How do you deal with an input stream that may or may not contain Unicode data?

devblogs.microsoft.com/oldnewthing/20120917-00

September 17, 2012

Raymond Chen

Dewi Morgan reinterpreted a question from a Suggestion Box of times past as "How do you deal with an input stream that may or may not contain Unicode data?" A related question from Dave wondered how applications that use `CP_ACP` to store data could ensure that the data is interpreted in the same code page by the recipient. "If I send a .txt file to a person in China, do they just go through code pages until it seems to display correctly?" These questions are additional manifestations of *Keep your eye on the code page*. When you store data, you need to have some sort of agreement (either explicit or implicit) with the code that reads the data as to how the data should be interpreted. Are they four-byte sign-magnitude integers stored in big-endian format? Are they two-byte ones-complement signed integers stored in little-endian format? Or maybe they are IEEE floating-point data stored in 80-bit format. If there is no agreement between the two parties, then confusion will ensue. That your data consists of text does not exempt you from this requirement. Is the text encoded in UTF-16LE? Or maybe it's UTF-8. Or perhaps it's in some other 8-bit character set. If the two sides don't agree, then there will be confusion. In the case of files encoded in `CP_ACP`, you have a problem if the source and destination have different values for `CP_ACP`. That text file you generate on a US-English system (where `CP_ACP` is 1252) may not make sense when decoded on a Chinese-Simplified system (where `CP_ACP` is 936). It so happens that all Windows 8-bit code pages agree on code points 0 through 127, so if you restrict yourself to that set, you are safe. The Windows shell team was not so careful, and they slipped some characters into a header file which are illegal when decoded in code page 932 (the `CP_ACP` used in Japan). The systems in Japan do not cycle through all the code pages looking for one that decodes without errors; they just use their local value of `CP_ACP`, and if the file makes no sense, then I guess it makes no sense. If you are in the unfortunate situation of having to consume data where the encoding is unspecified, you will find yourself forced to guess. And if you guess wrong, the result can be embarrassing. **Bonus chatter**: I remember one case where a customer asked, "We need to convert a string of chars into a string of wchars. What code page should we pass to the `MultiByteToWideChar` function?" I replied, "What code page is your char string in?"

There was no response. I guess they realized that once they answered that question, they had their answer.

Raymond Chen

Follow