data</td>
<td>81</td>
</tr>
<tr>
<td>Files</td>
<td>82</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>82</td>
</tr>
<tr>
<td>Horizontal 3D Slider Graph</td>
<td>82</td>
</tr>
<tr>
<td>Animation Data</td>
<td>82</td>
</tr>
<tr>
<td>Files</td>
<td>83</td>
</tr>
<tr>
<td>Resources</td>
<td>83</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>83</td>
</tr>
<tr>
<td>Vertical 3D Slider Graph</td>
<td>83</td>
</tr>
<tr>
<td>Animation Data</td>
<td>83</td>
</tr>
<tr>
<td>Files</td>
<td>84</td>
</tr>
<tr>
<td>Resources</td>
<td>84</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>84</td>
</tr>
<tr>
<td>Legend</td>
<td>84</td>
</tr>
<tr>
<td>Files</td>
<td>84</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>84</td>
</tr>
<tr>
<td>Bevels</td>
<td>84</td>
</tr>
<tr>
<td>Files</td>
<td>85</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>85</td>
</tr>
<tr>
<td>Gauge Widgets</td>
<td>85</td>
</tr>
<tr>
<td>Files</td>
<td>85</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>85</td>
</tr>
<tr>
<td>Clock Widgets</td>
<td>86</td>
</tr>
<tr>
<td>Files</td>
<td>86</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>86</td>
</tr>
<tr>
<td>Stopwatch Widget</td>
<td>86</td>
</tr>
<tr>
<td>Files</td>
<td>86</td>
</tr>
<tr>
<td>Resources</td>
<td>86</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>87</td>
</tr>
<tr>
<td>Thermometer Widget</td>
<td>88</td>
</tr>
<tr>
<td>Files</td>
<td>88</td>
</tr>
<tr>
<td>Resources</td>
<td>88</td>
</tr>
<tr>
<td>Resource Sets</td>
<td>89</td>
</tr>
<tr>
<td>Joystick Widgets</td>
<td>89</td>
</tr>
<tr>
<td>Files</td>
<td>89</td>
</tr>
</tbody>
</table>
Chapter 3   GLG Widget Resource Sets.................................99

Graph Widget Resource Sets ......................................................99
   Title .................................................................................99
   Data Area ........................................................................99
   Polar Data Area .................................................................99
   3D Data Area ..................................................................100
   Datagroup ........................................................................100
   Multiset Datagroup .............................................................100
   Packed Datagroup ...............................................................101
   Line Datagroup ................................................................101
   Multiline Datagroup ............................................................102
   XY Line Datagroup ...............................................................102
   Multiline XY Datagroup .......................................................103
   Filled Step Datagroup ........................................................103
   Multiline Step Datagroup .....................................................104
   Filled Multiline Step Datagroup ..........................................104
   Pie Datagroup ...................................................................105
   Polar XY Line Datagroup .....................................................105
   Multiline Polar XY Datagroup ............................................105
   3D Datagroup ...................................................................106
   Multiset 3D Datagroup .......................................................106
   3D Packed Datagroup ........................................................107
   Multiset 3D Packed Datagroup ...........................................107
   Surface Datagroup ..............................................................108
   Level ................................................................................109
   Symmetrical Level ..............................................................109
   Polar Level .........................................................................109
   X Level .............................................................................110
Chapter 1

Using the GLG Widgets

The GLG Toolkit includes an extensive library of pre-built widgets that may be incorporated into a wide variety of applications. Each widget is a GLG drawing (".g" file) containing a collection of GLG graphical objects that implement the widget’s graphical appearance and run-time behavior.

The widgets have no source code associated with them, and their functionality is completely defined in each widget’s drawing. Widgets’ drawings are cross-platform and may be used by GLG applications on various platforms and programming environments (C/C++, Java, C#/.NET, etc.) with no porting required.

Customizing GLG Widgets

The widgets are highly customizable. A widget drawing may be loaded into the GLG Graphics Builder and edited as any other GLG drawing. The Resource Browser of the Graphics Builder may be used to navigate and modify the widget resources using the Object Resources toolbar button. Unnecessary elements may be deleted and custom annotations and labels may be added to the widget. Custom dashboards may be created by combining several widgets into a single drawing, and new custom widget components may be created.

This chapter contains a general description of the categories of the GLG widgets: graphs, meters and dials, input widgets and process control objects, as well as the resource conventions in effect for each of these categories.

Using Widgets in the Graphics Builder

The graphs, dials and other widgets may be loaded into the Graphics Builder by loading corresponding "g" files, located in the widgets directory. However, the widget palettes are the most convenient way of adding the widgets to a drawing using the drag and drop functionality. To pop up a widget palette, click on the Palettes button of the main menu, then select the desired palette type.

Loading and Adding Widgets to the Drawing

There are two ways to load a widget drawing into the Graphics Builder:

- To add a widget to an existing drawing, simply click on the widget’s icon in the palette. The widget will be added at the current location of the editing focus. When creating a panel containing several widgets in a "$Widget" viewport, use the Hierarchy Down button to go into the viewport and then add widgets to it; otherwise, the widgets will be added at the top level outside of the viewport. When creating widget panels, individual widgets should be assigned unique names via the Properties dialog.

- To load the whole widget drawing into the Graphics Builder, Ctrl-click (click with the Control key held down) on the widget icon in the palette. This will discard the current drawing and replace it with the drawing of the selected widget. The widget drawing contains the widget, its icon and an animation command for prototyping the widget in the Run mode. The animation command is visible in the drawing as a text object.
Prototyping Widgets Using Predefined Animation Commands

All widgets may be prototyped using the Graphics Builder’s Run mode. To start the Run mode, select **Run, Start** from the main menu or click on the **Start** toolbar button. This will activate a dialog for entering an animation command.

To simplify the process, all graph and control widget drawings include proper animation commands. To animate a widget with the animation command included in the widget drawing, **Ctrl+click** on the widget icon in the palette to load the widget’s drawing, then start the Run mode and press the **OK** button in the animation command dialog to start the animation.

The Run mode toolbar provides controls for changing animation speed and pausing the animation. It also displays update performance data, such as updates per seconds and seconds per update.

For control widgets that can be used for both input and output, such as dials, sliders, toggles and other interface widgets, the Run mode may also be used to test the widget’s interactive behavior. Press the **Pause** button in the Run mode toolbar to stop animating the widget with output data and use the mouse to interact with the widget. For example, you can move a dial or slider with the mouse or change the state of a toggle widget.

Prototyping Dials and Meters With Custom Animation Commands

Custom animation commands may be used to animate any dial or meter widget in the drawing. For example, the following command will provide a smooth animation of a dial by using a sinusoidal wave in the 0-100 range as a datasource for the widget’s `Value` resource:

```plaintext
$datagen -sin d 0 100 $Widget/Value
```

`$Widget/Value` is a resource path for the resource that will be animated.

To enter the animation command, start the Run mode by selecting **Run, Start** from the main menu or clicking on the **Start** toolbar button. This will activate a dialog for entering an animation command. Enter the command and press **OK** to see animation.

If a dial is renamed or placed inside a panel containing several dials, the path may be different from the one used above. For example, the following path will animate a `Value` resource of a dial named **Dial2** placed inside a top level `$Widget` viewport:

```plaintext
$Widget/Dial2/Value
```

If the resource path is not correct, an error message will be generated. Use the Graphics Builder’s Resource Browser (the **All Resources** toolbar button) to browse the drawing’s resources and check the resource path.

To animate more than one dial, use several animation commands in one string. For example, the following command will animate dials in the drawing named **Dial1** and **Dial2**:

```plaintext
$datagen -sin d 0 100 $Widget/Value
```
$\text{datagen} -\text{sin d 0 100 Dial1/Value}
-\text{sin d 0 50 Dial2/Value}

Refer to the The datagen utility is embedded into the GLG Builder, which invokes the utility to generate data for prototyping the drawing in the Run mode using the $\text{datagen}$ command in the Run dialog. $\text{datagen}$ instructs the Builder to use an internal version of the datagen utility. chapter on page 376 of the GLG Programming Reference Manual for a complete list of the animation command options.

Prototyping Real-Time Charts with Custom Animation Commands

Real-time charts are animated by supplying data sample values and time stamps into the chart’s $\text{ValueEntryPoint}$ and $\text{TimeEntryPoint}$. If a time stamp is not supplied, the chart will automatically use current time to generate a time stamp. The following prototyping command may be used for animating a real-time chart with a single plot:

$\text{datagen} -\text{sin d 1 9 Widget/Chart/Plots/Plot#0/ValueEntryPoint}$

A default animation command included in each chart widget’s drawing provides a good starting point for experimenting with the chart’s options. A modified animation command may be stored in the drawing using the Run, Store Run Command option from the main menu.

Prototyping Graph Widgets with Custom Animation Commands

Prototyping Graph Widgets in the Graphics Builder requires more elaborate prototyping commands that animate both the graph’s labels and datasamples. If you want to use a custom animation command for a graph, use a default graph animation command defined in the graph’s drawing as a starting point and customize it to explore various graph animation options. The modified animation command may be stored in the drawing using the Run, Store Run Command option from the main menu. The customized drawing may be saved for later reuse or added to the Builder’s palettes.

The Store Run Command option stores the current animation command by setting the $\text{DatagenString}$ resource of the drawing, if it is present. In the graph widgets’ drawings, a text object at the top of the drawing displays the Run command. The TextString attribute of the text object is named $\text{DatagenString}$. If a graph widget’s drawing was loaded using Ctrl-click on the graph’s icon in a palette, the $\text{DatagenString}$ resource is already present in the drawing; if a graph widget was added using a mouse click, the $\text{DatagenString}$ resource may be added to the drawing by placing a text object in the drawing and naming its TextString attribute $\text{DatagenString}$.

Real-Time Charts and Graph Widgets

Graph and Chart Widgets are used to display a collection of dynamically changing numerical data in a graphical form, such as a bar graph, a line graph, a surface graph and others graphs. The graph widgets are animated in the same way as any other GLG drawings, by setting their resources. Each graph or chart has two types of resources: resources that control its appearance, such as a number of labels or datasamples, and resources for entry points that are used to supply dynamic data for the graph’s values and labels.
There are two types of chart and graph widgets:

- **Real-Time Chart widgets** are optimized for displaying a large number of lines with a huge number of data points. They also provide integrated zooming and scrolling functionality, tooltips and cursor feedback, as well as multiple Y axes, flexible tick labeling and many other features. Both time-based scrolling charts and XY Scatter widgets are provided.

- **2D and 3D graph widgets** provide a variety of graph types with presentation features, such as gradient shading or 3D presentation of multiple sets of data.

**Real-Time Charts**

All real-time chart widgets use the GLG Chart object which implements all real-time and interaction features of each chart. The chart object contains integrated axes, plots, legend and other objects and has a number of attributes that control its behavior.

Refer to the Chart Objects chapter of the GLG User’s Guide and Builder Reference Manual on page 137 for a detailed description of the chart’s attributes and features. Refer to the Using GLG Real-Time Charts chapter on page 77 of the GLG Builder and Animation Tutorial for information on editing real-time chart widgets.

In addition to chart properties, a `Chart` object of every real-time chart widget provides `OffsetTop`, `OffsetBottom`, `OffsetLeft` and `OffsetRight` resources that control layout of the chart inside the widget.

An application supplies data for a chart using the chart’s `ValueEntryPoint`, `TimeEntryPoint` and `Valid Entry Point` described in the the Chart Objects chapter. An application can either provide a time stamp for every data sample, or let the chart use current time as a time stamp. The chart automatically positions and scrolls its data samples according to their time stamps. The chart also supports the use of invalid data samples, which are displayed as gaps in the plot lines.

A real-time chart maintains a data buffer of a variable size that keeps data samples accumulated in the chart. Users can zoom and scroll the chart in both vertical and horizontal direction using the chart’s integrated zooming and scrolling features. The chart can also be scrolled by simply dragging it with the mouse.

The integrated zooming and scrolling behavior of the chart is controlled by the `ZoomMode` and the `Pan` attribute of the chart’s parent viewport. Refer to the Integrated Zooming and Panning chapter of the GLG User’s Guide and Builder Reference Manual on page 57 for more details.

The chart’s integrated tooltips and cursor feedback features may be used to display a cross-hair cursor that follows the mouse, as well as display a tooltip showing the information about the data sample located under the current mouse position. The chart object also provides an API for querying chart selection from a program.

To activate the chart’s tooltip, the `ProcessMouse` attribute of the chart’s parent viewport must be set to TOOLTIP. This is the default for chart widgets in the Real-Time Charts palette.
2D and 3D Graphs

2D and 3D graphs provide a collection of graph widgets with a variety of graph types, such as bar, pie, polar and other graphs, and with support for gradient and 3D shading.

While the Real-Time Charts are implemented using a single Chart object, the 2D and 3D graphs (including their axes and legends) are constructed using a collection of individual graphical objects, such as polygons, text, series and other objects. This allows for a greater flexibility in modifying their appearance by editing their drawings.

The following chapters describe components and resource hierarchies of the 2D and 3D graph widgets.

OpenGL Note

The 3D version of graph widgets is optimized for the OpenGL driver by default. It uses alpha-blending and transparency by setting the Visibility attribute of the graph’s DataGroup to a fractional value. Set DataGroup/Visibility to 1 when a 3D graph widget is used with a non-OpenGL driver.

Components of a GLG Graphs

The graphs of the GLG Widget Library are all constructed using similar elements to represent graph components such as axes, ticks, labels and data. A typical graph has a data area, axes, grids, titles, minor and major ticks, major tick labels for each axis, a data group containing graph data samples, a group of level lines and a status object, as shown in the picture below.
For convenience, these elements have identical names in all graphs, although some names may have additional suffixes in certain cases. This means that editing any particular element is similar for all graphs.

The widgets in the GLG Widget Library are intended for use as templates. They may be used as they are or edited with the GLG Graphics Builder to produce customized versions.

**Naming Conventions**

The names of resources play an important role, since they are used to access objects inside a widget. There are a few naming conventions used for the Graph Widget components:

- The names of the elements associated with an axis start with the name of the axis (X or Y), such as XLabel or YMajorTick.

- The *series object* is used to create a variable number of objects in a graph, such as a variable number of datasamples, labels and ticks. The name of the series object ends with the *Group* suffix, such as DataGroup, XLabelGroup or YMajorGroup for major ticks on the Y axis.

- Each series replicates an object defined as its template. The value of the series’ *Factor* attribute defines the number of the created template copies. For example, DataGroup/Factor defines the number of datasamples in a graph, and XLabelGroup/Factor defines the number of major tick labels. The label and tick series also have the *MinorFactor* attribute which controls the number of minor ticks, such as XLabelGroup/MajorFactor.

- The name of a template of a series object does not have any numerical suffix, such as DataSample. The names of the created instances of the template in a series object have a numerical suffix corresponding to the sequential number of the instance, such as DataSample0, DataSample1, and so on. All indexes are zero based, and DataSample1 is actually the second object in a series.

- Nested series may be used for creating several groups of groups with a variable number of elements. For example, a multi-line graph uses the first series to create a line with a variable number of points. This series is then used as a template to the second series to create a variable number of lines.

In the case of nested series objects, the name of top level series object is formed by adding a numerical suffix to the name of the bottom level composite object. This suffix indicates the level of the hierarchy and is spelled out as a word. For example, if there are two levels of series objects and the name of the inner series is DataGroup, the name of the outer series is DataGroupOne. The final hierarchy and the corresponding names look like this:

```
DataGroupOne     (series of lines)
DataGroup         (line template)
    DataSample  (datasample template)
DataGroup0        (first line instance)
    DataSample  (datasample template)
    DataSample0 (datasample instance)
    DataSample1 (datasample instance)
    ...
DataGroup1        (second line instance)
    DataSample  (datasample template)
    DataSample0 (datasample instance)
```
DataSample1 (datasample instance)

... ...

Performance Optimizations

Gradients

By default, the 2D version of graph widgets uses gradients for the graph’s DataArea, line markers, bars of the bar graphs and other graph elements. When a large number of datasamples needs to be displayed with a fast update rate, the gradients may be disabled by editing object’s rendering attributes to minimize CPU utilization and increase graph’s update rate.

RollBack Scrolling

The RollBack attribute of the graph’s DataGroup may be used in conjunction with the WRAPPED scrolling type for implementing scrolling behavior which scrolls the graph only once every n iterations, as defined by the value of the RollBack attribute.

For example, consider a graph with the WRAPPED scroll type, 200 datasamples, 10 X axis major ticks with labels, and 20 minor ticks per one major tick interval. Setting DataGroup/Rollback=40 and XLabelGroup/RollBack=2 will “roll” the graph back by 2 major tick and label intervals (which corresponds to 40 datasamples) when the graph gets completely filled with data.

The use of the RollBack limits the CPU-intensive scrolling operation to be performed only once on every 40th data update, compared with every data update in the regular scrolling graph with the SCROLLED scrolling type.

A special case of the rollback may be used to implement the graph which switches from the WRAPPED behavior to the SCROLLED behavior when the graph gets completely filled with data the first time. For example, for a graph with the WRAPPED scroll type, 200 datasamples, 10 X axis major ticks and labels, and 20 minor ticks per one major tick interval, the following settings may be used: DataGroup/Rollback=1 and XLabelGroup/Rollback=0.05 (which corresponds to one minor tick - 1/20).

Auxiliary Elements

The update speed of a graph also depends on the number and type of objects rendered on every update iteration. Deleting auxiliary elements of a graph such as grids or minor ticks, as well as filled data area can substantially increase the graph’s maximum update speed.

Batch Update

For graphs with a large number of data samples and fast update rates, rendering performance may be optimized by reducing the number of times the graph is redrawn. For example, if a graph receives a new data value every 10 milliseconds and redraws after each new data value, it will be redrawn 100 times per second. Instead of refreshing the graph each time a new data point is received, the application may update the graph just several times per second. Each update would push all data points accumulated since the last update and redraw the graph just once for all accumulated new data values.
Commonly Used Graph Resources

Before proceeding with the detailed description of all graph resources, let’s consider the typical resources used to animate a bar graph:

- **DataGroup/Factor**: Defines the number of datasamples in the graph.
- **DataGroup/DataSample/Low**: Defines the low range of data.
- **DataGroup/DataSample/High**: Defines the high range of data.
- **DataGroup/DataSample/Value**: Defines an initial value for graph’s samples.
- **DataGroup/ScrollType**: Defines the graph’s scroll type (WRAPPED or SCROLLED).
- **DataGroup/EntryPoint**: An entry point for pushing data values into the graph.
- **XLabelGroup/Factor**: Defines the number of X labels, major ticks and grids.
- **XLabelGroup/MinorFactor**: Defines the number of minor ticks per each major tick interval.
- **XLabelGroup/XLabel/String**: Defines an initial value for the label, usually set to an empty string.
- **XLabelGroup/EntryPoint**: An entry point for pushing X axis labels into the graph.
- **YLabelGroup/Factor**: Defines the number of Y labels, major ticks and grids.
- **YLabelGroup/MinorFactor**: Defines the number of minor ticks per each major tick interval.
- **YLabelGroup/Low**: Defines the low range of the Y axis labels, same as **DataGroup/DataSample/Low**.
- **YLabelGroup/High**: Defines the low range of the Y axis labels, same as **DataGroup/DataSample/High**.

A line graph has a slightly different resource hierarchy for its **DataGroup**:

- **DataGroup/Factor**: Defines the number of datasamples in the graph.
- **DataGroup/Marker/DataSample/Low**: Defines the low range of data.
- **DataGroup/Marker/DataSample/High**: Defines the high range of data.
- **DataGroup/Marker/DataSample/Value**: Defines an initial value for graph’s samples.
- **DataGroup/ScrollType**: Defines the graph’s scroll type.
- **DataGroup/EntryPoint**: An entry point for pushing data values into the graph.

A multi-line graph has a different **DataGroup** hierarchy as well:

- **DataGroupOne/Factor**: Defines the number of lines in the graph.
- **DataGroupOne/Persistent**: May be set to PERSISTENT to preserve attribute settings of individual lines.
- **DataGroupOne/DataGroup/Factor**: Defines a number of point in each line.
- **DataGroupOne/DataGroup/Marker/DataSample/Low**: Defines the low range.
- **DataGroupOne/DataGroup/Marker/DataSample/High**: Defines the high range.
- **DataGroupOne/DataGroup/Marker/DataSample/Value**: Defines an initial value.
- **DataGroupOne/DataGroup/ScrollType**: Defines the graph’s scroll type.
- **DataGroupOne/EntryPoint**: An entry point for pushing data values into all lines.
- **DataGroupOne/DataGroupN/EntryPoint**: An entry point for pushing values into the Nth line.
Chapter 1 Using the GLG Widgets

The top level `DataGroupOne/EntryPoint` resource may be used to push values into all lines of the graph. The entry points of each line (the `DataGroupOne/DataGroupN/EntryPoint` resources, where N is the index of a line) may be used to push datasamples into each line separately.

**Data Samples and the Data Group**

Data samples are objects used by a graph to display its data. The type of object used as a data sample may be different depending on the type of the graph. For example, a data sample in a bar graph is represented by a polygon whose height changes with the value of the data value displayed. For a line graph, the data value is represented by the position (usually just the Y-coordinate) of a point on the line.

