Subject: Re: Returning result from a widget program.
Posted by David Foster on Thu, 09 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Imanol Echave wrote:
>
> Hi people:
>
> I'm writting an IDL function that works with widgets. The user calls an IDL
> function that shows an widget interface to input data. When the user pushes the
> "OK" button the function has to return a result that depends on the input data.
> My problem is where to store this result to return it. I can't use the UVALUE of
> the widgets because when I want to return the result the widgets are destroyed.
> Any advice?

Hi Imanol -

I think the best thing you can do is to first create a "state info"
structure that contains contains useful information about your
widget, including status and especially the "result" from your
user input. Then create a pointer to this structure:

```
 statePtr = ptr_new( { status: 0, status_msg: '', result: 0 } )
```

Then store this pointer in the UVALUE of the TLB (top-level-base) of
your widget heirarchy, right before calling XMANAGER:

```
 widget_control, base, set_uvalue=statePtr

 XMANAGER, 'name', base
```

(Be SURE not to use /NO_BLOCK here! You want the function to wait
until you're done.)

Then at the beginning of your event handler you get this pointer
back from the UVALUE:

```
 widget_control, event.top, get_uvalue=statePtr
```

and you can access all of your structure elements within the event
handler like:

```
 temp = (*statePtr).result
```

Then, at the end of your function, to return the result(s) of the
function you add the next lines *after* the XMANAGER call:

```
 return_value = (*statePtr).result
```

```
ptr_free, statePtr          ; Be SURE to free memory!
return, return_value
END
```

Of course, if you need to return more items, you can make a new
structure from some of the fields of (*statePtr), and then return
that instead. The point is that you can use this method to return
whatever you like. The key is that I'm assuming that this will be
a "modal" widget, which will wait until the user presses "Ok"
before returning (keyword /NO_BLOCK is not used in XMANAGER call).

Hope this helps.

Dave

--

~~~~~~ ~~~~~~~
David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2240
La Jolla, CA 92037
~~~~~~ ~~~~~~~

Subject: Re: Returning result from a widget program.
Posted by davidf on Fri, 10 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Hi Folks,

I've just returned from a short holiday where I have resolved, once
again, to give up this strange IDL fetish I have and get on with the
rest of my life. But, alas, my work remains unfinished...

Imanol Echave (ccaeccai@sc.ehu.es) writes:

> I'm writting an IDL function that works with widgets. The user calls an IDL
> function that shows an widget interface to input data. When the user pushes the
> "OK" button the function has to return a result that depends on the input data.
> My problem is where to store this result to return it. I can't use the UVALUE of
> the widgets because when I want to return the result the widgets are destroyed.
> Any advice?

And he receives advice like this. Sigh...

> If you need the parameter var in an another program (or in an

> another subrutine) I suggest to put it in a common block.

Even Mirko Vukovic (who I sent my book to, for goodness sake!)
recommends a common block.

Only David Foster offers a word of caution about common
blocks in widget programs.

Let me put it this way. From time to time widget programs
are very useful. So useful that you might even want to use
several instances of that program at the same time. (A
dialog widget function that collects information about which
file to open comes to mind, or a program to load colors
in a particular window, or a program that processes images
in a particular way, etc, etc. In fact, just about every
widget program I write meets the criteria.)

But if you use a common block in that widget program, you
can only run ONE INSTANCE of that program at any particular
time. If you run more than one instance, your programs will
not work properly. Period. This makes common blocks in widget
programs a lousy choice in my humble opinion.

So it is important to know how to write widget programs
WITHOUT common blocks. In this case, Mr. Echave is absolutely
correct. He cannot use the UVALUE to store information he
collects from the user. Because by the time he collects it
and destroys the modal widget, the user value (and everything
stored there) is gone. He can't get it back to return it
as the result of the function. He must store the information
in a global location that is *external* to the widgets used
to create the program. A pointer location is *exactly* what
is called for, as David Foster indicates.

