Why do messages posted by PostThreadMessage disappear?

devblogs.microsoft.com/oldnewthing/20090930-00

September 30, 2009

Raymond Chen

The only thread message you can meaningfully post to a thread displaying UI is `WM_NULL`, and even then, it's only because you want to wake up the message loop for some reason. A common problem I see is people who use `PostThreadMessage` to talk to a thread that is displaying UI and then wonder why the message never arrives. Oh, the message arrived all right. It arrived and then was thrown away. This is actually a repeat of an earlier entry with the title *Thread messages are eaten by modal loops*, but I'm repeating it with a better subject line to help search engines. But since I'm here, I may as well augment the existing article. Obvious places where you have modal loops on a UI thread are functions that are explicitly modal like `DialogBox` or `MessageBox` or `TrackPopupMenuEx(TPM_RETURNCMD)` or `DoDragDrop`. But there are less obvious modal loops, like the modal loop that runs when you click on the caption bar and hold the button or the modal loop that runs when COM is waiting for a cross-thread call to complete. And since you don't control those modal loops, when they call `DispatchMessage`, your thread message will simply be thrown away.

If you need to communicate reliably with a thread that also displays UI, then create a hidden window and send or post messages to that window.

Raymond Chen

Follow