What does it mean for a window to be Unicode?

devblogs.microsoft.com/oldnewthing/20180906-00

September 6, 2018

Raymond Chen

The `IsWindowUnicode` function reports whether the current window procedure expects Unicode window messages. What specifically does this mean?

It means that messages like `WM_CHAR` will report characters as Unicode code units rather than as ANSI code units.

Okay, so what determines whether a window procedure receives Unicode messages?

When a window is created, its initial window procedure comes from the class registration. If the class was registered with `RegisterClass[Ex]W`, then it is a Unicode window procedure; otherwise it was registered with `RegisterClass[Ex]A` and is an ANSI window procedure.

If a window is subclassed with a function pointer, then the new window procedure is Unicode if it was set with `SetWindowLongPtrW(GWLP_WNDPROC)` and ANSI if it was set with `SetWindowLongPtrA(GWLP_WNDPROC)`.

As we saw some time ago, `GetWindowLongPtr(GWLP_WNDPROC)` returns a thunk if you use the ANSI version but the current window procedure is Unicode, or vice versa. Suppose the window procedure is Unicode and you call `GetWindowLongPtrA(GWLP_WNDPROC)`. This will return you a thunk, and if you later set that thunk back as the window procedure with `SetWindowLongPtrA(GWLP_WNDPROC)`, then the original Unicode window procedure is restored, and the window is a Unicode window again.

Note that calling `SetWindowLongPtrA(GWLP_WNDPROC)` can result in a window procedure that is Unicode.

As a window message travels from one window procedure to another window procedure (via `CallWindowProc`), the character set may flip between Unicode and ANSI. The `IsWindowUnicode` function tells you only about the window's current window procedure.

Raymond Chen

Follow