Subject: widgets and objects
Posted by mirko_vukovic on Thu, 13 Aug 1998 07:00:00 GMT
View Forum Message <> Reply to Message

this morning, while I was ... (well, it does not matter where I was)
where I was ... (hm, that does not matter either),

anyway, I was thinking widgets, and what is to me the biggest pain, i.e.
remembering information from one event to the other, the reason being the
addition of all the code to rememeber, and recall, and along the way to
introduce mistakes.

The currently recommended procedure is to use a pointer to a structure which
contains all the pertinent info. The "aesthetic" problem is that
you need to recall the pointer from some UVALUE, and then access the contents.
For example:

The event code either has a bunch of statements at the beginning saying
a=(*pInfo).a
b=(*pInfo).b
etc

or in the code, the statements contain

blah=func((*pInfo).Property1,(*pInfo).case2, ...)

and finally you have

(*pInfo).a=a
etc

at the end of the event where you are storing info for later use. And God
forbid if you are dealing with matrices. Then it is off course

*(pInfo).paImage

or even worse

(*(*pInfo).paData)[iRow,iCol]

This is my favorite, as I never fail to ... , well, it really does not matter
what I fail to. It is ugly.

In one sense, common blocks are nice, because, the variables are there
automatically, and the resulting code is much cleaner. (Mind you, I haven't
used a common block in widget programmin in a while).

Now, the combination of widgets and objects has been mentioned but not

described by several folks. Can one define a widget to be an object, and the events to be methods? Then self is naturally defined, and at least I do not have to worry about accessing it. The code does not get much cleaner, but at least one level of pointer access is eliminated. However, it is not clear to me that the event handlers can be methods. In that sense, one needs the event handler to call the method, yet another layer of complexity.

Ideally, when defining the widget, a special type of variable would be defined that exists in the widget space and in all the event handlers. It would be like a hidden common block, but would automatically exist in all the event handlers, and there would be a separate instance of these widget variables for separately created base widget (the biggest problem with common blocks).

Thus, a couple of hints to RSI. Allow the use of methods to be event handlers and/or allow the creation of these special widget common variables.

mirko

-----== Posted via Deja News, The Leader in Internet Discussion ==-----
http://www.dejanews.com/rg_mkgrp.xp Create Your Own Free Member Forum