Subject: Re: Widget Objects
Posted by Mirko Vukovic on Mon, 28 Feb 2000 08:00:00 GMT
View Forum Message <> Reply to Message

In article
<hamillnumerics-2502002135370001@user-37kafqa.dialup.mindspring.com>,
  hamillnumerics@mindspring.com (Jim Hamill) wrote:
> An object-oriented approach to IDL widgets works nicely. Readers of
this
> newsgroup who'd like to see what I'm talking about are invited to
check
> out ...
>
>
 http://hamillnumerics.home.mindspring.com/hamillnumerics/idl /javaconcept
s/widgetobject/widgetobjects1.html
>
> Has someone already done this?
>
My recent thoughts on the subject stem from the fact that I really
dislike building widgets, and from a need to build a GUI for a rather
complex object.

Now, as luck would have it, in this complex object, I had numerous lists
of attributes, for lack of a better name (not properties). While
objects have properties that are basically IDL variables, the attributes
are higher level stuff, stuff with meaning: Flags, range specifiers,
color table indices, one-of type variables (var can be one of A,B,C
say), variable scans (lin, log, exp, power, name it). These attributes
are all objects.

One advantage of attributes, is that they can check their own validity,
they can have their own documentation! This was actually the initial
motivator: no need to write documentation, _include_ it in the code.
The code could then print out the keywords necessary to set the
attributes, the constraints on the attributes, the works.

The attributes can also build their own GUI's on the fly. There is only
one or two ways to build a GUI to set a clear a flag after all.
Building a GUI then invloves setting a base widget, going through a list
of attributes and building it up.

My thoughts have evolved somewhat. Now I think of a GUI as a UI to a
routine or an object. There should be very little logic in it. The
logic should all be in the IDL routine or object. A GUI is composed of
groups (which themselves can contain elements of this self-referential
sentece), or data fields (by that I mean higher level stuff, like
flags, ranges, etc), user action initiators (run routine X), and GUI

state controllers (like more or less detail) and placeholders (I'm not sure about them yet, but they would be for objects that can change. For example an IDL function can accept a scalar or a vector).

I've been influence to a large extent by the LaTeX typesetting system in this regard. The GUI layout would be specified logically, and would deal mainly with the look of the thing. The appearance of individual elements would be controlled by the object that corresponds to its GUI field. It seems to me that this eases changes to the layout considerably.

Each of these object display properties that can be changed on the fly. It would then become possible to dynamically change the look of a GUI.

One very important thing to me would be that GUI components would be all parts of a large inheritance tree. I should be able to change the background color, or the base font size of the base GUI, and the changes to affect all the children. It would seem to me that building GUI's across platforms would be much easier, since one would need to change the paramaters at only one place.

One could make a group self-standing on the fly, so it is in a widget by itself, but still part of the whole logical group, and the re-incorporate it again into the base widget.

Caveat? None of the GUI stuff has been done yet! Mostly thinking and pondering, writing specs, doodling on paper, but no hard coding. Something of this size, I would approach carefully, after a whole lot more of doodling on paper.

Un-answered questions? Well, I'm not sure about the purpose of event-bubbling up in the widget world (when event_fun returns an event). I have not run across an application that needed it. I think that in the scheme of things laid out above, such a feature would not be needed. Since GUI is just a front end, if a result of the user action processing (first event) needs further processing, the underlying programs should take care of that.

I have not really thought hard about graphics and cursor actions in this scheme. And there are probably a few other issues that I can't think of right now.

Mirko

Before you buy.