The method used to create multiple instances of a data sample object depends on the type of the graph. For example, a bar graph uses a series object to create a variable number of bar datasamples, and a line graph uses a polyline object to create a line with a variable number of points. In both cases, the Factor resource controls the number of bars or line points. Regardless of the type of an object used to replicate graph’s datasamples, such a collection of data samples is called a data group, and these are identified with the `DataGroup` resource name.

If a series object is used to produce multiple copies of data samples, a template of the series object is usually named `DataSample`. The names of the created copies are formed by adding a consecutive digital index of the copy to the name of the template: `DataSample0`, `DataSample1`, and so on.

Each `DataGroup` has an `EntryPoint` resource which is used to push new data into the graph, while the graph automatically scrolls the data, via a history object attached to the `DataGroup`. The `ScrollType` resource allows changing the graph’s scroll type, while the `Inversed` resource controls the update direction.

**Resources of a DataSample**

The type of object used as a data sample depends on the type of the graph. However, there are three resources present in the `DataSample` resource no matter the type of object. These are:

**Value**

The value to be represented by the data sample object. The `Value` resource of the template `DataSample` defines the initial value of all datasamples, while the `Value` of other datasamples defines the value of the datasample displayed in the graph.

The graph always works in a linear coordinate system. If data has to be displayed in the logarithmic format, it must be converted to the logarithmic scale before setting the `Value` resource.

Changes made to the `Value` resource are not permanent, and will revert to an initial value when the graph is reset. To make `Value` resource changes permanent, explode the `DataGroup` object with the GLG Graphics Builder. This may be used to save the data in the graph, and not as general technique, as the graph looses the ability to change the number of its datasamples after its datagroup has been exploded.

**Low and High**

These resources define the range of data displayed by the graph. Data sample shapes that extend beyond these limits may be clipped. The default values are 0 and 1, respectively. These resources may be changed dynamically without losing data currently displayed in a graph. If the `Low` or `High` resources are changed, the graph automatically adjusts all currently displayed
data to the new range. This allows the application program to change the range of the data on
the fly when the amplitude of the data increases or decreases beyond the current graph’s data
range. It also allows the program to scale the data samples when the values get too small.

**Truncate**

Controls how the data samples with values outside of the graph’s range are plotted. If set to
YES, the values will be truncated to fit inside the graph’s range. For vertical graphs, the data
samples whose values are truncated will appear at the top or the bottom of *DataArea*.

The data sample objects displayed in a graph are dynamically created objects whose attributes are
inherited from the template *DataSample*. The attributes of the *DataSample* template may be
changed to define the appearance of all datasamples in a graph.

The program can change attributes of any datasample at any time as well, but as with any series object, changes to the
template instances are not permanent, and last only until the drawing is reset or reloaded, at which time the instances of
the datasamples are recreated by copying the template datasample. This behavior may be prototyped in the GLG Graphics
Builder by changing resources of datasample instances: the changes will be discarded after resetting the graph by pressing
the *Reset* toolbar button.

**Local and Global Attributes**

Any attribute of a datasample or any other template object may be made local or global by setting
the value of its *Global* flag. If the attribute is global, it is constrained for all datasamples in the
series. Changing the value of a global attribute will affect all datasamples as well as the series’
template. To change an attribute of individual datasamples independently, set the *Global* flag for
that attribute to *LOCAL*.

If a template attribute is *GLOBAL*, changing the attribute immediately affects all instances of the template in the series.
If a template attribute is *LOCAL*, the drawing needs to be reset by pressing the *Reset* toolbar button after changing the
attribute in the GLG Graphics Builder to see the new values. In a program, the values of local attributes must be set before
the hierarchy setup, since the hierarchy setup creates instances of the template and copies the settings of the template’s
attributes.

**Attribute Transformations**

In some graphs, various attributes of the data samples may have a transformation attached to them.
For example, in a multi-line graph, the line’s *EdgeColor* attribute has a *List* transformation to
change the color depending on the line index, so that each line is rendered in a different color. A
similar transformation is used to vary colors of the individual bars in packed and stacked bar graphs.

In this case, the value of the attribute can not be edited directly, as it is affected by the transformation
attached to it. Instead, the Color0, Color1, Color2, ... resources will be provided to define a color of
each line or bar.

To add more colors to the list of colors, select the line template of the polyline, select its *EdgeColor* attribute and edit a
transformation attached to the attribute.

**Nested Data Groups**

Some types of graphs contain data groups whose elements are themselves composite objects. For
example, a line graph with multiple lines or a packed bar graph that contain several “packs” of data,
each of which contain several data samples.
In a multi-line graph, the series object used to create multiple lines is named `DataGroupOne`. It contains a template object named `DataGroup` which is used as a template for each line. The created line instances are named `DataGroup0`, `DataGroup1`, and so on. Each line instance contains datasamples named `DataSample0`, `DataSample1`, etc., located inside the line’s `Points` group: `Points/DataSample0`, `Points/DataSample1`, and so on. Each line also contains `DataSample` template located inside its `Marker` object: `Marker/DataSample`.

In graphs such as a packed or stacked bar graph, the top level series object named `DataGroup` is used to replicate a desired number of “packs” or “stacks” of data, defined in its template named `Pack`. The created instances of packs will have names `Pack0`, `Pack1`, and so on. Each pack is a series object too, used to replicate its `DataSample` template to create several datasamples in each pack, named `DataSample0`, `DataSample1`, and so on.

For multi-line and other multi-set graphs, the `DataGroupOne/Persistency` attribute may be set to PERSISTENT to preserve attribute settings of individual lines. By default, the `Persistent` attribute is set to VOLATILE, and only the color setting is saved, while the rest of the line attributes have to be set at runtime. When the PERSISTENT setting is used, the attributes of individual lines may be set in the Builder.

**DataGroup Resource Hierarchy**

A `DataGroup` of a bar graph and other graphs that use the series object to replicate datasamples have the following resource hierarchy:

```
DataGroup   (series object)
  EntryPoint (entry point for pushing data values into the graph)
  Factor     (number of datasamples)
  DataSample (datasample template)
    Low      (low range of the graph)
    High     (high range of the graph)
    Truncate (controls how out of range values are plotted)
    Value    (initial value)
  DataSample0 (datasample instance)
  DataSample1 (datasample instance)
  ...       (more datasample instances)
```
A packed or stacked bar graph will have a slightly different resource hierarchy:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataGroup</td>
<td>(series object)</td>
</tr>
<tr>
<td>EntryPoint</td>
<td>(entry point for pushing values into the graph)</td>
</tr>
<tr>
<td>Factor</td>
<td>(number of datasample packs in the graph)</td>
</tr>
<tr>
<td>Pack</td>
<td>(template of a pack containing several datasamples)</td>
</tr>
<tr>
<td>Factor</td>
<td>(number of datasamples in a pack)</td>
</tr>
<tr>
<td>Persistent</td>
<td>(May be set to preserve attribute settings of individual samples in a pack)</td>
</tr>
<tr>
<td>DataSample</td>
<td>(datasample template)</td>
</tr>
<tr>
<td>Low</td>
<td>(low range of the graph)</td>
</tr>
<tr>
<td>High</td>
<td>(high range of the graph)</td>
</tr>
<tr>
<td>Truncate</td>
<td>(controls how out of range values are plotted)</td>
</tr>
<tr>
<td>Value</td>
<td>(initial value)</td>
</tr>
<tr>
<td>Pack0</td>
<td>(pack instance)</td>
</tr>
<tr>
<td>Pack1</td>
<td>(pack instance)</td>
</tr>
<tr>
<td>...</td>
<td>(more pack instances)</td>
</tr>
</tbody>
</table>

A typical line graph and other graphs that use the polyline object with a variable number of points have the following resource hierarchy:

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DataGroup</td>
<td>(polyline object)</td>
</tr>
<tr>
<td>EntryPoint</td>
<td>(entry point for pushing values into the graph)</td>
</tr>
<tr>
<td>Factor</td>
<td>(number of datasamples)</td>
</tr>
<tr>
<td>Marker</td>
<td>(template used for a point of the graph)</td>
</tr>
<tr>
<td>DataSample</td>
<td>(datasample)</td>
</tr>
<tr>
<td>Low</td>
<td>(low range of the graph)</td>
</tr>
<tr>
<td>High</td>
<td>(high range of the graph)</td>
</tr>
<tr>
<td>Truncate</td>
<td>(controls how out of range values are plotted)</td>
</tr>
<tr>
<td>Value</td>
<td>(initial value)</td>
</tr>
<tr>
<td>Points</td>
<td>(group containing all line’s datasamples)</td>
</tr>
<tr>
<td>DataSample0</td>
<td>(datasample instance)</td>
</tr>
<tr>
<td>DataSample1</td>
<td>(datasample instance)</td>
</tr>
<tr>
<td>...</td>
<td>(more datasample instances)</td>
</tr>
</tbody>
</table>
Finally, multi-line graphs that display several lines have the following resources:

- **DataGroupOne**: (series object used to replicate line instances)
- **EntryPoint**: (entry point for pushing values into all lines)
- **Factor**: (number of lines)
- **Persistent**: (May be set to preserve attribute settings of individual lines)
- **DataGroup**: (polyline object used as a line template)
- **Factor**: (initial number of points in each line)
- **Marker**: (template used for a point of the graph)
  - **DataSample**: (datasample)
    - **Low**: (low range of the graph)
    - **High**: (high range of the graph)
    - **Truncate**: (controls how out of range values are plotted)
    - **Value**: (initial value)
- **DataGroup0**: (line instance)
- **EntryPoint**: (entry point for pushing values into this line)
- **Factor**: (number of datasamples in this line)
- **Marker**: (template used for a point of this line)
- **DataSample**: (datasample)
  - **Low**: (low range of this line)
  - **High**: (high range of this line)
  - **Value**: (initial value)
- **DataGroup1**: (line instance)
- ... (line resources, same as in DataGroup0)
- ... (more lines instances)

**Data Area**

The Data Area is a filled polygon used as a background for 2D graphs. It allows the graph to paint the area behind data samples in a color different from the background color. In 3D versions of the graphs, there are several data areas, representing the three planes of the graph’s frame.

By selecting the Data Area polygon and editing it, you may change its fill color, border color and border width. You may also change the geometry of the whole graph by moving control points of the Data Area. The Data Area is rendered as a parallelogram object, and has three control points.

Like any parallelogram object, the three unconstrained control points of the Data Area may be named and then accessed as resources. This gives you the ability to change the geometry of the graph dynamically from a program. Changing the geometry of the Data Area adjusts the geometry of all datasamples displayed in the graph.

**Axes**

An axis of a graph is just a polygon. Axes are named by the corresponding coordinate: XAxis, YAxis and ZAxis (for 3D graphs).
**Major And Minor Ticks**

A major tick is represented by a polygon object named $X_{MajorTick}$ for the X axis, $Y_{MajorTick}$ for the Y axis, and $Z_{MajorTick}$ for the Z axis of 3D graphs. Everything described below in this chapter for the X axis ticks is also applicable to the ticks on the Y and Z axes.

The $X_{MajorTick}$ polygon is used as a template for the $X_{MajorGroup}$ series object to produce multiple copies of a major tick. The number of ticks is defined by the factor of the series object, and their appearance is defined by the series object template.

A minor tick is represented by a polygon named $X_{MinorTick}$ for the X axis. The graphs use two nested series objects to produce the minor ticks. The inner series object, named $X_{MinorGroup}$, uses $X_{MinorGroup}$ polygon as a template to produce the necessary number of minor ticks for one major tick interval. The series object on the top level, named $X_{MinorGroupOne}$, uses the $X_{MinorGroup}$ series as a template reproduce it the number of times equal to the number of major tick intervals.

The templates of both the major and minor ticks have the $EndPoint$ resource which controls the length of the tick. This resource may be used to adjust the tick length when the graph is stretched and ticks on one of the axes become too long or too short.

**Setting the Number of Major and Minor Ticks**

For convenience, the $Factor$ and $MinorFactor$ resource of both the major and minor ticks and the grids are constrained to the corresponding factors of the labels and may be edited in one place - the $X_{LabelGroup}$ series. Its $Factor$ resource controls the number of labels, major ticks and grids, and the $MinorFactor$ resource controls the number of minor ticks per each major tick. The total number of minor ticks, which is the product of the number of major ticks times the number of minor ticks, must match the number of the graph’s datasamples, or the ticks will be misaligned:

$$X_{LabelGroup}/Factor \times X_{LabelGroup}/MinorFactor = DataGroup/Factor$$

**Performance Optimization**

In graphs with a large number of datasamples, the minor ticks series may be deleted to simplify the graph and increase its update performance.

**Logarithmic Ticks**

Ticks can be positioned on a logarithmic scale if the graph displays logarithmic data. In this case, only minor ticks are drawn with uneven intervals. Major ticks are still drawn evenly, since they correspond to the integer powers of the logarithm base. It is the responsibility of the program or the process supplying data to provide logarithmic data sample values.

To make minor ticks logarithmic, set the $LogType$ attribute of the lower level series object of Y minor ticks ($Y_{MinorGroup}$) to $LOG$. The factor of this series should be set to a value of the base of the logarithm minus 1. For example, for the logarithm base ten, the factor should be set to nine.
Chapter 1 Using the GLG Widgets

Grid Lines

The background grid lines in a GLG Graph are rendered as polygons named XGrid and YGrid. The polygons are used as templates in the series objects named XGridGroup and YGridGroup, which produce the necessary number of grid lines. The factors of the grid’s series are constrained to have the same values as the number of major ticks on the corresponding axis.

Labels

Axis labels to be displayed at each major tick are represented by text objects named XLabel, YLabel and, for 3D graphs, ZLabel. These text objects are used as templates for the series objects named XLabelGroup, YLabelGroup and ZLabelGroup, which produce the desired number of labels for the corresponding axis. Each label is positioned at the corresponding major tick.

The values of the Factor attribute of each label series defines the number of labels, major ticks and grids. The MinorFactor resource defines the number of minor ticks per one major tick interval. For the time axis with scrolling labels (the X axis for horizontal graphs), the total number of minor ticks, which is the product of the number of major ticks times the number of minor ticks, must match the number of the graph’s datasamples, or the ticks and labels will be misaligned with the datasamples.

As with any other series, every individual label may be accessed with the index of the label as a suffix. For example, XLabelGroup/XLabel3 accesses the fourth label on the X axis (since the index is zero-based).

The XLabelGroup/XLabel and YLabelGroup/YLabel access the label templates which define the initial appearance of all labels for the X and Y axes. The attributes of each individual label of the label series may be changed dynamically at run time, but these changes are volatile and disappear after resetting or reloading the drawing.

The labels’ font can be changed by editing the FontType and FontSize attributes of the label’s template, i.e. XLabelGroup/XLabel/FontType.

By default, the labels use SCALED text and the FontSize resource controls the maximum size. The label will use a smaller font if the window is resized to a small size or if the label is too long. To use a label with a constant font size, change the label template to a FIXED font and adjust its control point (the XLabel/Point resource) to position it below the corresponding major tick.

If the type of the text object template for a label series is SCALED, the Global flag of the SizeConstraint attribute of the template label controls whether or not all axis labels are displayed using the same font size when labels have different length.

If the Global flag is set to LOCAL for the template label, label text strings are fitted into their boxes individually and may be displayed in different font sizes if labels have different length. If the flag is set to GLOBAL, fitting the text of one label into the corresponding box affects all other labels. It means that if the text of a label does not fit into its box, a font of the smaller size is chosen for that label. This change of a font size affects all labels on the axis, so that all of them are displayed in this smaller font size. After the conditions that caused the change of the labels’ font size are eliminated, the labels stay in their current small font size until the viewport is resized or reset.
Value Labels and the Format Resource

The labels on the value axis (the Y axis for horizontal graphs) are constrained to the values associated with the corresponding major ticks. The strings for the labels are supplied automatically depending on the graph’s data range and the factor of major ticks.

For example, if the data range of the graph is $[0,1]$ and the graph has six major ticks (the factor of major ticks is 5), then the strings displayed in the labels are “0.0”, “0.2”, “0.4”, “0.6”, “0.8” and “1.0”.

Every value label has resources named Low and High that define the range used for labeling. These resources are constrained to have the same values as the Low and High ranges of the graph’s DataSample. Changing these resources affects both the labeling range and the data range of the graph, rescaling currently displayed data and displaying new label values.

The String attribute of a value label has a format transformation attached to the attribute to produce the proper value. The Format attribute of the transformation is exported as the Format resource of the label and may be edited to change the label format.

The Format resource defines a C-style format used to display a label’s data value. The format string should have no more than one conversion specification requiring an argument. This conversion specification should take a double-precision value as an argument, therefore only f, g and e format specifications are allowed. The conversion specification is optional, and may be omitted, allowing you to use the Format resource to replace the label string.

For example, to specify two digits of precision for all labels in the Y label series, change the Format resource of the label series template ($YLabelGroup/YLabel/Format$) to “%.2f”. The Format may also be set to the following string “Value=%.2f” to display the “Value=” in front of each label value.

The “%” character has a special meaning and defines the start of a format specification. If you want to display a “%” character in the string, use “%%” in its place. Refer to any C compiler documentation for a complete explanation of string formatting capabilities.

NOTE: It is the responsibility of the programmer to supply a valid C format for the Format string. The GLG Toolkit does not check the format for validity before displaying a drawing. Any entry is treated as a valid C format, in exactly the same way as any C program does. Defining the wrong format may cause unpredictable results, including a program crash.

To disable automatic labelling and supply a string to be displayed in each label at run-time, set the value of the Global flag of the Format resource of the template label to LOCAL (and reset the drawing if it is done in the GLG Graphics Builder). Then set the Format resource of each label instance to a desired string. For example, setting a format string of a label instance to “Level 1” simply displays “Level 1” in the label.

Scrolling Time Axis Labels

The labels on the time axis (X axis horizontal graphs) annotate the datasamples of the graph and scroll with them. The XLabelGroup time label series is equipped with the EntryPoint resource for convenience in setting graph labels. There are two methods of supplying strings for updating time axis labels.
The first method is to set the *String* attribute of the label directly using the name of the label text object. For example, `XLabelGroup/XLabel3/String` would be the resource name used to access the string of the fourth label on the X axis. Although straightforward, this method requires the application program to keep track of the current update position and to update labels differently depending on the scrolling behavior of the graph.

The second method is to use the *EntryPoint* resource of the label group. This method allows the program to supply label strings without defining where to place them. This keeps the labels synchronous with the corresponding data samples.

The `XLabelGroup/EntryPoint` resource is used for updating X axis labels. Setting this resource to different values in a loop animates the X axis labels according to the graph scrolling behavior.Querying the value of the resource yields the last string used for updating.

To keep scrolling labels synchronized with the corresponding datasamples, the *Factor* and *MinorFactor* resources must be set to values that match the number of datasamples in the graph. The `XLabelGroup/Factor` resource controls the number of labels and major, while the `XLabelGroup/MinorFactor` resource controls the number of minor ticks per one major tick interval. The product of these two values yields the total number of minor ticks in the graph and should equal to the number of datasamples:

\[
DataGroup/Factor = XLabelGroup/Factor \times XLabelGroup/MinorFactor
\]

When this condition is met, the program can simply supply one time label per each datasample, and the graph will handle the scrolling, displaying only the label values that correspond to the major tick.

For example, let’s consider a graph with 200 datasamples, 10 major ticks and labels, and 20 minor ticks per each major tick interval. Since \(10 \times 20 = 200\), the program may supply one label and one datasample value per each update, and the graph will scroll the datasamples and labels automatically.

For a multi-line graph with the same resource settings and three lines, the program would supply three datasample values (one for each line) and one label per each update. The same would also be true for a packed or stacked bar graph with 3 datasamples in each pack.

For the graphs with a large number of datasamples, minor ticks may be deleted from the drawing to decrease the number of objects used by the graph. The *MinorFactor* of the label series should be still set to a proper value to maintain the equation \(Factor \times MinorFactor = Number\ of\ Data\ Samples\).

The *ScrollType* and *Inversed* resources of the `XLabelGroup` object is constrained to the corresponding resources of the graph’s `DataGroup` to ensure that the scrolling behavior of the labels is consistent with the scrolling behavior of the graph. The values of these resources may be set in either place.
Level Objects

In addition to grids, many of the GLG graphs have level objects. These are horizontal lines usually used to mark some special thresholds in the graph. While grid lines are always positioned automatically on the levels defined by the major ticks, level objects may be positioned arbitrarily to annotate application-specific thresholds.

A level object is represented by a polygon named LevelObject which is used as a template for the LevelObjectGroup series to produce several copies of the level object. A factor of the series defines the number of level objects created. Each level object has a Value resource, controlling its position, or level.

Attributes of polygons representing every level line may be changed to distinguish different levels, for example using different colors. As with all series, the Global flags of the template attributes that need to be changed must be set to LOCAL, otherwise all level objects are affected. The level’s series object may be exploded with the GLG Graphics Builder to make the changes permanent, otherwise the attributes have to be changed dynamically at run-time.

Level objects may be set invisible or deleted from the graph drawing if they are not used.

Status Object

The status object (the StatusObject resource) is used to indicate the current update position for the graphs with the WRAPPED scrolling type. It moves after every update to the next data sample, wrapping back to the starting position after reaching the last one.

