Use WM_WINDOWPOSCHANGED to react to window state changes

devblogs.microsoft.com/oldnewthing/20080115-00

January 15, 2008

Raymond Chen

The documentation for the `WM_SHOWWINDOW` message points out that the message is not sent under certain circumstances. But what if you want to know when the window is shown, including in the cases where you don't get `WM_SHOWWINDOW`?

The `WM_WINDOWPOSCHANGED` message is sent at the end of the window state change process. It sort of combines the other state change notifications, `WM_MOVE`, `WM_SIZE`, and `WM_SHOWWINDOW`. But it doesn't suffer from the same limitations as `WM_SHOWWINDOW`, so you can reliably use it to react to the window being shown or hidden. The handler would go something like this:

```
void OnWindowPosChanged(HWND hwnd, const WINDOWPOS *pwp)
{
    if (pwp->flags & SWP_SHOWWINDOW) {
       window_was_shown();
    }
    if (pwp->flags & SWP_HIDEWINDOW) {
       window_was_hidden();
    }
    if (!(pwp->flags & SWP_NOMOVE)) {
       window_moved_to(pwp->x, pwp->y);
    }
    if (!(pwp->flags & SWP_NOSIZE)) {
       window_resized_to(pwp->cx, pwp->cy);
    }
}
HANDLE_MSG(hwnd, WM_WINDOWPOSCHANGED, OnWindowPosChanged);
```

Note also that if you don't pass the `WM_WINDOWPOSCHANGED` message to `DefWindowProc`, then you won't get `WM_MOVE` or `WM_SIZE` messages, since it is `DefWindowProc` that converts `WM_WINDOWPOSCHANGED` into the `WM_MOVE` and `WM_SIZE` messages.

"If `WM_WINDOWPOSCHANGED` is redundant with `WM_MOVE`, `WM_SIZE`, and `WM_SHOWWINDOW`, then why do we have those other messages anyway?"

The `WM_WINDOWPOSCHANGED` message wasn't invented until Windows 3.1. Prior to that, you had no choice but to react to those other messages. You can think of those other three messages as legacy messages now. There's nothing wrong with them, but they're kind of old-fashioned now.

Next time, we'll look at the companion message `WM_WINDOWPOSCHANGING`.

Postscript: This entry was inspired by an actual customer question regarding the cases where `WM_SHOWWINDOW` message is not sent if the program is run with the `SW_SHOWMAXIMIZED` state. Unfortunately, one detail I missed in the customer's question was the remark that they need to know when the window is shown because it is "critical for the application to initialize its state." I didn't follow up on that little remark, but I should have, because it's very strange to do initialization work when a window is shown. What if the window is never shown? Does this mean that the program will never initialize itself? (For example, somebody might have run your program with the `SW_HIDE` state.) The `WM_NCCREATE` and `WM_CREATE` are the more traditional places to do window initialization.

Raymond Chen

Follow