Subject: Using WIDGET_EVENT to break into loops
Posted by andyhall on Wed, 07 Aug 1996 07:00:00 GMT
View Forum Message <> Reply to Message

Hi all.
I have been writing an application that uses a loop to perform animated data visualization, and I wanted to add the functionality to stop the loop before the animation was complete. In searching for a good way to do this, I came accross the following post from this group in answer to a similar question:

```
>  Check out the !NEWS IDL Newsletter Volume 6, Number 1 (Summer 96)
>  if you have it. It has an example illustratiing how to do what you
>  are asking.
>
>  Basically, within the loop in the event handler, you use
>  WIDGET_EVENT() to poll for events generated from the widget
>  heirarchy "represented" by the widget id argument to WIDGET_EVENT().
>
>  Something like:
>
>  quit_loop = 0
>  while ( quit_loop eq 0 ) do begin
>     ...
>          quit = WIDGET_EVENT( event.top, /nowait )
>          if ( quit.id eq state.cancel_button ) then $
>                quit_loop = 1
>     ...
>  endwhile
>
>  Hope this helps.
>  Dave Foster
>  foster@bial1.ucsd.edu
```

For reference, my routine uses a large graphical interface with an event handler processing all the user input. I use the xmanager routine to handle the widgets.

So, I tried the advice above. However, I had a few problems:

The above message (and the IDL help) seem to imply that WIDGET_EVENT(id) is a routine that will extract events for the hierarchy rooted at id and hand them over to the user to examine.

However, when I call WIDGET_EVENT, the procedure polls the hierarchy rooted at id for events, then passes those events to the event handler, processes the event, and _then_ returns the event to me.

Thus, when I implemented the above code, or rather this code:

```
WHILE NOT(quit_anim) DO BEGIN
    ...
    quit = widget_event(event.top, /nowait)
    IF quit.id NE 0 THEN quit_anim = 1
END
```

However, since the widget_event routine was processing the events
it found, by the time it got to the IF statement, the event struct
was cleared, and quit.id was always zero no matter what.

The only way I could find to break the loop was to add a button to
the same hierarchy such that the button's event exedcuted the command

```
    quit_anim = 1
```

This breaks the loop, but is not satisfactory for several reasons, chief
among them being that this requires that the application process ALL
events generated in that entire hierarchy during execution of the loop.

Sorry about the involved nature of this post, but I would greatly
appreciate anyone's advice.. I don't know if I should be structuring my
event handling differently, or if I am using widget_event wrong, or what.

Anyway, thanks in advance...

Andy Hall
