L2/18-185

Feedback related to taking security considerations into account in informative statements

Henri Sivonen (hsivonen@mozilla.com)

2018-05-02

In Unicode 10.0, there are some informative statements that could be improved in order to better take into account security considerations—particularly section *3.1.2 Substituting for Ill-Formed Subsequences* of UTR #36.

On page 83 in section *3.2 Conformance Requirements* in an example given in the first bullet point under C10, the standard says that a process could treat “an illegally terminated code unit sequence” “for example, by” “filtering the code unit out”. Just filtering the code unit out is more dangerous than the two alternatives presented: “signaling an error” or “representing the code unit with a marker such as U+FFFD REPLACEMENT CHARACTER”. For example, in the context of a data format such as HTML that can carry both inactive content (human-readable text) and active content (JavaScript program code), it is important that errors are not simply removed such that text before and after the error joins together to form code with a potentially dangerous function.

I suggest striking “, filtering the code unit out,” from the example and adding another sentence at the end of the paragraph saying: “While simply ignoring an ill-formed code unit sequence qualifies as not interpreting it as characters, silently ignoring ill-formed sequences is ill-advised, because joining text from before the ill-formed sequence and after the ill-formed sequence can cause the resulting text to take a new meaning, which might be especially dangerous in the context of textual formats that carry embedded program code, such as JavaScript.”

Similarly, on page 203 in section *5.3 Unknown and Missing Characters*, an introductory paragraph says that “simply display nothing” is an option for rendering unknown code points. While the same page goes on to discuss the concept of Default Ignorable Code Points, it would be prudent to qualify the “simply display nothing” option by scoping it to default ignorables in order not to suggest that display nothing is a generally valid fallback (as displaying nothing can mislead the human reader).

On page 257 in section *5.22 Best Practice for U+FFFD Substitution*, the last paragraph that covers “legacy character encodings and other character encodings that are defined externally” says vaguely: “Ultimately, that depends on the content of character mapping tables and their accompanying conversion algorithms.” It would be useful to point to the WHATWG Encoding Standard, which defines precise algorithms for Web-relevant legacy encodings, and to articulate the principle that the algorithms in the WHATWG Encoding Standard embody.

I suggest adding text along these lines: “For Web-relevant legacy encodings, the rules for U+FFFD substitution are defined in the WHATWG Encoding Standard[1]. The general principle for ASCII-compatible multi-byte encodings is that when an ASCII-range trail byte occurs in a multi-byte sequence that either doesn’t fit the general byte pattern of the encoding or that fits the pattern but doesn’t identify a mapped character according to the encoding’s mapping table, the trail byte in the ASCII range must not be subsumed into the U+FFFD substitution together with the non-ASCII lead byte but instead must be reprocessed as a single-byte (ASCII) sequence. This ensures that the introduction of an erroneous lead byte cannot mask and ASCII byte which might be security-relevant in a computer-readable syntax such as HTML or JavaScript.”

1 https://encoding.spec.whatwg.org/