A status object may be set invisible or deleted if not used. When a graph is just loaded or after a reset, the position of the status object coincides with the Y axis for horizontal graphs.

Titles

A few freely positioned titles are used to annotate a graph and its axes. The X and Y axis titles are named XAxisLabel and YAxisLabel, respectively. The graph’s title text object is named Title. A title’s position, as well as direction, text string, and other attributes may be changed by moving control points or editing attributes of the text object indicated by the resource name. An unlimited number of additional titles may be created in the GLG Graphics Builder by creating new text objects and positioning them.

Legends

Graphs with multiple datasets, such as multi-line graph or packed bar graph, contain legends that annotate the entities displayed by the datasets of the graph. Every legend item shows the name of the entity displayed by the corresponding data set and also the color or line attributes used to annotate it. The Special widget set also contains optional legends that may be added to any graph.
A legend consist of a series object named `LegendGroup` used to create multiple copies of a legend template. The legend template is made of a group named `Legend`. It contains a text object named `LegendLabel` and two polygons named `LegendBox` and `LegendLine`. Two marker objects named `LegendMarker1` and `LegendMarker2` are constrained to the ends of the `LegendLine` polygon.

The `LegendLabel` text object is used to name entities in the graph. The polygons and markers are used to display the color, line width or line type associated with a particular entity.

**Multiple Axes**

The `Special` widget set contains optional axis objects that may be added to any graph. Both the value and scrolling time axes are provided, including the vertical and horizontal versions. A rotary axis for dial widgets is also provided.

**3D viewing**

3D Graph Widgets have a few resources designed to control the user’s view of the graph in 3D space:

*ShearFactor, ShearX, ShearY*

Controls a shear transformation that may be applied to the whole drawing. The ShearFactor defines the degree of shear, and is meant to take values between 0 and 1 (although other values may be used for special effects). The ShearX and ShearY attributes determine the proportion of the shear in the X and Y direction. These two attributes should add to 1.

*XAngle, YAngle, ZAngle*

Control the rotation of the whole drawing around the specified axis.

To obtain the best effect, use either a rotate or shear transformation but not both, since they may interfere with each other. Most of the 3D graphs use the Shear viewing transformation by default, which provides the best 3D effect.

**Input Widgets, Dials and Meters**

The GLG Widget Library provides a wide selection of dial, meters, sliders, buttons, toggles, and other input widgets. Most of these widgets have an input handler attached to the widget’s viewport and may be used for both input and output. To disable the widget’s interactive capabilities and use it only for the output, delete or disable the widget viewport’s input handler.

This chapter describes common resources of various input widget types. Refer to the `Input Objects` chapter of the *GLG User’s Guide and Builder Reference Manual* for detailed information on resources of various input handlers.

Note that the input handlers attached to an input widget expects the widgets to have several predetermined resources depending on the type of the input handler. Altering the names of these resources may render the widget inoperable, and may generate an error message when the drawing is reset.
At run-time, the application’s Input callback receives a message when a widget receives some input. Refer to the *Input Objects* chapter on page 229 of the *GLG User’s Guide and Builder Reference Manual* for information on the message parameters of various input handlers.

Refer to the *Appendix B: Message Object Resources* section on page 427 of the *GLG User’s Guide and Builder Reference Manual* for information on the input actions for various types of input handlers.

**Meters, Dials and Avionics gauges**

Most of the meters, dials and avionics gauges use the *rotary axis* and the Knob resource set described below. Some of the avionics gauges use two sets of needles and/or two sets of rotary axes. Most of the meters and dials are available in two versions: with a viewport and viewport-less.

A viewport-based version of a dial displays each dial in a separate window by using a viewport object as a container for each widget. The *GlgKnob* input handler is attached to the widget’s viewport, enabling the widget to handle both the input and output. If the widget’s input handler is not disabled, the dial’s needle can be moved with the mouse.

A viewport-less version does not create a window and may be used as a non-rectangular dial on top of a custom background image. Most of the avionics dials provide only viewport-less version, but may easily be placed in a viewport if necessary.

**Rotary Axis**

The rotary axis is a part of a variety of GLG dials, meters, knobs and rotary switches, as well as most of the avionics gauges. The rotary axis includes two components: the rotary axis (named *Axis* in the drawing) with labels, major and minor ticks, and the alarm sector (named *AlarmSector*), consisting of colored arcs which indicate the normal and abnormal range of values (similar to the the alarm sectors commonly used in tachometers).

**Axis Resources**

Rotary axis contains the following resources that control its appearance:

- **AngleStart** (DDATA) The start angle of the rotary axis.
- **Angle** (DDATA) The angle between the start and end of the axis.
- **Center** (GDATA) The axis’ center position (may also be changed by moving the arc with the mouse).
- **High** (DDATA) The high range of the axis’ labels.
- **Low** (DDATA) The low range of the axis’ labels.
- **InnerBorder** (ARC) The inner arc outline (drawn if *Visibility*=ON).
- **OuterBorder** (ARC) The outer arc outline (drawn if *Visibility*=ON).
- **OuterCircle** (ARC) A full circle drawn around the axis (drawn if *Visibility*=ON).
- **InnerRadius** (DDATA) The inner radius of the axis’ ticks.
- **OuterRadius** (DDATA) The outer radius of the axis’ ticks.
- **LabelRadius** (DDATA) Controls the position of the axis’ labels.
NumMajorTicks (DDATA) The number of major tick intervals.
NumMinorTicks (DDATA) The number of minor tick intervals.
NumLabels (DDATA) The number of label intervals (one less than the number of labels).

MajorTick (POLYGON) The major tick template.
  InnerOffset (DDATA) The offset between the inner radius and the inner end of the major tick, used to control the tick length.
  OuterOffset (DDATA) The offset between the outer radius and the outer end of the major tick, used to control the tick length.

MinorTick (POLYGON) The minor tick template.
  InnerOffset (DDATA) The offset between the inner radius and the inner end of the minor tick, used to control the tick length.
  OuterOffset (DDATA) The offset between the outer radius and the outer end of the minor tick, used to control the tick length.

TickLabel (TEXT) The label template.
Format (SDATA) The C-style label format.

**Alarm Sector Resources**
Visibility (DDATA) Controls the visibility of the alarm sector.
AngleStart (DDATA) The start angle of the alarm sector.
AngleEnd (DDATA) The end angle of the alarm sector.
BgColor (GDATA) The background color of the alarm sector; is usually constrained to the background color of the dial.
Center (GDATA) The alarm sector’s center position (may also be changed by moving the arc with the mouse).

InnerRadius (DDATA) The inner radius of the alarm sector.
OuterRadius (DDATA) The outer radius of the alarm sector.
NumAlarms (DDATA) The number of arcs in the alarm sector.
Colors (GROUP) A group of the alarm sector’s colors.
  Color<N> (GDATA) A color of the <N>th alarm sector.

High (DDATA) The high range of the thresholds. The default value is 100, allowing the specification of thresholds as percentages of the dial’s range.

Low (DDATA) The low range of the thresholds. The default value is 0.
Thresholds (GROUP) A group of the alarm sector’s thresholds.
  Value<N> (DDATA) A thresholds of <N>th alarm sector. Controls’s the <N>th alarm sector’s angle, the range of the values must match the High and Low range of the thresholds.

**Slider**

A **Slider Widget** converts a linear mouse position into one or two scalar values, and provides some appropriate visual feedback by moving its active element. When the left mouse button is pressed over the Slider Widget, the slider reacts by moving its active element to the mouse position. If the left mouse button is held down and dragged, the active element of the slider follows the mouse until the button is released.
There are several different versions of slider widgets in the GLG Widget Library, each with a different look and feel. There are also two-dimensional versions that control two variables, one corresponding to the X coordinate and the other corresponding to the Y coordinate. For one-dimensional sliders, there are horizontal and vertical versions of each slider. Native sliders and scrollbars are also provided as separate widgets.

The slider contains $ValueX$ (for the horizontal slider) or $ValueY$ (for the vertical slider) resource which contains the slider’s value and is set by the slider each time its knob is moved. The two-dimensional slider contains both $ValueX$ and $ValueY$ resources, while horizontal and vertical sliders contain only one of them, depending on the slider direction. A native slider contains a $Value$ resource regardless of the slider direction.

An object called $ActiveElement$ provides a visual feedback for the current slider value. The range of positions of the active element is associated with the range of the slider values, defined by the $Low$ and $High$ resources of the slider. Some sliders also have an indicator showing the current slider value in digital form. Many sliders have the $SliderSize$, $StartPosition$ and $EndPosition$ resources that control the size of the $ActiveElement$ as well as the extent of its movement.

A slider may also be used for output, displaying a value of its $Value$ resource. When the slider’s $ValueX$, $ValueY$ or $Value$ resources are set, the slider positions its active element to show the new value. If the widget is used only for output, you may decide to delete or disable the input handler of the widget’s viewport, although this is not necessary. The $ActiveElement$’s $Truncate$ resource controls how out of range values are displayed. By default, it is set to ON to truncate the out of range values to force them in the range, so that the slider’s knob does not move outside its start and end positions.
Resources

These are the resources you are likely to see in a Slider Widget’s hierarchy. Note that while the ValueX and ValueY resources are optional, the slider widget must contain at least one of them. A native slider widget contains the Value resource instead of ValueX and ValueY.

ValueX (DDATA, optional)
The slider’s X value.

OutValueX (DDATA, optional)
The slider’s X output value. It can be used to constrain other objects to this resource.

ValueY (DDATA, optional)
The slider’s Y value.

OutValueY (DDATA, optional)
The slider’s Y output value. It can be used to constrain other objects to this attribute.

Value (DDATA, optional)
The native slider’s input value.

OutValue (DDATA, optional)
The native slider’s X output value. It can be used to constrain other objects to this attribute.

Low (DDATA, optional)
The lower limit of the slider’s output value.

High (DDATA, optional)
The upper limit of the slider’s output value.

LowX (DDATA, optional)
The lower limit of the 2D slider’s X value.

HighX (DDATA, optional)
The upper limit of the 2D slider’s X value.

LowY (DDATA, optional)
The lower limit of the 2D slider’s Y value.

HighY (DDATA, optional)
The upper limit of the 2D slider’s Y value.

Granularity (DDATA, optional)
If this resource is present and set to a non-zero value, the slider is limited to the number of allowed positions defined by the value of this resource.

DisableMotion (DDATA, optional)
If this resource is present and set to a non-zero value, the slider’s reaction to MotionNotify events is disabled.

StartPosition (GDATA, optional)
A point that defines the lower limit of the slider’s range.

EndPosition (MARKER, optional)
A point that defines the upper limit of the slider’s range.

ActiveArea (POLYGON, optional)
The screen cursor must be within this polygon for the slider to react to user input. This resource is not usually present in the default slider. Instead, the ActiveAreaSpare polygon (invisible by default) is defined. It may be renamed ActiveArea and positioned as desired to make only some area of the slider sensitive to mouse events.

Plane (POLYGON, optional)
The slider pointer slides on the plane defined with this polygon. The points of the polygon must be coplanar.

A slider may also have buttons. A user may click on these buttons to move the slider pointer the distance specified by the Increment resource in some direction. The buttons have the following names:

Increase (VIEWPORT, optional)
An “increase” button for a one-dimensional slider.

Decrease (VIEWPORT, optional)
A “decrease” button for a one-dimensional slider.

PageIncrease (VIEWPORT, optional)
A “page increase” button for a one-dimensional slider.

PageDecrease (VIEWPORT, optional)
A “page decrease” button for a one-dimensional slider.

Left (VIEWPORT, optional)
A directional button for a two-dimensional slider.

Right (VIEWPORT, optional)
A directional button for a two-dimensional slider.

Up (VIEWPORT, optional)
A directional button for a two-dimensional slider.
Down (VIEWPORT, optional) A directional button for a two-dimensional slider.
Increment (DDATA, optional) If the slider viewport contains motion buttons, each click on the button moves the slider by the amount specified by this resource. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the slider. If this resource is not present, the default increment of motion is 0.02 times the total slider range.

PageIncrement (DDATA, optional) Defines the amount of the page increment. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the slider. If this resource is not present, the default page increment is 0.1 times the total slider range.

Refer to the GIGSlider chapter on page 233 of the GLG User's Guide and Builder Reference Manual for information on other optional resources of the slider widgets.

Knob

A Knob Widget converts an angular mouse position into a value, providing visual feedback in the form of the angular position of the knob's active element. When the left mouse button is pressed over the Knob Widget, the knob reacts by moving its active element to the mouse position. If the left mouse button is held down and dragged, the active element of the knob follows the mouse until the button is released.

There are several different versions of Knob Widgets in the GLG Widget Library, each with a different look and feel. Each uses an object called ActiveElement to provide the visual feedback for the current knob value. The range of positions of the active element is associated with the range of the knob values, defined by the Low and High resources of the knob. Some knobs also have an indicator that shows the current knob value as a number.

A knob may also be used for output. When used this way, the widget converts a scalar values (specified with the Value resource) into the position of its active element. If the widget is used only for output, you may decide to delete or disable the input handler of the widget’s viewport, although this is not necessary. The ActiveElement's Truncate resource controls how out of range values are displayed. By default, it is set to ON to truncate the out of range values to force them in the range, so that the knob does not rotate outside its start and end positions.
### Resources

<table>
<thead>
<tr>
<th>Resource</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Value (DDATA, mandatory)</td>
<td>The knob’s value.</td>
</tr>
<tr>
<td>OutValue (DDATA, optional)</td>
<td>The knob’s output value. It can be used to constrain other objects to this resource.</td>
</tr>
<tr>
<td>Low (DDATA, optional)</td>
<td>The lower limit of the knob’s value.</td>
</tr>
<tr>
<td>High (DDATA, optional)</td>
<td>The upper limit of the knob’s value.</td>
</tr>
<tr>
<td>Granularity (DDATA, optional)</td>
<td>If this resource is present and set to a non-zero value, it specifies the number of allowed positions for the knob.</td>
</tr>
<tr>
<td>DisableMotion (DDATA, optional)</td>
<td>If this resource is present and set to a non-zero value, the knob’s reaction to MotionNotify events is disabled.</td>
</tr>
<tr>
<td>StartAngle (DDATA, optional)</td>
<td>The Value resource is at its lower limit when the knob is at this angle.</td>
</tr>
<tr>
<td>EndAngle (DDATA, optional)</td>
<td>The Value resource is at its upper limit when the knob is at this angle.</td>
</tr>
<tr>
<td>Center (MARKER, optional)</td>
<td>A marker placed at the rotation center of the knob.</td>
</tr>
<tr>
<td>ActiveArea (POLYGON, optional)</td>
<td>The screen cursor must be within this polygon for the knob to react to user input. This resource is not usually present in the default knob. Instead, the ActiveAreaSpare polygon (invisible by default) is defined. It may be renamed ActiveArea and positioned as desired to make only some area of the knob sensitive to mouse events.</td>
</tr>
<tr>
<td>Plane (POLYGON, optional)</td>
<td>The knob pointer slides on the plane defined with this polygon. The points of the polygon must be coplanar.</td>
</tr>
</tbody>
</table>

A Knob Widget may also have control buttons. A user may click on these buttons to move the knob pointer a preset distance in some direction. The buttons have the following names:

<table>
<thead>
<tr>
<th>Button</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Increase (VIEWPORT, optional)</td>
<td>An “increase” button for a knob.</td>
</tr>
<tr>
<td>Decrease (VIEWPORT, optional)</td>
<td>A “decrease” button for a knob.</td>
</tr>
<tr>
<td>PageIncrease (VIEWPORT, optional)</td>
<td>A “page increase” button for a knob.</td>
</tr>
<tr>
<td>PageDecrease (VIEWPORT, optional)</td>
<td>A “page decrease” button for a knob.</td>
</tr>
<tr>
<td>Increment (DDATA, optional)</td>
<td>If the knob viewport contains control buttons, each click on the button moves the knob by the amount specified by this resource. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the knob. If this resource is not present, the default increment of motion is 0.2 times the total knob range.</td>
</tr>
<tr>
<td>PageIncrement (DDATA, optional)</td>
<td>Defines the amount of the page increment. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the knob. If this resource is not present, the default page increment is 0.1 times the total knob range.</td>
</tr>
</tbody>
</table>

Refer to the GlgKnob chapter on page 237 of the GLG User’s Guide and Builder Reference Manual for information on other optional resources of the knob widgets.
Button

A Button Widget is a widget activated with a mouse press. Buttons can be push buttons that produce some action when they are pushed, but do not have a “state,” or toggle buttons, that have an internal state. The type and behavior of different buttons are based on the resources present in the button’s viewport. Native push button and toggle widgets are also provided.

Resources

OnState (DDATA, optional) The toggle state of the button. Buttons with this resource are toggle buttons. If the resource is not present, they are push buttons. If present, this resource may be 0 or 1. A toggle button will usually use this resource to create some indicator showing its state.

InState (DDATA, optional) This resource is 1 when the mouse cursor is on the button, and 0 otherwise. A button can use this resource to change some attribute when the mouse cursor enters the button viewport.

PressedState (DDATA, optional) This resource is 1 when the mouse cursor is on the button and the mouse button is depressed.

ArmedState (DDATA, optional) This resource is set to 1 when the mouse is on the button and the Control key is pressed. This functionality can be disabled by setting the resource to -1.

LabelString (SDATA, optional) The label for the button. This resource is used when the button is on a menu, and allows a button’s label to be changed from a program.

TokenValue (DDATA, optional) The button’s value. This resource is used when the button is a part of a menu widget.

ActOnPress (DDATA, optional) If this resource is present and non-zero, the button state will change with the down-click of the mouse button. Otherwise, the state changes with the up-click. This resource cannot be changed dynamically; the drawing must be reset to update it.

PressedLightCoeff (DDATA, optional) This value is added to the button’s ambient light coefficient. It is used to control how much darker the button becomes when pressed.

BevelSize (DDATA, optional) Specifies the width of the bevels, if there are any. The width is given in numbers of pixels.

Refer to the GlgButton chapter on page 240 of the GLG User’s Guide and Builder Reference Manual for information on other optional resources of the button widgets.

Menu

A Menu Widget is an array of button widgets. The Menu manages the buttons, allowing the application to manage the array of buttons as a single object. The use of the GlgMenu input handler is described on page 248 of the GLG User’s Guide and Builder Reference Manual.
Resources

*Button*<n> (VIEWPORT) These are the buttons on the menu. The <n> is a zero-based index. A menu must contain at least one button.

*ScrollObject* (VIEWPORT, optional) The menu scroll bar. Scrolling is enabled if *LabelList* contains the number of labels bigger than the number of buttons in the menu.

*SelectedIndex* (DDATA, optional) The index of the last selected button.

*SelectedString* (SDATA, optional) The label of the last selected button.

*SelectedValue* (DDATA, optional) The value of the last selected button.

*LabelList* (GROUP, optional) A list of S data objects containing button labels. This resource may not be modified dynamically. The drawing must be reset after each change to the *LabelList*.

*TooltipList* (GROUP, optional) A list of S data objects containing tooltip strings. This resource may not be modified dynamically. The drawing must be reset after each change to the *TooltipList*.

Refer to the *GlgMenu* chapter on page 248 of the *GLG User’s Guide and Builder Reference Manual* for information on other optional resources of the menu widgets.

Text

A *Text Widget* is used for entering a text string. There are versions of the text widget for entering numerical values, both integer and floating point.

Resources

*TextString* (SDATA, optional): The current value of the text string entered.

The following resources are present only in the numerical version of the text widget:

*Value* (DDATA) The current value.

*MinValue* (DDATA, optional) The minimum value.

*MaxValue* (DDATA, optional) The maximum value.

*ValueFormat* (SDATA, optional) The C-style format for displaying the value.

*InputInvalid* (DDATA) Displays the status of input parsing.

Refer to the *GlgNText* chapter on page 242 of the *GLG User’s Guide and Builder Reference Manual* for information on other optional resources of the text widgets.
**Spinner**

A **Spinner Widget** is used for entering a numerical value using arrows to increase or decrease it. An optional slider may be used for changing the value with the mouse.

**Resources**

- **Value** (DDATA) The spinner’s value.
- **MinValue** (DDATA, optional) The minimum allowed value.
- **MaxValue** (DDATA, optional) The maximum allowed value.
- **Increment** (DDATA, optional) The value increment.
- **PageIncrement** (DDATA, optional) The value page increment.
- **Wrap** (DDATA, optional) Enables wrap mode.
- **IncreaseKeys** (SDATA, optional) Lists keyboard accelerators for increasing the value.
- **DecreaseKeys** (SDATA, optional) Lists keyboard accelerators for decreasing the value.
- **TextInput** (VIEWPORT, optional) A text input viewport for entering the value.
- **Increase** (VIEWPORT, optional) An optional button for increasing the value by **Increment**.
- **Decrease** (VIEWPORT, optional) An optional button for decreasing the value by **Increment**.
- **PageIncrease** (VIEWPORT, optional) An optional button for increasing the value by **PageIncrement**.
- **PageDecrease** (VIEWPORT, optional) An optional button for decreasing the value by **PageIncrement**.
- **Slider** (VIEWPORT, optional) An optional slider object for changing the value with the mouse.

