WM_KILLFOCUS is the wrong time to do field validation

devblogs.microsoft.com/oldnewthing/20040419-00

April 19, 2004

Raymond Chen

“I’ll do my field validation when I get a WM_KILLFOCUS message.” This is wrong for multiple reasons. First, you may not get your focus loss message until it’s too late. Consider a dialog box with an edit control and an OK button. The edit control validates its contents on receipt of the WM_KILLFOCUS message. Suppose the user fills in some invalid data. Under favorable circumstances, the user clicks the OK button. Clicking the OK button causes focus to move away from the edit control, so the edit control’s WM_KILLFOCUS runs and gets a chance to tell the user that the field is no good. Since button clicks do not fire until the mouse is released while still over the button, invalid data will pop up a message box, which steals focus, and now the mouse-button-release doesn’t go to the button control. Result: Error message and IDOK action does not execute. Now let’s consider less favorable circumstances. Instead of clicking on the OK button, the user just presses Enter or types the keyboard accelerator for whatever button dismisses the dialog. The accelerator is converted by IsDialogMessage into a WM_COMMAND with the button control ID. Focus **does not change**. So now the IDOK (or whatever) handler runs and calls EndDialog() or performs whatever action the button represents. If the dialog exits, then focus will leave the edit control as part of dialog box destruction, and only then will the validation occur, but it’s too late now. The dialog is already exiting. Alternatively, if the action in response to the button is not dialog termination but rather starting some other procedure, then it will do it based on the unvalidated data in the dialog box, which is likely not what you want. Only when that procedure moves focus (say, by displaying a progress dialog) will the edit control receive a WM_KILLFOCUS, at which time it is too late to do anything. The procedure (using the unvalidated data) is already under way. There is also a usability problem with validating on focus loss. Suppose the user starts typing data into the edit control, and then the user gets distracted. Maybe they need to open a piece of email that has the information they need. Maybe they got a phone call and need to look up something in their Contacts database. Maybe they went to the bathroom and the screen saver just kicked in. The user does not want a “Sorry, that partial information you entered is invalid” error dialog, because they aren’t yet finished entering the data. I’ve told you all the places you shouldn’t do validation but haven’t said where you **should**. Do the validation when the users indicate that they are done with data entry and want to go on to the next step. For a simple dialog, this would mean

performing validation when the OK or other action verb button is clicked. For a wizard, it would be when the Next button is clicked. For a tabbed dialog, it would be when the user tabs to a new page.

(Warnings that do not change focus are permitted, like the balloon tip that apperas if you accidentally turn on Caps Lock while typing your password.)

Raymond Chen

Follow