Subject: Window without title bar
Posted by DoggyDog on Tue, 12 Feb 2008 15:48:02 GMT
View Forum Message <> Reply to Message

Being a novice programmer I am stuck with a problem I cannot solve. I am trying to create an invisible (i.e doesn't appear in task bar) window without a title bar and board. I also want to be able to control transparency (alpha blending). The only code snippets I find on the web are done with MFC . I am not trying to create something complicated, just a rectangle with some text, pretty much like a very simple splash screen, although this is not my objective. Any ideas?

DoggyDog

Subject: Re: Window without title bar
Posted by DoggyDog on Wed, 13 Feb 2008 11:37:34 GMT
View Forum Message <> Reply to Message

OK, after some more research I concluded that I need to dig into the win32 API to get this done and then create a U++ package for it. It seems to be possible to do without MFC, since wxWidgets has something similar (wxSplashScreen), although it only works with bitmaps (I want it more flexible). Daunting task for someone that sucks at programming , but what can I do! I would be very happy if some ace programmers out there could point me in the right direction. Anything is welcome!

DoggyDog

Subject: Re: Window without title bar
Posted by cbpporter on Wed, 13 Feb 2008 11:54:36 GMT
View Forum Message <> Reply to Message

You don't really need Windows API or anything else which is not provided by U++ in a cross-platform way.

You just need to use PopUp() on the control you want to pop-up (and maybe a manual event loop for a splash screen "experience") and has parameter for drop-shadow and other useful stuff . This is only to point you out in the right direction, I never used them myself but I'm going to look into it later and give you more information.

In the meantime, you could check out the "ide" package from uppsrc. It creates a splash screen somewhere (try "idewin.cpp") and you should be able to track down how to implement it.

Subject: Re: Window without title bar
Posted by DoggyDog on Wed, 13 Feb 2008 12:02:27 GMT
View Forum Message <> Reply to Message

Yeah, you are right. After thinking about it and going through some code with comments, I believe the correct path is to dig into the dark chambers of U++. Makes sense. Thanks!

DoggyDog

Subject: Re: Window without title bar
Posted by DoggyDog on Wed, 13 Feb 2008 12:54:46 GMT
View Forum Message <> Reply to Message

Great, I found it and it seems to be very straight-forward. You probably saved me weeks of agony. Thanks again!!

I will make an example so that other novices can make use of my experience from this. I will post it here in a few days or so. I would like to make a U++ package, but I think it is over my head at this point.

U++ is the way to go!!!

DoggyDog

Subject: Re: Window without title bar
Posted by Werner on Wed, 13 Feb 2008 15:41:56 GMT
View Forum Message <> Reply to Message

Some time ago I wrote a UPT ("StandardApplication") which also creates a splash screen. You

Application). The UPT creates, among other files, 2 files named "...Splash.hpp" and "...Splash.cpp". Maybe a closer look at these 2 files might help you.

Furthermore you might want to have a look at the following code which deals with a popup window. But beware, it's very simple. Just a proof of concept for another U++ feature (Single).

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

class PopUpWin : public LineEdit
{
private:
 bool poppedUp_;

 void RightDown(Point p, dword keyFlags)
 {
  Close();
  poppedUp_ = false;
```

```
 }

public:
 PopUpWin() : poppedUp_(false) { }
 void SetPoppedUp(bool yesNo) { poppedUp_ = yesNo; }
 bool IsPoppedUp() const { return poppedUp_; }
};

class MainWin : public TopWindow
{
private:
 void LeftDown(Point p, dword keyFlags)
 {
  if (Single<PopUpWin>().IsPoppedUp())
   return;
  Single<PopUpWin>().SetPoppedUp(true);
  Single<PopUpWin>().LeftPos(300, 200).TopPos(300, 150);
  Single<PopUpWin>().SetColor(TextCtrl::PAPER_NORMAL,Blue);
  Single<PopUpWin>().SetColor(TextCtrl::INK_NORMAL, White);
  Single<PopUpWin>().PopUp
   (
    this, // Ctrl* owner  = NULL
    true, // bool  savebits = true
    true, // bool  activate = true
    true, // bool  dropshadow = false
    false // bool  topmost  = false
   );
 }
};

