Subject: Re: Button_events and data
Posted by davidf on Wed, 25 Feb 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Bernard Puc (puc@gsfc.nasa.gov) writes:

> I'm looking for pointers on how to do the following: I have a line
> plot in a draw widget. I want to click the pointer on the plot and have
> a vertical line drawn at the nearest datapoint to the mouseclick.
> widget_draw returns the cursor location in device coordinates and I
> need to translate that into dataspace coordinates.
> Going one step further, ideally, I'd like to have a vertical line
> follow the cursor around the draw widget at all times, jumping from
> datapoint to datapoint. I suspect someone has already written such a
> thing...

Well, you hit me in a moment of weakness. :-)

I've been writing a program today that is out there on the
edge of my knowledge and experience. Writing something I
know how to write seemed kind of relaxing. I did this over
a couple of beers, so it may not be my *best* work. ;-)

The way I chose to implement your requirements is to draw
the vertical line as long as you hold the cursor down in the
draw widget. The program assumes regular "steps" in the
X direction, but the algorithm could easily be changed
to accommodate irregular steps.

Here you go. Save the code below as "example.pro". Type
"example" to see it work.

Cheers,

David

```
PRO Example_Cleanup, id
Widget_Control, id, Get_UValue=info, /No_Copy
IF N_Elements(info) NE 0 THEN WDelete, info.pixID
END


PRO Draw_Widget_Events, event

   ; Deal only with button up, button down, and motion events.

IF event.type GT 2 THEN RETURN
```

```
   ; What kind of event is this?

Widget_Control, event.top, Get_UValue=info, /No_Copy
eventType = ['Button Down', 'Button Up', 'Motion Events']
thisEvent = eventType(event.type)

CASE thisEvent OF

   'Button Down': BEGIN

      ; Turn motion events on.

      Widget_Control, event.id, Draw_Motion_Events=1
      ENDCASE

   'Button Up': BEGIN

      ; Turn motion events off.

      Widget_Control, event.id, Draw_Motion_Events=0

      ; Erase the last line.

      WSet, info.wid
      Device, Copy=[0,0,400,400,0,0,info.pixID]

      ENDCASE

   'Motion Events': BEGIN

      ; Erase the previous line.

      WSet, info.wid
      Device, Copy=[0,0,500,500,0,0,info.pixID]

      ; Set up plot and axes scaling.

      !P = info.p
      !X = info.x
      !Y = info.y

      ; Convert cursor location to data coordinates.

      coords = Convert_Coord(event.x, event.y, /Device, /To_Data)
      x = coords[0]
      y = coords[1]

      ; Make sure X value is within plot limits.
```

```
      x = !X.crange[0] > x
      x = !X.crange[1] < x

      ; Find the nearest data point.

      nearest = WHERE(info.indep GE (x - (info.step/2.0)), count)
      IF count EQ 0 THEN datapt = info.indep[info.last] ELSE BEGIN
         nearest = nearest[0]
         IF nearest EQ 0 THEN datapt = info.indep[0] ELSE $
            datapt = info.indep[nearest]
      ENDELSE

      ; Draw a vertical line through the nearest datapoint.

      PlotS, [datapt, datapt], !Y.CRange, Color=info.color
      format = '(F4.1)'
      XYOutS, datapt, 0.85, '(' + $
         String(info.indep[datapt],Format=format) + $
         ',' + String(info.data[datapt],Format=format) + ')', $
         Color=info.color
      ENDCASE

ENDCASE

Widget_Control, event.top, Set_UValue=info, /No_Copy
END


PRO Example

   ; Fake data.

data = Findgen(11)
data = Sin(data*360*!RaDeg)
indep = Findgen(11)

   ; Create the widgets.

tlb = Widget_Base(Title='Example Program', Column=1)
drawID = Widget_Draw(tlb, XSize=500, YSize=500, $
  Event_Pro='Draw_Widget_Events', Button_Events=1)

   ; Get the window index number.

Widget_Control, tlb, /Realize
Widget_Control, drawID, Get_Value=wid
WSet, wid
```

```
   ; Plot the data.

Plot, indep, data, PSym=-4, Position=[0.2, 0.2, 0.8, 0.8]

   ; Create a pixmap.

Window, /Free, /Pixmap, XSize=500, YSize=500
pixID = !D.Window
Plot, indep, data, PSym=-4, Position=[0.2, 0.2, 0.8, 0.8]

   ; The data step in the X direction. Assume regular steps.

step = indep[1] - indep[0]
TVLCT, 255, 255, 0, 1
color = 1

   ; Save information to run the program.

info = { indep:indep, $  ; Independent data.
         data:data, $    ; Dependent data.
         p:!P, $         ; Plotting system variable.
         x:!X, $         ; X Axis system variable.
         y:!Y, $         ; Y Axis system variable
         wid:wid, $      ; Window index number.
         pixID:pixID, $  ; The pixmap index number.
         step:step, $    ; The step in independent data.
         color:color, $  ; The line color.
         last:N_Elements(indep)-1 $  ; Last element in array.
         }

Widget_Control, tlb, Set_UValue=info, /No_Copy
XManager, 'example', tlb, /No_Block, Cleanup='Example_Cleanup'
END
```

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/