Subject: Dealing with background tasks elegantly in a userinterface
Posted by Alboni on Fri, 04 Mar 2022 15:35:09 GMT
View Forum Message <> Reply to Message

Hello,

I wonder if there is an elegant way to execute tasks in the background. There are often MySql queries or http or rpc calls or other tasks that can potentially take more time than one can spend in a callback if you want a responsive userinterface.
How does one go about that in u++?

Some points:
Executing a query in the background and get a notification to the ui when it is done or failed and how much more time in between. Tasks over 5 seconds should be cancellable.
Do I have to open a MySqlSession for every query that is executed in the background?? Is this an "expensive" or slow thing to do or doesn't it matter.
Can I copy SqlSessions and use the copy in another thread concurrently with the original? DO I have to open it with credentials every time?
Cross thread communication with queues or pipes?
Using callbacks from threads?

About serializing tasks from other threads, is there an example for that?
I don't want to reinvent the wheel and get unreliable code. Done that too many times already.
I want the quick, easy and robust way. I'm missing the howto here, the overview. A reference is not enough.

(Yes, I've found Upp::Thread)

Subject: Re: Dealing with background tasks elegantly in a userinterface
Posted by zsolt on Sun, 06 Mar 2022 00:41:45 GMT
View Forum Message <> Reply to Message

I use Thread class in my older code with Mutex protected containers as job queue.
Currently I prefer CoWork.
To notify GUI thread, I use PostCallback. It is a very easy way.
GUI is thread safe now, but I haven't tested GUI that way yet.
For HTTP, SMTP I created simple curl wrapper classes with progress support, but now you can use HttpRequest's WhenDo to notify user with a progress bar.
SQL: I use separate, per thread sessions, opened by the thread.
Serialization: Mutex and Mutex::Lock utility class are very useful.

Subject: Re: Dealing with background tasks elegantly in a userinterface

Posted by Oblivion on Sun, 06 Mar 2022 08:36:16 GMT
View Forum Message <> Reply to Message

Hello Alboni

Quote:
I wonder if there is an elegant way to execute tasks in the background. There are often MySql queries or http or rpc calls or other tasks that can potentially take more time than one can spend in a callback if you want a responsive userinterface.
How does one go about that in u++?

Try AsyncWork. It is simple, allows you to cancel tasks, and allows you to elegantly return values. (Based on CoWork). https://www.ultimatepp.org/src$Core$AsyncWork_en-us.html

Tutorial: https://www.ultimatepp.org/srcdoc$Core$Tutorial$en-us.html#S ection_7_5

Best regards,
Oblivion

Subject: Re: Dealing with background tasks elegantly in a userinterface
Posted by zsolt on Sun, 06 Mar 2022 17:11:54 GMT
View Forum Message <> Reply to Message

After seeing the tutorial, this AsyncWork seems to be a very useful thing for me.
I haven't tried it yet, but almost the same as async/await in Javascript.
Thanks!

Subject: Re: Dealing with background tasks elegantly in a userinterface
Posted by mirek on Fri, 18 Mar 2022 17:42:51 GMT
View Forum Message <> Reply to Message

zsolt wrote on Sun, 06 March 2022 01:41
To notify GUI thread, I use PostCallback. It is a very easy way.

You might enjoy checking out Ctrl::Call. Combined with lambda that is even easier.

Current rules for parallel GUI programming:

- only the main thread can manipulate windows (open, close, move) or run event loop (this is actually a limitation forced by win32 architecture)
- exception to this rule is Prompt* function which open window but can be called from non-main threads
- except this, non-main threads can call widget methods as long as they use GuiLock

Mirek