Subject: Re: "Visual Inheritance" possible?
Posted by mrjt on Fri, 08 Jun 2007 10:32:53 GMT
View Forum Message <> Reply to Message

In sense you mean it this it currently impossible as a layout class (one that inherits from WithxxxLayout) cannot inherit from another layout class without causing ambiguous function calls. To be honest, the way you suggest doesn't make sonse in some cases. For instance, if you move a control on the base layout what would happen to the sub-layouts?

However, I think the effect can be achieved quite well.
1) Copy/paste the controls from the base to the child, either in the Designer or in text mode. This gives a form of visual inheritace.
2) I assume you really want common control behaviour from the base class the be automatic on the sub-class windows? This can be done in one of two ways:
2a) Do not have the base class inherit from WithxxxLayout, but instead declare the controls explicitly as members of the class. You can then code the common bahaviour and use the base instead of TopWindow as the template parameter to WithxxxLayout when inheriting.
2b) Use a construct like this:

(op1/2 and result are all EditInt ctrls, calc andclose are buttons)

```
template <class P>
class AWindow : public P {
public:
 typedef AWindow CLASSNAME;

 void Calc()
 {
  if (!Accept()) return;
  result <<= ((int)~op1 + (int)~op2);
 }
protected:
 AWindow()
 {
  CtrlLayout(*this, "Base class");

  op1.NotNull() <<= 1;
  op2.NotNull() <<= 2;
 }
};

class BWindow : public AWindow< WithBWindowLayout<TopWindow> > {
public:
 typedef BWindow CLASSNAME;

 BWindow()
 {
  Title("Adder");
```

```
  calc <<= THISBACK(Calc);
  close <<= THISBACK(Close);
 }
};
```

This is neater in that you don't have to declare the controls explicitly, but does mean that you can't have multiple layers of inheritance. I've attached the package I tested the above code in.

Hope that helps,
James

File Attachments

1) LayoutTest.zip, downloaded 529 times