Subject: A question of C++
Posted by kbyte on Fri, 27 Mar 2009 09:06:24 GMT
View Forum Message <> Reply to Message

I have a SDI like application (developed using UPP) that has a menu and in it client are there is the main data where user makes their work. This application has lots of dialogs that are opened, using the main menu, in a modal fashion.

control (grids, labels, edits, etc) in all dialogs, including the client are of the main window.

Mirek had helped me in colorization (thank you again) of controls and the main Windows client are ready and sesnitive to colors stored in a binary file (windows background, label forecolor, edits textcolor, etc). Now I have to apply this to all modal dialogs too when they are invoked.

I know that C++ has means to make this without code copy-past, say, by deriving classes but has

All my modal dialogs classes are follow this pattern:

```
class CChangeUnitsDlg : public WithChangeUnitsDlgLayout<TopWindow>
{

}
```

Hence, all dialogs must store and be sensitive to the selected colors for background, labels and text on edit controls and on painting time the dialogs must paint using the selected colors.

Thanks very much

Alex

Subject: Re: A question of C++
Posted by kbyte on Fri, 27 Mar 2009 11:29:50 GMT
View Forum Message <> Reply to Message

I am trying this:

```
#include "ColoringSystem.h"
...

class CChangeUnitsDlg : public WithChangeUnitsDlgLayout<TopWindow> , public
CColoringSystem
{
```

}

and the CColoringSystem will store all needed colors and the paint of the CChangeUnitsDlg will paint every control 1 by one using the colors in the CColoringSystem class.

Alex

Subject: Re: A question of C++
Posted by mrjt on Fri, 27 Mar 2009 12:29:45 GMT
View Forum Message <> Reply to Message

kbyte wrote on Fri, 27 March 2009 11:29I am trying this:

#include "ColoringSystem.h"
...

class CChangeUnitsDlg : public WithChangeUnitsDlgLayout<TopWindow> , public CColoringSystem
{

}

and the CColoringSystem will store all needed colors and the paint of the CChangeUnitsDlg will paint every control 1 by one using the colors in the CColoringSystem class.

Alex

It doesn't work like that, Ctrls are respoionsible for drawing themselves.

You have several options here, but it depends on exactly what you are trying to do:
1- If you want to set colours globally (for all EditCtrls say) you can use EditField::StyleDefault().Write(). You can also do things like SColorFace_Write(Red()) to replace a particular colour.
2- If you need individual colouring for different ctrls of the same type (like username EditString is red, password is Blue) then you are going to have to do something more complicated. Unless you hard-code it this is going to be difficult to do, but possible.
One way would be to use your own version of the Layout macro that creates a string table from a .lay file and then use the table to read styling values from an Xml file. Or something.
3- Tell the user that they can set the colours themselves with ControlPanel->Display and that colours are deliberatley set to match the OS for a good reason

---

Subject: Re: A question of C++
Posted by kbyte on Fri, 27 Mar 2009 12:39:27 GMT
View Forum Message <> Reply to Message

Yes I want to change the colors for all controls of the same type.

I will try that. Thanks!

Alex

Subject: Re: A question of C++
Posted by mrjt on Fri, 27 Mar 2009 12:57:25 GMT
View Forum Message <> Reply to Message

In that case this thread might be useful:
Global style changes

Subject: Re: A question of C++
Posted by kbyte on Fri, 27 Mar 2009 15:55:52 GMT
View Forum Message <> Reply to Message

Very usefull that thread but no sample code and chameleon is complex to understand

Any simple code snipet for global edit or lable change?

Thank you

Alex

Subject: Re: A question of C++
Posted by kbyte on Fri, 27 Mar 2009 17:07:41 GMT
View Forum Message <> Reply to Message

I am trying this to changle edit boxes globally:

Exemple:

```
#include "DG.h"
DG::DG()
{
 CtrlLayout(*this, "Window title");
```

```
 EditField::Style edits =EditField::StyleDefault();

 edits.paper=Color(Color(255,0,0));
 edits.focus=Color(Color(0,0,255));
 edits.ink=Color(Color(0,255,0));

 EditField::SetStyle(edits);
}

GUI_APP_MAIN
{
 DG().Run();
}
```

But compiler says that i have to assign SetStyle to one object. So, I am confused. Can we set the global style or do I have to set control by control style?

Alex

Subject: Re: A question of C++
Posted by mrjt on Fri, 27 Mar 2009 17:39:46 GMT
View Forum Message <> Reply to Message

You can do both. To set it for a single ctrl you use SetStyle, which overrides StyleDefault() (which may be called something else on some ctrls).

To set it globally you change StyleDefault:

```
EditField::Style &s = EditField::StyleDefault().Write();
s.paper = Red;
s.focus = Green;
s.ink = Blue;
```

Incidentally, if you are going to set the style for a single ctrl you need to make sure that the Style instance you are using doesn't run out of scope (or gets deleted) before the Ctrl does. This is because the Ctrl stores a pointer to the Style internally (this is a pretty contrary to Upp style IMO. C'est la vie).
You can use a construct like this to make it work well:

```
void SetEditStyle(EditField &edit)
{
  static EditField::Style s = EditField::StyleDefault();
  s.paper = Red;
  s.focus = Green;
  s.ink = Blue;
  edit.SetStyle(s);
```

```
}
```

Subject: Re: A question of C++
Posted by kbyte on Sat, 28 Mar 2009 07:45:54 GMT
View Forum Message <> Reply to Message

Great!

Thank you very much

Alex