GUI_APP_MAIN
{
 MainWin mainWin;
 mainWin.SetRect(0, 0, 800, 600);
 mainWin.Run();
}
```
Werner

Subject: Re: Window without title bar
Posted by DoggyDog on Wed, 13 Feb 2008 16:16:21 GMT
View Forum Message <> Reply to Message

Werner,

thanks. Anything is welcome. I downloaded your UPT and will study the source.

Subject: Re: Window without title bar
Posted by Werner on Wed, 13 Feb 2008 20:06:04 GMT
View Forum Message <> Reply to Message

Meanwhile I finished my "single"-ruminations.

At the same time I improved my popup-window test application.

Perhaps the code is interesting to you because it deals with the possible GUI effects "slide" and "fade".

Here it is:

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

class PopUpWin : public LineEdit
{
private:
 bool poppedUp_;

 void RightDown(Point p, dword keyFlags)
 {
  Close();
  poppedUp_ = false;
 }

public:
 PopUpWin() : poppedUp_(false) { }
 void SetPoppedUp(bool yesNo) { poppedUp_ = yesNo; }
 bool IsPoppedUp() const { return poppedUp_; }
};

class MainWin : public TopWindow
{
private:
 typedef MainWin CLASSNAME;

 void NoEffect()
 {
  Single<PopUpWin>().SetPoppedUp(true);
  Single<PopUpWin>().SetRect(300, 300, 600, 400);
  Single<PopUpWin>().SetColor(TextCtrl::PAPER_NORMAL, White);
```

```
 Single<PopUpWin>().SetColor(TextCtrl::INK_NORMAL, Black);
 Single<PopUpWin>().PopUp
  (
   this,       // Ctrl* owner  = NULL
   true,       // bool  savebits = true
   true,       // bool  activate = true
   true,       // bool  dropshadow = false
   false       // bool  topmost  = false
  );
}

void SlideEffect()
{
 Single<PopUpWin>().SetPoppedUp(true);
 Single<PopUpWin>().SetRect(300, 300, 6, 4);
 Single<PopUpWin>().SetColor(TextCtrl::PAPER_NORMAL, White);
 Single<PopUpWin>().SetColor(TextCtrl::INK_NORMAL, Black);
 Single<PopUpWin>().PopUp
  (
   this,       // Ctrl* owner  = NULL
   true,       // bool  savebits = true
   true,       // bool  activate = true
   true,       // bool  dropshadow = false
   false       // bool  topmost  = false
  );
 Ctrl::ProcessEvents();
 Animate(Single<PopUpWin>(), RectC(300, 300, 600, 400), GUIEFFECT_SLIDE);
}

void FadeEffect()
{
 Single<PopUpWin>().SetPoppedUp(true);
 Single<PopUpWin>().SetRect(300, 300, 600, 400);
 Single<PopUpWin>().SetColor(TextCtrl::PAPER_NORMAL, White);
 Single<PopUpWin>().SetColor(TextCtrl::INK_NORMAL, Black);
 Single<PopUpWin>().PopUp
  (
   this,       // Ctrl* owner  = NULL
   true,       // bool  savebits = true
   true,       // bool  activate = true
   true,       // bool  dropshadow = false
   false       // bool  topmost  = false
  );
 Ctrl::ProcessEvents();
 Animate(Single<PopUpWin>(), RectC(300, 300, 600, 400), GUIEFFECT_FADE);
}

void local_menu(Bar& bar)
```

```
 {
  MenuBar local_menu;

  bar.Add("no effect", THISBACK(NoEffect));
  bar.Add("slide effect", THISBACK(SlideEffect));
  bar.Add("fade effect", THISBACK(FadeEffect));
  local_menu.Execute();
 }

 void LeftDown(Point p, dword keyFlags)
 {
  if (Single<PopUpWin>().IsPoppedUp())
   return;
  MenuBar::Execute(THISBACK(local_menu));
 }
};

GUI_APP_MAIN
{
 MainWin mainWin;
 mainWin.SetRect(0, 0, 800, 600);
 mainWin.Run();
}
```

Werner

Subject: Re: Window without title bar
Posted by DoggyDog on Thu, 14 Feb 2008 12:15:57 GMT
View Forum Message <> Reply to Message

Aaaahhh, this is good stuff. I built it and it is great! The only problem I have is Single<>. Could you give me a short explanation?

DoggyDog

Subject: Re: Window without title bar
Posted by Werner on Thu, 14 Feb 2008 12:38:53 GMT
View Forum Message <> Reply to Message

DoggyDog wrote on Thu, 14 February 2008 13:15... Single<>. Could you give me a short explanation?

