L2/18-187

# Feedback on the characterization of the UTF-32 encoding form

Henri Sivonen (hsivonen@mozilla.com)

2018-05-02

On page 35 of Unicode 10.0, the “Preferred Usage” section and figure 2-11 could be improved.

The “Preferred Usage” section says “UTF-32 may be a preferred encoding form where memory or disk storage space for characters is not a particular concern, but were fixed-width, single code unit access to characters is desired. UTF-32 is also a preferred encoding form for processing characters in most Unix platforms.”

The first sentence is problematic, because the desirability of single code unit access to characters is overvalued among people who are not yet properly familiar with Unicode—in particular with combining characters and UAX #29 making indexability by code point (as opposed to iterability) quite a bit less useful that it might first appear. In that sense, it's a bit dangerous to suggest that this property might be desirable without including a discussion about how rarely it actually is desirable. It seems to me that this section has the problem that it tries to find cases where UTF-32 is preferred, but there very rarely are reasons to prefer UTF-32 and the Standard doesn’t appear to want to say so.

The sentence about Unix platforms is quite misleading. While `wchar_t` is typically bound to UTF-32 on Unix platforms, Unix platforms tend to prefer UTF-8 both on disk and in system APIs.

I suggest rewriting the section along these lines: “**Preferred Usage.** UTF-32 is commonly used for APIs that deal with single code points. It is rare for UTF-32 to be the most appropriate encoding form for strings of multiple code points.” and adding text along the lines of “On Unix platforms the legacy `wchar_t` type in C and C++ is typically bound to UTF-32. (See *Section 5.2 Programming Languages and Data Types* for discussion of the portable `char16_t` and `char32_t` types.)” as the last paragraph under the “UTF-32” heading before the “Fixed Width” section so as to not frame `wchar_t` as “preferred” in any way.

As for figure 2-11, it is misleading for the three rows for UTF-32, UTF-16 and UTF-8 to have equal width. It would illustrate the encoding forms better if a byte was allocated equal width on each line so that the line for UTF-32 would be wider than the lines for UTF-16 and UTF-8 (as seen in Figure 2-12 when illustrating encoding schemes). It would probably be the best to keep the lines right-aligned to illustrate that the astral character is equally wide in all three encoding forms. (Right-aligning the rows would improve Figure 2-12 as well.)