

Re: WaitForSingleObject() will not deadlock

Re: WaitForSingleObject() will not deadlock

Source: <http://www.tech-archive.net/Archive/VC/microsoft.public.vc.mfc/2007-07/msg00235.html>

- *From:* Joseph M. Newcomer <newcomer@xxxxxxxxxxxxx>
 - *Date:* Mon, 02 Jul 2007 16:09:21 -0400
-

No, recursive mutex WILL solve the problem, as should be obvious from the nature of circular list detection. If you hit a node which has the visited-bit set, then you have done a circular walk. In the case of mutexes, if you try to lock the object to inspect the bit, and you already own the mutex, then you are able to acquire the lock (recursively), and you can legitimately inspect the visited bit. If it is set, you release the mutex and you now know you have a circular structure. If you are a thread other than the thread that owns the object, you will block, and therefore when the lock is finally released, and the waiting thread acquires it, the visited bit will NOT be set because the thread could not have previously walked into the node.

This is a rather elementary data structure algorithm, and it seems obvious that it cannot work without a recursively-acquirable lock of some sort.

joe

On Mon, 02 Jul 2007 11:57:44 -0700, Frank Cusack <fcusack@xxxxxxxxxxxxx> wrote:

Neither does a recursive mutex since a thread other than the one that acquired the first lock cannot acquire a lock either.

-frank

On Mon, 02 Jul 2007 10:47:10 -0400 Joseph M. Newcomer <newcomer@xxxxxxxxxxxxx> wrote:

No, trylock and its equivalents only say that the object is locked. It does not guarantee that you can lock an object and examine it, so it won't do the job of walking a circular structure when there are potentially concurrent walkers.

joe

On Sun, 01 Jul 2007 23:47:50 -0700, Frank Cusack <fcusack@xxxxxxxxxxxxx> wrote:

On Mon, 02 Jul 2007 01:37:50 -0400 Joseph M. Newcomer <newcomer@xxxxxxxxxxxxx> wrote:

I was not aware that anyone would design a non-recursive mutex,

Re: WaitForSingleObject() will not deadlock

since this has been an integral part of lock design for at least 20 years, if not longer.

I haven't known our good friend the mutex for so long, but I would hazard a guess that posix mutexes have ALWAYS had recursion as an option. Because posix has a non-recursive mutex (by default) doesn't mean it hasn't also always had a recursive mutex.

I was using recursive-acquisition locks in the late 1980s and they were a well-known technology at the time I was using them. A non-recursive mutex is not usable for doing things like traversing circular lists trying to detect circularity, for example.

Sure it is, use trylock() instead of lock().

-frank

Joseph M. Newcomer [MVP]
email: newcomer@xxxxxxxxxxxxx
Web: <http://www.flounder.com>
MVP Tips: http://www.flounder.com/mvp_tips.htm

Joseph M. Newcomer [MVP]
email: newcomer@xxxxxxxxxxxxx
Web: <http://www.flounder.com>
MVP Tips: http://www.flounder.com/mvp_tips.htm