"Single" is an Ultimate++ way to do jobs which are normally done by use of the singleton pattern. If you don't know what that means, just search the internet for "singleton". A good starting point to

read might be http://en.wikipedia.org/wiki/Singleton_pattern.

Werner

Subject: Re: Window without title bar
Posted by mr_ped on Thu, 14 Feb 2008 15:15:10 GMT
View Forum Message <> Reply to Message

Basically it's useless if you have other means of creating single local instance of certain class. Singleton's are replacement for global variables in pure-object languages where globals are "impossible", I'm not sure why singleton's concept got as far as into C++, where globals are possible.
Not that I suggest to use global variables, avoid them as much as possible, but if you need a global, I don't think there's big reason to mask them as singleton.

Anyway, I can't recall *ever* to use one in C++.

Subject: Re: Window without title bar
Posted by mrjt on Thu, 14 Feb 2008 15:18:16 GMT
View Forum Message <> Reply to Message

Isn't the difference that Singletons/Single<>s only get memory allocated on heap when they are first used?

Not that this is ever really useful

Subject: Re: Window without title bar
Posted by cbpporter on Thu, 14 Feb 2008 15:53:48 GMT
View Forum Message <> Reply to Message

Anyway, using singleton pattern here is really overkill. Just create and instance variable in MainWin called for example popUp, with type PopUpWin and replace all Single<PopUpWin>() with that given name.

For example, Single<PopUpWin>().SetPoppedUp(true);becomespopUp.SetPoppedUp(true);

Subject: Re: Window without title bar
Posted by mr_ped on Thu, 14 Feb 2008 17:07:59 GMT
View Forum Message <> Reply to Message

mrjt wrote on Thu, 14 February 2008 16:18Isn't the difference that Singletons/Single<>s only get

memory allocated on heap when they are first used?

Not that this is ever really useful

That's completely true.
And I'm still missing any real world scenario where you need global variable allocated as late as possible. I mean, I can't even imagine one, and I really tried for couple of minutes.
Even if such need would arise, you can still have global pointer to some memory huge object which is instantiated later into the program (at the moment when you find it appropriate).

I mean, the singletons are nice in the way you don't need to think what part of program you are in, and if that global variable has been already initialized. While some people may find this property as an advantage, I think it's actually major drawback instead. They sort of hide the structure of code and data-flow by making the initialization and usage to look the same.
It's better to understand when/how/why are executed parts of your application and when components need to be initialized, than fixing lack of understanding by using fool-proof classes.

I mean the fool-proof part of code is not a bad thing, but it should save you in case of your mistake (and detect+report that in ideal world so it gets noticed), not deliberately using it all the time and working without clue why you are doing that piece of code there but it "oh, just works anyway so who cares".

Subject: Re: Window without title bar
Posted by Werner on Thu, 14 Feb 2008 17:54:29 GMT
View Forum Message <> Reply to Message

cbpporter wrote on Thu, 14 February 2008 16:53Anyway, using singleton pattern here is really overkill. Just create and instance variable in MainWin called for example popUp, with type PopUpWin and replace all Single<PopUpWin>() with that given name.

For example, Single<PopUpWin>().SetPoppedUp(true);becomespopUp.SetPoppedUp(true);
In this case you're absolutely right. May I nevertheless point out a few issues?

1.
As Ultimate++ itself uses Single a lot (174 occurrences in uppsrc), I was doing some tests. Coincidentally I used PopUp for this when the popup window issue emerged. At that time I published my code without thinking much about it - it just worked.

2.
Ultimate++'s Single is definitely not a singleton. I just mentioned it because the reason to create these "singles" is the same. And indeed I find the theoretical foundations of the singleton pattern quite interesting.

3.
I do not share your opinion about singletons vs global variables etc.. As soon as you are contributing to large projects with many developers where you don't have complete control over all the code, you will try hard to avoid globals at any cost and do your best to guarantee that one and

only one object exists, namely a singleton, if that need arises.