The last several lines of the data collection function
might look like this:

```
  dataPtr = Ptr_New({cancel:1})
```

(My data pointers usually point to a structure that contains
the information I hope to collect from the user. This might
be the name of a data file, the type of data stored there,
the size of the data, etc. I like to have a field in that
structure that tells me if the user hit the CANCEL button
on my dialog. If so, the CANCEL field in the structure is set
to 1. When I set up the data pointer I usually turn this
CANCEL flag ON so that all I really have to worry about is
if the user hit the OK or ACCEPT button. This keeps me from

coming to grief if the user just kills the widget with his
or her mouse instead of using the thoughtfully provided
buttons.)

```
  info = {dataPtr:dataPtr, ..., ...}
  Widget_Control, tlb, Set_UValue=info, /No_Copy
  XManager, 'example', tlb   ; Modal widget blocks here.

     ; User killed widget. Get data and return it.

  data = *dataPtr
  Ptr_Free, dataPtr
  IF data.cancel THEN RETURN, -1 ELSE RETURN, data
  END
```

You can see how this works in more detail by looking at
the programs GETIMAGE or GETDATA from my anonymous ftp
site:

ftp://ftp.dfanning.com/pub/dfanning/outgoing/coyote/getimage .pro
ftp://ftp.dfanning.com/pub/dfanning/outgoing/coyote/getdata. pro

If you have trouble with the programs (they are well documented),
you can read the last two chapters in my book. They talk about
how to build both modal and non-modal widget dialogs without using
common blocks.

If I don't hear the words "common block" mentioned here for
at least two weeks, I know it will be safe to retire. :-)

Cheers,

David

--
David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438
Coyote's Guide to IDL Programming: http://www.dfanning.com/

Subject: Re: Returning result from a widget program.
Posted by David Foster on Fri, 10 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

mirko_vukovic@notes.mrc.sony.com wrote:
>

> Imanol Echave <ccaeccai@sc.ehu.es> wrote:
>> Hi people:
>>
>> I'm writting an IDL function that works with widgets. The user calls an
> IDL
>> function that shows an widget interface to input data. When the user pushes
> the
>> "OK" button the function has to return a result that depends on the input
> data.
>> My problem is where to store this result to return it. Any advice?
>
> common block, pointer variable?
>
> You can setup a widget cleanup routine (and frankly, I forget their syntax
> now), where you can do that kind of stuff.
>
> mirko
>

This is definitely the easiest approach, but it's messy and you'll
regret it later. There is almost always a way to avoid common blocks,
like my recent reply to Imanol.

Dave

--

~~~~~~ ~~~~~~
David S. Foster Univ. of California, San Diego
Programmer/Analyst Brain Image Analysis Laboratory
foster@bial1.ucsd.edu Department of Psychiatry
(619) 622-5892 8950 Via La Jolla Drive, Suite 2240
La Jolla, CA 92037
~~~~~~ ~~~~~~

Subject: Re: Returning result from a widget program.
Posted by DAMIANO ZILIO on Fri, 10 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

Imanol Echave wrote:

> Hi people:
>
> I'm writting an IDL function that works with widgets. The user calls an IDL
> function that shows an widget interface to input data. When the user pushes the
> "OK" button the function has to return a result that depends on the input data.
> My problem is where to store this result to return it. I can't use the UVALUE of

> the widgets because when I want to return the result the widgets are destroyed.
> Any advice?

Why don't you try with get_value=var?
If you need the parameter var in an another program (or in an ather subrutine) I suggest to
put it in a common block.

I hope that helps
--

Damiano Zilio

Subject: Re: Returning result from a widget program.
Posted by mirko_vukovic on Fri, 10 Jul 1998 07:00:00 GMT
View Forum Message <> Reply to Message

In article <35A4BF4B.5FE452DE@sc.ehu.es>,
Imanol Echave <ccaeccai@sc.ehu.es> wrote:
> Hi people:
>
> I'm writting an IDL function that works with widgets. The user calls an IDL
> function that shows an widget interface to input data. When the user pushes the
> "OK" button the function has to return a result that depends on the input data.
> My problem is where to store this result to return it. I can't use the UVALUE of
> the widgets because when I want to return the result the widgets are destroyed.
> Any advice?
>

(I guess it is my turn to show what I've learned about widget programming)

common block, pointer variable?

You can setup a widget cleanup routine (and frankly, I forget their syntax now), where you can do that kind of stuff.

mirko

you

-----== Posted via Deja News, The Leader in Internet Discussion ==-----