Hi,

I need to have several windows in an application, one for each part. Those windows are created on demand, several can be opened in the same time. When I need one, it must be either created, or focus must be given to it if existing. Let's say that it's somehow similar to MS Windows' MDI apps.

I tried this :

```
#ifndef _MultiWin_MultiWin_h
#define _MultiWin_MultiWin_h

#include <CtrlLib/CtrlLib.h>

using namespace Upp;


class Win1 : public TopWindow {
public:
 typedef Win1 CLASSNAME;
 Win1();
 ~Win1();
 void Close();
};

class Win2 : public TopWindow {
public:
 typedef Win2 CLASSNAME;
 Win2();
 ~Win2();
 void Close();
};

class MultiWin : public TopWindow {
protected:
 MenuBar menu;
public:
 typedef MultiWin CLASSNAME;
 MultiWin();
 void MainMenu(Bar& bar);
 void Window1();
 void Window2();
};
```

```
Win1 *win1;
Win2 *win2;

#endif

#include "MultiWin.h"

MultiWin::MultiWin()
{
 Title("Test multiple windows").MinimizeBox().Sizeable();
 AddFrame(menu);
 AddFrame(TopSeparatorFrame());
    menu.Set(THISBACK(MainMenu));
}

void MultiWin::MainMenu(Bar& bar) {
    bar.Add("Window 1", THISBACK(Window1));
    bar.Add("Window 2", THISBACK(Window2));
}

void MultiWin::Window1() {
 if (win1==NULL) {
 win1=new Win1();
 win1->OpenMain();
 }
 else {
 this->LostFocus();
 win1->SetFocus();
 }
}

void MultiWin::Window2() {
 if (win2==NULL) {
 win2=new Win2();
 win2->OpenMain();
 }
 else {
 this->LostFocus();
 win2->SetFocus();
 }
}

Win1::Win1() {
 Title("Window 1").MinimizeBox().Sizeable();
}

Win1::~Win1() {
 win1=NULL;
```

```
}

void Win1::Close() {
 delete this;
}

Win2::Win2() {
 Title("Window 2").MinimizeBox().Sizeable();
}

Win2::~Win2() {
 win2=NULL;
}

void Win2::Close() {
 delete this;
}

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

It works well, but when I try to re-open an existing window, it seems to get the focus but don't comes on the top of all others !

I tried a lot of things, but was unable find a solution... What can I do ? Is there a better way to do what I need to stay in "UPP spirit" ?

Thanks.

Subject: Re: Multiple windows and focus
Posted by mrjt on Wed, 04 Jul 2007 12:07:19 GMT
View Forum Message <> Reply to Message

```
 win->SetForeground();
 win->SetFocus();
```
Works fine for me. Also you don't need to call LostFocus, this will be done for you.

Now, some other suggestions:
'delete this' should never be used as it relies on the object being created by 'new'. Have the owner set the TopWindow::WhenClose callback to a function that deletes it instead.

You might want to use the Upp class Ptr<TopWindow> instead of the pointers. Then you don't need to worry about setting them back to NULL. Personally I'd keep them in an array of Ptrs.

Do you really need to create/delete the windows when they are opened/closed? Unless you will have many of them you could just have them as members and not have to worry about it.

Or you could google for 'Singleton class C++', which might do what you're looking for.

James

Subject: Re: Multiple windows and focus
Posted by jibe on Wed, 04 Jul 2007 14:10:19 GMT
View Forum Message <> Reply to Message

mrjt wrote on Wed, 04 July 2007 14:07 win->SetForeground();
  win->SetFocus();
Works fine for me. Also you don't need to call LostFocus, this will be done for you.
Yes, it's working fine, thanks I tried a lot of things, but I missed this method !

mrjt wrote on Wed, 04 July 2007 14:07You might want to use the Upp class Ptr<TopWindow> instead of the pointers. Then you don't need to worry about setting them back to NULL. Personally I'd keep them in an array of Ptrs.
I'm still new with UPP and don't know how to use this (even if I gess a little...). Is there a document explaining that, or I must study the sources ?

mrjt wrote on Wed, 04 July 2007 14:07Do you really need to create/delete the windows when they are opened/closed? Unless you will have many of them you could just have them as members and not have to worry about it.
Sometimes, I have few simple windows, so I could effectively create them all at the begining. But sometimes I need several complex windows, so I think that it's better to create/delete to save memory...

mrjt wrote on Wed, 04 July 2007 14:07Or you could google for 'Singleton class C++', which might do what you're looking for.
Yes, I could use a singleton. But is there a singleton base class in UPP ? For me, singleton is more especially interesting for a team of developpers, and in most cases I'm developping alone.

Anyway, if there is not yet, I think that a singleton base class would be interresting in UPP, even if it's not difficult to create one's own : no need to re-invent the wheel each time... Maybe to think for a next version

Thanks for your help and advices

Subject: Re: Multiple windows and focus
Posted by mrjt on Wed, 04 Jul 2007 14:45:30 GMT
View Forum Message <> Reply to Message

jibe wrote on Wed, 04 July 2007 15:10
Is there a document explaining that, or I must study the sources ?
In the manual. AFAIK they are basically just a container for pointers, and a bit simpler than the std:: variety.

Subject: Re: Multiple windows and focus
Posted by jibe on Wed, 04 Jul 2007 16:25:42 GMT
View Forum Message <> Reply to Message

Nice
Thanks.

Subject: Re: Multiple windows and focus
Posted by mirek on Thu, 05 Jul 2007 05:36:09 GMT
View Forum Message <> Reply to Message

mrjt wrote on Wed, 04 July 2007 08:07 win->SetForeground();
  win->SetFocus();
Works fine for me. Also you don't need to call LostFocus, this will be done for you.

Now, some other suggestions:
'delete this' should never be used as it relies on the object being created by 'new'. Have the owner set the TopWindow::WhenClose callback to a function that deletes it instead.

Actually, multiple peer windows scenario is one of exceptions to U++ common scope based resource management.

In this case is often the simplest thing to 'new' main windows and to use 'delete this' to close them. See UWord example.

Of course, it is also possible to deal with this using Array of windows or something like that, but we have found that in this particular situation, it is only more complex...