**List and Option Menu**

The List and Option Menu widgets provide access to the corresponding native list and option menu or combo box widgets. Refer to the corresponding chapters on page 245 and page 247 of the GLG User’s Guide and Builder Reference Manual for information on the resources of these widgets.

**Clock and Stopwatch**

The **Clock and Stopwatch Widgets** display the current time or elapsed time. Both widgets use the same input handler. The handler takes care of updating the time and reacting to the input events for the stopwatch. The Clock Widget continues updating even while being edited.

The Clock Widget does not react to input events from a user and serves to display the current time only. If the **TimerState** resource is present, the clock measures elapsed time instead of real time, and the widget becomes a stopwatch. The Stopwatch Widget can receive input through three buttons contained in the clock viewport.

If the input handler of the Clock Widget is deleted or disabled, the widget may be used to display user-defined time under control of an application program. In this case the application program uses the widget resources to drive the clock.

The hands and other elements of the widgets may be edited the same way as any other object.
Resources

- **Hour** (DDATA, optional) The number of hours. (0-11)
- **Min** (DDATA, optional) The number of minutes. (0-59)
- **Sec** (DDATA, optional) The number of seconds. (0-59)
- **ValueHour** (DDATA, optional) The scaled hour value. This is a value between 0 and 1, and is provided to simplify the positioning of clock hands.
- **ValueMin** (DDATA, optional) The scaled minute value. This is a value between 0 and 1, and is provided to simplify the positioning of clock hands.
- **ValueSec** (DDATA, optional) The scaled second value. This is a value between 0 and 1, and is provided to simplify the positioning of clock hands.
- **HourLabel** (TEXT, optional) A title for the hour display.
- **MinLabel** (TEXT, optional) A title for the minute display.
- **SecLabel** (TEXT, optional) A title for the second display.
- **TimeString** (SDATA, optional) A string describing the time in the “00:00” format.

The following resources are used for the stopwatch widgets.

- **TimerState** (DDATA, optional) If this resource is present, the Clock Widget measures elapsed time. When the resource is 0, the clock is stopped, and when it is 1, the clock is running.
- **Start** (VIEWPORT, optional) If a button with this name is present within the clock viewport, pressing it will cause the stopwatch to start.
- **Stop** (VIEWPORT, optional) If a button with this name is present within the clock viewport, pressing it will cause the stopwatch to stop.
- **Reset** (VIEWPORT, optional) If a button with this name is present within the clock viewport, pressing it will cause the stopwatch to reset to zero.

Palette

A **Palette Widget** is used for creating different palettes with which to present a user with choices of objects. The Palette Widget was created to help a user choose colors, but it can be used to choose one from any selection of objects. This widget is designed to be used within a program using the GLG API.

A palette contains a single resource, an object named **PaletteObject**. This is a group or series object that contains several objects, each corresponding to a different possible choice. The user selects one of the presented objects with the left mouse button, and the choice is returned to the calling program in the callback message object.

For more information about callbacks and message objects, see the *GLG User’s Guide and Builder Reference Manual*. 
Resources

*PaletteObject (ANY OBJECT)*
An object displaying the set of choices.

*SelectedObject (ANY OBJECT, message only)*
The selected object. This resource does not appear in the drawing resource hierarchy, but is passed to the user-supplied callback function inside the message object.

*DisableMotion (DDATE)*
Disables handling motion events. If motion events are enabled, selection messages are generated on both mouse click and mouse drag events.

**Font, Resource and Tag Browsers**

The **Font Browser Widget** is used to facilitate the font selection from the list of all fonts available for the X server used to display the widget. The Font Browser, in turn, uses other widgets such as Menu, Text and Slider as its components. The Font Browser Widget is not available on the Windows platform, where a native font browser is available.

The **Resource Browser** and Tag Browser Widgets provide the application with the resource and tag browsing functionality similar to the one in the GLG Graphics Builder.

Note that the browser’s buttons may have labels different from their resource names.

**Common Resources of Browser Widgets**

The following resources are shared by all types of the browser widgets:

*Path (VIEWPORT)*
A text widget for entering a resource path for the resource browser or tag name for the tag browser.

*Filter (VIEWPORT, optional)*
A text widget for entering and displaying the current value of the filter string. Only entries that match the filter string are displayed in the selection menu.

*Menu (VIEWPORT)*
A menu widget that displays a list of all available entries.

*Done (VIEWPORT, optional)*
A button for accepting the selection. This button is used to generate an “Activate” message when the selection is finished.

*Cancel (VIEWPORT, optional)*
A button for cancelling the selection. This button is used to generate a “Cancel” message to indicate the user’s desire to cancel the selection process without making a selection.

*Reset (VIEWPORT, optional)*
A button to reset the filter.

**Resource Browser’s Resources**

*ListNamed (VIEWPORT)*
A toggle that controls the display of named resources.

*ListAliases (VIEWPORT)*
A toggle that controls the display of aliases.

*ListDefault (VIEWPORT)*
A toggle that controls the display of default resources.
Tag Browser’s Resources

UniqueTags (VIEWPORT) A toggle that controls the display of unique tag names. If set to ON, all tags with the same name will be displayed. If set to OFF, only tags with unique names will be displayed.

Font Browser Resources

FontName (VIEWPORT) A text widget for entering and displaying the current font name selection.

FontSampleName (SDATA, optional) The name of the selected font.

FieldMenu (VIEWPORT, optional) A menu widget for constructing the filter. Clicking on one of the buttons shows a list of the available options for that particular field.

FontSample (VIEWPORT, optional) A viewport with a text object in it to display sample text using the selected font.

FontSampleName (SDATA, optional) The name of the selected font.

Process Control Objects

The process control symbols palettes provide the user with an easy way to create process control drawings by simply selecting the pre-built objects from a palette. Most of the objects have a built-in dynamic behavior to be used for displaying values or animation.

The resource named Value is the main dynamic resource of these objects. For example, setting the Value of a tank object to a value in the range from 0 to 1 changes the level of the tank’s filling. For a valve object, the Value resource controls its opening. For fans, pumps and other dynamic equipment, changing the Value resource from 0 to 1 in small increments animates the piece of equipment.

The objects may also have other resources to control their appearance:

• ForegroundColor and BackgroundColor control objects’ colors.

• OpenColor and CloseColor control the colors of valves.

• ColorIndex controls the displayed color of various indicator objects, and Color0, Color1, ... ColorN resources define available colors.

• LineWidth controls the line width of lines used to render the object.

• Units controls the unit label string used by various indicator objects.

• Format controls the C format used to display the object’s numerical value.

The process control objects also include connector-style objects such as various pipes and lines. While most of the objects are like nodes with just one control point, the connector-style objects have two control points which may be used to define their shape. Some of the connectors have various elements replicated along the connector and have the NumElements and ElementSize resources to control the elements.
The Resource Browser may be used to list all resources of the selected process control object. Tags may be assigned to the resource names to facilitate connecting dynamic data to the objects. Refer to the Using Tags chapter on page 291 of the GLG User’s Guide and Builder Reference Manual for more information about tags.

**Layout Templates**

Layout templates provide a collection of templates for an application’s top-level window. Each template contains a viewport named *DisplayArea*, as well as optional fixed-size (non-resizable) viewports on the left, right, top and bottom of *DisplayArea* that can be used to host a toolbar, a menu or a message area. The optional areas are named *AreaLeft*, *AreaRight*, *AreaTop* and *AreaBottom*. The *OffsetLeft*, *OffsetRight*, *OffsetTop* and *OffsetBottom* resources define the size of the optional areas in screen pixels.

For applications that use a menu to navigate between different drawings, the *DisplayArea* viewport can be replaced with a *SubWindow* object by selecting the viewport and using the Arrange, Replace Viewport with SubWindow menu option available in the Enterprise Edition of the Graphics Builder. The subwindow’s *Source* attribute can then be changed to FILE, and the *Source Path* attribute can be used to supply a filename of the drawing to be displayed in the subwindow.
Chapter 2

The GLG Widget Library

This chapter contains descriptions of each of the widgets in the GLG Widget Library. The description of each widget contains information about how to use that widget. You may also want to consult the Using the GLG Widgets chapter at the beginning of this manual.

The Process Control Objects are not listed in this chapter. Refer to the Process Control Objects section of the Using the GLG Widgets chapter for the list of common resources of the process control objects.

To test each widget, you can use the GLG Graphics Builder to animate the widget. The resource path of an entry point for animating each graph is listed in the graph’s resources. As an example, a graph widget might contain the following entry points:

**Data Values Entry Point**

\$Widget/DataGroup/EntryPoint

**Time Labels Entry Point**

\$Widget/XLabelGroup/EntryPoint

If the listed default data range is between 0 and 1, then the widget can be animated using the following animation command:

```
$datagen d 0 1 "$Widget/DataGroup/EntryPoint"
$s 0 0 "$Widget/XLabelGroup/EntryPoint"
```

All graph widgets contain default animation commands for prototyping them in the Graphics Builder. Refer to the Prototyping Graph Widgets with Custom Animation Commands section on page 17 of this manual for information on using the default animation command.

Refer to the the The datagen utility is embedded into the GLG Builder, which invokes the utility to generate data for prototyping the drawing in the Run mode using the $datagen command in the Run dialog. $datagen instructs the Builder to use an internal version of the datagen utility. chapter on page 376 of the GLG Programming Reference Manual for a complete list of the prototyping command options.

The rest of this chapter describes the resources of each widget. Because there is a lot of duplication between the widgets, the resources are grouped into “sets”, and the descriptions below just list the resource sets used by each widget. For lists of the resources that make up the resource sets, see the GLG Widget Resource Sets chapter on page 99 of this manual.
Bar Graph
Signed Bar Graph
Histogram Graph

The Bar Graph is used to display one set of data as an arrangement of vertical bars. The height of a bar corresponds to the value of the data it represents. The Signed Bar Graph is similar to the bar graph, but data samples may have negative values as well as positive. The Histogram Graph is similar to the Bar Graph, but with no gaps between the bars.

Animation Data

The entry points for these graphs are as follows:

Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default range for the Signed Bar Graph is between -1 and 1.

Files

Bar Graph: bar1.g
Signed Bar Graph: bar4.g
Histogram Graph: bar7.g

Resource Sets

These widgets use resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set
**Horizontal Bar Graph**

The Horizontal Bar Graph displays one set of data in 2D as horizontal bars. This graph is the same as the Bar Graph, except that the bars are horizontal. The graph resources are the same too, with the only distinction that X and Y prefixes are swapped.

*Animation Data*

The graph’s entry points are as follows:

**Data Entry**

$Widget/DataGroup/EntryPoint

**Label Entry**

$Widget/YLabelGroup/EntryPoint

The default data range is between 0 and 1.

*Files*

bar2.g

*Resource Sets*

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Value Axis Resource Set
- Y Time Axis Resource Set
- Zooming Resource Set

**Symmetrical Bar Graph**

The Symmetrical Bar Graph displays one set of data in 2D as vertical bars centered around X axis. The height of a bar corresponds to the value of the data it represents. This is not the same as a Signed Bar Graph. The height of a bar above the X axis is the same as it’s depth below it; it is symmetrical.

*Animation Data*

The graph’s entry points are as follows:
Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.

Files

bar3.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• Symmetrical Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set

Packed Bar Graph

The Packed Bar Graph displays several sets of data in 2D in clusters or “packs” of vertical bars. One pack of bars contains one sample of data that includes several datasamples. The height of each bar corresponds to the value of its data sample.

Animation Data

The graph’s entry points are as follows:

Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. There are three bars in each pack in the default graph.
Files

bar8.g

Resource Sets

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Packed Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Multiaxis Resource Set
- Zooming Resource Set
- Legend Resource Set

Stacked Bar Graph

The Stacked Bar Graph displays several sets of data in 2D as stacked vertical bars. This is comparable to a packed bar graph except that the bars that make up each cluster are placed on top of each other. Each bar, composed of a pack of several smaller bars, contains one iteration of data from each data set. The height of every component of a stack corresponds to the value of its data sample. The total height of a stack is equal to the sum of the heights of the individual components. The range of the graph is equal to the range of the sum of the individual components. This means that the sum of all values corresponding to one stack should be less than or equal to the range of the graph.

Animation Data

The graph’s entry points are as follows:

Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 0.2, and there are, by default, 5 bars in each stack.
Files

bar9.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Packed Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set
• Legend Resource Set

Step Graph

The Step Graph displays one set of data in 2D as a step line. The height of a step corresponds to the value of a data sample.

Animation Data

The graph’s entry points are as follows:

Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.

Files

step1.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set

**Filled Step Graph**

The Filled Step Graph is the same as the Step Graph (see page 52), but with a filled line.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry**

$Widget/DataGroup/EntryPoint

**Label Entry**

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.

**Files**

step4.g

**Resource Sets**

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Filled Step Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set
**Horizontal Step Graph**

The Horizontal Step Graph displays one set of data in 2D as a step line. The horizontal position of a step corresponds to the value of a data sample. This graph is the same as the Step Graph (see page 52), but is rotated on its side.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry**

`$Widget/DataGroup/EntryPoint`

**Label Entry**

`$Widget/YLabelGroup/EntryPoint`

The default data range is between 0 and 1.

**Files**

`step2.g`

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Value Axis Resource Set
- Y Time Axis Resource Set
- Zooming Resource Set

**Horizontal Filled Step Graph**

The Horizontal Filled Step Graph displays one set of data in 2D as a vertical step line. The horizontal position of a step corresponds to the value of a data sample. This graph is the same as the Filled Step Graph (see page 53), but is rotated on its side.

**Animation Data**

The graph’s entry points are as follows:
Data Entry
$Widget/DataGroup/EntryPoint

Label Entry
$Widget/YLabelGroup/EntryPoint

The default data range is between 0 and 1.

Files
step5.g

Resource Sets
This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Filled Step Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Value Axis Resource Set
- Y Time Axis Resource Set
- Zooming Resource Set

Multiline Step Graph
The Multiline Step Graph displays multiple sets of data in 2D as step lines. The height of a step corresponds to the value of a data sample.

Animation Data
The graph’s entry points are as follows:

Data Entry for All Lines
$Widget/DataGroupOne/EntryPoint

Label Entry
$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default number of data groups is two.
**Data Entry for One Line**

To animate just one of the data groups, use this resource in place of the above data entry point:

\[ $\text{Widget/DataGroupOne/DataGroup}<N>/\text{EntryPoint} \]

Use the number of the desired data group in place of \(<N>\), as in

\[ $\text{Widget/DataGroupOne/DataGroup3/EntryPoint} \].

**Files**

step3.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Multiset Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Zooming Resource Set
- Legend Resource Set

**Filled Multiline Step Graph**

The Filled Multiline Step Graph is the same as the Multiline Step Graph (see page 55), but with a filled step line.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry for All Lines**

\[ $\text{Widget/DataGroupOne/EntryPoint} \]

**Label Entry**

\[ $\text{Widget/XLabelGroup/EntryPoint} \]

The default data range is between 0 and 1. The default number of data groups is two.
Data Entry for One Line

To animate just one of the data groups, use this resource in place of the above data entry point:

$Widget/DataGroupOne/DataGroup<N>/EntryPoint

Use the number of the desired data group in place of $N$, as in $Widget/DataGroupOne/DataGroup3/EntryPoint$.

Files

step6.g

Resource Sets

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Filled Multiline Step Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Zooming Resource Set
- Legend Resource Set

Line Graph

Filled Line Graph
Line Segment Graph
Point Graph

The Line Graph displays one set of data in 2D as a polyline with marker objects identifying each data value. The Y coordinate of a polyline point corresponds to the value of a data sample. Markers may be switched off if desired.

The Filled Line Graph is the same as the Line Graph, but with a filled line. The Line Segment Graph is also similar, but the segments of the polyline may have different colors, widths and line types.

A Point Graph is simply a Line Graph without the lines. Only the markers are displayed.

Animation Data

The entry points for these graphs are as follows:
Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.

Files

Line Graph: line1.g
Filled Line graph: line3.g
Line Segment Graph: line5.g
Point Graph: point1.g

Resource Sets

These widgets use resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Line Datagroup Resource Set
- Level Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Zooming Resource Set

Multiline Graph
Filled Multiline Graph
Multiset Line Segment Graph
Multiset Point Graph

These four graphs are versions of the graphs on page 57 equipped to display data from multiple data sets. They have multiple lines and multiple Y axes with which to chart them.

Animation Data

These graphs’ entry points are as follows:

Data Entry for All Lines

$Widget/DataGroupOne/EntryPoint
Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default number of data groups is two.

Data Entry for One Line

To animate just one of the data groups, use this resource in place of the above data entry point:

$Widget/DataGroupOne/DataGroup<N>/EntryPoint

Use the number of the desired data group in place of <N>, as in $Widget/DataGroupOne/DataGroup3/EntryPoint.

Files

Multiline Graph: line2.g
Filled Multiline Graph: line4.g
Multiset Line Segment Graph: line6.g
Multiset Point Graph: point2.g

Resource Sets

These widgets use resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Multiline Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Multiaxis Resource Set
• Zooming Resource Set
• Legend Resource Set

XY Line Graph

XY Point Graph

The XY Line Graph displays one set of (X,Y) data pairs in 2D as a polyline with markers identifying each data pair. The markers may be switched off if desired. The X and Y coordinates of the polyline’s points correspond to X and Y values of the input data pairs. The XY Point Graph is the same as the XY Line Graph, but only the markers are displayed. The XY Graph is also known as a scatter graph.
Animation Data

The entry points for these graphs are as follows:

Data Entry

$Widget/DataGroup/XEntryPoint
$Widget/DataGroup/YEntryPoint

The default data range is between -1 and 1.

Files

XY Line Graph: scatt1.g
XY Point Graph: scatt3.g

Resource Sets

These widgets use resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Xy Line Datagroup Resource Set
• X Level Resource Set
• Y Level Resource Set
• Status Object Resource Set
• X Value Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set

Multiline XY Graph
Multiset Point XY Graph

These two graphs are versions of the graphs on page 59 equipped to display data from multiple data sets. There is only one set of axes with this graph, so the various data sets should have the same data range. For different data ranges, see page 61.

Animation Data

The entry points for these graphs are as follows:

Data Entry for All Lines

$Widget/DataGroupOne/XEntryPoint
$Widget/DataGroupOne/YEntryPoint

The default data range is between -1 and 1. The default number of data groups is two.
Data Entry for One Line

To animate just one of the data groups, use these resource in place of the above data entry points:

$\text{Widget/DataGroupOne/DataGroup}<N>/XEntryPoint$
$\text{Widget/DataGroupOne/DataGroup}<N>/YEntryPoint$

Use the number of the desired data group in place of $<N>$, as in
$\text{Widget/DataGroupOne/DataGroup3/EntryPoint}$.

Files

Multiline XY Graph: scatt2.g
Multiset Point XY Graph: scatt4.g

Resource Sets

These widgets use resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Multiline Xy Datagroup Resource Set
• X Level Resource Set
• Y Level Resource Set
• Status Object Resource Set
• X Value Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set
• Legend Resource Set

Multiaxis Multiline XY Graph

The Multiaxis Multiline XY Graph displays several sets of (X,Y) data pairs in 2D as polylines with markers identifying each data pair. Markers may be switched off if desired. The X and Y coordinates of the polyline’s points correspond to X and Y values of data pairs. The graph has several axes to display different sets of data in different ranges. See page 59 for the single data set version of this graph.

The Special widgets set contain additional axes objects that may be added to any graph.

Animation Data

The graph’s entry points are as follows:
Data Entry for All Graphs

$\text{Widget/DataGroupOne/XEntryPoint}
$\text{Widget/DataGroupOne/YEntryPoint}

The default data range is between -1 and 1. The default number of data groups is two.

Data Entry for One Line

To animate just one of the data groups, use these resource in place of the above data entry points:

$\text{Widget/DataGroupOne/DataGroup<N>/XEntryPoint}
$\text{Widget/DataGroupOne/DataGroup<N>/YEntryPoint}

Use the number of the desired data group in place of $<N>$, as in $\text{Widget/DataGroupOne/DataGroup3/EntryPoint}$. 

Files

scatt5.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Multiline Xy Datagroup Resource Set
• X Level Resource Set
• Y Level Resource Set
• Status Object Resource Set
• X Value Axis Resource Set
• Y Value Multiaxis Resource Set
• Zooming Resource Set
• Legend Resource Set

Polar Line Graph

Polar Filled Line Graph

Polar Line Segment Graph

Polar Point Graph

The Polar Line Graph displays one set of data in polar coordinates as a polyline with markers at each data value. The value of the data sample controls the radial distance of the polyline’s point from the center of the graph. Markers may be switched off if desired.
The Polar Filled Line Graph is the same as the Polar Line Graph, but with a filled line. The Polar Line Segment Graph is also similar, but the segments of the polyline may have different colors, widths and line types.

The Polar Point Graph is the same as the Polar Line Graph, but displays just the markers.

**Animation Data**

The entry points for these graphs are as follows:

**Data Entry**

$\text{Widget/DataGroup/EntryPoint}$

The default data range is between 0 and 1.

