# **Noncharacter Clarification**

Authors: Markus Scherer & Mark Davis Date: 2013-jan-16

#### Issue

Statements in the Unicode Standard about noncharacters can be misleading, and some implementations inappropriately reject Unicode text containing any noncharacters.

#### Proposal

Change the text in the standard on noncharacters to make it clearer how they should be used. Remove text that noncharacters "should never be interchanged" because it is unclear what "interchanged" means and invites over-rejection.

## Section 3.4 Characters and Encoding

Change D14 by removing text as indicated:

*Noncharacter*: A code point that is permanently reserved for internal use and that should never be interchanged. Noncharacters consist of the values U+nFFFE and U+nFFFF (where *n* is from 0 to 10<sub>16</sub>) and the values U+FDDO..U+FDEF.

## Section 16.7 Noncharacters

Change paragraph 1 by removing text as indicated:

Noncharacters are code points that are permanently reserved in the Unicode Standard for internal use. They are forbidden for use in open interchange of Unicode text data. See Section 3.4, Characters and Encoding, for the formal definition of noncharacters and conformance requirements related to their use.

Change paragraph 3 from

Applications are free to use any of these noncharacter code points internally but should *never* attempt to exchange them. If a noncharacter is received in open interchange, an application is not required to interpret it in any way. It is good practice, however, to recognize it as a noncharacter and to take appropriate action, such as replacing it with U+FFFD replacement character, to indicate the problem in the text. It is not recommended to simply delete ...

to

Applications are free to use any of these noncharacter code points internally. If a noncharacter is received in open interchange, an application is not required to interpret it in any way. It is good practice to support all characters, including private use and noncharacter code points, except when those are used with a different meaning internally. For example, if an implementation uses one of the noncharacters, then on input from external sources any instances of that noncharacter could be converted to a U+FFFD replacement character. However, tools (like a text editor), interchange formats like JSON or XML, and programming languages like Java, should support private use and noncharacter code points. It is not recommended to simply delete ...

Change paragraph 4 by removing text as indicated:

In effect, noncharacters can be thought of as application-internal private-use code points. Unlike the

private-use characters discussed in *Section 16.5, Private-Use Characters*, which *are* assigned characters <del>and</del> <del>which *are* intended for use in open interchange</del>, subject to interpretation by private agreement, noncharacters are permanently reserved (unassigned) and have no interpretation whatsoever outside of their possible application-internal private uses.

### Background

"Internal use" of noncharacters should not be limited to ephemeral occurrence during program execution. For example, CLDR collation uses contractions starting with U+FDDo and U+FDD1 to convey necessary data for "alphabetic indexes", for example in ICU's AlphabeticIndex class. This usage internal to CLDR and its implementations like ICU is supported by .xml, .txt and binary data files which are generated, edited, parsed, diff'ed, stored in version control systems, and released as part of Unicode libraries.

However, current versions of Linux software like gedit (default Gnome text editor), meld and kdiff3 (popular diff/merge tools) show error messages like "The file you opened has some invalid characters." for files containing noncharacters (as well as containing U+0000 [not helpful either] and surrogates [properly rejected in UTF-8 files]). gedit shows noncharacters in the same format as illegal byte sequences (as sequences of escaped bytes on red background).

Since "internal use" can require "interchange", it is problematic for the standard to say that noncharacters are not allowed in "open interchange". It invites over-rejection of noncharacters, to the point where "internal use" is more cumbersome than it should be.

It is also problematic, and unnecessary, to recommend without specifics that noncharacters be replaced with U+FFFD.

As 16.7 paragraph 4 says, "noncharacters can be thought of as application-internal private-use code points". They are "better than" using non-reserved unassigned code points (since they can be assigned in the future), they are "better than" using ill-formed sequences, and they are more appropriate for internal use than private-use code points. In particular, BMP PUA code points are widely used for CJK characters, symbols, and not-yet-encoded scripts. It is also desirable for internal use to take advantage of BMP code points; that is why U+FDDO..U+FDEF were reserved as noncharacters.

The Unicode Standard should be more clear about noncharacters, and provide proper guidance to implementers.

Comments from Mark:

What we finally ended up with in the UTC discussions was that noncharacters are "super" private use characters. They shouldn't be freely interchanged, and their usage should be restricted. We get that by saying that any process \*may\* replace them on input, and should if they are not coming from a trusted source. So if I am using three of them for sentinel values internally, when I get some random text on the web, to sanitize it I can and should replace \*those three\* in the text by U+FFFD (or some other mechanism).

However, that doesn't mean that tools (like a text editor) shouldn't support them, or that XML shouldn't, or that Java shouldn't, or that conversion tables should treat them as an error. Quite the contrary. Such restrictions just makes their intended usage more difficult; it is like the (stupid) restriction in XML that control codes can't be interchanged, which makes XML files like Apple's keyboard data or CLDR's data either non-compliant or clumsy.

So we need to be clearer on this point.