Subject: Re: widget layout
Posted by Ted Cary on Tue, 16 Jul 2002 17:41:45 GMT
View Forum Message <> Reply to Message

"David Fanning" <david@dfanning.com> wrote in message
news:MPG.179cf996c74af3a698992e@news.frii.com...
>
> This automatic resizing idea is interesting to me.
> I have a library of "widget objects", which wrap all
> the IDL widgets up into object wrappers. Then, rather
> then creating a widget hierarchy, you create an object
> hierarchy, based on IDL container objects.

> Putting the automatic resizing feature into one of these lower-level
> objects would be simple, I think.

Actually, I was thinking of implementing something like your object container widget hierarchy specifically for my resizer helper object, although it seemed like a lot of work for just this this routine. Typically, I had not realized the full potential of the method, although I now use object widgets frequently and even remember you hinting about your project before.

The way the resizer is supposed to work is by searching a widget hierarchy and finding all the resizable widgets in the TLB--by default, draw widgets. It resizes their windows around the other non-resizable realized widgets, whose dimensions remain fixed. The resizing depends upon the layout of the draw widgets in a base, which is why I needed to know this information. For example, if a TLB with three draw widgets in a column is resized, the default behavior is to distribute the change in base height evenly among the three draw windows. Sizing of TLBs with menubars is handled automatically, and there are options to fix individual draw widget window dimensions as well as to set maximum and minimum dimensions.

The problem with the routine is the "searching the widget hierarchy" part, since to be completely general it needs to find all resizable widgets and treat them differently according to the layouts of their bases. I used to manually call the object with the resizable IDs and layout info, but it's neater if it figures this information out for itself using just the TLB ID. Now I use WIDGET_INFO with /SIBLING and /CHILD to search through the hierarchy rooted at the TLB. The next step is to create some kind of object tree structure with "resizer" nodes corresponding to every resizable widget. A resize at the TLB root node should then work its up the tree to all the subnodes.

Not surprisingly this last part is not working well yet. Probably this would be a lot easier if instead of constructing an object tree hierarchy corresponding to an already-realized widget, the resizer could query a

widget hierarchy like yours, that was realized as an object tree in the first place. But aren't the widget hierarchies in IDL essentially stored internally as object trees anyway? If only there were just a WIDGET_WHERE function and a truly comprehensive WIDGET_GET_PROPERTY....

Anyway, what I can show you now is how the individual TLB resizer node works. If you already have the widget tree structure, maybe it's all you need, and probably you can improve upon it. In fact, the TLB node works fine for my typical application--some draw widgets in a TLB, resized around some labels and buttons, etc.... I'll remove the "print" and "help" statements and add some comments first, so give me a few days.

TC