Werner

Subject: Re: Window without title bar
Posted by Werner on Thu, 14 Feb 2008 21:06:31 GMT
View Forum Message <> Reply to Message

Ok. Just to reconcile those singleton-haters I have created a tiny demo, featuring multiple main windows and multiple subwindows (at most 1 subwindow in 1 main window lest Windows (!) crashes) with slide effect and fade effect - it goes without saying: WITHOUT using "Single", to say nothing of "singleton".

```
#include <CtrlLib/CtrlLib.h>

using namespace Upp;

class PopUpWin : public LineEdit
{
private:
 bool poppedUp_;

 void RightDown(Point p, dword keyFlags)
 {
  Close();
  poppedUp_ = false;
 }

public:
 PopUpWin() : poppedUp_(false) { }
 void SetPoppedUp(bool yesNo) { poppedUp_ = yesNo; }
 bool IsPoppedUp() const { return poppedUp_; }
};

class MainWin : public TopWindow
{
private:
 typedef MainWin CLASSNAME;

 static int mainWinCount;

 PopUpWin popUp;

 virtual void Paint(Draw& w)
 {
```

```
 w.DrawRect(GetSize(), WhiteGray);
 w.DrawText(0, 0, "Click left for context menu");
}

virtual void Close()
{
 delete this;          // window is on the heap
}

void AddMainWin()
{
 (new MainWin)->OpenMain();
}

void NoEffect()
{
 if (popUp.IsPoppedUp())
  return;
 popUp.SetPoppedUp(true);
 Rect mainRect = GetScreenRect();
 popUp.SetRect(mainRect.left + 20, mainRect.top + 20, 300, 225);
 popUp.SetColor(TextCtrl::PAPER_NORMAL, LtBlue);
 popUp.SetColor(TextCtrl::INK_NORMAL, Yellow);
 popUp.Set(String("no effect\nclick right to close"));
 popUp.PopUp
  (
   this,        // Ctrl* owner  = NULL
   true,        // bool  savebits = true
   true,        // bool  activate = true
   true,        // bool  dropshadow = false
   false        // bool  topmost  = false
  );
}

void SlideEffect()
{
 if (popUp.IsPoppedUp())
  return;
 popUp.SetPoppedUp(true);
 Rect mainRect = GetScreenRect();
 popUp.SetRect(mainRect.left + 20, mainRect.top + 20, 4, 3);
 popUp.SetColor(TextCtrl::PAPER_NORMAL, LtRed);
 popUp.SetColor(TextCtrl::INK_NORMAL, White);
 popUp.Set(String("slide effect\nclick right to close"));
 popUp.PopUp
  (
   this,        // Ctrl* owner  = NULL
   true,        // bool  savebits = true
```

```
    true,        // bool  activate = true
    true,        // bool  dropshadow = false
    false        // bool  topmost  = false
   );
  Ctrl::ProcessEvents();
  Animate(popUp, RectC(mainRect.left + 20, mainRect.top + 20, 300, 225), GUIEFFECT_SLIDE);
 }

 void FadeEffect()
 {
  if (popUp.IsPoppedUp())
   return;
  popUp.SetPoppedUp(true);
  Rect mainRect = GetScreenRect();
  popUp.SetRect(mainRect.left + 20, mainRect.top + 20, 300, 225);
  popUp.SetColor(TextCtrl::PAPER_NORMAL, LtGreen);
  popUp.SetColor(TextCtrl::INK_NORMAL, Blue);
  popUp.Set(String("fade effect\nclick right to close"));
  popUp.PopUp
   (
    this,        // Ctrl* owner  = NULL
    true,        // bool  savebits = true
    true,        // bool  activate = true
    true,        // bool  dropshadow = false
    false        // bool  topmost  = false
   );
  Ctrl::ProcessEvents();
  Animate(popUp, RectC(mainRect.left + 20, mainRect.top + 20, 300, 225), GUIEFFECT_FADE);
 }

 void local_menu(Bar& bar)
 {
  MenuBar local_menu;

  bar.Add("new main window", THISBACK(AddMainWin));
  bar.Add("subwindow with no effect", THISBACK(NoEffect));
  bar.Add("subwindow with slide effect", THISBACK(SlideEffect));
  bar.Add("subwindow with fade effect", THISBACK(FadeEffect));
  local_menu.Execute();
 }

 void LeftDown(Point p, dword keyFlags)
 {
  MenuBar::Execute(THISBACK(local_menu));
 }

public:
 MainWin()
```

```
 {
  ++mainWinCount;
  String title("Multi-windowed App with Subwindow: Window # ");
  Title(title + AsString(mainWinCount));
  SetRect(40 * mainWinCount, 40 * mainWinCount, 400, 300);
  NoCenter();
 }

 ~MainWin()
 {
  --mainWinCount;
 }
};

int MainWin::mainWinCount = 0;

GUI_APP_MAIN
{
 (new MainWin)->OpenMain();        // anytime deleteable
 Ctrl::EventLoop();
}
```

Enjoy!

Werner
