Subject: Re: Interactive Objects, Was: Simple GUI question
Posted by David Fanning on Wed, 23 Apr 2003 14:54:11 GMT
View Forum Message <> Reply to Message

Richard G. French (rfrench@wellesley.edu) writes:

> 3) A KEY POINT for me is that I want absolutely reproducible results without
> having to do any interaction. [...] In other words, I want to get rid of
> the 'I' in IDL as my final step, and be able to run a batch job that
> gives me the final result.

I would have to agree that there is little point
in writing widget programs if what you want to end
up with is a set-in-stone program with no interactivity.
Why waste your time adding something you don't want?

The whole point of widgets is interactively. I think a
case could be made that the whole point of objects is
interactivity with a memory. In other words, changes
you make to an object persist until you change it.
You can certainly save an object as it is, and call
it up years later (I presume from your described programming
style that you don't update your IDL version either, so I
am positive this will work) and have it do exactly the
same thing it just did. That's how persistent those suckers
are.)

(I took a natural history course with my youngest son
last summer and the first thing we learned about were these
strange little critters -- I can't recall their name now,
although I'm sure my son could -- that hang out in the
moss on desiccated boulders. They can stay dormant for
hundreds of years, but throw a little water on them and
they are up for a swim! You would like these little guys,
Dick. :-)

> Now, how about objects? I know that a lot of these are really cool! I like
> being able to display images in a resizable window and having the image
> follow along. I like being able to click on an axis of a plot and reformat
> the style. I like using objects that are self-contained and do a
> well-defined task. My problem, when thinking about actually writing one of
> these beasts, is that there seem to be too many choices!

Well, I can see how choices might be anathema to your
overall programming goals. :-)

I guess I would submit that you like interactivity, too.
You just prefer the slow method of interaction in which you

type commands over and over again at the IDL command line. (I presume you have no reservations about using the arrow keys.) It might well be that this method is overall much faster than writing a general purpose "fool-around-with-it-until-I-understand-the-damn-thing widget. But one advantage of objects over widgets, I think, is that it seems much easier to write that one new "fool around" method to explore a wild-hare idea.

> How many methods
> should I specify? What about classes?

Here is the real problem for people just starting out, I think. I've been extremely reluctant to write this object programming in IDL book I keep talking about because I didn't want real programmers to laugh at me. I was so dumb and excited about objects I just started using them! It wasn't until the past year or so that I began to understand the power and benefit of using them intelligently.

But Dave and I spend a couple of hundred dollars a month on transatlantic telephone calls arguing the merits of this design or that design. An image data object, to give you just one example, is particularly annoying. Should an image base class transparently handle 8-bit and 24-bit images? Or should these be two separate classes, primarily because they are handled differently in processing steps? We have been back and forth probably a thousand times on this one. It is probably one of the few times when we are *both* right!

We think one of the advantages of this Catalyst Library we are writing is that some of these choices have been made for you. In other words, we have done a lot of thinking about the design framework. We would be the first to tell you it is not perfect, but it does allow you to get on with it, without a lot of hemming and hawing about whether or not this is the *BEST* design. I think only lots and lots of experience can tell you that.

> As someone just beginning with
> objects, it is hard to get a good sense for how to break up a problem into
> manageable pieces, and to construct individual pieces that make sense.

I totally agree with this sentiment. My personal view (not shared, as far as I can tell, by the good folks at RSI) is that the only way you can get scientists migrating to objects is to give them objects that are similar to what they already know. Throwing them into objects by making them write object graphics programs is a fools errand, it seems to me. (But what do I know? I'm unemployed at the moment. :-)

> To follow David's analogy, I sometimes think that my widget programming
> style ends up being a design for an all-in-one blender, buzz saw, and CD-ROM
> burner, without doing any of them well. Object programming leaves me at a
> loss in terms of these fundamental design principles. The usual thing that I
> see is an example that says: "Start with an object that does one little
> thing, such as squaring a number or an array." But then I end up with about
> 100 lines of code which, in the end, just do what I can already do in one
> line of IDL - square a number or an array. What I need to see is an example
> of simple objects that can do something useful in IDL that I can't already
> do in a few lines of code. That might help me get over the hump so that I
> could join David's bandwagon.

I'm pretty sure writing objects does NOT save you time
up-front. Typically, a great deal of time and code goes
into producing the lowest-level objects. If time is saved,
it is on the back-end where applications can be built with
the lowest-level objects. If objects are built properly, you
find yourself spending less and less time with low-level
objects and more and more time with high-level objects.
It's then that the benefits kick in. I'm aware that by
this time it might be too late for someone who absolutely
has to have that NASA proposal done next week... :-)

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