How does PostQuitMessage know which thread to post the quit message to?

devblogs.microsoft.com/oldnewthing/20090112-00

January 12, 2009

Raymond Chen

Commenter bav016 asks how functions like `PostQuitMessage` and `SetTimer(NULL)` know which thread the messages should go to. Unlike some functions such as `InvalidateRect` which have a window handle parameter that lets you say which window you want to operate on, `PostQuitMessage` and `SetTimer(NULL)` don't say which thread the `WM_QUIT` or `WM_TIMER` message should go to. How do they decide?

The messages go to the current thread; that is, they are delivered to the thread that called the function in the first place.

There are many functions which operate on an implicit message queue, and those cases, they operate on the message queue associated with the thread making the call. If you call `GetKeyState` you retrieve the calling thread's keyboard state. If you call `GetMessage` you retrieve messages from the calling thread's message queue. If you call `InSendMessage`, you are told about the calling thread's message processing state. If you call `GetQueueStatus` you retrieve information about the calling thread's msesage queue. You get the idea.

If you want these functions to operate on a thread different from the one that is executing, you'll have to ask that thread to make the call for you.

Raymond Chen

Follow