Every window with the WS_SYSMENU style has a system menu, but it's not there until it needs to be

devblogs.microsoft.com/oldnewthing/20100528-00

May 28, 2010

Raymond Chen

I mentioned last time that there's an optimization in the treatment of the system menu which significantly reduces the number of menus in the system. When a window has the `WS_SYSMENU` window style, it has a system menu, but until somebody calls `GetSystemMenu` on that window, nobody knows what its menu handle is. Until that point, the window manager doesn't actually have to commit to creating a menu for the window; it can just pretend that the window has one. (This technique goes by the fancy name *lazy initialization*.) The window manager creates a global default system menu which contains the standard system menu items. If somebody presses Alt+Space or otherwise calls up the system menu for a window that has never had `GetSystemMenu` called on it, the window manager just uses the global default system menu, since it knows that nobody has customized the menu. (You can't customize a menu you don't have the handle to!) Since most people never customize their system menu, this optimization avoids cluttering the desktop heap with identical copies of the same menu. This was a particularly important optimization back in the 16-bit days, when all window manager objects had to fit into a single 64KB heap (known as *System Resources*). If you are really sneaky, you can catch a glimpse of the elusive *global default system menu* as it whizzes by: As with any other popup menu, the handle to the menu being displayed is passed to your window's `WM_INITMENUPOPUP`, and if your program has never called `GetSystemMenu`, the handle that you will see is the global default system menu. Mind you, you can't do much to this menu, since the window manager blocks any attempt to modify it. (Otherwise, your program's menu modification would have an unintended effect on the menus of other programs!)

Therefore, if your program is in the habit of modifying its system menu in its `WM_INITMENUPOPUP` handler, you should stick a dummy call to `GetSystemMenu` in your `WM_CREATE` handler to force your system menu to change from a pretend system menu to a real one.

Raymond Chen

Follow