Subject: Re: WhenAction() -> ProcessEvents() -> WhenAction() -> hang, crash :-)
Posted by xrysf03 on Sun, 08 Dec 2019 19:57:29 GMT
View Forum Message <> Reply to Message

Thanks for resonse Koldo, always polite and always swift :)

Hmm... I can see that U++ has its own class Mutex and ConditionVariable. And they appear to be less convoluted (objectified) than the C++11 std::mutex and std::condition_variable (geez... if I didn't know the bare C libpthread version, I would probably just shake my head in disbelief).

My favourite and perhaps "naive" mechanism for passing lumps of something to do, between a producer and a consumer, is using a queue, protected by a mutex+condvar. What is the most appropriate container in U++ to implement a FIFO queue? The online help mentions Vector, Array, BiVector, BiArray... and in this case, I don't need to index the "collection" on anything, all I need is FIFO operation. So I don't need a Set, let alone a Map.
Hmm. Or perhaps I'll just preallocate the buffers and use a simple Semaphore...

Actually none of this is probably relevant to the GUI. I'm just thinking forward :) I'll probably use one background thread to do the tuning and grabbing, and another background thread to do the crunching and ScatterCtrl updates. And I'll write the updates straight into the buffer that's been preallocated and passed to the ScatterCtrl. So all I need to ask the GUI foreground thread to do, is use Ctrl::Call() or PostCallback() to do a ScatterCtrl::Refresh() on my spectrogram object... any examples of this would be welcome. Or generally how to tell the GUI foreground thread to redraw a ScatterCtrl that has had its data buffer changed by a background thread...

Ahh, apologies, I'll have to check the reference/GuiMT . There's a nice example of PostCallback(). Ouch... anonymous functions in C++...