**Files**

- Polar Line Graph: polar1.g
- Polar Filled Line Graph: polar2.g
- Polar Line Segment Graph: polar6.g
- Polar Point Graph: polar7.g

**Resource Sets**

These widgets use resources from the following sets:

- Title Resource Set
- Polar Data Area Resource Set
- Line Datagroup Resource Set
- Polar Level Resource Set
- Status Object Resource Set
- Tangent Ticks And Labels Resource Set
- X Axis Resource Set
- Y Axis Resource Set
- Zooming Resource Set

**Multiline Polar Graph**

The Multiline Polar Graph displays several sets of data in polar coordinates as polylines with markers at each data value. The value of the data sample controls the radial distance of the polyline’s point from the center of the graph. Markers may be switched off if desired. This is a multiline version of the graph on page 62.
**Animation Data**

The entry points for these graphs are as follows:

**Data Entry for All Lines**

$\text{Widget/DataGroupOne/EntryPoint}$

The default data range is between -1 and 1. The default number of data groups is two.

**Data Entry for One Line**

To animate just one of the data groups, use these resource in place of the above data entry points:

$\text{Widget/DataGroupOne/DataGroup<N>/EntryPoint}$

Use the number of the desired data group in place of $<N>$, as in

$\text{Widget/DataGroupOne/DataGroup3/EntryPoint}$.

**Files**

polar4.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- Polar Data Area Resource Set
- Multiline Datagroup Resource Set
- Polar Level Resource Set
- Status Object Resource Set
- Tangent Ticks And Labels Resource Set
- X Axis Resource Set
- Y Axis Resource Set
- Zooming Resource Set

**XY Polar Line Graph**

XY Polar Line Graph: Displays one set of data pairs in 2D in polar coordinates as a polyline with markers identifying each data value. One value of a pair controls the angle and the other controls the radial distance of the polyline’s point from the center of the graph. Markers may be switched off if desired.

**Animation Data**

The graph’s entry points are as follows:
Data Entry

$Widget/DataGroup/AngleEntryPoint
$Widget/DataGroup/RadiusEntryPoint

The default radius range is between 0 and 1. The default angle range is between 0 and 360.

Files

polar3.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Polar Data Area Resource Set
• Polar Xy Line Datagroup Resource Set
• Polar Level Resource Set
• Radial Grid Resource Set
• Tangent Grid Resource Set
• Tangent Labels Resource Set
• Zooming Resource Set

Multiline Polar XY Graph

The Multiline Polar XY Graph is a multiple data set version of the Polar XY Graph described on page 64.

Animation Data

The graph’s entry points are as follows:

Data Entry for All Lines

$Widget/DataGroupOne/AngleEntryPoint
$Widget/DataGroupOne/RadiusEntryPoint

The default radius data range is between 0 and 1, while the angle spans between 0 and 360. The default number of data groups is two.

Data Entry for One Line

To animate just one of the data groups, use these resources in place of the above data entry points:

$Widget/DataGroupOne/DataGroup<N>/AngleEntryPoint
$Widget/DataGroupOne/DataGroup<N>/RadiusEntryPoint
Use the number of the desired data group in place of $<N>$, as in $\$Widget/DataGroupOne/DataGroup3/AngleEntryPoint$.

**Files**

polar5.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- Polar Data Area Resource Set
- Multiline Polar Xy Datagroup Resource Set
- Polar Level Resource Set
- Radial Grid Resource Set
- Tangent Grid Resource Set
- Tangent Labels Resource Set
- Zooming Resource Set

**Pie Chart**

The Pie Chart displays one set of data as arc segments. A pie segment selection may be indicated by shifting the segment outside of the pie chart.

**Animation Data**

The entry points for these graphs are as follows:

**Data Entry**

$\$Widget/DataGroup/EntryPoint$

The default data range is between 0 and any positive number.

**Files**

Pie Chart: pie1.g

**Resource Sets**

These widgets use resources from the following sets:

- Title Resource Set
- Pie Datagroup Resource Set
• Zooming Resource Set
• Legend Resource Set

**Combination Graph**

The Combination Graph is a Bar Graph Widget with a Multiset Line Graph overlaid on top of the bars.

**Animation Data**

The entry points for the bar graph part of the combination are as follows:

**Data Entry**

$\text{Widget/DataGroup/EntryPoint}$

**Label Entry**

$\text{Widget/XLabelGroup/EntryPoint}$

The default data range is between 0 and 1.

The entry point for the Multiset Line Graph component of the Combination Graph is as follows:

$\text{Widget/DataGroupOne/EntryPoint}$

The default number of data sets is two.

**Files**

comb1.g

**Resource Sets**

This widgets use resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• Multiline Datagroup Resource Set
• Level Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Zooming Resource Set
**Stock Graphs**

The Stock Graphs are used to show current and past values of a data series. In addition to displaying current data with a line or marker, a stock graph can show the range of data over one or two given time periods. There are two stock graphs. The simpler graph, `stock2.g`, displays a marker at some “current” value, and can display a high and low limit with a vertical line behind that marker. The more complex stock graph, `stock1.g`, displays a multiset line graph with bars and lines to display two different sets of limit information in the background.

**Animation Data**

The entry points for the `stock2.g` graphs are as follows:

**Data Entry**

$Widget/DataGroup/MarkEntryPoint
$Widget/DataGroup/HighEntryPoint
$Widget/DataGroup/LowEntryPoint

**Label Entry**

$Widget/XLabelGroup/EntryPoint

The default data range is between 20000 and 100000.

The entry points for the `stock1.g` graph are as follows:

**Data Entry**

$Widget/DataGroupOne/EntryPoint
$Widget/DataGroup/OpenEntryPoint
$Widget/DataGroup/ClosedEntryPoint
$Widget/DataGroup/HighEntryPoint
$Widget/DataGroup/LowEntryPoint
$Widget/DataGroup/MarkEntryPoint

The label entry point is the same as for the `stock2.g` graph.

**Files**

stock1.g

stock2.g

**Resource Sets**

These widgets use resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Multiline Datagroup Resource Set
- Level Resource Set
• Status Object Resource Set  
• X Time Axis Resource Set  
• Y Value Axis Resource Set  
• Zooming Resource Set

**Axes Widget**

The Axes Widget is a set of graph axes that may be added to any other widget.

**Files**

axes1.g

**Resource Sets**

These widgets use resources from the following sets:

• X Value Axis Resource Set  
• Y Value Axis Resource Set

**3D Bar Graph**  
**3D Signed Bar Graph**  
**3D Histogram Graph**  
**Pyramid Graph**  
**Prism Graph**  
**Cylinder Graph**

All the graphs in this group display one set of data as an array of 3D objects. The height of an object corresponds to the value of its data sample. The 3D Signed Bar displays negative values as well as positive. The difference between the other graphs is only the shape of the objects representing the data. The Pyramid Graph uses little pyramids, and the Cylinder Graph uses cylinders and so on.

**Animation Data**

The entry points for these graphs are as follows:

**Data Entry**

$Widget/DataGroup/EntryPoint

**Label Entry**

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.
Files

3D Bar Graph: bar101.g
3D Signed Bar Graph: bar109.g
3D Histogram Graph: bar119.g
Pyramid Graph: bar107.g
Prism Graph: bar103.g
Cylinder Graph: bar105.g

Resource Sets

These widgets use resources from the following sets:
- Title Resource Set
- 3D Data Area Resource Set
- 3D Datagroup Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Y1 Value Axis Resource Set
- Viewing Resource Set

Multiset 3D Bar Graph
Multiset 3D Signed Bar Graph
Multiset Pyramid Graph
Multiset Prism Graph
Multiset Cylinder Graph

These graphs are multiset variations of that graphs described on page 69.

Animation Data

The entry points for these graphs are as follows:

Data Entry for All DataGroups

$Widget/DataGroupOne/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default number of data groups is two.
**Data Entry for One DataGroup**

To animate just one of the data groups, use this resource in place of the above data entry point:

\[ \$\text{Widget/DataGroupOne/DataGroup}<N>/\text{EntryPoint} \]

Use the number of the desired data group in place of \(<N>\), as in
\[ \$\text{Widget/DataGroupOne/DataGroup3/EntryPoint}. \]

**Files**

- Multiset 3D Bar Graph: bar102.g
- Multiset 3D Signed Bar Graph: bar112.g
- Multiset Pyramid Graph: bar108.g
- Multiset Prism Graph: bar104.g
- Multiset Cylinder Graph: bar106.g

**Resource Sets**

These widgets use resources from the following sets:

- Title Resource Set
- 3D Data Area Resource Set
- Multiset 3D Datagroup Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Y1 Value Axis Resource Set
- Z Axis Resource Set
- Viewing Resource Set

**3D Packed Bar Graph**

**3D Packed Signed Bar Graph**

The 3D Packed Bar Graph displays several sets of data as packs of vertical 3D bars. One pack contains one iteration of data from each of the data sets. The height of a bar corresponds to the value of its data sample. The signed version accepts negative values as well as positive.

**Animation Data**

The entry points for these graphs are as follows:

**Data Entry**

\[ \$\text{Widget/DataGroup/EntryPoint} \]
Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1, and the default number of bars in a pack is three.

Files

3D Packed Bar Graph: bar115.g
3D Packed Signed Bar Graph: bar117.g

Resource Sets

These widgets use resources from the following sets:

- Title Resource Set
- 3D Data Area Resource Set
- 3D Packed Datagroup Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Y1 Value Axis Resource Set
- Viewing Resource Set

Multiset 3D Packed Bar Graph
Multiset 3D Packed Signed Bar Graph

Displays several sets of data as several series of packs of vertical 3D bars. One pack of bars keeps corresponding samples of one set. The height of a bar corresponds to the value of a data sample.

The signed version accepts negative values as well as positive.

Animation Data

The entry points for these graphs are as follows:

Data Entry for All DataGroups

$Widget/DataGroupOne/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default number of data groups is two, with three bars in each pack.
Data Entry for One DataGroup

To animate just one of the data groups, use this resource in place of the above data entry point:

$Widget/DataGroupOne/DataGroup<N>/EntryPoint

Use the number of the desired data group in place of <N>, as in
$Widget/DataGroupOne/DataGroup3/EntryPoint.

To animate just one of the data packs, you might use a resource that looks like this:

$Widget/DataGroupOne/DataGroup1/Pack2/EntryPoint

Files

Multiset 3D Packed Bar Graph: bar116.g
Multiset 3D Packed Signed Bar Graph: bar118.g

Resource Sets

These widgets use resources from the following sets:

• Title Resource Set
• 3D Data Area Resource Set
• Multiset 3D Packed Datagroup Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Z Axis Resource Set
• Viewing Resource Set

3D Stacked Bar Graph

The 3D Stacked Bar Graph displays several sets of data as packs of stacked vertical 3D bars. The corresponding samples of each set are placed on top of one another to form one pack of stacked bars. Each stack contains one iteration of data from each data set. The height of every component of a stack corresponds to the value of its data sample, and the total height of the stack equals the sum of the heights of its individual components.

The range of the graph should accommodate the range of the sum of the individual component data sets. This means that the sum of all values corresponding to one stack should be less than or equal to the total graph range.
Animation Data

The graph’s entry points are as follows:

Data Entry

$Widget/DataGroup/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 0.2, and there are five bars in a stack, by default.

Files

bar113.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• 3D Data Area Resource Set
• 3D Packed Datagroup Resource Set
• Status Object Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Viewing Resource Set

Multiset 3D Stacked Bar Graph

The Multiset 3D Stacked Bar Graph is a multiset version of the Stacked Bar Graph described on page 73.

Animation Data

The graph’s entry points are as follows:

Data Entry for All DataGroups

$Widget/DataGroupOne/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 0.2. The default number of bars in a stack is 5 and the number of data groups is 2.
Data Entry for One DataGroup

To animate just one of the data groups, use this resource in place of the above data entry point:

\$Widget/DataGroupOne/DataGroup\<N>/EntryPoint

Use the number of the desired data group in place of \(<N>\), as in
\$Widget/DataGroupOne/DataGroup3/EntryPoint.

Files

bar114.g

Resource Sets

This widget uses resources from the following sets:

- Title Resource Set
- 3D Data Area Resource Set
- Multiset 3D Packed Datagroup Resource Set
- Status Object Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Y1 Value Axis Resource Set
- Z Axis Resource Set
- Viewing Resource Set

3D Step Graph

The 3D Step Graph displays one set of data in 3D as a step line. The height of a step corresponds to the value of a data sample.

Animation Data

The graph’s entry points are as follows:

Data Entry

\$Widget/DataGroup/EntryPoint

Label Entry

\$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.
Files

step101.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• 3D Data Area Resource Set
• 3D Datagroup Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Viewing Resource Set

Multiset 3D Step Graph

The Multiset 3D Step Graph displays several sets of data in 3D as step lines. The height of a step corresponds to the value of a data sample.

Animation Data

The graph’s entry points are as follows:

Data Entry for All DataGroups

$Widget/DataGroupOne/EntryPoint

Label Entry

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. The default number of data groups is two.

Data Entry for One DataGroup

To animate just one of the data groups, use this resource in place of the above data entry point:

$Widget/DataGroupOne/DataGroup<N>/EntryPoint

Use the number of the desired data group in place of <N>, as in $Widget/DataGroupOne/DataGroup3/EntryPoint.
**Files**

step102.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- 3D Data Area Resource Set
- Multiset 3D Datagroup Resource Set
- X Time Axis Resource Set
- Y Value Axis Resource Set
- Y1 Value Axis Resource Set
- Z Axis Resource Set
- Viewing Resource Set

**Ribbon Graph**

The Ribbon Graph displays one set of data in 3D as a set of points connected by a ribbon. The height of a point corresponds to the value of a data sample. You can adjust the width of the ribbon by editing the polygon that is its series template.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry**

$Widget/DataGroup/EntryPoint

**Label Entry**

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1.

**Files**

ribbon101.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- 3D Data Area Resource Set
• Datagroup Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Viewing Resource Set

**Multiset Ribbon Graph**

The Multiset Ribbon Graph displays several sets of data in 3D as data points connected by a ribbon. The height of a point corresponds to the value of a data sample.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry for All DataGroups**

$\text{Widget/DataGroupOne/EntryPoint}$

**Label Entry**

$\text{Widget/XLabelGroup/EntryPoint}$

The default data range is between 0 and 1. The default number of data groups is two.

**Data Entry for One DataGroup**

To animate just one of the data groups, use this resource in place of the above data entry point:

$\text{Widget/DataGroupOne/DataGroup<N>/EntryPoint}$

Use the number of the desired data group in place of $<$N$>$, as in $\text{Widget/DataGroupOne/DataGroup3/EntryPoint}$.

**Files**

ribbon102.g

**Resource Sets**

This widget uses resources from the following sets:

• Title Resource Set
• 3D Data Area Resource Set
• Multiset Datagroup Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Z Axis Resource Set
• Viewing Resource Set

**Framed Surface Graph**

**Unframed Surface Graph**

These graphs display one set of data in 3D as a surface. The surface area is defined by a parallelogram. The parallelogram is then divided into number of nodes by the grid lines parallel to its sides. Every node is elevated by the amount corresponding to the value of a data sample. Drawing markers can be drawn on the graph. Markers are always drawn on the top of the surface.

To form a surface graph, every set of four nodes is connected by a line, creating a parallelogram. This parallelogram is close to a planar one only if the number of data samples is big and the value changes from one data sample to the next is small in both X and Y directions.

The difference between two graphs is in the number of context objects. The Unframed Surface Graph has only the axes, ticks and labels in addition to the surface. The Framed Surface Graph also has a frame made from the polygons and horizontal grids.

**Animation Data**

The entry points for these graphs are as follows:

**Data Entry**

`$Widget/DataGroup/EntryPoint`

The default data range is between 0 and 1. The default graph has 11 column and row nodes, giving 121 total nodes. Use these parameters to control the `datagen` test program.

**Files**

Framed Surface Graph: surf101.g
Unframed Surface Graph: surf102.g

**Resource Sets**

These widgets use resources from the following sets:

• Title Resource Set
• 3D Data Area Resource Set
• Surface Datagroup Resource Set
• X Time Axis Resource Set
• Y Value Axis Resource Set
• Y1 Value Axis Resource Set
• Z Value Axis Resource Set
• Viewing Resource Set

**Gantt Chart**

Displays constrained time intervals as horizontal bars. The beginning of the next interval is constrained to coincide with the end of the previous interval.

The sum of all time interval values of the graph should be less than or equal to the total range of the graph.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry**

$Widget/DataGroup/EntryPoint

**Label Entry**

$Widget/XLabelGroup/EntryPoint

The default data range is between 0 and 1. However, the default graph has 10 time intervals in it, so you should test the program with a data range between 0 and 0.1 so that the sum of all the time intervals is less than 1.

**Files**

gantt1.g

**Resource Sets**

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• X Axis Resource Set
• Y Axis Resource Set
• Zooming Resource Set

**Horizontal Slider Graph**

The Horizontal Slider Graph displays one data value in 2D as a horizontal bar. The horizontal dimension of the bar corresponds to the value of the data sample.
Animation Data

The graph’s entry points are as follows:

Data Entry

\$Widget/DataGroup/ValueX

The default data range is between 0 and 1.

The Slider Graph may also be used as a control widget, and animated without any data by using its reaction to mouse movement.

Files

bar5.g

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Data Area Resource Set
• Datagroup Resource Set
• 4 Level Resource Sets
• X Value Axis Resource Set
• Legend Resource Set
• Value Label Resource Set

Vertical Slider Graph

The Vertical Slider Graph displays one data value in 2D as a vertical bar. The height of the bar corresponds to the value of a data sample.

Animation Data

The graph’s entry points are as follows:

Data Entry

\$Widget/DataGroup/ValueY

The default data range is between 0 and 1.

The Slider Graph may also be used as a control widget, in which case it may be animated without any data by using its reaction to mouse movement.
**Files**

bar6.g

**Resource Sets**

This widget uses resources from the following sets:

- Title Resource Set
- Data Area Resource Set
- Datagroup Resource Set
- 4 Level Resource Sets
- Y Value Axis Resource Set
- Legend Resource Set
- Value Label Resource Set

**Horizontal 3D Slider Graph**

The Horizontal 3D Slider Graph is identical to the Horizontal Slider Graph described on page 80, except that the active element is a 3D bar. It displays one data value at a time, and the horizontal dimension of the bar corresponds to the value of the data.

**Animation Data**

The graph’s entry points are as follows:

**Data Entry**

$Widget/DataSample/ValueX

The default data range is between 0 and 1.

The 3D Slider Graph may also be used as a control widget, in which case it can be animated without any data by using its reaction to mouse movement.
File

bar110.g

Resources

ActiveElementGroup (GROUP): data sample
ActiveElement (POLYGON): one polygon of a data sample bar
ValueX (DDATA): the value defining a position of a data sample
Low (DDATA): low range of graph values
High (DDATA): high range of graph values
StartColor (GDATA): color for the minimum value
EndColor (GDATA): color for the maximum value

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• X Value Axis Resource Set
• Value Label Resource Set

Vertical 3D Slider Graph

The Vertical 3D Slider Graph is identical to the Vertical Slider Graph described on page 81, except that the active element is a 3D bar. It displays one data value at a time, and the horizontal dimension of the bar corresponds to the value of the data.

Animation Data

The graph’s entry points are as follows:

Data Entry

$Widget/DataSample/ValueY

The default data range is between 0 and 1. The widget may also be used as a control widget, in which case it can be animated without any data by using its reaction to mouse movement.
Files

bar111.g

Resources

ActiveElementGroup (GROUP): data sample
ActiveElement (POLYGON): one polygon of a data sample bar
ValueY (DDATA): the value defining a position of a data sample
Low (DDATA): low range of graph values
High (DDATA): high range of graph values
StartColor (GDATA): color for the minimum value
EndColor (GDATA): color for the maximum value

Resource Sets

This widget uses resources from the following sets:

• Title Resource Set
• Y Value Axis Resource Set
• Value Label Resource Set

Legend

The Legend Widget is an optional legend that may be pasted into any other graph widget.

Files

legend1.g

Resource Sets

This widget uses resources from the Legend Resource Set.

Bevels

The Bevel Widget is an optional beveled area to use for decoration. You can use it to add a border to another widget by placing that widget inside the Bevel Widget. You can constrain resources of the contained widget to the bevel lighting or color to provide a status indicator, as for a toggle button.

The Bevel Widget comes with a text object already inside it for ease of editing. You can use the text object to change the bevel colors easily. Delete or move the text object before inserting the next widget.
Files
bevel1.g

Resource Sets
This widget uses resources from the Bevel Resource Set

Gauge Widgets
The Gauge Widgets provide several graphical dials and gauges with different look and feel. They can all be used to display a value or to enter one using the mouse. Several gauges may be combined together with other widgets to create custom control panels.

Files
meter1.g
meter2.g
meter3.g
meter4.g
Left Quadrant Meter: meter5.g
Right Quadrant Meter: meter6.g
Left Quadrant Meter with Bevels: meter7.g
meter8.g
Semicircle Meter: meter9.g

Resource Sets
These widgets use resources from the following sets:

- Knob Resource Set
- Active Element Resource Set
- Unit Label Resource Set
- Scale Area Resource Set
- Center Marker Resource Set
- Value Axis Resource Set
- Value Label Group Resource Set
- Zooming Resource Set
Clock Widgets

The Clock Widgets are used to tell time. A clock widget may be used for output only, in which case you can delete the input handler, then use the ValueHour, ValueMin and ValueSec resources for moving the corresponding clock hands. Use the Hour, Min and Sec resources for supplying data to be displayed as numbers.

