The WM_GETDLGCODE message is a query message and should not modify state

devblogs.microsoft.com/oldnewthing/20230329-00

March 29, 2023

Raymond Chen

A customer had a dialog box that had initial focus on a combo box, and they found that many users were pressing Enter in a mistaken belief that it would open the combo box dropdown, but instead it was dismissing the dialog box. The code then proceeded with the value in the combo box, which was typically not the value the user wanted. (We know that because the user pressed Enter in a failed attempt to open the combo box dropdown.)

The customer wanted to customize the dialog box so that pressing Enter on the combo box opened the combo box dropdown rather than dismissing the dialog.

The customer figured that they would subclass the combo box and respond to the `WM_GETDLGCODE` message with `DLGC_WANTMESSAGE` if the message is a `WM_KEYDOWN` of `VK_RETURN`. The code went something like this:

```
// Code in italics is wrong.
LRESULT CALLBACK ComboBoxDropDownOnEnterSubclassProc(
    HWND hwnd,
    UINT message,
    WPARAM wParam,
    LPARAM lParam,
    UINT_PTR uIdSubclass,
    DWORD_PTR /* dwRefData */)
{
    switch (message) {
    case WM_NCDESTROY:
        RemoveWindowSubclass(hwnd,
            ComboBoxDropDownOnEnterSubclassProc,
            uIdSubclass);
        break;

    case WM_GETDLGCODE:
        if (auto msg = (MSG*)lParam; msg &&
            msg->message == WM_KEYDOWN &&
            msg->wParam == VK_RETURN &&
            SendMessage(hwnd, CB_GETDROPPEDSTATE, 0, 0) == 0) {
            // The user pressed Enter on a closed combo box.
            // Open the dropdown and say that we want the message.
            SendMessage(hwnd, CB_SHOWDROPDOWN, TRUE, 0);
            return DLGC_WANTMESSAGE;
        }
        break;
    }

    return DefSubclassProc(hwnd, message, wParam, lParam);
}

INT_PTR CALLBACK DialogProc(
    HWND hdlg,
    UINT message,
    WPARAM wParam,
    LPARAM lParam)
{
    switch (message) {
    case WM_INITDIALOG:
        SetWindowSubclass(GetDlgItem(hdlg, IDC_OPTION),
            ComboBoxDropDownOnEnterSubclassProc, 0, 0);
        /* other initialization... */
        return TRUE;

    /* other message handlers... */
    }
    return FALSE;
}
```

The idea here is that we customize the `WM_GETDLGCODE` message handler by checking if the message being processed is a press of the Enter key. If so, and if the combo box is not dropped down, we drop it down and say that we want the message.

However, the customer found that this did not work. When the user pressed the Enter key, the drop down did show, but it immediately closed. What happened?

The problem here is that the `WM_GETDLGCODE` message is a *query* message, not an *action* message. Anybody can send the `WM_GETDLGCODE` message to ask the window, "So, *if* I were processing this message, how *would* you like the dialog manager to behave?" You don't actually *do* anything yet, because the caller might just be asking for information unrelated to processing the message. For example, maybe it's asking if you support the `EM_SETSEL` message or whether you are the default pushbutton because it's managing focus and wants to know what your control prefers.

What happens is that the code is responding to the query by opening the dropdown, and then the dialog manager says, "Oh, so you want that message? Great. Let me send you that message." And then the combo box gets the Enter key, and the default behavior for hitting Enter in an open combo box is to close it!

If you want to process a message in a special way, you have to do it in two steps.

- Respond to `WM_GETDLGCODE` by saying, "Yes, I want this message."
- Respond to the message by doing the thing.

Another mistake the code makes is getting so excited about the message that it forgets to answer the other questions, like "Do you support `WM_SETSEL`?"

Let's fix both of the problems.

```
LRESULT CALLBACK ComboBoxDropDownOnEnterSubclassProc(
    HWND hwnd,
    UINT message,
    WPARAM wParam,
    LPARAM lParam,
    UINT_PTR uIdSubclass,
    DWORD_PTR /* dwRefData */)
{
    switch (message) {
    case WM_NCDESTROY:
        RemoveWindowSubclass(hwnd,
            ComboBoxDropDownOnEnterSubclassProc,
            uIdSubclass);
        break;

    case WM_GETDLGCODE:
        if (auto msg = (MSG*)lParam; msg &&
            msg->message == WM_KEYDOWN &&
            msg->wParam == VK_RETURN &&
            SendMessage(hwnd, CB_GETDROPPEDSTATE, 0, 0) == 0) {
            // The user pressed Enter on a closed combo box.
            // Say that we want the message, in addition to whatever
            // else the combo box wants to say.
            return DLGC_WANTMESSAGE |
                   DefSubclassProc(hwnd, message, wParam, lParam);
        }
        break;

    case WM_KEYDOWN:
        // Pressing Enter on a closed combo box opens the dropdown.
        if (wParam == VK_RETURN &&
            SendMessage(hwnd, CB_GETDROPPEDSTATE, 0, 0) == 0) {
            SendMessage(hwnd, CB_SHOWDROPDOWN, TRUE, 0);
            return 0;
        }
        break;
    }

    return DefSubclassProc(hwnd, message, wParam, lParam);
}
```

We add the `DLGC_WANTMESSAGE` flag to any other flags which the original combo box wanted to return for this message. That way, we preserve the other combo box custom behaviors.

Only when we actually receive the `WM_KEYDOWN` with `VK_RETURN` when the combo box is closed do we show the dropdown.

Now, this code is working a little bit too hard, because the combo box itself will say that it wants the `VK_RETURN` if the combo box is dropped. Therefore, we can just say we want all `VK_RETURN` keys, avoiding the need to check whether the dropdown is shown.

```
LRESULT CALLBACK ComboBoxDropDownOnEnterSubclassProc(
    HWND hwnd,
    UINT message,
    WPARAM wParam,
    LPARAM lParam,
    UINT_PTR uIdSubclass,
    DWORD_PTR /* dwRefData */)
{
    switch (message) {
    case WM_NCDESTROY:
        RemoveWindowSubclass(hwnd,
            ComboBoxDropDownOnEnterSubclassProc,
            uIdSubclass);
        break;

    case WM_GETDLGCODE:
        if (auto msg = (MSG*)lParam; msg &&
            msg->message == WM_KEYDOWN &&
            msg->wParam == VK_RETURN) {
            // && SendMessage(hwnd, CB_GETDROPPEDSTATE, 0, 0) == 0
            // The user pressed Enter on a closed combo box.
            // Say that we want the message, in addition to whatever
            // else the combo box wants to say.
            return DLGC_WANTMESSAGE |
                   DefSubclassProc(hwnd, message, wParam, lParam);
        }
        break;

    case WM_KEYDOWN:
        // Pressing Enter on a closed combo box opens the dropdown.
        if (wParam == VK_RETURN &&
            SendMessage(hwnd, CB_GETDROPPEDSTATE, 0, 0) == 0) {
            SendMessage(hwnd, CB_SHOWDROPDOWN, TRUE, 0);
            return 0;
        }
        break;
    }

    return DefSubclassProc(hwnd, message, wParam, lParam);
}
```