Why does the MoveWindow function let you suppress repainting?

devblogs.microsoft.com/oldnewthing/20090316-00

March 16, 2009

Raymond Chen

Commenter Phil Quirk asks via the suggestion box why the `MoveWindow` function lets you suppress repainting. “Shouldn’t the OS be able to figure out if the window needs to be repainted?”

Indeed the window manager does do a very nice job of figuring it out if you pass `bRepaint = TRUE`, which is the expected value of the parameter. But if you think you’re smarter than the window manager, then you can pass `bRepaint = FALSE` and tell the window manager, “Even though you think the window needs to be repainted, don’t repaint it. Trust me on this.”

Why would you try to outwit the window manager? Maybe you have special knowledge about how your application behaves. For example, you might exploit special properties about the source and destination coordinates and decide that certain portions of the window should not be redrawn but rather should be shared between the old and new locations, sort of like the advanced tricks you can play with the `WM_NCCALCSIZE` message. Or you might know that your program is going to invalidate the entire client rectangle soon anyway, so a repaint immediately after the move would just be a waste of time. The `bRepaint` parameter provides an escape hatch, a throwback to the days when the window manager let you do strange things because you might be doing them as part of a clever thing.

Mind you, these are pretty advanced concerns and most of the time you would be best off just passing `bRepaint = TRUE` and letting the window manager do its job of deciding what needs to be repainted.

Raymond Chen

Follow