Why does Ctrl+ScrollLock cancel dialogs?

devblogs.microsoft.com/oldnewthing/20080211-00

February 11, 2008

Raymond Chen

Commenter Adam Russell asks why Ctrl+ScrollLock cancels dialogs. Easy. Because Ctrl+ScrollLock is the same as Ctrl+Break, and Ctrl+Break cancels dialogs. Okay, that answer actually raises two more questions. First, why is Ctrl+ScrollLock the same as Ctrl+Break? This is a consequence of the backward compatibility designed into the Enhanced Keyboard layout which is in widespread use today. If you go back and look at the original PC/XT keyboard layout, you'll see that many of the keys we're familiar with today simply didn't exist back then, and one key (PrtSc) existed in a very different form. Let's start with PrtSc. Observe that it was originally a shifted key, atop the asterisk. In the PC/AT keyboard layout, the PrtSc/* key migrated into the numeric keypad, and with the introduction of the enhanced keyboard layout, the two functions PrtSc and multiplication were split into two separate keys. The asterisk stayed with the numeric keypad, while the PrtSc function moved to the top row as a function key (sharing a key with the SysRq key). Okay, that's all nice, but what about Ctrl+ScrollLock and Ctrl+Break? Well, in the original PC/XT keyboard layout, there was no Break key. The key sequence for Break was Ctrl+ScrollLock. (And for completeness, the key sequence for Pause was Ctrl+NumLock.) Even though the enhanced keyboard moved the Pause and Break functions to their own key, pressing the Pause key internally generated scan codes that simulated a press of Ctrl+NumLock. In other words, when you pressed Pause, the keyboard hardware actually tells the computer, "The user pressed the Ctrl key and then pressed the NumLock key." Similarly, when you pressed Ctrl+Break, the keyboard hardware tells the computer, "The user pressed the Ctrl key and then pressed the ScrollLock key." Therefore, Ctrl+ScrollLock acts like Ctrl+Break because at the hardware level, they are *the same thing*. That the two functions exist on separate keys is just a fake-out by the keyboard hardware. Okay, so now that we understand why Ctrl+ScrollLock is the same as Ctrl+Break, the next question is why Ctrl+Break cancels a dialog box. If you look at the list of virtual key codes in `winuser.h`, you'll find lots of virtual keys that don't exist on the PC keyboard: `VK_CLEAR`, function keys `VK_F13` through `VK_F24`, and the long-forgotten `VK_PA1`. What are all these things? In the beginning, Windows and MS-DOS ran on more than just the IBM PC. I know for certain that it also ran on the NEC PC-98, and there were probably other architectures that were either explicitly supported or for which support was planned or reserved. The designers of the original Windows keyboard input driver model wanted to cover all of the likely bases and included support for keys beyond the basic 84-key PC/XT

keyboard, keys that could be found on then-popular keyboard layouts such as the VT-100 and even the 3270. I don't know which keyboard had a Cancel key, but presumably one did, or at least the people who designed the input driver model wanted to be prepared for the possibility of one showing up.

Given that you had a Cancel key, it seemed natural for the dialog manager to support it by treating a press of the Cancel key as the same as clicking the Cancel button on a dialog.

Raymond Chen

Follow