
Subject: Re: widget cleanup problem

Posted by [Braedley](#) on Tue, 05 Dec 2006 15:50:17 GMT

[View Forum Message](#) <> [Reply to Message](#)

David Fanning wrote:

> Paul van Delst writes:

>

>> I'm having a bit of issue with cleaning up after myself in a widget program.

>>

>> I have a main GUI that, alongside "regular widgets" also contains a bunch of standalone
>> compound widgets. The main GUI info structure and loading looks something like:

>>

```
>> Info = { Debug      : Debug  , $  
>>      cw1Id       : cw1Id   , $ ; compound widget 1 base id  
>>      cw2Id       : cw2Id   , $ ; compound widget 2 base id  
>>      cw3Id       : cw3Id   , $ ; compound widget 3 base id  
>>      ....etc....  
>>      cwNId       : cwNId   } ; compound widget N base id
```

```
>> InfoPtr = PTR_NEW( Info )
```

```
>> WIDGET_CONTROL, tIBaseId, SET_UVALUE = InfoPtr
```

>>

>> To get a look at the compound widgets info structure I do the following:

>>

```
>> WIDGET_CONTROL, Info.cw3Id, GET_UVALUE = cw3_InfoPtr
```

>>

>> My problem is that when an exit event occurs and the cleanup routine is called, the child
>> compound widgets are cleared first so the reference to their top-level-based is gone and I
>> now have dangling pointers.

>>

>> In my main "exit" event handler, all the various compound widget base ids are still valid.
>> However, by the time the "cleanup" routine is called, they are not -- and thus I can't
>> free the info pointers.

>>

>> I wanted the main "cleanup" routine to handle all the child compound widget pointer
>> free'ing. Does it *have* to be done in the main "exit" event handler?

>>

>> I hope my explanation above makes sense.

>

> People here are just entirely too optimistic today!

>

> With these kind of compound widgets, what we usually do
> is use KILL_NOTIFY to assign a callback to the TLB of the
> compound widget. (This isn't really a TLB, but you know
> what I mean.) When that base widget dies, you enter your
> "cleanup" routine for that compound widget. (This means
> you will have to write one for it.) This is where you
> free your pointer.

>

> This is *identical* to the CLEANUP routine you assign to
 > the application's TLB, except that you can't (supposedly)
 > use KILL_NOTIFY to assign a cleanup routine to a widget that
 > is being managed directly by XManager. You have to use the
 > CLEANUP keyword on XManager. (I see ITTVIS programmers break
 > this rule all the time, so I presume it's not enforced like
 > it used to be, but I still teach it this way. And it still
 > works.)
 >
 > Cheers,
 >
 > David
 > --
 > David Fanning, Ph.D.
 > Fanning Software Consulting, Inc.
 > Coyote's Guide to IDL Programming: <http://www.dfanning.com/>
 > Sepore ma de ni thui. ("Perhaps thou speakest truth.")

I have come to realize that I have a similar problem with a compound widget I created. Some sample code:

```
pro cw_listSelect_cleanup, id
widget_control, widget_info(id, /child), get_uvalue=state
;pointer cleanup
ptr_free, state.initial_value, state.eding_value
```

```
widget_control, /destroy, id
end
```

```
function cw_listSelect, parent, value=value, $
  uvalue=uvalue, uname=uname, $
  sensitive=sensitive, frame=frame, $
  title=title, xsize=xsize, ysize=ysize
```

```
base=widget_base(parent, col=1, $
  uvalue=uvalue, sensitive=sensitive, $
  frame=frame, uname=uname, title=title, $
  kill_notify='cw_listselect_cleanup', $
  pro_set_value='cw_listSelect_set', $
  func_get_value='cw_listSelect_get')
```

```
label_base=widget_base(base, /row)
if n_elements(title) eq 1 then $
  wTitle=widget_label(label_base, value=title)
```

```
;some irrelevant code
```

```
initial_value=ptr_new(value)
```

```
ending_value=ptr_new([''])
move_info={initial_list:initial_list, ending_list:ending_list, $
initial_value:initial_value, ending_value:ending_value}
```

```
;some more irrelevant code
```

```
widget_control, label_base, set_uvalue=move_info
```

```
return, base
end
```

My problem comes when I try to destroy the widget. I actually do make it into the cleanup routine, but apparently label_base has already been destroyed because I get the following error:

WIDGET_CONTROL: Invalid widget identifier: 0.

when I try to get the uvalue. I still want the user to be able to set a uvalue for this compound widget, so setting the uvalue of base to move_info won't work. Any suggestions?
