Subject: how to use timer id? and how to kill a timer
Posted by bonami on Tue, 20 Jul 2010 08:03:20 GMT
View Forum Message <> Reply to Message

first question

in Ctrl::SetTimeCallback() there is,
  ASSERT(id >= 0 && (size_t)id < (int)sizeof(Ctrl));

in doc, it says,
  id. Should be in range 0..80. U++ defines compile-time protocol for distributing these ids. If Ctrl wants to use non-zero time callback id, it should define it using
 enum { TIMEID_PERIODIC = baseclass::TIMEID_COUNT,
           TIMEID_COUNT };

i'm confused by the doc. what's the enum for? my TopWindow::TIMEID_COUNT is 2. that means i can only use id 1 & 2?

second question

how can i add procedure for creating a timer?
i have a TopWindow and it create a Thread. the thread controls when to create and delete a timer. what should i do in TopWindow? after i Run() it, there is no place to insert code to wait for the Thread's timer creation request. is there a way to send a message to TopWindow?

Thank you.

Subject: Re: how to use timer id?
Posted by mrjt on Wed, 21 Jul 2010 08:46:45 GMT
View Forum Message <> Reply to Message

First question:
The enum is to ensure that you don't use ids that are already used by your base classes. For instance, if TopWindow had:
enum { TIMEID_TIMER1 = Ctrl::TIMEID_COUNT,
TIMEID_TIMER2,
TIMEID_TIMER3,
TIMEID_COUNT };
and your class
enum { TIMEID_PERIODIC_TIMER = TopWindow::TIMEID_COUNT,
TIMEID_COUNT };

Then TIMEID_PERIODIC_TIMER would be 3, and TopWindow would use ids 0,1,2. In practice I've never found a situation where this is a problem but it's good practice and not much work.

Second question:
I'm not 100% sure I understand you, but in this situation I would just pass a Callback the the

Thread as a parameter. The Thread can then execute the Callback when it's ready to create the timer.

Subject: Re: how to use timer id?
Posted by bonami on Wed, 21 Jul 2010 09:28:11 GMT
View Forum Message <> Reply to Message

I think you understood me. Since Thread is not in TopWindow procedure, I cannot create a timer in it and have to let TopWindow do it.
However I do not understand you. Can you please post sample code on how to write this callback? I'm never familiar with this callback functionality.

Subject: Re: how to use timer id?
Posted by mrjt on Wed, 21 Jul 2010 10:54:40 GMT
View Forum Message <> Reply to Message

This code opens a window and uses a Thread to wait 5 seconds before starting window timer counting up:

```
class MyWindow : public TopWindow
{
 typedef MyWindow CLASSNAME;
 int count;
public:
 MyWindow() : count(0)     {}
 virtual void Paint(Draw& w)   { TopWindow::Paint(w); w.DrawText(4, 4, AsString(count)); }
 void Count()       { ++count; Refresh(); }
 void CreateCounter(int interval)  { SetTimeCallback(interval, THISBACK(Count)); }
};

void MyThread(Callback1<int> TimerCB)
{
 for (int i = 0; i < 50; ++i) {
  if (Thread::IsShutdownThreads())
   return;
  Sleep(100);
 }
 TimerCB(-1000);
}

GUI_APP_MAIN
{
 MyWindow wnd;
 Thread thrd;

 wnd.SetRect(RectC(0, 0, 200, 200));
```

```
 wnd.CenterScreen();

 thrd.Run(callback1(MyThread, callback(&wnd, &MyWindow::CreateCounter)));
 wnd.Run();

 Thread::ShutdownThreads();
}
```
You'll need to give more information about what you're trying to achive if you need anything more complex than this.

Subject: Re: how to use timer id?
Posted by bonami on Thu, 22 Jul 2010 02:09:14 GMT
View Forum Message <> Reply to Message

my problem is not how to envoke the TopWindow's functions in the Thread, but how to separate these threads. I traced your code. TimerCB(-1000) is not thread safe, it calls CreateCounter() synchronously. Since the timer is executed in TopWindow thread, this is not what I want. In fact, timer creation is not a problem. Killing it is.
Here's a demo of my code,

```
class e : public TopWindow {
  class a {
    Thread t;
    void f(void);
  };
  void cb(void);
  a aa;
};
void e::a::f(void)
{
  Sleep(1000);
  SetTimeCallback(1000, cb, 10);
  Sleep(1000);
  KillTimeCallback(10);
}
void e::cb(void)
{
}
GUI_APP_MAIN
{
  e ee;
  ee.aa.t.Run(callback(&ee.aa, &e::a::f));
  ee.Run();
}
```

This code cannot build. it just shows my intention. in the Thread function f, I create a timer expiring after 1 second. Then it sleeps 1 second and kills the timer. The problem is, I do not know

whether the timer is already envoked when I kill it. That is why I want to put all the timer operations in TopWindow thread and I think that is what should be done. So I need to send a message to TopWindow and let it create or kill the timer, from an MS view.
Plus, why did you test Thread::IsShutdownThreads()?