Files

time2.g
time3.g

Resource Sets

These widgets use resources from the following sets:

- Clock Resource Set
- Clock Hands Resource Set
- Scale Area Resource Set
- Center Marker Resource Set
- Value Axis Resource Set
- Zooming Resource Set

Stopwatch Widget

The Stopwatch Widget measures the time elapsed between pressing its Start and Stop button. The widget does not reset the time count until its Reset button is pressed. This allows the total time for several time intervals to be measured. The time is measured with the precision of one second.

The stopwatch widget displays the elapsed time in the following format:

<hours>:<minutes>:<seconds>

If the strings corresponding to the hours or minutes are deleted, the display shows the total number of expired minutes or seconds, without going back to zero at 59. That is, if the hour string is deleted, the clock will read “61:00” when 61 minutes have elapsed.

Files

time1.g

Resources

Meter (VIEWPORT): stopwatch viewport
Buttons (VIEWPORT): button panel
Resource Sets

This widget uses resources from the following sets:

- Clock Resource Set
- Active Element Resource Set
- Scale Area Resource Set
- Center Marker Resource Set
- Value Axis Resource Set
- Value Label Group Resource Set
- Zooming Resource Set
Thermometer Widget

The Thermometer Widget displays a temperature value in both C° and F°; allows the value to be changed using the mouse.

Files

gauge1.g

Resources

The Thermometer Widget has its own resource set, as follows:

FAxis (POLYGON, optional): F° axis
FAxisLabel (TEXT, optional): F° axis title
FMajorGroup (SERIES): F° axis major ticks
   FMajorTick (POLYGON): template major tick
   Factor (DDATA): number of major ticks
FMajorGroupOne (SERIES): F° axis minor ticks
   FMajorGroup (SERIES): template for one set of minor ticks
   FMajorTick (POLYGON): template minor tick
   Factor (DDATA): number of minor ticks
   LogType (DDATA): controls logarithmic positioning of minor ticks
   Factor (DDATA, CONSTR): number of major ticks
FLabelGroup (SERIES): F° axis major tick labels
   FLabel (TEXT): template major tick label
   Format (SDATA): format to use for displaying label values
   Low (DDATA): low range of F° values
   High (DDATA): high range of F° values
   Factor (DDATA, CONSTR): number of major ticks
CAxis (POLYGON, optional): C° axis
CAxisLabel (TEXT, optional): C° axis title
CMajorGroup (SERIES): C° major ticks
   CMajorTick (POLYGON): template major tick
   Factor (DDATA): number of C° major ticks
CMajorGroupOne (SERIES): C° minor ticks
   CMajorGroup (SERIES): template for one set of minor ticks
   CMajorTick (POLYGON): template minor tick
   Factor (DDATA): number of C° minor ticks
   LogType (DDATA): controls logarithmic positioning of minor ticks
   Factor (DDATA, CONSTR): number of C° major ticks
CLabelGroup (SERIES): C° major tick labels
   CLabel (TEXT): template major tick label
   Format (SDATA): format to use for displaying C° label values
   Low (DDATA, CONSTR): low range of C° values
   High (DDATA, CONSTR): high range of C° values
Factor (DDATA, CONSTR): number of ° major ticks

**Resource Sets**

This widget uses resources from the following sets:

- Slider Resource Set
- Active Element Resource Set
- Scale Area Resource Set
- Zooming Resource Set

**Joystick Widgets**

The Joystick Widgets provide several joystick controls to enter pairs of X and Y values using the mouse. The joysticks also provide a visual feedback displaying the current values.

**Files**

joystick1.g
joystick2.g
joystick3.g
joystick4.g
joystick5.g
joystick6.g

**Resources**

X1Label (TEXT, optional): X value title
Y1Label (TEXT, optional): Y value title

**Resource Sets**

These widgets use resources from the following sets:

- Slider Resource Set
- Active Element Resource Set
- X Value Axis Resource Set
- Y Value Axis Resource Set
- Scale Area Resource Set
- X Value Label Group Resource Set
- Y Value Label Group Resource Set
- Pointer Resource Set (optional)
• Zooming Resource Set

**Indicator Widgets**

The Indicator Widget is a bevelled area displaying a numerical value. The Color Indicator is rendered as an alarm light that changes its color.

**Files**

Numerical Indicator: indicator1.g, indicator3.g, indicator4.g  
Color Indicator: indicator2.g, indicator7.g, indicator8.g  
Vertical Equalizer Indicator: indicator3.g  
Horizontal Equalizer Indicator: indicator4.g

**Resources**

Element (POLYGON, optional): label highlight element

**Resource Sets**

These widgets use resources from the following sets:

• Bevel Resource Set (optional)  
• Value Label Group Resource Set (optional)

**Button Widgets**

The Push Button Widget is a button with a visual feedback on press. It may be added to any other widget to provide an interface for triggering some action.

The GLG Widget Library also includes a set of directional buttons. This is a set of four buttons indicating up, down, left and right movements. These buttons are intended for use with the joystick widgets, GIS maps, etc. Simply paste the whole set into the joystick's viewport. To disable dithering, set the patterns factor of the button’s viewport colortable to 0.

**Files**

Push Button: button1.g, button4.g, etc.  
Set of Directional Buttons: button5.g, button6.g, button7.g, button8.g, etc.  
Round Button: button3.g, etc.

**Resource Sets**

These widgets use resources from the following sets:

• Button Resource Set
• Active Element Resource Set (optional)
• Bevel Resource Set (optional)

Two-Position Switch Widgets

These widgets compose a set of discrete-position switches with various appearances. A switch toggles between two discrete positions on a mouse click, providing visual feedback. Several switches may be combined together with other widgets to create custom control panels.

Files

switch1.g
switch2.g
switch3.g
switch4.g
switch5.g
switch6.g

Resources

OnLabel (TEXT): On State label
OffLabel (TEXT): Off State label
OnTick (POLYGON, optional): On tick
OffTick (POLYGON, optional): Off tick

Resource Sets

These widgets use resources from the following sets:

• Button Resource Set
• Active Element Resource Set
• Zooming Resource Set

Multipositional Knob Widgets

These widgets compose a set of multipositional rotational switches with varying appearances. A switch reacts to the mouse clicks by moving its active element to indicate the discrete position closest to the mouse pick. Several switches may be combined together with other widgets to create custom control panels.
**Files**

switch7.g  
switch8.g  
knob9.g  
knob10.g

**Resources**

- LabelGroup (SERIES): position labels  
  - Label (TEXT): template label  
  - Factor (DDATA): number of labels  
- TickGroup (SERIES): position ticks  
  - Tick (MARKER): template tick  
  - Factor (DDATA, CONSTR): number of labels

**Resource Sets**

These widgets use resources from the following sets:

- Knob Resource Set
- Active Element Resource Set
- Unit Label Resource Set (Optional)
- Center Marker Resource Set
- Zooming Resource Set

**Multipositional Slider Widgets**

These widgets compose a set of multipositional linear switches with varying appearances. A switch reacts to the mouse clicks by moving its active element to the discrete position closest to the mouse pick. Several switches may be combined together with other widgets to create custom control panels.
Files

switch9.g
switch10.g

Resources

LabelGroup (SERIES): position labels
  Label (TEXT): template label
  Factor (DDATA): number of labels
TickGroup (SERIES): position ticks
  Tick (MARKER): template tick
  Factor (DDATA, CONSTR): number of labels

Resource Sets

These widgets use resources from the following sets:
  • Slider Resource Set
  • Active Element Resource Set
  • Unit Label Resource Set (optional)
  • Center Marker Resource Set
  • Zooming Resource Set

Knob Widgets

The Knob Widgets provide a set of knobs for entering a value using the mouse. The knobs can also provide a visual feedback for the current value. Several knobs may be combined together with other widgets to create custom control panels.

Files

knob1.g
knob2.g
knob3.g
knob4.g
knob5.g
knob6.g
knob7.g
knob8.g

Resources

MaxValueLabel (TEXT): maximum value label
MinValueLabel (TEXT): minimum value label
Resource Sets

These widgets use resources from the following sets:

- Knob Resource Set
- Active Element Resource Set
- Value Axis Resource Set
- Value Label Resource Set (optional)
- Unit Label Resource Set
- Center Marker Resource Set
- Zooming Resource Set

Slider Widgets

The Slider Widgets provide a set of one-dimensional linear sliders for entering a value using the mouse. The sliders can also provide a visual feedback for the current value. Several sliders may be combined together with other widgets to create custom control panels.

Files

slider1.g
slider2.g
slider3.g
slider4.g
slider5.g
slider6.g
slider7.g
slider8.g
slider9.g
slider10.g
slider11.g
slider12.g
slider13.g
slider14.g

Resources

MaxValueLabel (TEXT): maximum value label
MinValueLabel (TEXT): minimum value label
Resource Sets

These widgets use resources from the following sets:

- Slider Resource Set
- Active Element Resource Set
- Value Axis Resource Set
- Value Label Resource Set (optional)
- Unit Label Resource Set
- Zooming Resource Set

Palette Widgets

There are several Palette Widgets in the Library. The Color Palette displays a palette of colors, allowing the RGB color values to be selected with the mouse. To change the set of colors displayed in the palette, change the colortable of the Palette Widget’s viewport and adjust the RowFactor and ColumnFactor attributes of the palette to match the NumColors and NumGrades attributes of the viewport’s colortable. The palette’s drawing contains two palettes for both the horizontal and the vertical layouts. The RowFactor attribute should correspond to the NumColors for the vertical color palette and to the NumGrades for the horizontal one.

The other palette widgets include the Line Width, Line Type, Font Size and Font Type Palettes. The number of palette entries may be changed using the Factor attribute of the palette’s series object.

Files

Color Palette: palette1.g
Line Width Palette: palette2.g

Resource Sets

These widgets use resources from the Palette Resource Set.

Menu Widget

The Menu Widget displays a graphical menu object, presenting rows and columns of buttons and allowing a selection to be made with the mouse. The number of buttons may be changed by changing the values of the RowFactor and ColumnFactor attributes of the menu’s square series object.

Any attribute of the graphical appearance as well as the layout of buttons may be customized using the GLG Graphics Builder. Once the button layout is established, the menu manages it by changing the buttons’ positions and dimensions when the menu is resized.
The buttons of the menu can be made inactive by setting the \textit{DisableInput} attribute of the button’s viewport.

As with all the other GLG objects, buttons and menus may be made visible or invisible by simply changing the \textit{Visibility} attribute of their viewports. This may be used to pop up and then erase a menu. Use the menu viewport control points to define the menu geometry. The control points may be given a name in order to access them programmatically.

\textit{Files}

menu1.g

\textit{Resource Sets}

This widget uses resources from the Menu Resource Set

\textbf{Text Widget}

The Text Widget provides a viewport into which a user can enter a one line text string. There are text widgets for both the string and numerical text entries.

\textit{Files}

String text entry: text1.g
Integer number text entry: text2.g
Floating number text entry: text3.g

\textit{Resource Sets}

This widget uses resources from the Text Resource Set

\textbf{Font Browser Widget}

The Font Browser Widget allows a user to choose from the variety of fonts on a given X server. A “filter” for using wildcard characters as well as a menu of possible choices for every field of the X font specification are provided, simplifying the font selection. This widget is not available on the Windows platform where a standard font browser is available.

Any attribute of the graphical appearance as well as the layout of the Font Browser Widget may be customized using the GLG Graphics Builder.
Files

menu2.g

Resource Sets

This widget uses resources from the Font Browser Resource Set.
Chapter 3

GLG Widget Resource Sets

The resources of the widgets in the GLG Widget Library are arranged in groups for the convenience of their users. These groupings are only conventions established by the widget creators; only a few of them have a functional meaning.

For example, if it is noted that a widget has the resources from the “Datagroup Resource Set,” this means that all the resources named in that set are used in the resource hierarchy of that widget. Since many of the widgets are similar in nature, they share the same sets of resources.

The lists of resources in this chapter list the name of the resource, and then (in parentheses) the type of the object named by the resource. If the object has children objects, these are listed immediately below the parent. The description of the Datagroup Resource Set is shown below:

*DataGroup* (SERIES): The graph data samples.

  *DataSample* (POLYGON): The data sample template object.

    *Value* (DDATA): A value defining the height of a data sample.

    *Low* (DDATA): The lower limit of graph values.

    *High* (DDATA): The upper limit of graph values.

    *Truncate* (DDATA): Controls rendering values outside of the range.

  *Factor* (DDATA): The number of data samples.

  *ScrollType* (DDATA): The scrolling type of the graph.

  *EntryPoint* (DDATA): The entry point for updating graph data samples.

The *DataGroup* resource indicates a series containing four resources: *DataSample*, *Factor*, *ScrollType*, and *EntryPoint*. The *DataSample* object itself contains four resources: *Value*, *Low*, and *High*. The *Value* resource may thus be indicated by *DataGroup/DataSample/Value*. The *Value* resource is a double-precision scalar value, indicated by the “DDATA” in the following parentheses. (Geometrical values are indicated by GDATA, and string values by SDATA.) If a resource is optional or constrained to equal another value, these facts are indicated in the parentheses.

### Graph Widget Resource Sets

The resource sets used in the GLG Graph Widget Library are listed below:

**Title**

*Title* (TEXT): The title of the graph.

**Data Area**

*DataArea* (POLYGON): The area where data are displayed.

**Polar Data Area**
DataArea (ARC): The area where data are displayed.

3D Data Area

To specify a three-dimensional data area, three mutually perpendicular rectangles are used.

DataArea (POLYGON): The back data area. In its normal position, this is a rectangle in the XY plane.

DataAreaLeft (POLYGON): The left data area, a rectangle in the YZ plane.

DataAreaBottom (POLYGON): The bottom data area, a rectangle in the XZ plane.

Datagroup

DataGroup (SERIES): The graph data samples.

DataSample (POLYGON): The data sample template object.

Value (DDATA): A value defining the height of a data sample.

Low (DDATA): The lower limit of graph values.

High (DDATA): The upper limit range of graph values.

Truncate (DDATA): Controls rendering values outside of the range.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

EntryPoint (DDATA): The data entry point for updating graph data samples.

Multiset Datagroup

A Multiset datagroup is simply a series of Datagroups.

DataGroupOne (SERIES): The graph’s data sets.

DataGroup (SERIES): The graph data samples.

DataSample (POLYGON): The template polygon for the data samples.

Value (DDATA): A value defining the height of a data sample.

Low (DDATA): The lower limit of graph values.

High (DDATA): The upper limit of graph values.

Truncate (DDATA): Controls rendering values outside of the range.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

Inversed (DDATA): Controls the scrolling direction.

EntryPoint (DDATA): The data entry point for updating graph data samples within a single Datagroup.

Factor (DDATA): The number of datagroups in the series.

Persistent (DDATA): May be set to preserve attribute settings of individual lines.

EntryPoint (DDATA): The entry point for updating all datagroups.


**Packed Datagroup**

*DataGroup* (SERIES): The graph data samples.

*Pack* (SERIES): The template for one cluster (pack) of data samples.

*DataSample* (POLYGON): The data sample template polygon.

*Value* (DDATA): A value defining the height of a data sample.

*Low* (DDATA): The lower limit of graph values.

*High* (DDATA): The upper limit of graph values.

*Truncate* (DDATA): Controls rendering values outside of the range.

*Factor* (DDATA): The number of data samples in a pack.

*Persistent* (DDATA): May be set to preserve attribute settings of individual samples in a pack.

*Factor* (DDATA): The number of data sample clusters.

*ScrollType* (DDATA): The scrolling type of the graph.

*Inversed* (DDATA): Controls the scrolling direction.

*EntryPoint* (DDATA): The entry point for updating graph data samples.

**Line Datagroup**

These resources are used for simple line graphs.

*DataGroup* (POLYLINE): The graph data samples.

*Marker* (MARKER): A template object defining the marker to use at each data point.

*DataSample* (GDATA): The position of the marker.

*Value* (DDATA): A value defining the height of a data sample.

*Low* (DDATA): The lower limit of graph values.

*High* (DDATA): The upper limit of graph values.

*Truncate* (DDATA): Controls rendering values outside of the range.

*Polygon* (POLYGON): A template polygon defining the characteristics of the polyline line segments.

*Markers* (GROUP, optional): The created copies of the template marker. These will not appear unless the *Marker* resource is present.

*Polygons* (GROUP, optional): The created instances of line segments. These will not appear unless the *Polygon* resource is present.

*Points* (GROUP): The group of dynamically created points.

*DataSample<n>* (GDATA): The created instances of the data sample template.

*Angle* (DDATA, optional): The angle spanned by a polar polyline.

*Factor* (DDATA): The number of data samples on the polyline.

*ScrollType* (DDATA): The scrolling type of the graph.

*Inversed* (DDATA): Controls the scrolling direction.

*EntryPoint* (DDATA): The entry point for updating graph data samples.
Multiline Datagroup

These resources are used for graphs with more than one line of data drawn in the same data area.

DataGroupOne (SERIES): The graph’s data sets.
DataGroup (POLYLINE): The template for one data set.
Marker (MARKER): The template for the parent object’s markers.

DataSample (GDATA): The template point of the polyline.
Value (DDATA): A value defining the height of a data sample.
Low (DDATA): The lower limit of graph values.
High (DDATA): The upper limit of graph values.
Truncate (DDATA): Controls rendering values outside of the range.

Polygon (POLYGON): The template for the parent object’s line segments.
Markers (GROUP, optional): The dynamically created copies of the template marker. These will not appear unless the Marker resource is present.

Polygons (GROUP, optional): The created instances of line segments. These will not appear unless the Polygon resource is present.

Points (GROUP): The group of dynamically created points.

DataSample<\n> (GDATA): The dynamically created copies of the data sample template.

Angle (DDATA, optional): The angle spanned by a polar polyline.
Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.
Inversed (DDATA): Controls the scrolling direction.

EntryPoint (DDATA): The entry point for updating the data samples of one datagroup.

Factor (DDATA): The number of datagroups in the graph.
Persistent (DDATA): May be set to preserve attribute settings of individual lines.

EntryPoint (DDATA): The data entry point for updating the data samples of all the datagroups in the graph.

XY Line Datagroup

DataGroup (POLYLINE): The graph data samples.
Marker (MARKER): The template for the parent object’s markers.

DataSample (GDATA): The position of the marker.

XValue (DDATA): The X coordinate of a data sample.
XLow (DDATA): The lower limit of X values.
XHigh (DDATA): The upper limit of X values.

YValue (DDATA): The Y coordinate of a data sample.
YLow (DDATA): The lower limit of Y values.
YHigh (DDATA): The upper limit of Y values.

Truncate (DDATA): Controls rendering values outside of the range.

Polygon (POLYGON): The template for the parent object’s line segments.
Markers (GROUP, optional): The dynamically created copies of the template marker. These will not appear unless the Marker resource is present.

Polygons (GROUP, optional): The created instances of line segments. These will not appear
unless the Polygon resource is present.

Points (GROUP): The group of dynamically created points.

DataSample<\text{n}> (GDATA): The dynamically created copies of the data sample template.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

XEntryPoint (DDATA): The data entry point for the X values.

YEntryPoint (DDATA): The data entry point for the Y values.

**Multiline XY Datagroup**

Title (TEXT): The title of the graph.

DataArea (POLYGON): The area in which data are displayed.

DataGroupOne (SERIES): The series containing all the graph’s data sets.

DataGroup (POLYLINE): The template datagroup for the graph.

Marker (MARKER): The marker template for the polyline.

DataSample (GDATA): The position of the marker.

XValue (DDATA): The X coordinate of a data sample.

XLow (DDATA): The lower limit of X values.

XHigh (DDATA): The upper limit of X values.

YValue (DDATA): The Y coordinate of a data sample.

YLow (DDATA): The lower limit of Y values.

YHigh (DDATA): The upper limit of Y values.

Truncate (DDATA): Controls rendering values outside of the range.

Polygon (POLYGON): The template for the parent object’s line segments.

Markers (GROUP, optional): The dynamically created copies of the template marker. These will not appear unless the Marker resource is present.

Polygons (GROUP, optional): The created instances of line segments. These will not appear unless the Polygon resource is present.

Points (GROUP): The group of dynamically created points.

DataSample<\text{n}> (GDATA): The dynamically created copies of the data sample template.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

XEntryPoint (DDATA): The data entry point for the X values of one set.

YEntryPoint (DDATA): The data entry point for the Y values of one set.

Factor (DDATA): The number of datagroups in the graph.

Persistent (DDATA): May be set to preserve attribute settings of individual lines.

XEntryPoint (DDATA): The data entry point for the X values of all sets.

YEntryPoint (DDATA): The data entry point for the Y values of all sets.

**Filled Step Datagroup**

DataGroup (SERIES): The graph data samples.

DataSample (GROUP): The template group.

Fill (POLYGON): The filled part of a data sample.
*Edge* (POLYGON): The edge of a data sample.

*Value* (DDATA): A value defining the height of a data sample.

*Low* (DDATA): The lower limit of graph values.

*High* (DDATA): The upper limit of graph values.

*Truncate* (DDATA): Controls rendering values outside of the range.

*Factor* (DDATA): The number of data samples.

