Multithreaded UI code may be just as hard as multithreaded non-UI code, but the consequences are different

devblogs.microsoft.com/oldnewthing/20110516-00

May 16, 2011

Raymond Chen

Commenter Tim Smith claims that the problems with multithreaded UI code are not significantly more than plain multithreaded code. While that may be true on a theoretical level, the situations are quite different in practice. Regardless of whether your multithreaded code does UI or not, you have to deal with race conditions, synchronization, cache coherency, priority inversion, all that mulitthreaded stuff. The difference is that multithreaded problems with non-UI code are often rare, relying on race conditions and other timing issues. As a result, you can often get away with a multithreaded bug, because it may shows up in practice only rarely, if ever. (On the other hand, when it does show up, it's often impossible to diagnose.) If you mess up multithreaded UI code, the most common effect is a hang. The nice thing about this is that it's easier to diagnose because everything has stopped and you can try to figure out who is waiting for what. On the other hand, the problems also occur with much more frequency.

So it's true that the problems are the same, but the way they manifest themselves are very different.

Raymond Chen

Follow