Subject: Re: Multiple widgetized windows in one application
Posted by steinhh on Thu, 29 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:

> More to the point, how many widget programmers like to write
> programs that are easy to extend and maintain? And how many
> like to write programs that they have to futz around with
> for hours to make a simple change?
>
> Event handlers should be assigned logically according to what
> they do, in my opinion. I quite often assign an event handler
> to a "menu button" so that events caused by this button's
> children will bubble up and all be handled by this one event
> handler. This is especially the case if all the different
> events are similar in some way. Perhaps they all do some
> kind of image processing.

Most widget programs that I've written have been using a
single event handling routine, structured along the lines
of:

```
PRO program_event,ev
  widget_control,ev.top,get_uvalue=info,/no_copy
  widget_control,ev.id,get_uvalue=uval

  CASE uval OF

  "QUIT":BEGIN
    widget_control,ev.top,/destroy
    return
    ENDCASE

  "SAYHELLO":program_sayhello,info,ev

  "DRAW":program_draw,info,ev
  :
  :
  END

  widget_control,ev.top,set_uvalue=info,/no_copy
END
```

Most of these were written before pointers were invented,
hence the /no_copy switch to avoid memory copying when
fetching the info structure. This also creates the
need to put the info structure back.

For complicated tasks I write separate *subroutines*, but
the events go through the same handler.

This is to keep things simple, and to minimize the number
of places where things can go wrong, like forgetting to
put back the info structure, doing "refresh" operations
if necessary etc..

It also makes it easier to put in a single line like

    help,info,ev,/structure

in just *one* place to do debugging of the event handling.

In my opinion, if an event influences or uses the state
of the application (i.e. the info structure) directly, it
should go through the same event handler as the
others.

If it *doesnt* influence or use the info structure, make
a compound widget!

So, in my opinion, the event_pro/event_func mechanism should
only be used when implementing compound widgets.

And if you find yourself using the event_pro/event_func
for groups of widgets controlling a subsystem of your
event application, I'd say chances are that you might be
better off implementing a compound widget - if nothing
else it will at least exercises the brain into thinking
object orientation (the compound widget being the object).

Regards,

Stein Vidar
