```
Function ExportSelectedTrace()
   GetLastUserMenuInfo
   // Do something with S_graphName, S_traceName
End

Function DrawXYHere()
   GetLastUserMenuInfo

   // Figure out mouse position within graph (requires Igor 8)
   String info = traceinfo(S_graphName, S_traceName, 0)
   String xaxis = stringbykey("XAXIS",info)
   String yaxis = stringbykey("YAXIS",info)
   Variable x_pos = AxisValFromPixel(S_graphName, xaxis, V_mouseX)
   Variable y_pos = AxisValFromPixel(S_graphName, yaxis, V_mouseY)

   // Draw
   GetAxis/W=$S_graphName/Q $yaxis
   Variable miny=V_min, maxy=V_max
   GetAxis/W=$S_graphName/Q $xaxis
   Variable minx=V_min, maxx=V_max
   SetDrawLayer/W=$S_graphName/K userFront
   SetDrawEnv/W=$S_graphName push
   SetDrawEnv/W=$S_graphName xcoord= $xaxis,ycoord=$yaxis,save
   DrawLine/W=$S_graphName minx, y_pos, maxx, y_pos      // horizontal line
   DrawLine/W=$S_graphName x_pos, miny, x_pos, maxy      // vertical line
   SetDrawEnv/W=$S_graphName pop
End
```

See Also

Chapter IV-5, **User-Defined Menus** and especially the sections **Optional Menu Items** on page IV-130, **Multiple Menu Items** on page IV-131, and **Specialized Menu Item Definitions** on page IV-132.

Trace Names on page II-282, **Programming With Trace Names** on page IV-87.

GetMarquee

GetMarquee [**/K/W=*winName*/Z**] [***axisName*** [**,** ***axisName***]]

The GetMarquee operation provides a way for you to use the marquee as an input mechanism in graphs and page layout windows. It puts information about the marquee into variables.

Parameters

If you specify *axisName* (allowed only for graphs) the coordinates are in axis units. If you specify an axis that does not exist, Igor generates an error.

If you specify only one axis then Igor sets only the variables appropriate to that axis. For example, if you execute "`GetMarquee left`" then Igor sets the V_bottom and V_top variables but does *not* set V_left and V_right.

Flags

/K	Kills the marquee. Usually you will want to kill the marquee when you call GetMarquee, so you should use the /K flag. This is modeled after what happens when you create a marquee in a graph and then choose Expand from the Marquee menu. There may be some situations in which you want the marquee to persist. Igor also automatically kills the marquee anytime the window containing the marquee is deactivated, including when a dialog is summoned.
/W=*winName*	Specifies the named window or subwindow. When omitted, action will affect the active window or subwindow. When identifying a subwindow with *winName*, see **Subwindow Syntax** on page III-92 for details on forming the window hierarchy.
/Z	No runtime error generated if the target window isn't a graph or layout, but V_flag will be zero. /Z does not prevent other kinds of problems from generating a runtime error.

Details

GetMarquee is intended to be used in procedures invoked through user menu items added to the graph Marquee menu and the layout Marquee menu.

GetMarquee sets the following variables and strings:

V_flag	0: There was no marquee when GetMarquee was invoked. 1: There was a marquee when GetMarquee was invoked.
V_left	Marquee left coordinate.
V_right	Marquee right coordinate.
V_top	Marquee top coordinate.
V_bottom	Marquee bottom coordinate.
S_marqueeWin	Name of window containing the marquee, or "" if no marquee. If subwindow, subwindow syntax will be used.

When called from the command line, GetMarquee sets global variables and strings in the current data folder. When called from a procedure, it sets local variables and strings.

In addition, creating, adjusting, or removing a marquee may set additional marquee global variables (see the **Marquee Globals** section, below).

The target window must be a layout or a graph. Use /Z to avoid generating a runtime-error (V_flag will be 0 if the target window was not a layout or graph).

If the target is a layout then Igor sets the variables in units of points relative to the top/left corner of the paper.

If the target is a graph then Igor sets V_left and V_right based on the specified horizontal axis. If no horizontal axis was specified, V_left and V_right are set relative to the left edge of the base window in points.

If the target is a graph then Igor sets V_bottom and V_top based on the specified vertical axis. If no vertical axis was specified, V_top and V_bottom are set relative to the top edge of the base window in points.

