Subject: Re: /ALIAS and drawing in different draw_widgets
Posted by David Fanning on Sat, 03 Aug 2002 20:24:14 GMT
View Forum Message <> Reply to Message

lyubo (lzagorch@cs.wright.edu) writes:

> I have the following situation:
> Model M1 contains objects A,B, and C
> Model M2 contains an alias of A
> Model M3 contains an alias of B
> Model M4 contains an alias of C
> The models are added to different views and the views are drawn in
> different draw widgets.
>
> The problem comes when I update the objects (A,B, and C) and try
> to redraw the views. It looks like I can't redraw M1,M2,M3,and M4
> correctly in that sequence. Redrawing only M1 works fine, redrawing
> M2,M3, and M4 (without M1) also works, but if I try to redisplay M1,
> M2,M3,and M4, M1 has only Cs.
> Furthermore, there is a memory leakage when I redraw M1. I guess
> that it has something to do with the aliases, but I have no idea what
> could be causing that. What exactly happens when you add an alias
> of an object to a model? Isn't it just a copy of the same object? If it
> isn't which one gets updated first - the alias or the object?

I don't have too many specific suggestions, except that
in my experience about 99% of the odd things that happen
when you are programming in object graphics are caused by
programming mistakes rather than than the software. I would
always look there first.

Having said that, this whole notion of aliases is interesting
to me. It occurred to me that Dave Burridge and I have
developed something very much like it in the object widget
system we have developed. In fact, it is so much like it that
I wonder if we didn't independently come up with the same
solution RSI uses.

(This is not the first coincidence that has caused me to think
this. Just watching the system perform sometimes brings the
object graphics system to mind. At the very least, I like to
think building this system has brought a little deeper insight
into how an object system is suppose to work in general.)

In any case, in our system we have the notion of an object
hierarchy, based on container objects. There is a container
at the top, it has children, they have children, etc. When
you create an object you tell it who its "parent" is. All this

is very much like widget programming.

But, of course, you might create an object, say an image object, that you would like to belong to two different draw widget objects, to give a simple example. You wouldn't want to duplicate or copy the object, because then you would have to duplicate the data and this would be poor memory management. (Let's say, for sake of argument, that the image was 3.4 GBytes in size.)

In our system, only one object can be the "parent" object, but you can add a reference to that object to another container. In the IDL vernacular, this would be the "alias" object. But now we have the problem of who controls this object, and more specifically, who destroys it? Clearly you don't want the image object destroyed until everyone else is finished using it.

We have solved the problem by reference counting. Every time the object is "added" to a container, be it the parent object or some other object, its reference count is increased by one. When the object is "destroyed" (say by having one of the containers that holds it destroyed), we intercept this and decrement the object's reference count. An object is only really destroyed when its reference count goes to zero.

So, if the IDL system works anything like ours, what happens when you add an alias object to a model is that you simply add the object reference to the model container. I suspect that both models are then working with the same object. If you change it one place, you change it somewhere else.

Why this fails to do what you want it to do in your program is a mystery to me. But then a LOT of object programming is a mystery to me, even when I've just spent the whole day writing the damn things. :-)

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