Subject: Re: IDL Runtime
Posted by David Fanning on Thu, 15 Nov 2001 01:21:32 GMT
View Forum Message <> Reply to Message

Andre Kyme (nak@imag.wsahs.nsw.gov.au) writes:

> Appreciate the reply, but as yet nobody has given me a good explanation of how
> XMANAGER works and I don't find the documentation provided by RSI insightful at
> all.

Yes, well, no one wants to commit themselves
because no one is really sure *how* it works.
The documentation was written in 1946. I think
it's safe to say, however, that most of us are
pretty sure it doesn't work the way you seem
to think it works. :-)

> I'm quite happy to adjust my program design - but I'm not sure exactly how
> to do this. People keep replying that "Your program must be so complicated..."
> but it is really quite simple! Or at least the IDEA is! I expect the idea would
> be a very typical widget programming endeavour!

It's the experience of this newsgroup (or, at the
very least, this correspondent) that the hardest
questions come from the simplest things.

> To briefly describe what my design structure is:
>
> About 5 widget buttons get created and realized to form a menu
> Each has an event handler
> I call XMANAGER to manage the created widgets
> I wait...
>
> Oh, button number 1 was pressed!
> Well we better do what event handler number 1 tells us to do.
>
> The event-handler is just a couple of programs that follow one after the other
> - is that complicated!! Some of them call other programs - But execution is
> just moving sequentially through the event handler routine. Pretty standard I
> think.

Then, no problem. When you get to the end of all the
sequential programs, you exit the event handler and
you are ready to process the next event. (Which has
already queued up, no doubt, because users are
murder on slow event handlers. But that's another
story.)

> The actual programs in the event handler involve the user drawing a region of
> interest (I don't use draw widgets, just a standard graphics window and a
> routine involving the cursor command) and involve moving it around, and also
> confirming they are happy with its location.

Well, IMHO, this is a huge mistake. And this
part is going to have to be re-written if you
want to run this as a run-time program, I think.
Run-time IDL will not work with "hybrid" programs
like this, it seems to me, although I haven't tried
it.

> Now suppose they get fed up with the whole business (like I'm feeling right
> now) and want to go and have a coffee - so when they are asked to confirm the
> region (I use dialogue_message with the /cancel keyword), they click "CANCEL" -

Here is the problem. How do you know when they
are finished? What they are doing is not in your
"widget space", so you can't get any feedback from
them. What should be happening is that the user is
generating events that you are handling in your widget
program. Here they are working outside the box, and you
can't get any feedback from them.

> Why should it be so complicated to just return to your menu options when I
> detect the cancellation? This would have to be a pretty standard widget
> application surely - menu, option, back to menu, option, back to menu...? What
> is the conventional philosophy for doing 'this'?

RETURN, simple as that. The code would probably look
like this:

```
ans = Dialog_Message('Are you finished?', /Question, /Cancel)
IF Strupcase(ans) EQ 'CANCEL' THEN RETURN
```

> So again, the design involves having options initially (menu). Some options
> aren't relevant initially such as "Save" and "Print", so if the user clicks on
> these I issue messages for the user telling them there's no point clicking
> these ones yet. The most likely thing they will want to do initially is
> "Analysis". "Analysis" is the button with the event handler I described above:
> programs that just simply get executed one after the other. During execution of
> the event-handler programs, the menu buttons can't be clicked, mainly due to
> the fact that the region drawing/moving uses the CURSOR command - so the
> computer is waiting for mouse clicks inside the graphics window. (You might
> prefer draw widgets w/o CURSOR etc etc, but there's really no problem I can see
> with doing it the way I have.

Well, I can see problems upon problems. But I'll

let it go here.

> It essentially creates MODAL functionality which
> ensures the user gets to the end. Anyway I think this is beside the point).

I'm not sure it's beside the point. I'm more
inclined to think it is *exactly* the point.
It is certainly what makes your program NOT a
widget program. It is hard to make the point
that widgets don't *work* right, when what you
use as an example is a program that is not
written as a widget program. What you have,
I think, is a program that uses some widget
functionality, and not always in the appropriate
way, even then.

> Also, exactly what DOES happen when instead, the user doesn't get fed up,
> finishes the whole event handling routine, and we arrive at the "END" command
> at the conclusion of the event-handler? Where is program execution? What is
> happening? I'm thinking that we're ready for another menu-button press??

You are indeed ready for another button press.
It is the "where am I part" that confuses me.
I guess it depends on whether you used the
NO_BLOCK keyword when you called XManager.

I'd guess from what you say happens that you
are probably at the stop where the interpreter
is looking for something to happen. That is
probably the main-level, or something very
much like it.

I'd be curious to know if your program still
works *without* the XManager call.

Cheers,

David

--
David W. Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438, E-mail: david@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155