What is the deal with the ES_OEMCONVERT flag?

devblogs.microsoft.com/oldnewthing/20050719-12

July 19, 2005

Raymond Chen

The `ES_OEMCONVERT` edit control style is a holdover from 16-bit Windows. This ancient MSDN article from the Windows 3.1 SDK describes the flag thus:

> ES_OEMCONVERT causes text entered into the edit control to be converted from ANSI to OEM and then back to ANSI. This ensures proper character conversion when the application calls the **AnsiToOem** function to convert a Windows string in the edit control to OEM characters. ES_OEMCONVERT is most useful for edit controls that contain filenames.

Set the wayback machine to, well, January 31, 1992, the date of the article.

At this time, the predominant Windows platform was Windows 3.0. Windows 3.1 was still a few months away from release, and Windows NT 3.1 was over a year away. The predominant file system was 16-bit FAT, and the relevant feature of FAT of this era for the purpose of this discussion is that file names were stored on disk in the OEM character set. (We discussed the history behind the schism between the OEM and ANSI code pages in an earlier article.)

Since GUI programs used the ANSI character set, but file names were stored in the OEM character set, the only characters that could be used in file names from GUI programs were those that exist in both character sets. If a character existed in the ANSI character set but not the OEM character set, then there would be no way of using it as a file name; and if a character existed in the OEM character set but not the ANSI character set, the GUI program couldn’t manipulate it.

The `ES_OEMCONVERT` flag on a edit control ensures that only characters that exist in both the ANSI and OEM character sets are used, hence the remark “ES_OEMCONVERT is most useful for edit controls that contain filenames”.

Fast-forward to today.

All the popular Windows file systems support Unicode file names and have for ten years. There is no longer a data loss converting from the ANSI character set to the character set used by the file system. Therefore, there is no need to filter out any characters to forestall the user typing a character that will be lost during the conversion to a file name. In other words, the `ES_OEMCONVERT` flag is pointless today. It's a leftover from the days before Unicode.

Indeed, if you use this flag, you make your program worse, not better, because it unnecessarily restricts the set of characters that the user will be allowed to use in file names. A user running the US-English version of Windows would not be allowed to enter Chinese characters as a file name, for example, even though the file system is perfectly capable of creating files whose names contain those characters.

Raymond Chen

Follow