WaitForInputIdle should really be called WaitForProcessStartupComplete

devblogs.microsoft.com/oldnewthing/20100325-00

March 25, 2010

Raymond Chen

The `WaitForInputIdle` function waits for a process to finish its initialization, which is determined when it reaches a state where it is just sitting around waiting for messages. The documentation for `WaitForInputIdle` doesn't even get around to the initialization part until the **Remarks** section. If all you read is the one-sentence summary, *Waits until the specified process is waiting for user input with no input pending, or until the time-out interval has elapsed*, it would not be unreasonable for you to conclude that a process goes into and out of the *input idle* state each time it processes a message. But no, it's a one-time transition. If you call `WaitForInputIdle` on a process which had previously gone input idle, but is now busy and not processing pending input messages, the function will still return immediately, because `WaitForInputIdle` only checks whether the process has gone input idle *at all* and not whether it is input idle *right now*. As the **Remarks** section notes, the purpose of the `WaitForInputIdle` function is for a process to determine whether another process (which is recently launched) has reached a state where it is okay to send that process messages. This is important to know when the form of communication between two processes is a message-based mechanism, and the two processes otherwise have no real way of knowing what the other is doing. (If the two processes had been written by the same author, then you could come up with some more expressive interface for the two to communicate through, one which avoids the need for one process to guess when the other one is ready.) The specific scenario that `WaitForInputIdle` was created to address is DDE. Back in the old 16-bit days, you didn't need a `WaitForInputIdle` function, because scheduling was co-operative. You know that the other process was sitting idle, because if it were busy, your code wouldn't be running in the first place. It's like waiting for the talking stick to be handed to you so that you can ask the question, "Are you ready to give up the talking stick?" The `WaitForInputIdle` function assisted in the porting of these 16-bit applications by allowing a process to wait and simulate the "Wait for the other person to stop talking" operation which had previously been implicit in a co-operative system. What would it mean for `WaitForInputIdle` to wait on a program that has already completed its initialization, when the program has multiple threads? Suppose one thread is sitting around waiting for messages, but another is busy and still has unprocessed input messages. Would a call to this `WaitForInputIdleAgain` function wait, or should it return immediately?

According to the description, it would return immediately, because there is a thread in the process which is "waiting for user input with no input pending." So even if `WaitForInputIdle` worked like this imaginary `WaitForInputIdleAgain` function, it still wouldn't help you, because it wouldn't actually wait in cases where you probably wanted it to.

Actually, the above analysis applies to `WaitForInputIdle` as well; we'll pick up this discussion next time.

Raymond Chen

Follow