Subject: Re: global variables again - and passing out common blocks
Posted by J.D. Smith on Tue, 28 Apr 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Martin Schultz wrote:
>
> Hi,
>
> This is how fast things change. Here I was doubting the necessity to
> pass global variables back from a routine, and now this is what I am
> stuck with:
>
> I tried to add two features to my EXPLORE tool, but it turns out that
> only one of them works at a time:
> (1) upon exit: extract the data of the last active window, store it as a
> heap variable in a common block and return it to the main program (if a
> named variable was passed as a keyword)
> (2) use the /NO_BLOCK keyword to xmanager to have the command line
> available
>
> The trouble lies in the program flow: without /NO_BLOCK, IDL
> continues program execution after the call to xmanager when the widget
> is closed, so there is a chance to do additional things in the
> mainprogram (like retrieving the pointer information). However, with
> /NO_BLOCK set, IDL runs through the main program (pro explore)
> completely, and when exiting the widget application, you only get a call
> to the cleanup routine, but there is no way to pass anything out any
> more except "activating" the common block EPOINTERS in $MAIN$ which
> leads to a bit of trouble described below and makes it mandatory to
> manually free the pointers if you don't need them. Any better idea is
> greatly appreciated.
>
>
> Another problem, which I consider a real bug, is related to common
> blocks: if you define a common block inside a subroutine and you want to
> make it accessible to $MAIN$, you are in bad luck if you had issued a
> HELP command on any variable in the common block before. Example:
>
> pro test
> common testcom,a
> a=10
> end
>
> IDL>test
> IDL>help,a
> A UNDEFINED = <Undefined>
> IDL>common testcom
> % A is already defined with a conflicting definition.

> ; if it says "undefined" before, how can it be "defined" now ?
> ; It works after delvar,a !
>

A solution similar to the one I'm forced to use is to create a procedure, call it "import", which sources the common block, looks inside to see if a valid data structure was put there, and pushes the data back to main through a passed named variable.

Of course, this requires an additional procedure call by the user, and points out another valid example of when it would be useful to be able to push and pull data from the main level. Imagine if a simple call, say in the "cleanup" routine for your widget, could export the relevant data onto $MAIN$, if requested.

JD

--

J.D. Smith |*| WORK: (607) 255-5842
Cornell University Dept. of Astronomy |*| (607) 255-4083
206 Space Sciences Bldg. |*| FAX: (607) 255-5875
Ithaca, NY 14853 |*|