Subject: Re: sharing information across widget hierarchies
Posted by Robbie on Tue, 10 Aug 2010 01:15:28 GMT
View Forum Message <> Reply to Message

I'd say that if your widget is much more complicated than a few text boxes then you should always use a pointer or object to store the program state. Using a pointer means that you can get your data into the place where you want it, without copying it. Whatever you might loose with poor syntax, you gain with code re-use.

Robbie

Subject: Re: sharing information across widget hierarchies
Posted by penteado on Tue, 10 Aug 2010 01:28:43 GMT
View Forum Message <> Reply to Message

On Aug 6, 11:23 pm, Chris <beaum...@ifa.hawaii.edu> wrote:
> Ok, I can get around this by attaching the _same_ event function to
> widgets 2 and 3. Now at least a single function gets called whenever
> an event happens; but there's no single widget in which to store state
> information. So then I have to store the state information as a
> _pointer_ in the uvalue to widgets 2 and 3. This means that I'm
> constantly dereferencing the state information pointer, which is much
> more awkward (syntactically) than working with a single, non-pointer
> variable (does anyone else hate how ugly x =
> (*(*state_ptr).big_array_ptr)[first_col, *] looks?)

With IDL 8, you can both have simpler code and only pass around a reference, using a hash instead of a structure. This removes two levels of indirection: the one used to pass a pointer instead of copying the structure, and the other to have dynamic fields in it. For instance, the fragment above could be just

(state_hash['big_array'])[first_col,*]