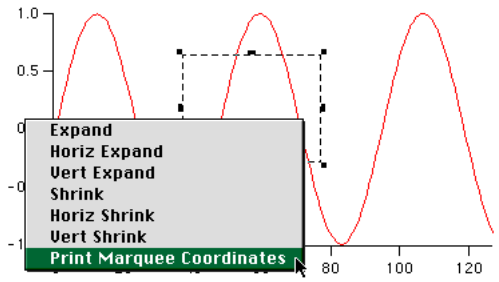
If there is no marquee when you invoke GetMarquee then Igor sets V_left, V_top, V_right, V_bottom based on the last time the marquee was active.

GetMarquee Example

```
Menu "GraphMarquee"
    "Print Marquee Coordinates", PrintMarqueeCoords()
End

Function PrintMarqueeCoords()
    GetMarquee left, bottom
    if (V_flag == 0)
        Print "There is no marquee"
    else
        printf "marquee left in bottom axis terms: %g\r", V_left
        printf "marquee right in bottom axis terms: %g\r", V_right
        printf "marquee top in left axis terms: %g\r", V_top
        printf "marquee bottom in left axis terms: %g\r", V_bottom
    endif
End
```

You can run this procedure by putting it into the procedure window, making a marquee in a graph, clicking in the marquee and choosing Print Marquee Coordinates:



The procedure calls GetMarquee to set the local marquee variables and then prints their values in the history area:

```
PrintMarqueeCoords()
  marquee left in bottom axis terms: 32.1149
  marquee right in bottom axis terms: 64.7165
  marquee top in left axis terms: 0.724075
  marquee bottom in left axis terms: -0.131061
```

Marquee Globals

You can cause Igor to update global marquee variables whenever the user adjusts the marquee (without the need for you to invoke GetMarquee) by creating a global variable named V_marquee in the root data folder:

```
Variable/G root:V_marquee = 1     //Creates V_marquee and sets bit 0 only
```

When the user adjusts the marquee Igor checks to see if root:V_marquee exists and which bits are set, and updates (and creates if necessary) these globals:

`Variable/G root:V_left`	Marquee left coordinate.
`Variable/G root:V_right`	Marquee right coordinate.
`Variable/G root:V_top`	Marquee top coordinate.
`Variable/G root:V_bottom`	Marquee bottom coordinate.
`String/G root:S_marqueeWin`	Name of window that contains marquee, or "" if no marquee. Set only if root:V_Marquee has bit 15 (0x8000) set.

Unlike the local variables, for graphs these global variables are never in points. Root:V_left and V_right will be axis coordinates based on the first bottom axis created for the graph (if none, then for the first top axis). The axis creation order is the same as is returned by **AxisList**. Similarly, root:V_top and root:V_bottom will be axis coordinates based on the first left axis or the first right axis.

Igor examines the global root:V_marquee for bitwise flags to decide which globals to update, and when:

root:V_marquee Bit Meaning	Bit Number	Bit Value
Update global variables for graph marquees	0	1
Update global variables for layout marquees	2	4
Update S_marqueeWin when updating global variables	15	0x8000

Marquee Globals Example

By creating the global variable root:V_marquee this way:

```
Variable/G root:V_marquee = 1 + 4 + 0x8000
```

whenever the user creates, adjusts, or removes a marquee in any graph or layout Igor will create and update the global root:V_left, etc. coordinate variables and set the global string root:S_marqueeWin to the name of the window which has the marquee in it. When the marquee is removed, root:S_marqueeWin will be set to "".

This mechanism does neat things by making a **ValDisplay** or **SetVariable** control depend on any of the globals. See the Marquee Demo experiment in the Examples:Feature Demos folder for an example.

You can also cause a function to run whenever the user creates, adjusts, or removes a marquee by setting up a dependency formula using **SetFormula** to bind one of the marquee globals to one of the function's input arguments:

```
Variable/G root:dependencyTarget

SetFormula root:dependencyTarget, "MyMarqueeFunction(root:S_marqueeWin)"

Function MyMarqueeFunction(marqueeWindow)
   String marqueeWindow          // this will be root:S_marqueeWin

   if( strlen(marqueeWindow) )
      NVAR V_left= root:V_left, V_right= root:V_right
      NVAR V_top= root:V_top, V_bottom= root:V_bottom
      Printf marqueeWindow + " has a marquee at: "
      Printf "%d, %d, %d, %d\r", V_left, V_right, V_top, V_bottom
   else
```