Subject: Re: A problem with UPP application scale . . .
Posted by jlfranks on Fri, 20 Feb 2009 16:25:59 GMT
View Forum Message <> Reply to Message

Well, I'm discussing the UI Class header file(s) that make up a UPP UI application.

Starting at the top of the hierarchy we have a header file that inherits from a layout file (.lay) of type TopWindow.

Note in the example below, the #includes for more UI class files for next level of hierarchy. This is the problem. Those files are structured in similar manner to top level except the inheritance is from their layout files and perhaps of type ParentCtrl.

Carry this down a four or five more levels until you have a populated UI Class hierarchy of 70 some odd classes. Any change anywhere in the layout files or the header files causes massive re-compilation. Now add multiple UI developers for a big project and progress becomes grindingly slow.

And, the clue that something is really not right is when the content of a label is changed at the bottom of the UI class hierarchy. This should not affect many files, but it does because the layout file changes. Or, added somemore callbacks to the implementation file and declare their signature in the corresponding UI Class header file. Same thing happens, i.e., compile the world.

There really is no problem for small projects. This is a problem related to scale.

The UI Classes exhibit the inheritance (is-a) relationship instead of containment (has-a). The latter has the opportunity for abstraction by using pointers to UI classes, forward references, and moving the #includes of other UI classes to the .cpp implementation file. This header file is less encumbered with dependencies to the rest of the UI structure.

With the current UPP implementation I cannot figure out how to break that dependency. The required macros are coupling other parts of the UI class hierarchy via the layout file directly into the header files. I've tried for about a day to come up with some scheme to remove this characteristic but without success.

Does this make the problem statement clearer?

Examples UI Class header file:

...guards...

```
#include <CtrlLib/CtrlLib.h>
using namespace Upp;

// The next includes is where trouble begins. It is
// needed because the layout file includes items to get
// to those next levels of the class hierarch (other screens).
//
#include <...more UI Class files for next level of hierarchy...>
#include <...more UI Class files for next level of hierarchy...>
#include <...more UI Class files for next level of hierarchy...>
#include <...more UI Class files for next level of hierarchy...>
#include <...more UI Class files for next level of hierarchy...>
#include <...more UI Class files for next level of hierarchy...>

#define LAYOUTFILE <here/MyUIClass.lay>
#include <CtrlCore/lay.h>

class MyUIClass : public WithMyUIClassLayout<TopWindow>
{
  public:
    typedef MyUIClass CLASSNAME;
    MyUIClass();

  // ... other stuff for this class ...
};
```