Subject: Re: Standalone IDL applications?
Posted by davidf on Mon, 10 Jul 2000 07:00:00 GMT
View Forum Message <> Reply to Message

Michael Cugley (mjcugley@tigger.medschool.dundee.ac.uk) writes:

>> I'm guessing that you haven't make this dialog a MODAL
>> widget. You are probably relying on its ability to BLOCK
>> in your non-runtime version, but that isn't happening in
>> the run-time version (where by definition the "main" program
>> always blocks the IDL command line). Set the DIALOG_PARENT
>> keyword to the ID of your top-level base (or the widget you
>> are calling the dialog from).
>
> Okay, I have *no* idea why this should make the difference it does,
> nor why it would fail in such a completely uninformative way, but this
> does appear to be the solution :) Thank you very much!

Well, it was 5AM and I was hurrying to get to my tennis game. :-)

Let me see if I can explain it better now with a cup of coffee
in hand.

The DIALOG_READ_IMAGE has to "stop" and get input from
the user. Then it is suppose to (I guess, I've given up
on the documentation long ago) go read the image data
and return it when you destroy the widget.

In your non-runtime version of the program, the program
that calls DIALOG_READ_IMAGE is a non-blocking widget.
That is, you have access to the IDL command line when
you run it. When you get to the DIALOG_READ_IMAGE part
of your code, DIALOG_READ_IMAGE blocks the command line
and waits for input, because it has been written to be
a blocking widget if you don't supply a group leader for
its top-level base. (This is the purpose of the DIALOG_PARENT
keyword.)

So far, so good. Everything works normally. But it is only
the FIRST blocking widget that actually blocks the IDL
command line (naturally, there is no point in blocking the
IDL command line multiple times).

What happens in a run-time version of the program is that
the program that calls DIALOG_READ_IMAGE is "made" a blocking
widget by virtue of its being a run-time program. That is
to say, *all* main programs are by definition blocking widgets,
since you never see an IDL command line in a run-time program.

Hence, DIALOG_READ_IMAGE is the *second* blocking widget program
and it runs right through its block. It doesn't stop at all!

The only way to get it to stop is to make it a *modal*
widget program, rather than a *blocking* widget program.
But you can only make a modal widget if you have a group
leader specified for it. (Some people, even programmers
at RSI who should know better, create a "fake" top-level
base for this purpose, but there are rules on some
operating systems that says a group leader must be a
realized widget, so there will be a little tiny window
somewhere on the display while the real widget program is
on the display. Ugly, IMHO.)

But, as I say, this is all explained with examples, etc.
in several widget articles on my web page. :-)

Cheers,

David

P.S. I presume it fails in such an uninformative way because
it is not really a *mistake* to write a program like this.
It's just not a very good idea. And certainly not if you
want it to work properly. :-)

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

Subject: Re: Standalone IDL applications?
Posted by Michael Cugley on Mon, 10 Jul 2000 07:00:00 GMT

View Forum Message <> Reply to Message

davidf@dfanning.com (David Fanning) writes:

> I'm guessing that you haven't make this dialog a MODAL
> widget. You are probably relying on its ability to BLOCK
> in your non-runtime version, but that isn't happening in
> the run-time version (where by definition the "main" program
> always blocks the IDL command line). Set the DIALOG_PARENT
> keyword to the ID of your top-level base (or the widget you
> are calling the dialog from).

Okay, I have *no* idea why this should make the difference it does,
nor why it would fail in such a completely uninformative way, but this
does appear to be the solution :) Thank you very much!

--
Michael Cugley (mjcugley@medphys.dundee.ac.uk)

Subject: Re: Standalone IDL applications?
Posted by davidf on Mon, 10 Jul 2000 07:00:00 GMT

View Forum Message <> Reply to Message

Michael Cugley (mjcugley@tigger.medschool.dundee.ac.uk) writes:

> I'm developing an IDL GUI application; it's pretty much finished, I'd
> just like to turn it into a standalone application, mainly so my
> supervisor can play with it without taking up IDL licences.

Uh, well, *that's* not gonna happen. :-(

Even run-time applications take licenses. Sorry. No such
thing as an IDL executable for rift-raft like ourselves.
(Although the less cynical of us still hold out feeble hope.)

> I've followed the instructions in the hyperhelp, using IDLDE,
> compiling the project to a save file, and saving the runtime bits when
> asked. The thing starts up as expected, but there's a problem. The
> application is an image analysis one, and I can't seem to load in any
> images. The DIALOGUE_READ_IMAGE comes up, and I can pick the image
> and all (and a thumbnail appears), but then the application itself
> doesn't actually recieve the image. No error messages.
>
> Should this be important, it's IDL Version 5.3 on Sun Sparc Solaris.
>
> Is this a licencing thing? Has anyone else had any experience in this
> area?

This is no licensing thing. All my experience screams "PROGRAMMER
ERROR". :-)

I'm guessing that you haven't make this dialog a MODAL
widget. You are probably relying on its ability to BLOCK
in your non-runtime version, but that isn't happening in
the run-time version (where by definition the "main" program
always blocks the IDL command line). Set the DIALOG_PARENT
keyword to the ID of your top-level base (or the widget you

are calling the dialog from).

See this article for a fuller explanation of the difference
between blocking and modal widget behavior:

   http://www.dfanning.com/tips/modal_blocking.html

This is not your fault, by the way. The documentation and
code for Dialog_Read_Image (never on my "must use" list in
any case) is even worse than usual. :-(

Cheers,

David

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

Subject: Re: Standalone IDL applications?
Posted by Michael Asten on Tue, 11 Jul 2000 07:00:00 GMT

View Forum Message <> Reply to Message

Michael Cugley wrote:

> davidf@dfanning.com (David Fanning) writes:
>
>> I'm guessing that you haven't make this dialog a MODAL
>> widget. You are probably relying on its ability to BLOCK
>> in your non-runtime version, but that isn't happening in
>> the run-time version (where by definition the "main" program
>> always blocks the IDL command line). Set the DIALOG_PARENT
>> keyword to the ID of your top-level base (or the widget you
>> are calling the dialog from).
>
> Okay, I have *no* idea why this should make the difference it does,
> nor why it would fail in such a completely uninformative way,

A trap when using a runtime version (by this I mean literally the
Idlrt.exe Runtime module, not the idlde.exe development module with a
SAVEd program file) is that the command and log windows do not show on the
screen under RT. Thus any error in the runtime program is invisible. The
only solution as I understand it, is to trap errors explicitly and write
to a text widget, which is a lot of programming work. It is quite a

disincentive to using RT licences, and I have suggested to RSI that they
add a log window to the RT system to make error output (and any other
printed output) easier to achieve.

That said, the RT licences are cheap at a few hundred dollars, and provide
a reasonable solution to your need to put an executable version of your
code "on the bosses desk". Once the RT licence is installed, I have had
no difficulty in having colleagues run my SAVEd files just by double
clicking on the icon for the *.sav file. (Windows recognises a *.sav file
as being linked to idlrt.exe - I dont know if the solaris OS would need
further tweaking). Thus from the user viewpoint, the existence of the
idl runtime software is quite transparent.

Regards,
Michael Asten

> but this
> does appear to be the solution :) Thank you very much!
>
> --
> Michael Cugley (mjcugley@medphys.dundee.ac.uk)