On the fuzzy definition of a "Unicode application"

devblogs.microsoft.com/oldnewthing/20060315-16

March 15, 2006

Raymond Chen

Commenter mpz wondered why the IME cannot detect whether it is sending characters to a Unicode or non-Unicode application and generate the appropriate character accordingly. But what exactly is a Unicode application? Actually, let me turn the question around: What is a non-Unicode application? Suppose you write a program and don't `#define UNICODE`, so you'd think you have a non-Unicode application. But your program uses a control provided by another library, and the authors of that library defined `UNICODE`. The controls created by that library are therefore Unicode, aren't they? Now you type that frustrating character to a control created by that library. Should it generate a U+00A5 or a U+005C? To know the answer to that question requires psychic powers. If the control takes the character and uses it exclusively internally, then presumably the IME should generate U+00A5. But if the control takes the character and returns it back to your program (say the control is a fancy edit control), then presumably the IME should generate U+005C. How does it know? It's not going to do some sort of analysis of the code in the helper library to decide what it's going to do with that character. Even human beings with access to the source code may have difficulty deciding whether the character will ever get converted to the `CP_ACP` code page in the future. Indeed, if the decision is based on the user's future actions, then you will need to invoke some sort of clairvoyance (and relinquishing of free will) to get the correct answer.

Note that this helper library might be in the form of a static library, in which case your application is really neither Unicode nor ANSI, but rather a mix of the two. Parts of it are Unicode and parts are ANSI. What's a poor IME to do?

Raymond Chen

Follow