Subject: Re: Saving an application state
Posted by David Fanning on Mon, 21 Jun 2004 15:16:40 GMT
View Forum Message <> Reply to Message

Rob Dimeo writes:

> We have had a number of requests from our software users to be able to
> save the "state" of the GUI application prior to closing it so that
> users can restore it at a later time. The GUI applications are
> typically written as object widgets that can contain many other
> objects. The relationship with the the GUI and other objects can be
> has-a and is-a. I know that there are certain tricks related to
> saving objects and restoring them (as discussed on this NG) but I
> don't have that much experience doing this myself. I am probably
> asking too much but....is there a straightforward way to modify
> existing object widgets that allows the user to save and restore the
> widget state?

I wish. :-(

I think the short answer is iTools.

What RSI has done in iTools is decouple the GUI representation of the object from the object itself. This is a fantastic idea. It means you can easily store the state of an object application and restore it later without having to think about widget identifiers. Of course, the price of this feature is an enormous increase in complexity, which I think makes iTools so complicated no one outside of RSI can use it. (Which, I have been thinking lately, must really be the Plan. :-)

In simple object widget applications, it is easy enough to simply divorce the graphical part of the object from the rest of the program (probably via a GUI method that simply builds the graphical user interface). "Saving" the state then simply becomes saving everything *except* the GUI.

I find, though, that it works better in theory than it does in practice. In practice, at least using the widget objects I use daily, objects are so intertwined that saving an object often means creating save files 12-15 MBytes in size! Mostly because IDL is doing the right thing and saving the *connections* between objects as well as the objects themselves. It's just that that's not what I want!
Well, it's what I want. I just don't want it right now. :-)

I've thought about this a lot (and JD has written about his method for doing this several times in this newsgroup).

I think I actually have a method for doing this in my
Catalyst Library that would work with a LOT less complexity
than the iTools model. But, unfortunately, it would involve
some rather major modifications that I have not found time
to implement yet.

In the meantime, I think the alternatives are to invest a
couple of years learning iTools, or just muddle our way
through using what we already know. :-)

Cheers,

David

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