DrawOval

`DrawOval` `[`***`/W=winName`***`]` ***`left, top, right, bottom`***

The DrawOval operation draws an oval in the target graph, layout or control panel within the rectangle defined by *left*, *top*, *right,* and *bottom*.

Flags

/W=*winName*	Draws to the named window or subwindow. When omitted, action will affect the active window or subwindow. This must be the first flag specified when used in a Proc or Macro or on the command line. When identifying a subwindow with *winName*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.

Details

The coordinate system as well as the oval's thickness, color, dash pattern and other properties are determined by the current drawing environment. The oval is drawn in the current draw layer for the window, as determined by SetDrawLayer.

See Also

Chapter III-3, **Drawing**.

The **SetDrawEnv**, **SetDrawLayer** and **DrawAction** operations.

DrawPICT

`DrawPICT` `[`***`/W=winName`***`][`**`/RABS`**`]` ***`left, top, hScaling, vScaling, pictName`***

The DrawPICT operation draws the named picture in the target graph, layout or control panel. The *left* and *top* parameters set the position of the top/left corner of the picture. *hScaling* and *vScaling* set the horizontal and vertical scale factors with 1 meaning 100%.

Flags

/RABS	Draws the named picture using absolute scaling. In this mode, it draws the picture in the rectangle defined by *left* and *top* for point (x0,y0), and by *hScaling* and *vScaling* for point (x1,y1), respectively.
/W=*winName*	Draws to the named window or subwindow. When omitted, action will affect the active window or subwindow. This must be the first flag specified when used in a Proc or Macro or on the command line. When identifying a subwindow with *winName*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.

Details

The coordinate system for the left and top parameters is determined by the current drawing environment. The PICT is drawn in the current draw layer for the window, as determined by SetDrawLayer.

See Also

Chapter III-3, **Drawing**.

The **SetDrawEnv**, **SetDrawLayer** and **DrawAction** operations.

DrawPoly

`DrawPoly` `[`***`/W=winName`*** **`/ABS`**`]` ***`xorg, yorg, hScaling, vScaling, xWaveName, yWaveName`***

`DrawPoly` `[`***`/W=winName`*** **`/ABS`**`]` ***`xorg, yorg, hScaling, vScaling,`*** **`{`**x_0,y_0,x_1,y_1 …**`}`**

`DrawPoly/A` `[`***`/W=winName`***`]` **`{`**x_n, y_n, x_{n+1}, y_{n+1} …**`}`**

The DrawPoly operation draws a polygon in the target graph, layout or control panel.

Parameters

(*xorg*, *yorg*) defines the starting point for the polygon in the currently active coordinate system.

hScaling and *vScaling* set the horizontal and vertical scale factors, with 1 meaning 100%.

The *xWaveName*, *yWaveName* version of DrawPoly gets data from those X and Y waves. This connection is maintained so that changes to either wave will update the polygon.

The DrawPoly operation is not multidimensional aware. See **Analysis on Multidimensional Waves** on page II-95 for details.

To use the version of DrawPoly that takes a literal list of vertices, you place as many vertices as you like on the first line and then use as many /A versions as necessary to define all the vertices.

Flags

/A	Appends the given vertices to the currently open polygon (freshly drawn or current selection).
/ABS	Suppresses the default subtraction of the first point from the rest of the data.
/W=*winName*	Draws to the named window or subwindow. When omitted, action will affect the active window or subwindow. This must be the first flag specified when used in a Proc or Macro or on the command line. When identifying a subwindow with *winName*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.

Details

Because *xorg* and *yorg* define the location of the starting vertex of the poly, adding or subtracting a constant from the vertices will have no effect. The first XY pair in the {*x0, y0, x1, y1,…*} vertex list will appear at (*xorg,yorg*) *regardless* of the value of *x0* and *y0*. *x0* and *y0* merely serve to set a reference point for the list of vertices. Subsequent vertices are relative to (*x0,y0*).

To keep your mental health intact, we recommend that you specify (*x0,y0*) as (0,0) so that all the following vertices are offsets from that origin. Then (*xorg,yorg*) sets the position of the polygon and all of the vertices in the list are relative to that origin.

An alternate method is to use the same values for (*x0,y0*) as for (*xorg,yorg*) if you consider the vertices to be “absolute” coordinates.

You can include the /ABS flag to suppress the subtraction of the first point.

To change just the origin and scale of the currently open polygon — without having to respecify the data — use:

```
DrawPoly xorg, yorg, hScaling, vScaling,{}
```

The coordinate system as well as the polygon’s thickness, color, dash pattern and other properties are determined by the current drawing environment. The polygon is drawn in the current draw layer for the window, as determined by SetDrawLayer.

It is possible to separate a polygon into segments by adding coordinate pairs that are NaN. For details, see **Segmented Polygons** on page III-70.

Examples

Here are some commands to draw some small triangles using absolute drawing coordinates (see **SetDrawEnv**).

```
Display              // make a new empty graph
//Draw one triangle, starting at 50,50 at 100% scaling
SetDrawEnv xcoord= abs,ycoord= abs
DrawPoly 50,50,1,1, {0,0,10,10,-10,10,0,0}
//Draw second triangle below and to the right, same size and shape
SetDrawEnv xcoord= abs,ycoord= abs
DrawPoly 100,100,1,1, {0,0,10,10,-10,10,0,0}
```

For another example using polygons, see **Segmented Polygons** on page III-70.

See Also

DrawPoly and DrawBezier Operations on page III-75

SetDrawEnv, **SetDrawLayer**, **DrawBezier**, **DrawAction**, **PolygonOp**.