Subject: Re: Widget Event_Pro question
Posted by David Fanning on Mon, 31 Mar 2008 13:45:55 GMT
View Forum Message <> Reply to Message

Spon writes:

> As part of a little image display programme, I've currently got a draw
> widget:
>
> Display = WIDGET_DRAW(Base, XSIZE = S[0], YSIZE = S[1], $
> /MOTION_EVENTS, EVENT_PRO = 'SLIDESHOWWIDGET_GETVALUES')
>
> And I'd like to add button events to it. I don't want to get rid of
> the motion events though. The easiest way that I can think of adding
> functionality is to (ideally) have two Event_Pro strings, one to be
> run if a motion event is detected, another if a mouse click is
> detected.
>
> Is there a way of doing this?
>
> Or am I just stuck making my SLIDESHOWWIDGET_GETVALUES programme big,
> confusing and unwieldy?

This will depend entirely on what kind of IDL programmer you
are. :-)

You can't have two different event handlers assigned to a
widget at the same time, but there is nothing that prevents
that single event handler from being an event dispatcher.
The event comes in. The event handler figures out what kind
of event it is (e.g., button up, button down, motion,
expose, etc.), and then the event is passed onto some other
IDL procedure or function for further processing. Often,
the info pointer is also needed, along with the event structure,
to completely handle the event.

```
  PRO MyProg_DrawWidgetEventProcessing, event

  ; Get info pointer.
  Widget_Control, event.top, GET_UVALUE=infoPtr

  ; What kind of event is this?
  kind =
['DOWN','UP','MOTION','VIEWPORT','EXPOSE','CH','KEY','WHEEL' ]
  CASE kind[event.type] OF
    'DOWN': MyProg_HandleButtonDownEvents, event, infoPtr
    'UP': MyProg_HandleButtonUpEvents, event, infoPtr
    'MOTION': MyProg_HandleButtonMotionEvents, event, infoPtr
```

```
    ELSE: ; Don't care.
  ENDCASE

  END
```

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Widget Event_Pro question
Posted by Spon on Mon, 31 Mar 2008 14:14:46 GMT
View Forum Message <> Reply to Message

On Mar 31, 2:45 pm, David Fanning <da...@dfanning.com> wrote:
> This will depend entirely on what kind of IDL programmer you
> are. :-)
>
> You can't have two different event handlers assigned to a
> widget at the same time, but there is nothing that prevents
> that single event handler from being an event dispatcher.
> The event comes in. The event handler figures out what kind
> of event it is (e.g., button up, button down, motion,
> expose, etc.), and then the event is passed onto some other
> IDL procedure or function for further processing. Often,
> the info pointer is also needed, along with the event structure,
> to completely handle the event.
>
>   PRO MyProg_DrawWidgetEventProcessing, event
>
>   ; Get info pointer.
>   Widget_Control, event.top, GET_UVALUE=infoPtr
>
>   ; What kind of event is this?
>   kind =
> ['DOWN','UP','MOTION','VIEWPORT','EXPOSE','CH','KEY','WHEEL']
>   CASE kind[event.type] OF
>   'DOWN': MyProg_HandleButtonDownEvents, event, infoPtr
>   'UP': MyProg_HandleButtonUpEvents, event, infoPtr
>   'MOTION': MyProg_HandleButtonMotionEvents, event, infoPtr
>   ELSE: ; Don't care.
>   ENDCASE

>
>   END
>
> Cheers,
>
> David
> --
> David Fanning, Ph.D.
> Fanning Software Consulting, Inc.
> Coyote's Guide to IDL Programming:http://www.dfanning.com/
> Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Hey, that's a neat trick, thanks David! Looks like it could keep
things a little more organised in my event handler structure; and God
knows I need all the help I can get when it comes to being
organised! :-)

As it happened, I just chickened out and went for for an
  IF Event.Release THEN BEGIN
statement in this case, but it'd only be a short cut-'n'-paste from
there to calling a separate handler programme to do what I've
shoehorned in after the BEGIN...

...which includes the following, and has me struggling with widget
geometry (and this is _before_ I'm even going near cross-platform
stuff :-\ )

```
    ; Prepare a second widget base:
   GraphBase = WIDGET_BASE(/COLUMN, $
     GROUP_LEADER = Event.Top, $
     TITLE = "Press 'Done' to close window.", $
     /ALIGN_RIGHT)
```

to which I add a simple 'done' button, a couple of labels to remind me
where in the image I'd clicked to get this graph, and a draw widget to
hold my graph.

Now, if instead of Group_Leader I go for Event.Top as my *parent*, the
graph window appears in a sort of added-on blob to my original window,
with the result that the 'Done' button kills the whole shebang,
instead of just closing my graph window and letting me get on with
wasting CPU processing power dragging my mouse over the image window.

But if I keep Event.Top as my Group_Leader, then the /ALIGN_RIGHT
keyword doesn't seem to do a whole lot, and my graph widget sits smack
bang on top of my image window widget.

Typically, I want to have my cake *and* eat it ;-)
I'd like my second widget base to be independent enough of its leader

to be a separate entity, but to be *aware* enough of the leader to
know where I want it to sit, namely glued to its right edge. Is this
possible?

Thanks for the help so far,
regards,

Chris

Subject: Re: Widget Event_Pro question
Posted by David Fanning on Mon, 31 Mar 2008 14:45:03 GMT
View Forum Message <> Reply to Message

Spon writes:

> But if I keep Event.Top as my Group_Leader, then the /ALIGN_RIGHT
> keyword doesn't seem to do a whole lot, and my graph widget sits smack
> bang on top of my image window widget.
>
> Typically, I want to have my cake *and* eat it ;-)
> I'd like my second widget base to be independent enough of its leader
> to be a separate entity, but to be *aware* enough of the leader to
> know where I want it to sit, namely glued to its right edge. Is this
> possible?

Well, ALIGN_RIGHT has the right name to give you hope,
but none of the functionality, unfortunately. That is to
say, it doesn't do what you think it does. :-)

If you want to position a top-level base next to
another top-level base, that is easily done. Find
the position and size of the first TLB:

```
  Widget_Control, TLB_1, TLB_GET_OFFSET=offsets, $
     TLB_GET_SIZE=sizes
```

Then, position the second TLB next to the first:

```
  Widget_Cotnrol, TLB_2, XOFFSET=offsets[0] + sizes[0] + 10, $
     YOFFSET = offsets[1]
```

The window is not "glued" to anything, and if either window
is moved, you won't know about it unless you have TLB_MOVE_EVENTS
set for the TLBs. If you do, you can figure out where the
window moved to and reposition things.

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting, Inc.
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Sepore ma de ni thui. ("Perhaps thou speakest truth.")

Subject: Re: Widget Event_Pro question
Posted by Spon on Wed, 02 Apr 2008 15:21:04 GMT
View Forum Message <> Reply to Message

On Mar 31, 3:45 pm, David Fanning <n...@dfanning.com> wrote:

Hello once again,

I'm now trying to find the best way to tell widget A how many copies of widget B are open, and what their IDs are, so that widget A knows what to kill when the user presses the 'Graph Deletion' button. I'm doing this by passing a pointer to an array around my top-level UVAL structure, and concatenating the new graph widget IDs onto the array as they get created, and then removing these IDs when the widget gets killed.

I need to define this array before any instances of widget B are open. Can I use a value of [-1L] for this? Or is -1 a valid widget ID number that might cause problems later if IDL decides to label one of my widgets -1 at some point?

Regards,

Chris
