Subject: Re: Global variables and IDL
Posted by Martin Schultz on Wed, 21 Apr 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan wrote:
>
> In article <371CD72E.C77A38A7@io.harvard.edu>
> Martin Schultz <mgs@io.harvard.edu> writes:
> [...]
> Why not simply create a detached widget_base containing a widget_draw
> instead ...
Thanks Stein! That's sort of what I have in mind for a future version of
that program - including proper widget handling and a couple more
options
like selecting plot symbols etc. I might even end up venturing into
object
space and come up with some plot object as David adevrtizes on his page.
But first I need to finish this and that and ...

>
> But it's ok (in my humble opinion) to use common blocks to
> implement a system to keep track of such "global" data. I.e.,
> make two-three routines that share one (private) common block,
> along the lines of:
>
> REGISTER_ITEM,"NAME",DATA ;; Will "undefine" DATA,
> ;; to avoid copying
> DATAPTR = RETRIEVE_ITEM("NAME") ;; Returns a pointer to
> ;; the registered DATA item.

May not be as clean as this, but in principle that comes close to what I
am doing. I
should add one more tip for use of common blocks which is - as you say -

* limit common blocks to a few routines that handle the input/output of
data

Thinking a little more about this, true "global" variables may be a
better concept
although not supported in IDL. I think it would be nice to have the
flexibility of
normal IDL variables available for global variables as well (including
UNDEFINE ;-).
One example are option settings that are personal but (mostly) stable:
printer
paper size, preferred standard character size and font, maybe even a
sequence of

preferred plotting symbols and colors. You might argue that one can do this with
system variables, but I found them a little too inflexible: it's hard to re-assign
values (if you change the type or number of elements), and it is somewhat clumsy
to test if a system variable was already defined? If you type
   if (n_elements(!undefined) eq 0) then defsysv,'!undefined',0
your program will stop (you have to use
defsysv,'!undefined',exists=answer and then
query the result of answer).

  In my oppinion it would be nice to be able to explicitely declare variables as
global, such as
    GLOBAL glbarray=findgen(20)
This would also help the problem discussed some months ago of how to retrieve information
from widgets that are/were running in NO_BLOCK mode (e.g. data that has been manipulated).
Currently, this is another situation where you must use common blocks. The GLOBAL approach
would be more flexible in that the widget application could define the variables when needed,
and the program that wants to use that data can as always test for n_elements() gt 0. Also,
if there are several instances of the widget application, the latest (current) application
could overwrite the global variable by redefining it, or it could append to it, etc. These
things are not easily done with common blocks.

   For me, a computer program is somewhat like a house: you have certain fixed structures
like the doors and windows (which you can change but with some effort) and you have furniture
that you can move around as you wish. When you settle in a new home, you are likely to
change some of the fixed structures to accomodate your needs, but afterwards you will mostly
rearrange furniture. Yet, you still wnt to be able to use all the doors and windows that
are there.

   Thanks for this helpful discussion,
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/
-- -------