Subject: Re: Basic MT queries
Posted by dolik.rce on Tue, 21 Sep 2010 11:36:57 GMT
View Forum Message <> Reply to Message

Hi Javier
281264 wrote on Tue, 21 September 2010 12:261.- Atomic and associated functions, are they similar to INTERLOCKED?
If I'm not mistaken, Atomic*() functions use platform specific routines for atomic access to variables. INTERLOCKED macro gives you similar effect on entire block. The main difference is that INTERLOCK uses mutex to achieve this, so it is much slower.

281264 wrote on Tue, 21 September 2010 12:262.- ShutDownThreads function: what is it for? How a thread can be paused and resumed?
Calling Thread::ShutDownThreads() sets an internal flag that can be checked using Thread::IsShutDownThreads(). If I remember correctly, ShutDownThreads waits till all the threads end. To stop/resume thread you should use a flag for which you check in the thread. It is a good idea to to mark it volatile, e.g. "volatile bool running;".

281264 wrote on Tue, 21 September 2010 12:263.- Apparently there are numerous global functions in U++ not documented. One of them is GuiLock__? What is this for?.
GuiLock is one of the ways how to safely update GUI from threads. BTW: There is actually a space, the "__" is used just as a variable name The idea behind GUILock is that as long as the object __ (or gl) exists, the main thread is blocked from changing GUI, so it prevents dead-locks. The lock is actually acquired in constructor and released in destructor, so it can be easily limited using brackets.

Honza

PS: Looking at you sources from first post, I noticed that you use the thread1 variable to call the static functions (which can be called without an object). Actually the whole code in this case could be done without using the variable at all. To start thread in such case you can use Thread::Start(callback). The only downside is that you can't Wait() for such thread, but in many cases that is not necessary.