Why does my thread handle suddenly go bad? All I did was wait on it!

devblogs.microsoft.com/oldnewthing/20170929-00

September 29, 2017

Raymond Chen

A customer reported that they had a very strange bug, where waiting on a thread handle causes it to become invalid. Here's a code fragment:

```
DWORD waitResult = WaitForSingleObject(hThread, INFINITE);
assert(waitResult == WAIT_OBJECT_0); // assertion passes

DoSomeCleanup();

CloseHandle(hThread);
```

That final call to `CloseHandle` results in a `STATUS_ INVALID_ HANDLE` exception when run in the debugger. How did the handle become invalid? We sucessfully waited on it just a few lines earlier.

There wasn't enough information to go on, so we had to make some guesses. Perhaps `hThread` was already closed, and it got recycled to refer to some unrelated kernel object, and it's that unrelated object that you're waiting on when you call `WaitForSingleObject`. And then when you do some cleanup, that causes the unrelated handle to be closed, which means that the numeric value of `hThread` now refers to an invalid handle.

The customer did some investigation and discovered that they were obtaining the thread handle from the `_beginthread` function. The handle returned by the `_beginthread` function is explicitly documented as being closed by the `_endthread` function.

> `_endthread` automatically closes the thread handle, whereas `_endthreadex` does not. Therefore, when you use `_beginthread` and `_endthread`, do not explicitly close the thread handle by calling the Win32 `CloseHandle` API. This behavior differs from the Win32 `ExitThread` API.

Basically, the deal is that the `_beginthread` function returns a handle to the created thread, but does not give you ownership of the handle. Ownership of that handle remains with the thread itself, and the thread automatically closes the handle when it exits. (Not mentioned in the MSDN documentation for `_beginthread` is that the runtime automatically calls `_endthread` if the thread function returns normally. This detail is mentioned in the documentation for `_endthread`, which is probably a better place for it anyway.)

Basically, the handle returned by the `_beginthread` function is useless. You don't know how long it's valid, and it might even be invalid by the time you even receive it!

Switching to `_endthreadex` fixed the problem.

Raymond Chen

Follow