*ScrollType* (DDATA): The scrolling type of the graph.

*Inversed* (DDATA): Controls the scrolling direction.

*EntryPoint* (DDATA): The entry point for updating graph data samples.

**Multiline Step Datagroup**

*DataGroupOne* (SERIES): The series of the data sets that compose the graph.

*DataGroup* (SERIES): The template data set for the graph.

*DataSample* (POLYGON): The data sample template object.

*Value* (DDATA): A value defining the height of a data sample.

*Low* (DDATA): The lower limit of graph values.

*High* (DDATA): The upper limit of graph values.

*Truncate* (DDATA): Controls rendering values outside of the range.

*Factor* (DDATA): The number of data samples.

*ScrollType* (DDATA): The scrolling type of the graph.

*Inversed* (DDATA): Controls the scrolling direction.

*EntryPoint* (DDATA): The data entry point for updating samples of one datagroup.

*Factor* (DDATA): The number of datagroups in the graph.

*Persistent* (DDATA): May be set to preserve attribute settings of individual lines.

*EntryPoint* (DDATA): The data entry point for updating samples of all the datagroups.

**Filled Multiline Step Datagroup**

*DataGroupOne* (SERIES): The collection of datagroups that make up the graph.

*DataGroup* (SERIES): The template datagroup.

*DataSample* (GROUP): The data sample template object.

*Fill* (POLYGON): The filled part of a data sample.

*Edge* (POLYGON): The edge of a data sample.

*Value* (DDATA): A value defining the height of a data sample.

*Low* (DDATA): The lower limit of graph values.

*High* (DDATA): The upper limit of graph values.

*Truncate* (DDATA): Controls rendering values outside of the range.

*Factor* (DDATA): The number of data samples.

*ScrollType* (DDATA): The scrolling type of the graph.

*Inversed* (DDATA): Controls the scrolling direction.

*EntryPoint* (DDATA): The data entry point for updating samples of one datagroup.

*Factor* (DDATA): The number of datagroups in the graph.

*Persistent* (DDATA): May be set to preserve attribute settings of individual lines.
EntryPoint (DDATA): The data entry point for updating samples of all the datagroups.

**Pie Datagroup**

DataGroup (SERIES): The graph data samples.

DataSample (GROUP): The data sample template object.

Sector (ARC): The arc segment of a pie chart.

Label (TEXT): A label showing the value of that segment.

Format (SDATA): The format (C-style) for displaying the Label value.

SplitFactor (DDATA): When set to 1, the pie segment will be drawn separated from the rest of the pie. When set to zero, the pie will contact its neighbors.

Factor (DDATA): The number of data samples (pieces of the pie).

EntryPoint (DDATA): The entry point for updating data samples.

**Polar XY Line Datagroup**

DataGroup (POLYLINE): The graph data samples.

Marker (MARKER): The marker to use for the series template.

DataSample (GDATA): The position of the marker.

RadValue (DDATA): The data point’s distance from the center of the graph.

AngleValue (DDATA): The angle described by the data point, the center of the graph, and the X axis.

Low (DDATA): The lower limit of the data point’s radius value.

High (DDATA): The upper limit of the data point’s radius value.

Truncate (DDATA): Controls rendering values outside of the range.

Polygon (POLYGON): The polyline’s template polygon.

Markers (GROUP, optional): The dynamically created copies of the template marker. These will not appear unless the Marker resource is present.

Polygons (GROUP, optional): The created instances of line segments. These will not appear unless the Polygon resource is present.

Points (GROUP): The group of dynamically created points.

DataSample<n> (GDATA): The dynamically created copies of the data sample template.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

AngleEntryPoint (DDATA): The entry point for updating the AngleValue resource of a the collection of data samples

RadiusEntryPoint (DDATA): The entry point for updating the RadValue resource of the collection of data samples.

**Multiline Polar XY Datagroup**

DataGroupOne (SERIES): The collection of polylines that make up the graph.

DataGroup (POLYLINE): The polyline template.

Marker (MARKER): The template for the parent object’s markers.

DataSample (GDATA): The position of the marker.
RadValue (DDATA): The data point’s distance from the center of the graph.

AngleValue (DDATA): The angle described by the data point, the center of the graph, and the X axis.

Low (DDATA): The lower limit of the data point’s radius value.

High (DDATA): The upper limit of the data point’s radius value.

Truncate (DDATA): Controls rendering values outside of the range.

Polygon (POLYGON): The template for the parent object’s line segments.

Markers (GROUP, optional): The dynamically created copies of the template marker. These will not appear unless the Marker resource is present.

Polygons (GROUP, optional): The created instances of line segments. These will not appear unless the Polygon resource is present.

Points (GROUP): The group of dynamically created points.

DataSample<n> (GDATA): The dynamically created copies of the data sample template.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

AngleEntryPoint (DDATA): The entry point for updating the AngleValue resource of one set of data samples.

RadiusEntryPoint (DDATA): The entry point for updating the RadValue resource of one set of data samples.

Factor (DDATA): The number of datagroups (polylines) in the graph.

Persistent (DDATA): May be set to preserve attribute settings of individual lines.

AngleEntryPoint (DDATA): The angle entry point for updating all of the data samples.

RadiusEntryPoint (DDATA): The radius entry point for updating all of the data samples.

3D Datagroup

DataGroup (SERIES): The graph data samples.

DataSample (GROUP): The data sample template object. This is a collection of several polygons arranged in three-dimensional space.

Element (POLYGON): The template polygon whose attributes are constrained to the attributes of all the polygons in the DataSample group.

Value (DDATA): A value defining the height of a data sample.

Low (DDATA): The lower limit of graph values.

High (DDATA): The upper limit of graph values.

Truncate (DDATA): Controls rendering values outside of the range.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

Inversed (DDATA): Controls the scrolling direction.

EntryPoint (DDATA): The entry point for updating graph data samples.

Multiset 3D Datagroup

DataGroupOne (SERIES): The collection of the graph’s data sets.

DataGroup (SERIES): A template datagroup.
DataSample (GROUP): The data sample template object. This is a collection of several polygons arranged in three-dimensional space.

Element (POLYGON): The template polygon whose attributes are constrained to the attributes of all the polygons in the DataSample group.

Value (DDATA): A value defining the height of a data sample.

Low (DDATA): The lower limit of graph values.

High (DDATA): The upper limit of graph values.

Truncate (DDATA): Controls rendering values outside of the range.

Factor (DDATA): The number of data samples.

ScrollType (DDATA): The scrolling type of the graph.

Inversed (DDATA): Controls the scrolling direction.

EntryPoint (DDATA): The entry point for updating one data set.

Factor (DDATA): The number of datagroups in the graph.

EntryPoint (DDATA): The entry point for updating all data sets.

3D Packed Datagroup

DataGroup (SERIES): The graph data samples.

Pack (SERIES): The template for one cluster (pack) of data samples.

DataSample (GROUP): The data sample template object.

Element (POLYGON): One polygon component of a composite data sample object. The attributes of all the component polygons are constrained to attributes of this element.

Value (DDATA): A value defining the height of a data sample.

Low (DDATA): The lower limit of graph values.

High (DDATA): The upper limit of graph values.

Truncate (DDATA): Controls rendering values outside of the range.

Factor (DDATA): The number of data samples in a pack.

Factor (DDATA): The number of data sample clusters.

ScrollType (DDATA): The scrolling type of the graph.

Inversed (DDATA): Controls the scrolling direction.

EntryPoint (DDATA): The entry point for updating graph data samples.

Multiset 3D Packed Datagroup

DataGroupOne (SERIES): The collection of data groups that make up the graph.

DataGroup (SERIES): The collection of data samples making up one component of the graph.

Pack (SERIES): The template for one cluster (pack) of data samples.

DataSample (GROUP): The data sample template object.
**Element** (POLYGON): One polygon component of a composite data sample object. The attributes of all the component polygons are constrained to the attributes of this element.

**Value** (DDATA): A value defining the height of a data sample.

**Low** (DDATA): The lower limit of graph values.

**High** (DDATA): The upper limit of graph values.

**Truncate** (DDATA): Controls rendering values outside of the range.

**Factor** (DDATA): The number of data samples in a pack.

**ScrollType** (DDATA): The number of data sample clusters.

**Inversed** (DDATA): The scrolling type of the graph.

**EntryPoint** (DDATA): The scrolling direction.

**EntryPoint** (DDATA): The entry point for updating graph data samples.

**Factor** (DDATA): The number of datagroups in the graph.

**EntryPoint** (DDATA): The data entry point for updating all data sets.

---

**Surface Datagroup**

**DataGroup** (SERIES): The graph data samples.

**DataSample** (GDATA): The data sample for a surface graph is the 3-D coordinates of a vertex point of the surface.

**Value** (DDATA): A value defining the height (Y-coordinate) of a data sample.

**Low** (DDATA): The lower limit of graph values.

**High** (DDATA): The upper limit of graph values.

**Truncate** (DDATA): Controls rendering values outside of the range.

**Polygon** (POLYGON): The template polygon for the surface facets. The colors and line characteristics are copied from this template.

**Polygons** (GROUP): The group that contains the dynamically created facets of the surface.

**Polygon<n>** (POLYGON): Each facet of the surface is individually addressable with this resource. Note that, as with all GLG indexes, this index starts at zero.

**Marker** (MARKER): The template for the markers that appear at the surface vertices.

**Markers** (GROUP): The group containing the dynamically markers (if markers are enabled).

**Marker<n>** (MARKER): Each marker of a surface is individually addressable with this resource.

**Points** (GROUP): The group containing the positions of each vertex in the graph surface.

**DataSample<n>** (GDATA): An instance of the data sample with index <n> (zero based).

**Value** (DDATA): A value defining the height of a data sample.

**Low** (DDATA): The lower limit of graph values.

**High** (DDATA): The upper limit of graph values.

**Rows** (DDATA): The number of rows of data samples.

**Columns** (DDATA): The number of columns of data samples.

**EntryPoint** (DDATA): The entry point for updating graph data samples.
**Level**

Level objects are used to draw horizontal lines on a bar graph. They will often be used to indicate some externally defined data level.

*LevelObjectGroup* (SERIES): This is the series of level lines for the graph.
*LevelObject* (POLYGON): The template level line provides the line characteristics for the line.
   *Value* (DDATA): The height of a level line
   *Low* (DDATA, constrained): The lower limit of graph values.
   *High* (DDATA, constrained): The upper limit of graph values.
*Factor* (DDATA): The number of level lines in the graph.
*LevelObject<n>* (POLYGON): The n<sup>th</sup> level line.
   *Value* (DDATA): The height of a level line.
   *Low* (DDATA, constrained): The lower limit of graph values.
   *High* (DDATA, constrained): The upper limit of graph values.

**Symmetrical Level**

These level lines are used to mark pairs of level lines that are symmetrical around a given level.

*LevelObjectOne* (SERIES): A series of level line pairs.
*LevelObjectGroup* (GROUP): The template for a pair of level lines.
*LevelObject1* (POLYGON): The upper level line.
   *Value* (DDATA): The height of the upper level line.
   *High* (DDATA, constrained): The upper limit of graph values.
*LevelObject2* (POLYGON): The lower level line.
   *Value* (DDATA, constrained): The height of the lower level line.
   *High* (DDATA, constrained): The upper limit of graph values.
*Factor* (DDATA): number of level line sets

**Polar Level**

These lines are used on polar graphs.

*LevelObjectGroup* (SERIES): A series of level lines.
*LevelObject* (ARC): The template level line. The line’s characteristics, like weight and type, are taken from this template.
   *Value* (DDATA): The radius of a level line.
   *Low* (DDATA, constrained): The lower limit of graph values.
   *High* (DDATA, constrained): The upper limit of graph values.
*Factor* (DDATA): The number of level lines.
**X Level**

*XLevelObjectGroup* (SERIES): A series of vertical (X) level lines.

*XLevelObject* (POLYGON): The template X level line.

Value (DDATA): The X-coordinate of the line.

*XLow* (DDATA, constrained): The lower limit of X values in the graph.

*XHigh* (DDATA, constrained): The upper limit of X values in the graph.

*Factor* (DDATA): The number of X level lines in the graph.

**Y Level**

*YLevelObjectGroup* (SERIES): A series of horizontal (Y) level lines.

*YLevelObject* (POLYGON): The template Y level line.

Value (DDATA): The Y-coordinate of the line.

*YLow* (DDATA, constrained): The lower limit of Y values in the graph.

*YHigh* (DDATA, constrained): The upper limit of Y values in the graph.

*Factor* (DDATA): The number of Y level lines.

**Status Object**

A status object is used to point to the update position for a wrapped graph.

*StatusObject* (POLYGON): Shows current update state. The origin of the polygon object is placed at the axis intercept of the current update position.

**Value Axis**

This is the general-purpose axis resource group. More specific resource groups (e.g. X-axis, Y-axis, X-axis for data values, X-axis for time values, and so on) are described below.

*Axis* (POLYGON): The line representing the axis itself.

*AxisLabel* (TEXT, optional): The label for the axis.

*Base* (POLYGON, optional): A polygon defining the positioning of axis ticks. The position of this polygon’s first point coincides with the first tick on the axis.

*MajorGroup* (SERIES): The series of major ticks.

*MajorTick* (POLYGON): A template for the axis major ticks.

*Factor* (DDATA): The number of major ticks.

*MinorGroupOne* (SERIES, optional): The series that contains all the axis minor ticks.

*MinorGroup* (SERIES): The template for one set of minor ticks (between two major ticks).

*MinorTick* (POLYGON): The template for an individual minor tick.

*Factor* (DDATA): The number of minor ticks between major ticks.

*LogType* (DDATA): Controls logarithmic positioning of minor ticks.

*Factor* (DDATA, constrained): The number of X major ticks.

*LabelGroup* (SERIES): The series of major tick labels.

*Label* (TEXT): A template for the major tick labels.

*Format* (SDATA): The C-style format to use for displaying label values.
Low (DDATA, constrained): The lower limit of the graph’s data.
High (DDATA, constrained): The upper limit of the graph’s data.
Factor (DDATA, constrained): The number of major ticks.

**X Axis**

These resources control the appearance of a graph’s X axis. Note that the axis labels in this group and other similar ones are arranged in a series with an enclosed history object. (For more about the history object, see [*].) This arrangement is provided as a convenient way to set individual attributes of series members, and has no functional relation to the graph’s EntryPoint resource.

*XAxis* (POLYGON): The line representing the X axis itself.

*XMajorGroup* (SERIES): The series of X major ticks.

*XMajorTick* (POLYGON): The template for the X major ticks.

Factor (DDATA): The number of X major ticks.

*XMinorGroupOne* (SERIES): The series of X minor ticks.

*XMinorGroup* (SERIES): template for one set of X minor ticks

*XMinorTick* (POLYGON): The template for the X minor ticks.

Factor (DDATA): The number of X minor ticks.

Factor (DDATA, constrained): The number of X major ticks.

*XGridGroup* (SERIES, optional): The X axis grid lines.

*XGrid* (POLYGON): The template grid line.

Factor (DDATA, constrained): The number of X major ticks.

*XLabelGroup* (SERIES, optional): The series of major tick labels.

*XLabel* (TEXT): A template major tick label.

Factor (DDATA, constrained): The number of major ticks.

EntryPoint (SDATA, optional): Entry point for simplified setting of label strings.

**Y Axis**

*YAxis* (POLYGON): The line representing the Y axis itself.

*YMajorGroup* (SERIES): The series of Y major ticks.

*YMajorTick* (POLYGON): The template for the Y major ticks.

Factor (DDATA): The number of Y major ticks.

*YMinorGroupOne* (SERIES): The series of Y minor ticks.

*YMinorGroup* (SERIES): template for one set of Y minor ticks

*YMinorTick* (POLYGON): The template for the Y minor ticks.

Factor (DDATA): The number of Y minor ticks.

Factor (DDATA, constrained): The number of Y major ticks.

*YGridGroup* (SERIES, optional): The series of Y axis grid lines.

*YGrid* (POLYGON): The template grid line.

Factor (DDATA, constrained): The number of Y major ticks.

*YLabelGroup* (SERIES, optional): The series of major tick labels.

*YLabel* (TEXT): The template major tick label.

Factor (DDATA, constrained): The number of major ticks.
EntryPoint (SDATA, optional): Entry point for simplified setting of label strings.

**Z Axis**

ZAxis (POLYGON): The line representing the Z axis itself.
ZMajorGroup (SERIES): The series of Z major ticks.
ZMajorTick (POLYGON): The template for the Z major ticks.
Factor (DDATA): The number of Z major ticks.
ZMinorGroupOne (SERIES, optional): The series of Z minor ticks.
ZMinorGroup (SERIES): The template for one set of Z minor ticks.
ZMinorTick (POLYGON): The template for a single Z minor tick.
Factor (DDATA): The number of Z minor ticks.
Factor (DDATA, constrained): The number of Z major ticks.
ZLabelGroup (SERIES, optional): The set of major tick labels.
ZLabel (TEXT): A template major tick label.
Factor (DDATA, constrained): The number of major ticks.
EntryPoint (SDATA, optional): Entry point for simplified setting of label strings.

**X Value Axis**

XAxis (POLYGON, optional): The line representing the X axis itself.
XAxisLabel (TEXT, optional): The title for the X axis.
XMajorGroup (SERIES): The series of X major ticks.
XMajorTick (POLYGON): The template for the X major ticks.
Factor (DDATA): The number of X major ticks.
XMinorGroupOne (SERIES): The series of X minor ticks.
XMinorGroup (SERIES): template for one set of X minor ticks
XMinorTick (POLYGON): The template for the X minor ticks.
Factor (DDATA): The number of X minor ticks.
LogType (DDATA): Controls logarithmic positioning of minor ticks.
Factor (DDATA, constrained): The number of X major ticks.
XLabelGroup (SERIES): X major tick labels
XLabel (TEXT): The template for the X major ticks. label
Format (SDATA): The C-style format to use for displaying label values.
Low (DDATA, constrained): The lower limit of the graph’s data.
High (DDATA, constrained): The upper limit of the graph’s data.
Factor (DDATA, constrained): The number of X major ticks.
XGridGroup (SERIES): The series of X axis grid lines.
XGrid (POLYGON): The template grid line.
Factor (DDATA, constrained): The number of X major ticks.
Chapter 3  GLG Widget Resource Sets

Y Value Axis

YAxis (POLYGON, optional): The line representing the Y axis itself.
YAxisLabel (TEXT, optional): The title for the Y axis.
YMajorGroup (SERIES): The series of Y major ticks.
YMajorTick (POLYGON): The template for the Y major ticks.
Factor (DDATA): The number of Y major ticks.
YMinorGroupOne (SERIES): The series of Y minor ticks.
YMinorGroup (SERIES): template for one set of Y minor ticks
YMInorTick (POLYGON): The template for the Y minor ticks.
Factor (DDATA, constrained): The number of Y major ticks.
YLabelGroup (SERIES): Y major tick labels
YLabel (TEXT): The template for the Y major ticks. label
Format (SDATA): The C-style format to use for displaying label values.
Low (DDATA, constrained): The lower limit of the graph’s data.
High (DDATA, constrained): The upper limit of the graph’s data.
Factor (DDATA, constrained): The number of Y major ticks.
YGridGroup (SERIES): The series of Y axis grid lines.
YGrid (POLYGON): The template grid line.
Factor (DDDATA, constrained): The number of Y major ticks.

Y1 Value Axis

An auxiliary set of ticks and labels. It is the same as Y Value Axis Resource Set except that all the resources at the top level of the resource hierarchy that begin with “Y” are replaced with names that begin with “Y1” instead.

Y Value Multiaxis

This set of several Y-axes is designed to be used in graphs that display several different sets of data at the same time.

YAxisGroupOne (SERIES): A series containing several vertical axes.
Factor (DDATA): The number of Y axes displayed.
YAxisGroup (GROUP): The template for one vertical axis with ticks and labels.
YAxis (POLYGON): The line representing the Y axis itself.
YAxisLabel (TEXT): The title for the Y axis.
YMajorGroup (SERIES): The series of Y major ticks.
YMInorTick (POLYGON): The template for the Y major ticks.
Factor (DDATA): The number of Y major ticks.
YMinorGroupOne (SERIES): The series of Y minor ticks.
YMinorGroup (SERIES): template for one set of Y minor ticks
YMinorTick (POLYGON): The template for the Y minor ticks.
Factor (DDATA): The number of Y minor ticks.
LogType (DDATA): controls logarithmic positioning of minor ticks
Factor (DDATA, constrained): The number of Y major ticks.
YLabelGroup (SERIES): Y major tick labels
YLabel (TEXT): The template for the Y major ticks. label
Format (SDATA): The C-style format to use for displaying label values.
Low (DDATA, constrained): The lower limit of the graph’s data.
High (DDATA, constrained): The upper limit of the graph’s data.
Factor (DDATA, constrained): The number of Y major ticks.
YGridGroup (SERIES): The Y axis grid lines.
YGrid (POLYGON): The template grid line.
Factor (DDATA, constrained): The number of Y major ticks.

