Subject: Re: When should objects be used?
Posted by davidf on Thu, 24 Jun 1999 07:00:00 GMT
View Forum Message <> Reply to Message

Uh, well, gosh. If people are going to be serious about this...

I have two main criteria I use for creating an object:

1. I'm writing a compound widget. I write these as objects
   because they are so much easier to control. Getting the
   value of a compound widget means getting its object reference.
   Then I can do anything to the compound widget I like by calling
   a simple object method. When I remember I wanted to do something
   else to the compound widget, I just write a 3-4 line new method
   and it's done. No fooling around.

2. I want a "thing" to have some intelligence. For example, I
   want a plot that knows how to position itself in a window
   and what linestyle and symbol I want to use. I want a smart
   image that knows and can remember what kind of processing
   I have applied to it. I want a contour plot that knows how
   to position itself on a map projection or not and whether I
   want a colorbar with it or not. That kind of thing.

Most of the objects I write are of the second type. The
real advantage to me of these "smart" objects is that they
can be easily controlled by widget programs. For example,
the user can choose the plot background and foreground colors
from a palette of drawing colors, just by clicking on a
pull-down menu item. I don't have to hardcode a couple of
colors into my program and listen to how my user doesn't like
my aesthetic sensibilities. "Do it yourself", I say.

I do the same thing with smart images. Sure, I have an *idea*
how this image should be processed. But what if my user wants
to smooth the image *before* the edge enhancement step, when I
was certain they would do it *after*. What if they want to smooth
twice before this step? My program can accommodate any ol' thing
the user is stupid enough to want to try, even though I know
better. A smart image can be processed any way at all, and if
you don't like what you just did, you can even undo it, because
an UNDO method is part of what a smart image is.

The very first object I wrote was a smart window that was
simply told what kind of grid it should use to lay things
out (I.e., 2 column by 3 row). Given a "plot object" it
could lay those things out in its grid. The "plot" could be
any old thing at all. The window didn't know or care. This

made it possible for me to write Copy, Cut, and Delete methods for
things that were in the window. I had never thought of doing
anything like this before. Now the user could set up the
window the way *he* wanted it set up. Not the way I imagined
he wanted it set up. He could even drag something from this
position and put it over there.

I guess this is the thing about objects that has most amazed
me. Building objects tends to *generate* more ideas about what
to do with them than I ever got when I was writing programs
in the old linear way. All of a sudden whole new ways of
working and interacting with data is possible. The problem is
often not "what would I use an object for", but "how can I stop
having ideas so I can stop writing this damn program and get
some work done". They have been liberating for me, but then
I'm not the world's most innovative programmer, although
I've always been a pretty good mimic.

I like objects because they make me feel brighter than I
really am. (And tossing the jargon around really intimidates
people who ask you a question about IDL you don't know how
to answer. :-)

I know because about half the time when I read these object
posts in this newsgroup I think to myself, "Fanning, *you*
are writing a book about objects!? Who do you think you're
kidding!"

Cheers,

David

P.S. Let's just say objects are a whole lot cheaper than cocaine. ;-)

--
David Fanning, Ph.D.
Fanning Software Consulting
Phone: 970-221-0438 E-Mail: davidf@dfanning.com
Coyote's Guide to IDL Programming: http://www.dfanning.com/
Toll-Free IDL Book Orders: 1-888-461-0155