Should I translate my window class names?

devblogs.microsoft.com/oldnewthing/20220331-00

March 31, 2022

Raymond Chen

The default Windows Desktop project template used by Visual Studio loads the window class name from resources:

```
WCHAR szWindowClass[MAX_LOADSTRING];

LoadStringW(hInstance, IDC_PROJECTNAME, szWindowClass, MAX_LOADSTRING);

ATOM MyRegisterClass(HINSTANCE hInstance)
{
    WNDCLASSEXW wcex;

    wcex.cbSize = sizeof(WNDCLASSEX);

    wcex.style          = CS_HREDRAW | CS_VREDRAW;
    wcex.lpfnWndProc    = WndProc;
    wcex.cbClsExtra     = 0;
    wcex.cbWndExtra     = 0;
    wcex.hInstance      = hInstance;
    wcex.hIcon          = LoadIcon(hInstance, MAKEINTRESOURCE(IDI_PROJECTNAME));
    wcex.hCursor        = LoadCursor(nullptr, IDC_ARROW);
    wcex.hbrBackground  = (HBRUSH)(COLOR_WINDOW+1);
    wcex.lpszMenuName   = MAKEINTRESOURCEW(IDC_PROJECTNAME);
    wcex.lpszClassName  = szWindowClass;
    wcex.hIconSm        = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(IDI_SMALL));

    return RegisterClassExW(&wcex);
}
```

Do window class name need to be a translatable resource?

Window class names are never shown to the user, so there's no need for them to be translated.

Okay, but even though there's no requirement that they be translated, *should* they be translated?

No, they shouldn't.

Window class names are often the subject of automation. Screen readers and other assistive technologies use class names to identify windows so that they can apply custom functionality for that window. For example, if they see a window whose class name is `CabinetWClass`, then they know that it is an Explorer window and can activate Explorer-specific behaviors, like maybe enabling special commands for opening and closing the Preview Pane. If the name of the window class changed as the user changed languages, the assistive technology tools would have to keep updating to catch up with the different translations.

Back in the days of 16-bit Windows, it was common for the programmatic interface to parts of the system to be based on window class names. For example, if you wanted to open a particular Control Panel, you launched `control.exe`, and then looked for a window whose class name is `Control Panel` and sent it a specific message. If these class names were translated for each locale, it would be nearly impossible to write an installer: You would have to write something like this:

```
HWND FindControlPanelWindow()
{
    // English?
    HWND hwnd = FindWindow("Control Panel", NULL);
    if (hwnd) return hwnd;

    // German?
    HWND hwnd = FindWindow("Systemsteuerung", NULL);
    if (hwnd) return hwnd;

    // French?
    HWND hwnd = FindWindow("Panneau de configuration", NULL);
    if (hwnd) return hwnd;

    // Vietnamese? (Điều khiển)
    // Note that Vietnamese uses code page 1258, but our source
    // code is in code page 1252, so this string is intentional
    // mojibake.[1]
    HWND hwnd = FindWindow("ĐiêÌu khiêÒ", NULL);
    if (hwnd) return hwnd;

    //... and so on ...
}
```

And then each time Windows added support for another language (over 100 of them now), you'd have to ship an updated installer.

For these reasons, window class names should be treated as programmatic names and should not be localized.

[1] By an interesting coincidence, code unit D0 in code page 1258 is U+0110 (LATIN CAPITAL LETTER D WITH STROKE) "Đ" which looks very similar to code unit D0 in code page 1252: U+00D0 (LATIN CAPITAL LETTER ETH) "Ð".

Raymond Chen

Follow