
Subject: UI window threading problem

Posted by [crydev](#) on Sun, 28 Apr 2013 20:28:39 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello,

I ran into a problem that I initially fixed, but I'm not convinced that there isn't a better way of solving it. My problem is as following.

I have a main window that is capable of opening another window that runs non-modal above the main window, meaning the main window is not clickable nor focusable while the opened window is still opened. The non-modal window enables the user to select parameters for a memory scan, and by clicking OK the memory scan has to be started. The problem I ran into in the first place was that I had to be able to make the non-modal window access a function in the main window. That isn't too hard to fix, but the next problem is, that the non-modal window keeps on top, non-responsive, until the memory scan that is started is finished. My fix to that problem was creating a new thread that started the memory scan, in order to keep any possible piece of UI responsive. Downside: I had many exceptions that were very hard to find, due to the threading between the UI parts.

Now that I all fixed it, I wonder: isn't there a better solution to achieve the non-modal window to be closed, without the need of a thread?

Thanks in advance!

crydev

Subject: Re: UI window threading problem

Posted by [Sender Ghost](#) on Mon, 29 Apr 2013 00:03:27 GMT

[View Forum Message](#) <> [Reply to Message](#)

Hello, Crydev.

crydev wrote on Sun, 28 April 2013 22:28: Now that I all fixed it, I wonder: isn't there a better solution to achieve the non-modal window to be closed, without the need of a thread?

The simple answer is using of PostCallback before closing non-modal window:

```
void DoingScan()
{
    // Doing some scanning
    Sleep(1000);
}

void Window::OnOK()
{
    PostCallback(callback(DoingScan));
    Close();
}
```

But after the reconstruction of what you did, I came to the following solution:
Toggle Spoiler

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

using namespace Upp;

class Window : public TopWindow {
public:
    typedef Window CLASSNAME;
    Window();
    // Ctrls
    EditString text;
    Button btnOK, btnCancel;
    // Events
    void OnOK();
};

Window::Window()
{
    Title("Window");
    Sizeable().Zoomable();
    const Size sz(320, 240);
    SetRect(sz); SetMinSize(sz);

    text.SetData("Some text");

    btnOK.SetLabel(t_("OK")) <=> THISBACK(OnOK);
    btnCancel.SetLabel(t_("Cancel"));
    Rejector(btnCancel, IDCANCEL);

    Add(text.HSizePosZ(4, 4).TopPosZ(4, 20));
    Add(btnOK.LeftPosZ(4, 50).TopPosZ(28, 20));
    Add(btnCancel.LeftPosZ(58, 50).TopPosZ(28, 20));
}

void Window::OnOK()
{
    AcceptBreak(10);
}

class App : public TopWindow {
private:
    bool doing;
    Thread work;
public:
    typedef App CLASSNAME;
```

```

App();
~App();
// CtrlIs
Window window;
Button btnOpen;
// Events
void OnOpen();
void OnScan();
};

App::App() : doing(false)
{
    Title("Application");
    Sizeable().Zoomable();
    const Size sz(640, 480);
    SetRect(sz); SetMinSize(sz);

    btnOpen.SetLabel(t_("Open")).Tip(t_("Open another window")) <=<= THISBACK(OnOpen);

    Add(btnOpen.LeftPosZ(4, 50).TopPosZ(4, 20));
}

App::~~App()
{
    work.ShutdownThreads();
    work.Wait();
}

void App::OnOpen()
{
    if (window.Execute() != 10)
        return;

    work.Run(THISBACK(OnScan));
}

void App::OnScan()
{
    if (!doing) doing = true;
    else return;
    // Getting some data
    String text = window.text;
    // Doing some scanning
    Sleep(1000);

    if (work.IsShutdownThreads())
        return;
}

```

```
GuiLock __;  
PromptOK(Format("The scan is complete with words:&%s", DeQtf(text)));  
doing = false;  
}  
  
GUI_APP_MAIN  
{  
    Ctrl::GlobalBackPaint();  
  
    App app;  
    app.Run();  
}
```

Where you have access to the contents of the non-modal window after closing it, while run the scanning process on the right exit code.

Subject: Re: UI window threading problem
Posted by [crydev](#) on Mon, 29 Apr 2013 08:53:44 GMT
[View Forum Message](#) <> [Reply to Message](#)

Thanks a lot Sender Ghost! I used the solution you gave and it works very well. It is even less error prone now. Although I still had to start the scan on another thread to keep the UI responsive, but that thread could be started from the main window, which is perfect.

Subject: Re: UI window threading problem
Posted by [Sender Ghost](#) on Mon, 29 Apr 2013 10:52:03 GMT
[View Forum Message](#) <> [Reply to Message](#)

Actually, there are more solutions, like using ProcessEvents between iterations of your algorithm(s) or even using "Alternative Multithreading" by Pavel Ostapenko.