Subject: Re: how to use timer id?
Posted by mrjt on Thu, 22 Jul 2010 08:58:14 GMT
View Forum Message <> Reply to Message

I just cannot understand what it is you are trying to do

You are correct that my previous example wasn't threadsafe. Here is a threadsafe version. As far as I can tell ALL timers will executed in the GUI thread.

```
#include "CtrlLib/CtrlLib.h"

using namespace Upp;

class MyWindow : public TopWindow
{
 typedef MyWindow CLASSNAME;
 int count;
public:
 MyWindow() : count(0)     {}
 virtual void Paint(Draw& w)   { TopWindow::Paint(w); w.DrawText(4, 4, AsString(count)); }
 void Count()       { ASSERT(Thread::IsMain()); ++count; Refresh(); }
};

void MyThread(Callback cb)
{
 TimeCallback timer;
 for (int i = 0; i < 50; ++i) {
  if (Thread::IsShutdownThreads())
   return;
  Sleep(100);
 }
 timer.Set(-1000, cb);
 for (int i = 0; i < 50; ++i) {
  if (Thread::IsShutdownThreads())
   return;
  Sleep(100);
 }
 timer.Kill();
}

GUI_APP_MAIN
{
 MyWindow wnd;
```

```
 Thread thrd;

 wnd.SetRect(RectC(0, 0, 200, 200));
 wnd.CenterScreen();

 thrd.Run(callback1(MyThread, callback(&wnd, &MyWindow::Count)));
 wnd.Run();

 Thread::ShutdownThreads();
}
```

Quote:The problem is, I do not know whether the timer is already envoked when I kill it
Of course. You can be sure that the timer has been created, but not whether it has been executed. If you need guaranteed excution than you should call the function directly (using GuiLock if it needs GUI access)

Quote:Plus, why did you test Thread::IsShutdownThreads()?
Because I call Thread::ShutDownThreads before closing. This ensures that all threads are finished, otherwise I get heap leaks from dangling threads. I check IsShutdownThreads so that I can terminate the thread prematurely, if you take them out you'd have to wait for the thread to finished naturally.

Subject: Re: how to use timer id?
Posted by bonami on Thu, 22 Jul 2010 09:32:40 GMT
View Forum Message <> Reply to Message

I need the timer creation and killing to be executed in main thread.

When I kill the timer, there are two situations.
One, the callback has already been executing. Then I cannot tell in the callback whether it should execute. This is why I need the killing operation to be executed in main thread.
Two, the timer has not timed out yet. This way, the callback will never be executed. This is all right.

My real implementation is an auto processor.
A timer can be asked for. Before it ends, network packets may ends it. Thread is blocked by a semaphore. Network packets minus this semaphore. Timer ending minus it, too. If network packets received, the timer needs to be killed. And later packets needs to be neglected. After a while, this all may run again. So the timer callback cannot be executed again after I called timer killing. Otherwise, it will mess up the next round.

Subject: Re: how to use timer id?
Posted by mrjt on Thu, 22 Jul 2010 10:10:05 GMT
View Forum Message <> Reply to Message

You are never going to be able to guarantee that the timer won't execute between receiving the last packet and destroying the timer.

Even if you execute it in the main thread you have to await on the GUI Mutex and the timer could execute while you're doing that. You need to modify your design so that this is not a problem.

Do that and the example above should work fine. You don't need the timer creation and destruction to be in the main thread (and they are mutex locked and threadsafe).

Subject: Re: how to use timer id?
Posted by bonami on Fri, 23 Jul 2010 02:28:24 GMT
View Forum Message <> Reply to Message

i add delays into your code and the timer killing is asynchronous with timer callback execution. I regard this as a flaw. I can kill it, but I cannot know whether it is executed. I have to choose other implementations.

```
class MyWindow : public TopWindow
{
 typedef MyWindow CLASSNAME;
public:
 void Count()
 {
 //here 2nd.
 Sleep(10000);
 //here 5th.
 }
};
void MyThread(Callback cb)
{
 TimeCallback timer;
 timer.Set(1000, cb);
 //here 1st.
 Sleep(5000);
 //here 3rd.
 timer.Kill();
 //here 4th.
}
...
```

Subject: Re: how to use timer id?
Posted by mrjt on Fri, 23 Jul 2010 09:32:41 GMT
View Forum Message <> Reply to Message

Of course it's asyncronous, the timer's running in a different thread

There simply isn't anyway to guarantee that the timer will not execute between you receiving the last packet and killing the timer. It's not a flaw, it's just the nature of MT.

Subject: Re: how to use timer id?
Posted by bonami on Mon, 26 Jul 2010 02:16:11 GMT
View Forum Message <> Reply to Message

IC. thank you anyway.
