# If I have a modeless dialog box with custom accelerators, which should I call first: IsDialogMessage or TranslateAccelerator

devblogs.microsoft.com/oldnewthing/20160818-00

August 18, 2016

Raymond Chen

A customer had a modeless dialog box with custom accelerators.

If their window had been a modeless dialog box without custom accelerators, then their message dispatch would be

```
if (!IsDialogMessage(hdlg, &msg)) {
  TranslateMessage(&msg);
  DispatchMessage(&msg);
}
```

On the other hand, if their window with accelerators had been a plain window rather than a dialog box, then their message dispatch would be

```
if (!TranslateAccelerator(hwnd, hacc, &msg)) {
  TranslateMessage(&msg);
  DispatchMessage(&msg);
}
```

But since they have both, the question arises: Which should I do first, the `IsDialog-Message` or the `TranslateAccelerator`?

The customer experimented and found that they had to call `TranslateAccelerator` first:

```
if (!TranslateAccelerator(hwnd, hacc, &msg) &&
    !IsDialogMessage(hdlg, &msg)) {
  TranslateMessage(&msg);
  DispatchMessage(&msg);
}
```

If they flipped the order, they found that accelerators were not being translated:

```
// Code in italics is wrong.
if (!IsDialogMessage(hdlg, &msg) &&
    !TranslateAccelerator(hwnd, hacc, &msg)) {
  TranslateMessage(&msg);
  DispatchMessage(&msg);
}
```

The customer empirically determined that you have to translate the accelerator first. (Or they could have read my article on the subject of custom accelerators in dialog boxes and seen the correct order.) But why is that the correct order?

The answer goes back to the return values of the two function. The `TranslateAccelerator` function returns a nonzero value if it recognized the message as an accelerator (and posted a `WM_COMMAND` message). The `IsDialogMessage` function returns a nonzero value if it recognized the message as a message for the dialog (and dispatched it).

Now look at what happens if you have a message that is *both* a message for the dialog *and* an accelerator. For example, focus is on a button control in the dialog box and you press, say, `Alt`+`F2`.

Let's say you call `IsDialogMessage` first. The `IsDialogMessage` function says, "Why yes, this message is for the dialog box, so I dispatched it to the button control. Mission accomplished. I'm so awesome!" The `IsDialogMessage` function returns a non-zero value, and the `TranslateAccelerator` never gets a chance to run.

On the other hand, if you call `TranslateAccelerator` first, then the `TranslateAccelerator` function sees the accelerator key and posts the `WM_COMMAND` function to the dialog window, then it returns a non-zero value to say "Why yes, this message is an accelerator, so I posted a `WM_COMMAND` message. Mission accomplished. I'm so awesome!" The `TranslateAccelerator` function returns a non-zero value, and the `IsDialogMessage` never gets a chance to run.

The question of which one to call first is therefore a matter of priority. If the user presses the accelerator key while focus is on a control in the dialog box, which is more important to you: The fact that it is an accelerator, or the fact that it is a message that targets the dialog box? Whichever is more important to you goes first.

But wait, that's not the end of the story. Note that the above code calls `TranslateAccelerator` unconditionally, which means that the accelerator keys are active even if focus is not on the dialog box at all. For example, focus may be on another window on the same thread (say, the owner of the modeless dialog box). You probably don't want the modeless dialog box stealing accelerators from the owner. To avoid this problem, you need to translate accelerators for your dialog box only if the focus is somewhere in your dialog box.

```
if (!((hdlg == msg.hwnd || IsChild(hdlg, msg.hwnd)) &&
      !TranslateAccelerator(hdlg, hacc, &msg)) &&
    !IsDialogMessage(hdlg, &msg)) {
 TranslateMessage(&msg);
 DispatchMessage(&msg);
}
```

This code should look familiar, since I copied it from my original article.

Raymond Chen

**Follow**