Once you go input-idle, your application is deemed ready to receive DDE messages

devblogs.microsoft.com/oldnewthing/20140620-00

June 20, 2014

Raymond Chen

Feel free to stop using DDE. There was one customer who confessed that they were still using DDE, and they asked for help debugging a DDE problem. They found that sometimes, when their application was launched for DDE, it never received the `WM_DDE_INITIATE` message. Instead, the `ShellExecute` function returned `ERROR_DDE_FAIL`. If launched from Explorer, the error message shown to the user was "There was a problem sending the command to the program." It took a long time to figure out what was going on, and there were a number of dead ends, but I'll cut to the chase: The problem was that one of the features they added to their program included code that ran during process startup, and that code pumped messages as part of its initialization. Message pumping was not expected there, which is why it took so long to isolate. The `WaitForInputIdle` function was created for DDE. It's how a DDE client determines that the DDE server is ready to accept commands. And as soon as any thread in your process goes input-idle, the entire process is declared to be input-idle, and you end up becoming eligible to receive DDE messages, *even if you're not really ready for them*. In the case of this program, the accidental message pump caused the application to be considered ready to accept DDE commands even though the main DDE server hadn't gotten off the ground yet. The initiation message went to the splash screen, and the splash screen said, "Why are you bothering me with stupid DDE messages? I'm just a splash screen!"

TL;DR: If your application includes a DDE server, make sure not to pump messages until your DDE server is ready to receive commands.

Raymond Chen

Follow