Subject: Re: Global variables and IDL
Posted by Martin Schultz on Tue, 20 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

David Fanning wrote:
>
> Walter Roberson (roberson@ibd.nrc.ca) writes:
>
>> Jim Russell wrote:
>>
>> :I'm certain that IDL has a global variable (I remember it being mentioned in
>> :Fanning's book), but don't remember how to invoke it. Maybe someone else can
>> :provide that for you.
>>
>> Use a common block. That requires adding only one extra statement to each
>> routine that uses an element of the common, and requires no other code
>> changes.
>
> I'm quite sure you didn't find a common block recommendation in
> Fanning's book. :-(
>

Here is my confession: yes, I use common blocks (and goto statements), and I don't even
feel too bad about it -- although it happens probably mostly for lack of knowledge of
better ways (perhaps Davids' third book will be able to change this?). It is defintively
true that you should avoid common blocks (and gotos) whenever possible (where "possible"
has to be defined in a feasible manner considering the available time etc.), and that
you should resort to UVALUES with pointers in widget applications as soon as there is
the slightest possibility that any one using these widgets will ever run more than one of
them (very few people probably write widgets when they intend a program exclusively for
personal use). But here are two examples where I used common blocks -- and I would be happy
to learn how I could have avoided them:

* in my EXPLORE tool (which can handle several "instances" at least if opened from within), I use
a common block to keep track of the drawing windows that have been opened and used. This is
needed to kill a window when the associated widget is closed as well as to open a new window

for a new widget instance. At first it would seem that I could simply
use the /free keyword to WINDOW and store the window number locally with
the widget, but I have to close *all* windows when I exit the program.
Should I use the event notification method? Sounds like a viable thing
-- but I would have to rewrite a large part of a running program which
my boss never likes ...

* in our new 3D model output analysis tool, we store data descriptors
for all data that has been read in a common block. This makes these data
available to all instances of the tool (although there currently is only
one and it is not even widgetized), and the common block avoids multiple
copies of large arrays which take up a lot of space and time when read.

Here are two little practice tips for common blocks:

(1) Make sure they are initialized correctly!
If you know where you encounter them the first time, you can write
something like

```
COMMON bla, thisdata

if (n_elements(thisdata) eq 0) then thisdata = ptr_new()
```

If not (i.e. you include your common blocks via @my_common , you can
call a specific init routine
with the above test from within your include file. To prevent an
infinite loop, you can write
something like this in my_common.pro:

```
COMMON myprobablyunnecessarycommonblock, thisdata, thatdata

if (strupcase(routine_name()) ne 'GAMAP_INIT') then gamap_init
```

Where the "routine_name()" function is available from my library (see
web site below).

(2) The use of @include files helps a great lot to avoid conflicting
definitions of common blocks which IDL doesn't like (you always have to
restart IDL when you change the structure of a common block).

(3) Become smarter than me and learn to use singletons and other methods
to avoid them ;-)

Martin.

--
---------- -------

Dr. Martin Schultz
Department for Engineering&Applied Sciences, Harvard University
109 Pierce Hall, 29 Oxford St., Cambridge, MA-02138, USA

phone: (617)-496-8318
fax : (617)-495-4551

e-mail: mgs@io.harvard.edu
Internet-homepage: http://www-as.harvard.edu/people/staff/mgs/
---------- -------