Z Value Axis

ZAxis (POLYGON, optional): The line representing the Z axis itself.
ZAxisLabel (TEXT): The title for the Z axis.
ZMajorGroup (SERIES): The series of Z major ticks.
ZMajorTick (POLYGON): The template for the Z major ticks.
Factor (DDATA): The number of Z major ticks.
ZMinorGroupOne (SERIES): The series of Z minor ticks.
ZMinorGroup (SERIES): The template for one set of Z minor ticks.
ZMinorTick (POLYGON): The template for a single Z minor tick.
Factor (DDATA): The number of Z minor ticks.
Factor (DDATA, constrained): The number of Z major ticks.
ZLabelGroup (SERIES): The Z major tick labels.
ZLabel (TEXT): The template for the Z major ticks.
Format (SDATA): format to use for displaying label values
Low (DDATA): The lower limit of the Z axis.
High (DDATA): The upper limit of the Z axis.
Factor (DDATA, constrained): The number of Z major ticks.

X Time Axis

XAxis (POLYGON, optional): The line representing the X axis itself.
XAxisLabel (TEXT): The title for the X axis.
XMajorGroup (SERIES): The series of X major ticks.
XMajorTick (POLYGON): The template for the X major ticks.
Factor (DDATA): The number of X major ticks.
ScrollType (DDATA, constrained): The scrolling type of the graph.
Inversed (DDATA, constrained): Controls the direction of the graph’s scrolling behavior.
XMinorGroupOne (SERIES): The series of X minor ticks.
XMinorGroup (SERIES): The template for one set of X minor ticks.
Chapter 3 GLG Widget Resource Sets

XMinorTick (POLYGON): The template for the X minor ticks.

Factor (DDATA): The number of X minor ticks.

Factor (DDATA, constrained): The number of X major ticks.

XLabelGroup (SERIES): X major tick labels

XLabel (TEXT): The template for the X major ticks labels.

Factor (DDATA, constrained): The number of X major ticks.

MinorFactor (DDATA, constrained): The number of X minor ticks.

ScrollType (DATA, constrained): The scrolling type of the graph.

Inversed (DDATA, constrained): Controls the direction of the labels’ scrolling behavior.

EntryPoint (SDATA, optional): The entry point for setting labels’ strings.

XGridGroup (SERIES): The series of X axis grid lines.

XGrid (POLYGON): The template grid line.

Factor (DDATA, constrained): The number of X major ticks.

MinorFactor (DDATA, constrained): The number of X minor ticks.

ScrollType (DATA, constrained): The scrolling type used for the grid lines.

Inversed (DDATA, constrained): Controls the direction of the grid lines’ scrolling behavior.

Y Time Axis

YAxis (POLYGON, optional): The line representing the Y axis itself.

YAxisLabel (TEXT): The title for the Y axis.

YMajorGroup (SERIES): The series of Y major ticks.

YMajorTick (POLYGON): The template for the Y major ticks.

Factor (DDATA): The number of Y major ticks.

ScrollType (DATA, constrained): The scrolling type of the graph.

Inversed (DDATA, constrained): Controls the direction of the axis ticks’ scrolling behavior.

YMinorGroupOne (SERIES): The series of Y minor ticks.

YMinorGroup (SERIES): template for one set of Y minor ticks

YMinorTick (POLYGON): The template for the Y minor ticks.

Factor (DDATA): The number of Y minor ticks.

Factor (DDATA, constrained): The number of Y major ticks.

YLabelGroup (SERIES): Y major tick labels

YLabel (TEXT): The template for the Y major ticks label

Factor (DDATA, constrained): The number of Y major ticks.

MinorFactor (DDATA, constrained): The number of Y minor ticks.

ScrollType (DATA, constrained): The scrolling type of the graph.

Inversed (DDATA, constrained): Controls the direction of the labels’ scrolling behavior.

EntryPoint (SDATA, optional): The data entry point provided for simplified setting of label strings.

YGridGroup (SERIES): The Y axis grid lines.

YGrid (POLYGON): The template grid line.

Factor (DDATA, constrained): The number of Y major ticks.

MinorFactor (DDATA, constrained): The number of Y minor ticks.

ScrollType (DATA, constrained): The scrolling type of the graph.

Inversed (DDATA, constrained): Controls the direction of the labels’ scrolling behavior.
**Tangent Ticks**

The tangent and radial groups are used for polar graph grids. The tangent axes are the ones that go in a circle around the origin, while the radial axes radiate outwards from the origin.

*TangentMajorGroup* (SERIES): The series of tangent major ticks.

*Factor* (DDATA): The number of tangent major ticks.

*TangentMinorGroupOne* (SERIES): The series of tangent minor ticks.

*TangentMinorGroup* (SERIES): The template for one set of tangent minor ticks (to be drawn between an adjacent pair of major ticks).

*Factor* (DDATA): The number of tangent minor ticks between adjacent major ticks.

*Factor* (DDATA, constrained): The number of tangent major ticks.

**Tangent Labels**

*TangentLabelGroup* (SERIES): The series of tangent labels.

*TangentLabel* (TEXT): The template tangent label.

*Format* (SDATA): The C-style format to use for displaying label values.

*Low* (DDATA, constrained): The lower limit of the label values.

*High* (DDATA, constrained): The upper limit of the label values.

*Factor* (DDATA, constrained): The number of tangent labels.

**Tangent Grid**

*TangentGridGroup* (SERIES): A series of tangent grid lines.

*TangentGrid* (POLYGON): The template grid line.

*Factor* (DDATA): The number of tangent grid lines.

**Radial Grid**

*RadialGridGroup* (SERIES): A series of radial grid lines.

*RadialGrid* (ARC): The template grid line.

*Factor* (DDATA): The number of radial grids.

**Legend**

*LegendObject* (VIEWPORT): legend viewport

*LegendGroup* (GROUP): a group of legend elements

*Legend* (GROUP): template legend element

*LegendBox* (POLYGON, optional): legend box

*LegendLine* (POLYGON, optional): legend line

*LegendMarker1* (POLYGON, optional): legend line marker

*LegendMarker2* (POLYGON, optional): legend line marker

*LegendLabel* (TEXT): legend text object
**Bevel**

*Body (GROUP):* set of bevels  
  *Element (POLYGON, optional):* bevel polygon  
  *BevelSize (DDATA, optional):* the width of button bevels in pixels  
*Insert (VIEWPORT, optional):* a viewport to place objects inside bevels  
  *FrameSize (DDATA):* the width of the gap between the bevels and the Insert viewport.  
  *InverseShading (DDATA, optional):* inverses bevel shading when set to 1.

**View Resources**

These resource sets control the view a user has of the widget.

**Viewing**

The angles used to specify the viewing transformation are defined relative to the “main view.” This view of a drawing has the X-axis pointing to the right, the Y-axis pointing up, and the Z-axis pointing directly at the viewer.

*PanCenter (GDATA):* This point is drawn in the center of the top-level viewport.  
*Scale (DDATA):* The zoom factor for a drawing. A value of 1.0 indicates that the corners of the top-level viewport will have the coordinates (-1000 -1000) and (1000 1000).  
*XAngle (DDATA):* The angle of rotation around the X axis from the main view.  
*YAngle (DDATA):* The angle of rotation around the Y axis from the main view.  
*ZAngle (DDATA):* The angle of rotation around the Z axis from the main view.  
*ShearFactor (DDATA):* The degree of shear along the Z axis.  
*ShearX (DDATA):* The proportion of the specified shear in the X direction.  
*ShearY (DDATA):* The proportion of the specified shear in the Y direction.

**Zooming**

*PanCenter (GDATA):* This point is drawn in the center of the top-level viewport.  
*Scale (DDATA):* The zoom factor for a drawing. A value of 1.0 indicates that the corners of the top-level viewport will have the coordinates (-1000 -1000) and (1000 1000).
Control And Interface Widget Common Resources

This section lists the sets of resources common for control and interface widgets. For lists of the resources used by the widget input handlers, and for more information about the input widgets, see also the Input Widgets, Dials and Meters section of the Using the GLG Widgets chapter.

Slider

ValueX (DDATA, optional): The slider’s X value.

OutValueX (DDATA, optional): The slider’s X output value. You may constrain other objects to this attribute.

ValueY (DDATA, optional): The slider’s Y value.

OutValueY (DDATA, optional): The slider’s Y output value. You may constrain other objects to this resource.

Value (DDATA, optional): The native slider’s input value.

OutValue (DDATA, optional): The native slider’s X output value. You may constrain other objects to this attribute.

Low (DDATA, optional): The lower limit of the slider’s output value.

High (DDATA, optional): The upper limit of the slider’s output value.

LowX (DDATA, optional): The lower limit of the 2D slider’s X value.

HighX (DDATA, optional): The upper limit of the 2D slider’s X value.

LowY (DDATA, optional): The lower limit of the 2D slider’s Y value.

HighY (DDATA, optional): The upper limit of the 2D slider’s Y value.

Granularity (DDATA, optional): If this resource is present and set to a non-zero value, the slider is limited to this number of allowed positions.

DisableMotion (DDATA, optional): If this resource is present and set to a non-zero value, the slider’s reaction to MotionNotify events is disabled.

Start (MARKER, optional): A marker placed at the lower limit of the slider’s range.

XEnd (MARKER, optional): A marker placed at the upper limit of the slider’s range of X values.

YEnd (MARKER, optional): A marker placed at the upper limit of the slider’s range of Y values.

ActiveArea (POLYGON, optional): The screen cursor must be within this polygon for the slider to react to user input. This resource is not usually present in the default slider. Instead, the ActiveAreaSpare polygon (invisible by default) is defined. It may be renamed ActiveArea and positioned as desired to make only some area of the slider sensitive to mouse events.

Plane (POLYGON, optional): The slider pointer slides on the plane defined with this polygon. The points of the polygon must be coplanar.

Increase (VIEWPORT, optional): An “increase” button for a one-dimensional slider.

Decrease (VIEWPORT, optional): A “decrease” button for a one-dimensional slider.

Left (VIEWPORT, optional): A directional button for a two-dimensional slider.

Right (VIEWPORT, optional): A directional button for a two-dimensional slider.
Up (VIEWPORT, optional): A directional button for a two-dimensional slider.
Down (VIEWPORT, optional): A directional button for a two-dimensional slider.
Increment (DDATA, optional): If the slider viewport contains motion buttons, each click of the button moves the slider by the amount specified by this resource. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the slider. If this resource is not present, the default increment of motion is 0.1 times the total slider range.

Knob

Value (DDATA, mandatory): The knob’s value.
OutValue (DDATA, optional): The knob’s output value. You may constrain other objects to this resource.
Low (DDATA, optional): The lower limit of the knob’s value.
High (DDATA, optional): The upper limit of the knob’s value.
Granularity (DDATA, optional): If this resource is present and set to a non-zero value, it specifies the number of allowed positions for the knob.
DisableMotion (DDATA, optional): If this resource is present and set to a non-zero value, the knob’s reaction to MotionNotify events is disabled.
StartAngle (DDATA, optional): The Value resource is at its lower limit when the knob is at this angle.
EndAngle (DDATA, optional): The Value resource is at its upper limit when the knob is at this angle.
Center (MARKER, optional): A marker placed at the rotation center of the knob.
ActiveArea (POLYGON, optional): The screen cursor must be within this polygon for the knob to react to user input. This resource is not usually present in the default knob. Instead, the ActiveAreaSpare polygon (invisible by default) is defined. It may be renamed ActiveArea and positioned as desired to make only some area of the knob sensitive to mouse events.
Plane (POLYGON, optional): The knob pointer slides on the plane defined with this polygon. The points of the polygon must be coplanar.

Increase (VIEWPORT, optional): An “increase” button for a knob.
Decrease (VIEWPORT, optional): A “decrease” button for a knob.
Increment (DDATA, optional): If the knob viewport contains control buttons, each click of the button moves the knob by the amount specified by this resource. The value of the resource is between 0 and 1, and refers to a proportion of the total range of the knob. If this resource is not present, the default increment of motion is 0.1 times the total knob range.
Active Element

An input widget’s active element may be a simple object, or a group of objects:

\textit{ActiveElement} (POLYGON or ARC, optional): The moving part of a control widget.

or

\textit{ActiveElementGroup} (GROUP, optional): The moving parts of a control widget.

\textit{ActiveElement} (POLYGON or ARC, optional): The main part of the active element group.

\textit{Pointer} (POLYGON, optional): The end part of a meter needle.

The active element may have the \textit{Truncate} resource that controls how input values outside of the range are displayed. When it is set to ON, the value is truncated to fit into the \textit{High-Low} range of the widget. When the value is truncated, the active element is positioned at either the start or the end of the scale, depending on the input value.

Clock Hands

\textit{HourHandGroup} (GROUP, optional): The hour hand.

\textit{HourHand} (POLYGON): A polygon element of the hour hand.

\textit{MinHandGroup} (GROUP, optional): The minute hand.

\textit{MinHand} (POLYGON): A polygon element of the minute hand.

\textit{SecHandGroup} (GROUP, optional): The second hand.

\textit{SecHand} (POLYGON): A polygon element of the second hand.

Unit Label

\textit{UnitLabel} (TEXT): The axis label of a control widget.

Scale Area

\textit{ScaleArea} (POLYGON): The area behind the moving part of a control widget or dial.

\textit{ScaleAreaBorder} (POLYGON, optional): A polygon surrounding the \textit{ScaleArea}.

\textit{ScaleAreaBorder1} (POLYGON, optional): A polygon surrounding the polygon surrounding the \textit{ScaleArea}.

Center Marker

\textit{Element} (POLYGON, ARC or GROUP): A graphical object providing additional decoration to a meter’s rotation center.

Value Label

\textit{ValueLabel} (TEXT): The value label of a control widget.

\textit{Low} (DDATA, constrained): The lower limit of the meter’s value.

\textit{High} (DDATA, constrained): The upper limit of the meter’s value.

\textit{Format} (SDATA): The C-style label display format.
Value Label Group

ValueLabelGroup (GROUP): A value label with decorations.

LabelArea (POLYGON): The background area of the label.

LabelBorder (POLYGON): One side of the bevels decorating the label.

ValueLabel (TEXT): The value label of a control widget.

Value (DDATA, constrained, optional): The meter value.

Low (DDATA, constrained, optional): The lower limit of the meter value.

High (DDATA, constrained, optional): The upper limit of the meter value.

Format (SDATA): The display format for the label.

X Value Label Group

XValueLabelGroup (GROUP): A value label with decorations.

XLabelArea (POLYGON): The background area of the label.

XLabelBorder (POLYGON): One side of the bevels decorating the label.

XValueLabel (TEXT): The meter value label.

Low (DDATA, constrained): The lower limit of the meter value.

High (DDATA, constrained): The upper limit of the meter value.

Format (SDATA): The C-style label display format.

Y Value Label Group

YValueLabelGroup (GROUP): A value label with decorations.

YLabelArea (POLYGON): The background area of the label.

YLabelBorder (POLYGON): One side of the bevels decorating the label.

YValueLabel (TEXT): The meter value label.

Low (DDATA, constrained): The lower limit of the meter value.

High (DDATA, constrained): The upper limit of the meter value.

Format (SDATA): The C-style label display format.

Pointer

XPointer (POLYGON): A graphical shape that indicates the current X value on an XY graph.

YPointer (POLYGON): A graphical shape that indicates the current Y value on an XY graph.
Index

3D bar graph, 69
3D color datagroup resource set, 106
3D data area resource set, 100
3D datagroup resource set, 106
3D histogram graph, 69
3D packed bar graph, 71
3D packed signed bar graph, 71
3D signed bar graph, 69
3D stacked bar graph, 73
3D step graph, 75

A
active element resource set, 119, 120
axis, 27

B
bar graph, 48
3D, 69
3D multiple data sets, 70
3D packed, 71
3D stacked, 73
horizontal, 49
multiset 3D packed, 72
packed, 50
stacked, 51
symmetrical, 49
bevel resource set, 117
bevels, 84
button widget, 40
button widgets, 90

C
center marker resource set, 120
clock hands resource set, 120
clock widget, 42
clock widgets, 86
Commonly Used Graph Resources, 22
control widget resources, 118
cylinder graph, 69

D
data area, 27
data area resource set, 99
data samples, 23
datagroup resource set, 100

F
filled line graph, 57
filled multiline graph, 58
filled multiline step datagroup resource set, 104
filled multiline step graph, 56
filled step graph, 53
font browser widget, 44, 96
framed surface graph, 79

G
gantt chart, 80
gauge widgets, 85
graph components, 19
graph widgets, 17

H
histogram graph, 48
horizontal 3D slider graph, 82
horizontal bar graph, 49
horizontal filled step graph, 54
horizontal slider graph, 80
horizontal step graph, 54

I
indicator widgets, 90
interface widget resources, 118

J
joystick widgets, 89

K
knob
  multiposition widget, 91
knob widget, 38
knob widgets, 93

L
labels, 29
Layout Templates, 46
legend resource set, 84, 116
legends, 32
level objects, 32
level resource set, 109
line datagroup resource set, 101
line graph, 57
multiline polar, 65
multiple lines, 58
polar, 62, 63
line segment graph, 57
logarithmic ticks, 28

M
major ticks, 28
menu widgets, 95
minor ticks, 28
multi-axis multiline XY graph, 61
multiline datagroup resource set, 102
multiline graph, 58
multiline polar graph, 63
multiline polar XY datagroup resource set, 105
multiline polar XY graph, 65
multiline step datagroup resource set, 104
multiline step graph, 55
multiline XY datagroup resource set, 103
multiline XY graph, 60
Multiple Axes, 33
multipositional knob widgets, 91
multipositional slider widgets, 92
multiset 3D bar graph, 70
multiset 3D datagroup resource set, 106
multiset 3D packed bar graph, 72
multiset 3D packed datagroup resource set, 107
multiset 3D packed signed bar graph, 72
multiset 3D signed bar graph, 70
multiset 3D stacked bar graph, 74
multiset 3D step graph, 76
multiset cylinder graph, 70
multiset datagroup resource set, 100
multiset line segment graph, 58
multiset point graph, 58
multiset point XY graph, 60
multiset prism graph, 70
multiset pyramid graph, 70
multiset ribbon graph, 78

N
naming conventions, 20

P
packed bar graph, 50
packed color datagroup resource set, 101
palette widget, 43
palette widgets, 95
pie chart, 66
pie datagroup, 105
point graph, 57
pointer resource set, 121
polar data area resource set, 99
polar filled line graph, 62
polar level resource set, 109
polar line graph, 62
polar line segment graph, 62
polar point graph, 62
polar XY line datagroup resource set, 105
prism graph, 69
pyramid graph, 69

R
radial grid resource set, 116
resource
naming conventions, 20
Resource Browser Widget, 44
resource set
3D data area, 100
3D datagroup, 106
3D packed datagroup, 107
active element, 118, 119, 120
bevel, 117
center marker, 120
clock hands, 120
data area, 99
datagroup, 100
filled multiline step datagroup, 104
legend, 116
level, 109
line datagroup, 101
multiline datagroup, 102
multiline polar XY datagroup, 105
multiline step datagroup, 104
multiline XY datagroup, 103
multiset 3D datagroup, 106
multiset 3D packed datagroup, 107
multiset datagroup, 100
packed color datagroup, 101
pie datagroup, 105
pointer, 121
polar data area, 99
polar level, 109
polar XY line datagroup, 105
radial grid, 116
scale area, 120
status object, 110
surface datagroup, 108
symmetrical level, 109
tangent grid, 116
tangent labels, 116
tangent ticks and labels, 116
title, 99
unit label, 120
value axis, 110
value label, 120
value label group, 121
viewing, 117
X axis, 111
X level, 110
X time axis, 114
X value axis, 112
X value label group, 121
Xy line datagroup, 102
Y axis, 111
Y level, 110
Y time axis, 115
Y value axis, 113
Y value multiaxis, 113
Y1 value axis, 113
Z axis, 112
Z value axis, 114
zooming, 117
ribbon graph, 77

S
scale area resource set, 120
Scrolling Time Axis Labels, 30
signed bar graph, 48
slider
  horizontal 3D graph, 82
  horizontal graph, 80
  multiposition widget, 92
  vertical 3D graph, 83
  vertical graph, 81
slider widget, 35
slider widgets, 94
spinner widget, 42
stacked bar graph, 51
status object resource set, 110
status objects, 32
step graph, 52
stopwatch widget, 42, 86
surface datagroup resource set, 108
symmetrical bar graph, 49
symmetrical level resource set, 109

T
Tag Browser Widget, 44
tangent grid resource set, 116
tangent labels resource set, 116
tangent ticks and labels resource set, 116
text objects, 29
text widget, 41, 96
thermometer widget, 88
title resource set, 99
titles, 32
Truncate, 24

U
unframed surface graph, 79
unit label resource set, 120
updating time axis labels, 30

V
value axis resource set, 110
value label group resource set, 121
value label resource set, 120
value labels, 30
vertical 3D slider graph, 83
vertical slider graph, 81
viewing resource set, 117

X
X axis resource set, 111
X level resource set, 110
X time axis resource set, 110
X value label group resource set, 121
Xy line datagroup resource set, 102
XY line graph, 59
XY point graph, 59
XY polar line graph, 64

Y
Y axis resource set, 111
Y level resource set, 110
Y time axis resource set, 115
Y value axis resource set, 113
Y value multiaxis resource set, 113
Y1 value axis resource set, 113

Z
Z axis resource set, 112
Z value axis resource set, 114
zooming resource set, 117