If I call GetExitCodeThread for a thread that I know for sure has exited, why does it still say STILL_ACTIVE?

devblogs.microsoft.com/oldnewthing/20180302-00

March 2, 2018

Raymond Chen

A customer reported that when they took the thread handle returned by the `_beginthread` function and passed it to `GetExitCodeThread`, the function reported that the thread was `STILL_ACTIVE`, even though the thread is known to have already exited.

The customer found this text in the documentation for `_beginthread` and wants more information about the case where the function can return an invalid handle if the thread exits quickly.

> If the thread that's generated by **_beginthread** exits quickly, the handle that's returned to the caller of **_beginthread** might be invalid or point to another thread.

The point is that the `_beginthread` function returns a handle to a thread, but that handle is valid only as long as the thread is running. Once the thread exits, the handle spontaneously becomes invalid, at which point it becomes eligible for recycling.

The upshot of this is that the handle returned by `_beginthread` is useless, because it can go invalid for reasons outside your control.

The source code for the C runtime library is included with Visual Studio. You can read the source code for the `_beginthread` function if you want to understand what's going on.

The customer asked whether this behavior of `_beginthread` would explain the problem they are seeing.

Yes, it is exactly the problem you are seeing. Calling `GetExitCodeThread` with the handle returned by `_beginthread` is useless.

- If the thread is running, then you will get `STILL_ACTIVE`.
- If the thread is not running, then you are using an invalid parameter and you will get garbage. The value `STILL_ACTIVE` is one possible manifestation of garbage.

In general, the return value from `_beginthread` is not useful and should not be used for anything other than determining whether the thread was started successfully.

We recommend that if the customer wants to use the thread handle, they should switch to `_beginthreadex` and remember to close the handle when they are done.

The customer explained that their application was originally developed by another company. They had considered switching from `_beginthread` to `_beginthreadex`, but didn't want to do so unless absolutely necessary, because they would have to justify the time and money required to fix the problem to their management.

Yes, switching from `_beginthread` to `_beginthreadex` will fix the problem. As noted, the handle returned by `_beginthread` is useless, and any code that tries to do anything beyond test it for success is like to run into problems exactly like the one you describe.

We learned about some people who want documentation that a bad idea is a bad idea. But that documentation is already there in MSDN. They already have the paperwork they need. I couldn't quite figure out what the customer was looking for. Did they just want a personalized version of the documentation customized just for them?

If that's what you need, you can copy/paste the following paragraph.

> To whom it may concern,
>
> It is my recommendation that the return value from the `_beginthread` function should be used only to determine whether the function succeeded. In particular, the handle returned by the function is of indeterminate lifetime and cannot be reliably used.
>
> Sincerely,
>
> Raymond Chen

Raymond Chen

Follow