Subject: Re: WaitForMultipleObjects() analog?
Posted by Mindtraveller on Wed, 28 May 2008 20:19:43 GMT
View Forum Message <> Reply to Message

hojtsy wrote on Wed, 28 May 2008 12:16Mindtraveller,

What are you trying to achieve with multiple Semaphores? I would expect that an alternative implementation could use Monitors. These are fancy mutexes which also have Wait and Pulse methods.
I am working on a U++ example code which uses Monitors to implement a Producer-Consumer queue.

See the concept explained in http://www.albahari.com/threading/part4.html

Hojtsy
I just didn`t fully understand about Monitors usage as we don`t have ones in U++ for now.
Then about waiting a number of Semaphores. Let`s imagine the real world application. Your program must heavily work with a number of devices attached to serial port (or ethernet or something else). Of course it is solved by adding a class which creates new thread and have cycle which is sleeping most of time. But there is a number of awake signal types. And these signals could be activated from within: this thread, OS and main thread. These signals are:

application close event. sleeping i/o thread should do finalization and close.
OS i/o event. sleeping thread should handle the fact of newly sent or received data.
queue event. sleeping thread just got a new task to do. this task was added by another thread.
timer event (timeout). happens when system i/o functions didn`t respond in time or final value of some deferred OS call didn`t arive in time.

These events are dramatically different from each other and when my thread is awakened I should know the type of this event to handle it. So I use one synchronization object (let`s say Semaphore for simplicity) for each event type. Doing this with one Semaphore would be a nightmare, wouldn`t it?