Subject: Re: Problem breaking loop (with close button) in main thread
Posted by Oblivion on Thu, 20 Oct 2022 18:10:59 GMT
View Forum Message <> Reply to Message

Hi Lance,

Quote:
probably should not modify GUI from within other than the GUI thread.

You are allowed to modify GUI from within other threads. For this purpose, you need to use

a) EnterGuiMutex() & LeaveGuiMutex();
b) GuiLock, which is basicly the above functions wrapped in a class for convenience:

Rudimentary example:

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

using namespace Upp;

#define LAYOUTFILE <DeadMutex/DeadMutex.lay>
#include <CtrlCore/lay.h>

static std::atomic<bool> fin;

class DeadMutex : public WithDeadMutexLayout<TopWindow> {

 typedef DeadMutex CLASSNAME;

public:
 DeadMutex()
 {
  CtrlLayout(*this, "Window title");
  start.WhenPush = THISFN(Start);
  stop.WhenPush = [this]{ fin = true; };
  WhenClose = [this]{ ShutdownThreads(); Break(); };
 }

 void Start(){
  Thread().Run([=] {
   {
    GuiLock __; // EnterGuiMutex()
    log.Append("Badguy instance running & owned the gui mutex...\n");
    // LeaveGuiMutex();
   }
   while(!fin && !IsShutdownThreads() ){
```

```
    Sleep(50);
   }
   GuiLock __;
   log.Append("Stop requested, Badguy instance exiting...\n");
   fin = false;
  });
 }
};

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

There is also a GuiUnlock class which basically reverses the order of LeaveGuiMutex and EnterGuiMutex, so you can force the main thread to temporarily unlock its GUI lock.

What you are not allowed to do is, creating windows and prompts within other threads.

Best regards,
Oblivion
