Why does my single-threaded program have multiple threads?

devblogs.microsoft.com/oldnewthing/20191115-00

November 15, 2019

Raymond Chen

You've written a simple single-threaded program, but when you look in Task Manager, it says that the program has two or even more threads. What's going on?

Even though your program doesn't create any threads, a library used by your program might create threads, and the system itself might create threads.

For example, if you call the `SHFileOperation` function to copy some files, the shell may create additional threads to assist with the file copy operation. For example, the progress UI could be shown on the UI thread, with a separate thread used to perform the disk access.

Even after the multithreaded operation is complete, you may see threads lingering in the process because the multithreaded operation may have used the thread pool. Every process has a default thread pool which is created upon demand, and is destroyed at process termination.

If you are a console application, then the system creates an additional thread in your process in order to handle and deliver console control notifications.

In more recent versions of Windows 10 (I forget exactly when it started), the loader takes advantage of the thread pool to speed up loading DLLs into memory. This means that in practice, by the time the first line of code in your application starts to execute, the process default thread pool has already been created in order to load the DLLs your application uses.

Raymond Chen

Follow