Subject: Re: Widgets: group leader and procedures
Posted by Robert Moss on Sun, 04 Jan 2004 12:45:17 GMT
View Forum Message <> Reply to Message

In the case you have described, you can simply use event.top as the group leader for the new top level base. Here event.top is the tlb created in your main procedure.

```
;---------------------------------------
  PRO analyse_simset_data, event
     widget_control, event.top, get_uvalue = pstate

     base2=widget_base( title = 'Simset Analysis', $
   group_leader = event.top )
     <...etc...>
  END
;---------------------------------------
```

A bit of unsolicited advice: you had the following:

```
     state={ bunch of stuff}
     pstate = Ptr_new(state, /no_copy)
     widget_control, tlb, set_uvalue = pstate
```

I would do this instead:

```
     state = {bunch of stuff}
     widget_control, tlb, set_uvalue = state, /no_copy
```

There is no real need to stuff your entire widget state into a pointer. If you have dynamic data structure elements or large array elements in the state, make those individual elements pointers, but not the entire state structure. This will save you from typing a lot of extra asterisks when you access the state later. Your mileage may vary. Batteries not included. May contain nuts.

Robert Moss, PhD

Karthik wrote:
> Hello All,
>
>   I have a main programme with a widget base and it includes a widget
> button called "Alalyse Data" - when the user clicks on it I would like
> to execute the bit of code called "alalyse_simset_data", so the thing
> looks like this (not exactly the same but the code below is illustrative)
>
> ;---------------------------------------

```
> Pro main_programme
>    tlb      = widget_base()
>    analyse  = widget_button(row5, value = 'Analyse Data', $
>          Event_PRO= 'analyse_simset_data')
>    <etc>
>       widget_control, tlb, /realize
>
>    state={ bunch of stuff}
>    pstate = Ptr_new(state, /no_copy)
>    widget_control, tlb, set_uvalue = pstate
>       xmanager, 'make_random_activity', tlb
> END
> ;----------------------------------------
>
>
> Now I would like, for clarity, the analyse_simset_data bit of code to be
> a completely independent file and I would like that to contain a widget
> as well.
>
> ;----------------------------------------
> PRO analyse_simset_data, event
>    widget_control, event.top, get_uvalue = pstate
>
>    base2=widget_base(  ?????????? )
> END
> ;----------------------------------------
>
>
>    The ????? marks refer to my doubts on the subject: how do you pass
> on the group leader information in the state pointer so that killing the
> main widget from main_programme kills the widget created in
> analyse_simset_data as well? Is the group leader information a string?
> Are there any rules I should follow when a procedure invoked my a main
> programme creates a widget? Any advice/suggestion would be greatly
> appreciated.
>
>    Thanks,
>
> Karthik.
>
```

Subject: Re: Widgets: group leader and procedures
Posted by David Fanning on Sun, 04 Jan 2004 15:26:18 GMT
View Forum Message <> Reply to Message

Karthik writes:

> I have a main programme with a widget base and it includes a widget
> button called "Alalyse Data" - when the user clicks on it I would like
> to execute the bit of code called "alalyse_simset_data", so the thing
> looks like this (not exactly the same but the code below is illustrative)
>
> ;--
> Pro main_programme
> tlb   = widget_base()
> analyse = widget_button(row5, value = 'Analyse Data', $
>   Event_PRO= 'analyse_simset_data')
> <etc>
>   widget_control, tlb, /realize
>
> state={ bunch of stuff}
> pstate = Ptr_new(state, /no_copy)
> widget_control, tlb, set_uvalue = pstate
>   xmanager, 'make_random_activity', tlb
> END
> ;--
>
>
> Now I would like, for clarity, the analyse_simset_data bit of code to be
> a completely independent file and I would like that to contain a widget
> as well.
>
> ;--
> PRO analyse_simset_data, event
> widget_control, event.top, get_uvalue = pstate
>
> base2=widget_base(??????????)
> END
> ;--
>
>
> The ????? marks refer to my doubts on the subject: how do you pass on
> the group leader information in the state pointer so that killing the
> main widget from main_programme kills the widget created in
> analyse_simset_data as well? Is the group leader information a string?
> Are there any rules I should follow when a procedure invoked my a main
> programme creates a widget? Any advice/suggestion would be greatly
> appreciated.

You could simply store the group leader ID in the state structure
so you could retrieve it and use it when you create your second
top-level base, but this is not usually necessary. Almost always
the new group leader is either event.top or event.id. In your
case, event.top will do (although you could use either, the group
leader widget does not have to be a base widget):

base2=widget_base(group_leader=event.top, ..._

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
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: Widgets: group leader and procedures
Posted by Jeff Guerber on Wed, 07 Jan 2004 03:56:53 GMT
View Forum Message <> Reply to Message

On Sun, 4 Jan 2004, Robert Moss wrote:

> A bit of unsolicited advice: you had the following:
>
> state={ bunch of stuff}
> pstate = Ptr_new(state, /no_copy)
> widget_control, tlb, set_uvalue = pstate
>
> I would do this instead:
>
> state = {bunch of stuff}
> widget_control, tlb, set_uvalue = state, /no_copy
>
> There is no real need to stuff your entire widget state into a pointer.

I disagree: If you use a pointer for the widget state, you can change its contents and not have to worry about stuffing it back into the uvalue before you exit the event handler. Plus you won't be constantly making copies of everything in your state structure, which can be an advantage if it's large. Just remember to have the TLB's cleanup routine free pstate (I call it statep, myself). After I started writing widgets, I very quickly switched over to state pointers (until I discovered object widgets, that is!).

Jeff Guerber

Subject: Re: Widgets: group leader and procedures
Posted by David Fanning on Wed, 07 Jan 2004 04:20:49 GMT

Jeff Guerber writes:

> I disagree: If you use a pointer for the widget state, you can change
> its contents and not have to worry about stuffing it back into the uvalue
> before you exit the event handler. Plus you won't be constantly making
> copies of everything in your state structure, which can be an advantage if
> it's large. Just remember to have the TLB's cleanup routine free pstate
> (I call it statep, myself). After I started writing widgets, I very
> quickly switched over to state pointers (until I discovered object
> widgets, that is!).

If you are going to the trouble of putting it back
in the UValue of the TLB, then a simple NO_COPY
will prevent any copying of the state structure
as you move it into and out of an event handler.

I personally prefer to write the NO_COPYs for
simple programs rather than deal with the cumbersome
pointer syntax. But there are distinct advantages to
a pointer in more complicated programs where the state
structure sometimes has to be passed around to programs
outside the confines of the normal single widget program.

But if the pointer syntax doesn't push a user quickly to
object widgets, then the user has a LOT more patience than
I have. :-)

Cheers,

David
--
David W. Fanning, Ph.D.
Fanning Software Consulting, Inc.
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