Subject: multithread access to the same control
Posted by hojtsy on Tue, 31 Jan 2006 07:42:25 GMT
View Forum Message <> Reply to Message

I am writing a software which downloads and parses 4 XML files in paralel from different HTTP servers. When each thread finishes the parsing it should update a field in an ArrayCtrl. I don't find any docs about multithreading in u++. Does the HttpClient support paralel downloads of diffetent files in different threads? Is it OK to access the ArrayCtrl from multiple threads? Do I need mutex for it? Is the multithreading support available under Unix?

Subject: Re: multithread access to the same control
Posted by mirek on Tue, 31 Jan 2006 14:35:52 GMT
View Forum Message <> Reply to Message

Well, multithreading model is quite simple:

GUI must runs in single thread and is not serialized.

What IS serialized (locked) is "timer" queue (" because it is in fact used for many other things as well). Means you can safely post callbacks to GUI thread.

Also, Core types ARE serialized.

So the right thing to do is to run HttpClient in different threads (I hope Tom will protest here if there is any problem with it, but I do not thing so) and update your ArrayCtrl using PostCallback.

Multithreading support is available in Unix (but is tested even less than the one on Win32... in fact I doubt that any U++ app ever run on real multiprocessor machine, and until then, some problems might be left unrevealed).

Mirek

Subject: Re: multithread access to the same control
Posted by hojtsy on Tue, 31 Jan 2006 23:11:04 GMT
View Forum Message <> Reply to Message

I have a question about the Thread object.
Suppose that every time a button is pressed, I would like to start a new worker thread which terminates after a processing task is finished by it.

```
void onButton()
{
  Thread t;
  t.Run( THISBACK(ProcessingFunction) );
}
```

Will the destructor of the Thread object terminate the thread? If yes, then I need dynamic allocation of Thread, but I won't know when to release that memory.

Subject: Re: multithread access to the same control
Posted by mirek on Tue, 31 Jan 2006 23:18:48 GMT
View Forum Message <> Reply to Message

No, thread must terminate itself. Terminating via Thread variable is not possible.

Subject: Re: multithread access to the same control
Posted by hojtsy on Wed, 01 Feb 2006 21:59:40 GMT
View Forum Message <> Reply to Message

I created a small example multithreaded application. You can add numbers to a list, triggering a new thread which counts the number of divisors, and updates the table when it is finished. Even though a slow implementation was choosen you still need to use large numbers to be able see the multithreading in action. The example uses Thread and PostCallback. As no example is present for these features, maybe my application, or a modified version could be added to the examples or reference directory.

File Attachments
1) Divisors.zip, downloaded 1664 times

Subject: Re: multithread access to the same control
Posted by mirek on Thu, 02 Feb 2006 22:31:12 GMT
View Forum Message <> Reply to Message

A very good idea.

I have spent all the evening working on it

There were two problems with original example:

* I think that it is quite error-prone to run several threads in single instance. Chances that you make unserialized access to some member variable are high

* However, the more serious problem is with application exit. When you press Close button, chances are that thread will be running and will invoke dangling callback.

I have tried to fix both and to improve the example (using uint64 etc...) - it is now in "reference".

Mirek

File Attachments
1) GuiMT.zip, downloaded 1798 times