Subject: Re: EGAD! A common block question.....
Posted by David Fanning on Wed, 22 May 2002 15:27:34 GMT
View Forum Message <> Reply to Message

paul (psisk@ball.com) writes:

> I realize common blocks are kind of a nasty subject around here,
> anyway, I'm a newbie to this whole IDL thing, so here is my
> question.....

Don't listen to all the nonsense. Common blocks are fine.
RSI uses them all the time in the IDL code they write. :-)

> I have multiple widget based tools that need to share properties
> (mostly directories and such), my first thought was to create a common
> block that holds a hashtable. Upon initialization, properties are
> read from a file and set in the common block. Also I wanted each tool
> to be able to set its own properties into the common block, therefore,
> the hashtable can grow with each tool initialization. This process
> seems to work fine in the development environment, but not in the
> runtime environment. Any suggestions? Comments? Alternatives?

Is your initialization routine a procedure, or is it a
main-level program. You won't be able to have any
main-level programs in a run-time environment. If it
is a procedure, I can't see why this won't work as
a run-time program. Your "main" routine calls
the initialization routine, then fires off the
tools, right?

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

Subject: Re: EGAD! A common block question.....
Posted by psisk on Thu, 23 May 2002 13:48:50 GMT
View Forum Message <> Reply to Message

The initialization routine for the common block is a procedure.
Basically there is a wrapper around the common block, the tools simply
call a procedure <Properties, filename> to initialize it. When data

is needed they call the function <PropertiesGet, prop>. It seems that whenever a new tool loads properties, it destroys any other properties in the common block, but only in the runtime environment.

I've tried another implementation using a system variable, although I have not had a chance to test it. Are system variables less error prone than common blocks?

thanks, paul.

Subject: Re: EGAD! A common block question.....
Posted by David Fanning on Thu, 23 May 2002 14:10:42 GMT
View Forum Message <> Reply to Message

paul (psisk@ball.com) writes:

> The initialization routine for the common block is a procedure.
> Basically there is a wrapper around the common block, the tools simply
> call a procedure <Properties, filename> to initialize it. When data
> is needed they call the function <PropertiesGet, prop>. It seems that
> whenever a new tool loads properties, it destroys any other properties
> in the common block, but only in the runtime environment.

Strange. I don't know what to make of that. But
the whole common block thing sounds like a red herring
to me.

> I've tried another implementation using a system variable, although I
> have not had a chance to test it. Are system variables less error
> prone than common blocks?

Error prone!? I don't think of common blocks as error
prone. I think of them as "limiting in the context of
widget programs". I doubt the problem is with common
blocks, though. I'd be looking somewhere else. Where,
I'm not sure, since my knowledge of the whole system
is so limited. :-(

You may have to call in a professional consultant,
but be prepared to pay the big bucks. No one wants
to look at someone else's code. :-)

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

Subject: Re: EGAD! A common block question.....
Posted by Pavel A. Romashkin on Thu, 23 May 2002 16:00:00 GMT
View Forum Message <> Reply to Message

paul wrote:
>
> I realize common blocks are kind of a nasty subject around here,
> anyway, I'm a newbie to this whole IDL thing, so here is my
> question.....

Allright, caught cheating. How'd you know that common blocks are nasty
if you were so new to IDL and this group? :-)
Anyway, like David said, there's nothing wrong with commons. There may
be a lot of inefficiency though with how one uses them :-)

> I have multiple widget based tools that need to share properties
> (mostly directories and such), my first thought was to create a common
> block that holds a hashtable. Upon initialization, properties are
> read from a file and set in the common block. Also I wanted each tool
> to be able to set its own properties into the common block, therefore,
> the hashtable can grow with each tool initialization. This process
> seems to work fine in the development environment, but not in the
> runtime environment. Any suggestions? Comments? Alternatives?

Again, in this sort of environment one thing I can suggest is trying not
to use common blocks for storing properties of widgets. Use a State
structure. If you are feel like you are up to it, use a State object or
a static shared pointer - if your widgets contain a lot of data to be shared.
Use common block only to pass information about *where* that State is,
not *what* it is.
Good luck,
Pavel

Subject: Re: EGAD! A common block question.....
Posted by psisk on Fri, 24 May 2002 15:34:56 GMT
View Forum Message <> Reply to Message

"Pavel A. Romashkin" <pavel_romashkin@hotmail.com> wrote in message
news:<3CED11FF.1A2E0026@hotmail.com>...

> paul wrote:
>>
>> I realize common blocks are kind of a nasty subject around here,
>> anyway, I'm a newbie to this whole IDL thing, so here is my
>> question.....
>
> Allright, caught cheating. How'd you know that common blocks are nasty
> if you were so new to IDL and this group? :-)
> Anyway, like David said, there's nothing wrong with commons. There may
> be a lot of inefficiency though with how one uses them :-)
>
Ok so I've done a bit of wandering around here the past week or
so.....

Anyway, I guess my question boils down to: Are there any fundamental
differences between IDL in the development environment and IDL in the
runtime environment. I have noticed in other posts that the !Path
variable is not defined in runtime, though this does not affect my
application...are there any other quirks?

Thanks again, paul.
