What if I need to wait for more than MAXIMUM_WAIT_OBJECTS threads?

devblogs.microsoft.com/oldnewthing/20240823-00

August 23, 2024

Raymond Chen

A customer had a test that created a lot of threads, and they wanted their test to wait for all of the threads to exit before proceeding to the next step. However, the number of threads exceeded the maximum number of handles, more than `MAXIMUM_WAIT_OBJECTS` of them, so what is the best way to wait for all of them if they can't do a single `WaitForMultipleObjects`?

The customer noted that the documentation had a few suggestions. One is to divide the objects into groups of size at most `MAXIMUM_WAIT_OBJECTS` and for each group, create a thread to call `WaitForMultipleObjects`. Another suggestion is to call `RegisterWaitForSingleObject` on each handle.

The customer thought these approaches were unnecessarily complicated. What about just dividing the objects into groups of size at most `MAXIMUM_WAIT_OBJECTS` and just going into a loop calling `WaitForMultipleObjects` on each group? "Is there some subtlety that we're missing?"

Process handles and thread handles have the property that waits on them are idempotent. Waiting on a process or thread handle waits for the process or thread to exit, but it has no effect on the process or thread itself. This is different from some other types of handles: Waiting on semaphores, mutexes, and auto-reset events has side effects: Consuming a semaphore token, taking ownership of the mutex, and resetting an auto-reset event.

Furthermore, process and thread handles also have the property that once they become signaled, they never become unsignaled. This means that once you have successfully waited on them to become signaled, you don’t have to worry about the possibility that in the future, they might not be signaled any more.

Therefore, if you are waiting for a group of process and thread handles all to be signaled, you have the liberty to wait for them in any order and not require the special behavior of `WaitForMultipleObjects` where it doesn’t create any wait side-effects until all the objects become signaled simultaneously.

So yes, you can wait for them in blocks of `MAXIMUM_WAIT_OBJECTS`. But really, even that is too much work. You can just wait for them one at a time.

```
for (auto&& handle : m_threadHandles)
{
    REQUIRE(WaitForSingleObject(handle, INFINITE)
            == WAIT_OBJECT_0);
}
```