Why is CLIPFORMAT defined to be a WORD rather than a UINT?

devblogs.microsoft.com/oldnewthing/20111128-00

November 28, 2011

Raymond Chen

Commenter Ivo wants to know if the `RegisterClipboardFormat` function returns a `UINT`, why is the `CLIPFORMAT` data type defined to be a `WORD`? Since a `WORD` is smaller than a `UINT`, you have to stick in a cast every time you assign the result of `RegisterClipboard-Format` to a `CLIPFORMAT`. Rewind to 16-bit Windows. Back in those days, a `UINT` and a `WORD` were the same size, namely, 16 bits. As a result, people got lazy about the distinction. Six of one, a half dozen of the other. (People are lazy about this sort of distinction even today, assuming for example that `UINT` and `DWORD` are the same size, and in turn forcing `UINT` to remain a 32-bit integer type even on 64-bit Windows.) The `RegisterClipboardFormat` function came first, and when the OLE folks wanted to define a friendly name for the data type to hold a clipboard format, they said, “Well, a clipboard format is a 16-bit integer, so let me use a 16-bit integer.” A `WORD` is a 16-bit integer, so there you go. This mismatch had no effect in 16-bit code, but once Win32 showed up, you had a problem since 32-bit Windows expanded the `UINT` type to 32 bits. Not only does keeping a `CLIPFORMAT` in a `WORD` create the need for all this casting, it also leaves two bytes of padding in the `FORMATETC` structure. Strike two.

Yeah, basically, it sucks.

Raymond Chen

Follow