# Paint messages will come in as fast as you let them

devblogs.microsoft.com/oldnewthing/20111219-00

December 19, 2011

Raymond Chen

There is a class of messages which are generated on demand rather than explicitly posted into a message queue. If you call `GetMessage` or `PeekMessage` and the queue is empty, then the window manager will look to see if one of these generated-on-demand messages is due, messages like `WM_TIMER`, `WM_MOUSEMOVE`, and `WM_PAINT`. Neil wonders, "In that program that called `InvalidateRect` 100,000 times, how many paint messages were generated?" The Zen answer to this question is "Yes." A more practical answer is "As many as you can get." When somebody calls `InvalidateRect`, the window manager adds the specified rectangle to the window's invalid region (or invalidates the entire client area if no rectangle is provided) and sets a flag that says "Yo, there's painting to be done!" (It's not actually a flag, but you can think of it that way.) When a message retrieval function finds that there are no incoming sent messages to be dispatched nor any applicable messages in the queue to be retrieved, it looks at these extra flags to see if it should generate a message on the fly. If the "Yo, there's painting to be done!" flag is set on a window that the thread is responsible for, a `WM_PAINT` message is generated for that window. (Similarly, a `WM_TIMER` is generated if a timer has elapsed, and a `WM_MOUSEMOVE` is generated if the mouse has moved since the last time this thread retrieved a mouse message.) Therefore, the number of `WM_PAINT` messages by 100,000 invalidations is not deterministic, but it'll be at least one and may be as high as 100,000. It's basically just a race between the invalidation thread and the paint thread.

| | |
|---|---|
| `InvalidateRect` | |
| `InvalidateRect` | |
| | `GetMessage` (retrieves `WM_PAINT`) |
| | `WM_PAINT` dispatched |
| | `GetMessage` (waits for a message) |
| `InvalidateRect` | |
| | `GetMessage` (returns with `WM_PAINT`) |

| | |
|---|---|
| `InvalidateRect` | |
| | `WM_PAINT` dispatched |
| `InvalidateRect` | |
| | `GetMessage` (retrieves `WM_PAINT`) |
| `InvalidateRect` | |
| `InvalidateRect` | |
| | `WM_PAINT` dispatched |
| | `GetMessage` (retrieves `WM_PAINT`) |
| `InvalidateRect` | |
| | `WM_PAINT` dispatched |
| | `GetMessage` (retrieves `WM_PAINT`) |
| | `WM_PAINT` dispatched |
| | `GetMessage` (waits for a message) |

If the thread doing the painting manages to call `GetMessage` between each call to `InvalidateRect`, then it will see every invalidation. On the other hand (which is more likely), it only manages to call `GetMessage` after a few invalidations have taken place, it will see the accumulated invalidation in a single `WM_PAINT` message.

Now that you understand how generated messages work, you can answer this question which sometimes comes in:

> If the user is continuously moving the mouse, how many `WM_MOUSEMOVE` messages will I get?

Raymond Chen

**Follow**