Subject: OOP Programming, Was: Something Else
Posted by davidf on Wed, 04 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Reinhold Schaaf (Kakadu42@gmx.net) writes:

> As a second point, one
> really has to get used to the fact that objects cannot be created on the
> stack, (stack objects are destructed automatically on exit from the
> program unit in which the object was constructed). Hence one is forced
> to destruct any object manually, which is highly error-prone and makes
> life really uncomfortable.

I am finding this to be the case too. Having learned my lesson with the object graphics programs I naively offered up as "examples", I am trying to be extra careful with memory management. But I currently have six pointers to objects hanging around and I will tell you they are damned hard to track down! There must be a better way, or at least some way to get more information about them besides sprinkling a thousand Print statements over your code.

> A third remark concernes event handling: It is not possible to define a
> method of a class as the event handler of a widget.
> As a consequence, one is forced to make the event handler a global
> function. But global functions cannot access members of objects, so one
> has to add methods to the class which would be unnecessary otherwise.

I have grappled with this, too. Having all of a sudden become enamored with object programming, it is not much of a leap to believe that this is the way widget programs should work. Especially compound widgets. Like Reinhold, I've found imperfect, interim solutions.

> Things could be ways simpler if global functions were allowed to be
> event handling routines! I wonder whether this could be changed in
> future versions of IDL.

I haven't specifically discussed this with the folks at RSI (although I plan to), but I think OOP is probably in the same state that widgets were in when they were first introduced. That is to say, the first effort was pretty good, but nowhere near where they had to get to eventually.

As we gain experience with object programming I think we need to let RSI know what works and what can be done better with these objects. (J.D. Smith has already offered a number of excellent suggestions.) I really do think RSI is listening to customers in a more focused way than in the past. This

could be the time to really take IDL several steps forward
as a language.

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: OOP Programming, Was: Something Else
Posted by davidf on Fri, 06 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Stein Vidar Hagfors Haugan (steinhh@ulrik.uio.no) writes:

> The observant reader will of course have noticed that if this
> scheme is followed *consistently*, you only need *one* event
> function (outside the individual class scopes, that is!) for
> *all* of your "objectified" compound widget types!!

The really observant reader will also notice that the widget
programming sections of my recently released book are rapidly
becoming obsolete. :-(

On the other hand, I am thinking there may be a few people
who would pay to have this method explained to them. Especially
when they see the powerful new programs that can be written
in a few days time using it. :-)

I'm suppose to go skiing next week and I promised my wife
I'd leave my computer home, but...I don't know. This is
awfully exciting stuff.

Cheers,

David

P.S. Let's just say I've changed my mind this morning about
getting the house cleaned up. :-)

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: OOP Programming, Was: Something Else
Posted by steinhh on Fri, 06 Mar 1998 08:00:00 GMT
View Forum Message <> Reply to Message

Just one thing about this kind of solution (from Mark Rivers):

```
> pro mca_display_event, event ; event processing for MCA application
>
>     widget_control, event.top, get_uvalue = mca_display
>     mca_display->event, event
> end
```

I realize(d) (after I started to write this message :-) that this is
the event processing for the entire application.... But for compound
widgets, etc, it is customary (and a good custom, too!) to leave
the compound widget ID's UVALUE to the "owner's" discretion. Thus,
in general, it is wise to store the object pointer on the uvalue
of the *first child* of the event.id.

I.e., in a compound widget event *function* (allowing the event to be
modified and sent further up the hierarchy, or being gobbled up by
sending along a zero value) one should do something like (untested
code follows!):

```
function cw_object_event,event
     widget_control,widget_info(event.id,/child),get_uvalue=objec t
    return, object->event_func(event)
end
```

The observant reader will of course have noticed that if this
scheme is followed *consistently*, you only need *one* event
function (outside the individual class scopes, that is!) for
all of your "objectified" compound widget types!!

Neat, if I may say so myself.

Darn, why can't I have some time to spend actually *doing* this
OO stuff, instead of just writing about it!

Regards,

Stein Vidar