Using the TAB key to navigate in non-dialogs

devblogs.microsoft.com/oldnewthing/20031021-00

October 21, 2003

Raymond Chen

The IsDialogMessage function works even if you aren't a dialog. As long as your child windows have the WS_TABSTOP and/or WS_GROUP styles, they can be navigated as if they were part of a dialog box. One caveat is that IsDialogMessage will send DM_GETDEFID and DM_SETDEFID messages to your window, which are message numbers WM_USER and WM_USER+1, so you should avoid using those messages in your window procedure for some other purpose.

These changes to our scratch program illustrate how you can use the TAB key to navigate within a non-dialog.

```
HWND g_hwndLastFocus;
void OnSetFocus(HWND hwnd, HWND hwndOldFocus)
{
    if (g_hwndLastFocus) {
        SetFocus(g_hwndLastFocus);
    }
}
void OnActivate(HWND hwnd, UINT state,
                HWND hwndActDeact, BOOL fMinimized)
{
    if (state == WA_INACTIVE) {
        g_hwndLastFocus = GetFocus();
    }
}
// Just display a messagebox so you can see something
void OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify)
{
    switch (id) {
    case 100:
        MessageBox(hwnd, TEXT("Button 1 pushed"),
                   TEXT("Title"), MB_OK);
        break;
    case 101:
        MessageBox(hwnd, TEXT("Button 2 pushed"),
                   TEXT("Title"), MB_OK);
        break;
    case IDCANCEL:
        MessageBox(hwnd, TEXT("Cancel pushed"),
                   TEXT("Title"), MB_OK);
        break;
    }
}
BOOL
OnCreate(HWND hwnd, LPCREATESTRUCT lpcs)
{
    HWND hwndChild =
        CreateWindow(
        TEXT("button"),                 /* Class Name */
        TEXT("Button &1"),              /* Title */
        WS_CHILD | WS_VISIBLE | WS_TABSTOP |
        BS_DEFPUSHBUTTON | BS_TEXT,     /* Style */
        0, 0, 100, 100,                 /* Position and size */
        hwnd,                           /* Parent */
        (HMENU)100,                     /* Child ID */
        g_hinst,                        /* Instance */
        0);                             /* No special parameters */
    if (!hwndChild) return FALSE;
    g_hwndLastFocus = hwndChild;
    hwndChild =
        CreateWindow(
        TEXT("button"),                 /* Class Name */
        TEXT("Button &2"),              /* Title */
        WS_CHILD | WS_VISIBLE | WS_TABSTOP |
        BS_PUSHBUTTON | BS_TEXT,        /* Style */
        100, 0, 100, 100,               /* Position and size */
        hwnd,                           /* Parent */
        (HMENU)101,                     /* Child ID */
        g_hinst,                        /* Instance */
        0);                             /* No special parameters */
    if (!hwndChild) return FALSE;
    hwndChild =
        CreateWindow(
        TEXT("button"),                 /* Class Name */
        TEXT("Cancel"),                 /* Title */
        WS_CHILD | WS_VISIBLE | WS_TABSTOP |
        BS_PUSHBUTTON | BS_TEXT,        /* Style */
        200, 0, 100, 100,               /* Position and size */
        hwnd,                           /* Parent */
        (HMENU)IDCANCEL,                /* Child ID */
        g_hinst,                        /* Instance */
        0);                             /* No special parameters */
    if (!hwndChild) return FALSE;
    return TRUE;
}
//  Add to WndProc
    HANDLE_MSG(hwnd, WM_COMMAND, OnCommand);
    HANDLE_MSG(hwnd, WM_ACTIVATE, OnActivate);
    HANDLE_MSG(hwnd, WM_SETFOCUS, OnSetFocus);
    // Add blank case statements for these to ensure we don't use them
    // by mistake.
    case DM_GETDEFID: break;
    case DM_SETDEFID: break;
//  Change message loop
    MSG msg;
    while (GetMessage(&msg, NULL, 0, 0)) {
        if (IsDialogMessage(hwnd, &msg)) {
            /* Already handled by dialog manager */
        } else {
            TranslateMessage(&msg);
            DispatchMessage(&msg);
        }
    }
```

One subtlety is the additional handling of the WM_ACTIVATE and WM_SETFOCUS messages to preserve the focus when the user switches away from the window and back. Notice also that we picked Button 1 as our initial default button by setting it with the BS_DEFPUSHBUTTON style.

Observe that all the standard dialog accelerators now work. The TAB key navigates, the Alt+1 and Alt+2 keys act as accelerators for the two buttons, the Enter key presses the default button, and the ESC key pushes the Cancel button since its control ID is IDCANCEL.

Raymond Chen

Follow