Subject: Re: How to get values from a widget
Posted by davidf on Mon, 24 Nov 1997 08:00:00 GMT
View Forum Message <> Reply to Message

I~nigo Garcia (iruiz@astro.rug.nl) provides us with our Monday
morning puzzle when he writes:

```
>  I have a program that can plots several things in a Postcript file, and
>  I want to ask the user which ones he wants to plot, via a widget
>  interface. I want to make a call like to this widget, and know which
>  plots are desired, but I can not get it.
>
>  Here follows a part of the code:
>
>>  n=10
>>  base = widget_base(title='OPTIONS FOR THE ANALYSIS',/modal,/col,group_le=par,$
>>              /floating,frame=3)
>>  but = widget_base(base,/col,/nonexclusive,/frame)
>>  for iii=0,n-1 do begin
>>     b(iii)=  widget_button(but,event_pro='widget_std_event',val=b_str(iii ),uval=1)
>>     widget_control,b(iii),/set_button
>>     options(iii) = 1
>>  endfor
>>  ok = widget_base(base,/row)
>>  for iii=0,1 do fin(iii) = widget_button(ok,val=fin_str(iii),uval=0)
>>  state = {b:b,options:options,fin:fin}
>>  widget_control,/realize,base,set_uvalue=state
>>  xmanager,'widget_std',base
>>  print,state.options
>
>  In the "options" variable I store which figures are going to be plotted
>  (option(i)=1, or not (option(i)=0). This variable is changed via the
>  buttons in the event handler. But when I destroy the widget, and print
>  the state.options, I get the initial values (all of them 1). How can I
>  get information that was created within a widget ??
```

I'm hoping that there are at least 17 more people on the
newsgroup this week who can answer this question for I~nigo
after taking my class last week. :-)

The problem I~nigo faces is a common one for people who
write what I call "pop-up dialog widgets" without common
blocks. Where in the world can he store the information he
wants to collect from the user so he can get it back into
his program?

The answer is clearly not in the state structure, because
this structure is destroyed when the modal dialog widget is

destroyed. In fact, the information the program collects from the user has to be stored *outside* the widget somehow. The answer is clearly common blocks. (Just kidding. I wanted to see if anyone is paying attention. :-)

The answer is that information collected from the user has to be stored in some global area of memory, like the heap memory allocated with a pointer (or handle if you are using IDL 4.0).

Instead of putting the "options" array in the state structure, I~nigo should put it at a pointer location and put the *pointer* in the state structure. Like this:

```
ptr = Ptr_Free(options)
state = {b:b,ptr:ptr,fin:fin}
```

Then, as the user clicks on buttons to select options, his error handler code should be updating this array:

```
(*state.ptr)[selection] = 1
```

Now, when his widget is destroyed, releasing the modal property, he comes back into his widget definition module, just following the XManager command. But this time, the user-selected information is available to him. His code will look something like this:

```
xmanager,'widget_std',base
goodOptions = *ptr
Ptr_Free, ptr
Print, goodOptions
```

That should do it. Look at the program GetImage from my web page for an example of how this can be done.

You can get more details in my book, which--God and weather willing--should be available by the first of next week. :-)

Cheers,

David

David Fanning, Ph.D.
Fanning Software Consulting
E-Mail: davidf@dfanning.com
Phone: 970-221-0438