Subject: Re: Object Madness or Restoring Nightmares
Posted by David Fanning on Wed, 03 Mar 2004 02:52:40 GMT
View Forum Message <> Reply to Message

Rick Towler writes:

> Could you maybe speculate as to what was/is happening?
>
> I was writing a response to the effect that I have seen this before too when
> saving and instance of IDLgrModel which contains IDLgrGraphic atoms. When I
> destroyed the model, (some of? all of? can't remember) the atoms were left
> strewn about.
>
> My "fix" was to extract the atoms and then save them. When restoring I add
> them to a fresh instance of IDLgrModel.
>
> I am curious as to why this is...

Well, this gets curiouser and curiouser. I guess the problem
is back. I don't know why I thought it had gone away.
Wishful thinking, probably. Or maybe a time warp of some
kind. Anyway... it's back.

Now here is the thing. This is a large application.
Everything is an object including the "application"
object.

So, the object that I am saving is a small object. It
contains three other container objects inside it,
and one of those containers contains three image objects.
So, all together, maybe a dozen objects and pointers.

Now, recall that I save it like this from within a SaveSession
method:

```
theStudy = self.currentStudy
Save, theStudy, Filename='somename.sav'
```

I restore the object like this in a RestoreSession method.

```
Obj_Destroy, self.currentStudy
Restore, Filename='somename.sav'
self.currentStudy = theStudy
```

I can get to the same place in the program either by running
a new study, in which I read some data files, etc. Or by
restoring a study. The same objects are created. The interface
looks identical.

I can destroy the main object at this point. If I read the
data to get here, I am completely clean. If I restored the
small study object to get here, I have--are you ready for this--
908 pointers and 784 objects left on the heap!!!!!

If I look at these objects and pointers I notice that my
entire application is left on the heap, even though I have
just destroyed it! In fact, the very first object on the
heap is my main program object. How can this be?

Here is a clue that may be too gruesome to contemplatee.
Every object in this system is a subclass of an IDLgrComponent
object. Lord help us, if *that* turns out to have anything to
do with this!

Cheers,

David
--
David Fanning, Ph.D.
Fanning Software Consulting
Coyote's Guide to IDL Programming: http://www.dfanning.com/