The eutofont font format: additions a of 2004

William Overington

22 March 2004

The eutofont format is an experimental font format which permits a font to be encoded as a sequence of Private Use Area characters in the range U+EAC0 to U+EAFF within a Unicode plain text file.

This document adds some codes in the range U+EAB0 to U+EABF so as to provide some glyph substitution facilities. This document needs to be read as an addition to the eutofont 2003 format document eutofont.PDF as it does not repeat all of the information in that document.

However, one paragraph needs to be repeated here.

quote

Please know that at the time of writing the eutofont format has not been tested in use. It is a theoretical development. However, it has been designed with the experience of producing the eutocode graphics format, which has been tested in practice.

end quote

The codes introduced in this document are classified as follows.

U+EAB0 .. U+EABF Codes for preprocessing text which is to be displayed.

The intention is that the text in a document which is to be displayed may be preprocessed using rules specified within a eutofont format font using these additional codes and the text then displayed using the glyphs of the same eutofont format font.

The numbering of the glyphs used in a eutofont format font is not referenced in these glyph substitution rules: all references in these rules are to code points. The thinking is that typically a eutofont format font will map substitute glyphs to the Unicode Private Use Area. However, that is not essential as the eutofont format can be used to map a glyph outside of the Unicode range of U+0000 to U+10FFFF and thus make the glyph not accessible directly from a UTF16 character stream.

The reason for using code points rather than glyph numbers in the glyph substitution rules is that collections of rules are then potentially portable from font to font rather than having to be written specially for each font.

Here are the code point allocations.

U+EAB0, decimal 60080, SUBSTITUTE ALWAYS
U+EAB1, decimal 60081, SUBSTITUTE IF DISCRETIONARY SUBSTITUTIONS ARE SELECTED BY THE END USER
U+EAB2, decimal 60082, PLEASE NOTE THE FOLLOWING ALTERNATIVE GLYPH SUBSTITUTION POSSIBILITY OR POSSIBILITIES

U+EAB4, decimal 60084, START OF A SUBSTITUTION RULE CONDITION STRING

U+EAB8, decimal 60088, SUBSTITUTE IF THE CURRENT SUBSTITUTION RULE CONDITION STRING IS THE SAME AS THE CURRENTLY SELECTED LANGUAGE

U+EABC, decimal 60092, DECIMAL ENTER CODEPOINT VALUE
U+EABD, decimal 60093, HEXADECIMAL ENTER CODEPOINT VALUE
U+EABE, decimal 60094, SUBSTITUTION RULE SEPARATION MARKER
U+EABF, decimal 60095, END OF CURRENT SUBSTITUTION RULE

There are four types of glyph substitution rules. For each, a string of one or more character code points can be replaced by a sequence of zero or more character code points. The difference as between the four types of rules is as to whether the rule is unconditional, conditional upon a discretionary ligatures use choice having been made by the end user, conditional upon a specific language being used, conditional upon user choice of an alternative glyph for a particular character or sequence of characters.

Some examples follow, after a listing of the Unicode characters which are used within the examples.

U+0026 AMPERSAND
U+0065 LATIN SMALL LETTER E
U+0066 LATIN SMALL LETTER F
U+0069 LATIN SMALL LETTER I
U+006E LATIN SMALL LETTER N
U+FB01 LATIN SMALL LIGATURE FI

Always substitute f followed by i with an fi ligature.

U+EAB0 U+0066 U+0069 U+EABE U+FB01 U+EABF

Discretionary substitution of f followed by i with an fi ligature.

U+EAB1 U+0066 U+0069 U+EABE U+FB01 U+EABF

If the language is English then substitute f followed by i with an fi ligature.

U+EAB4 U+0065 U+006E U+EAB8 U+0066 U+0069 U+EABE U+FB01 U+EABF

Suppose that the font has two alternative forms of the ampersand glyph, located at U+E000 and U+E001. The following sequence shows that they are alternative glyph choices.

U+EAB2 U+0026 U+EABE U+E000 U+EABE U+E001 U+EABF

None, some or all of the characters in the above examples which are not in the range U+EAB0 to U+EABF may, if so desired, be represented by a code point stated as a sequence of one or more characters in the range U+EAE0 to U+EAE9 followed by U+EABC or by a code point stated as a sequence of one or more characters in the range U+EAF0 to U+EAFF followed by U+EABD.

For example, suppose that in the first example, the U+FB01 were expressed as a hexadecimal data sequence.

The original example is as follows.

U+EAB1 U+0066 U+0069 U+EABE U+FB01 U+EABF

The version with U+FB01 expressed as a hexadecimal data sequence is as follows.

U+EAB1 U+0066 U+0069 U+EABE U+EAFF U+EAFB U+EAF0 U+EAF1 U+EABD U+EABF

The sequence U+EAFF U+EAFB U+EAF0 U+EAF1 U+EABD expressing to the eutofont system the same as U+FB01. Please note that in the sequence U+EAFF U+EAFB U+EAF0 U+EAF1 that the rightmost characters of the four code points are F, B, 0 and 1.

Please note that the hexadecimal data sequence can be used to express a number outside of the Unicode range of U+0000 to U+10FFFF if so desired.

Here is a display of the authoring-time glyphs for these additions a of 2004 to the eutofont 2003 format which are included in the Quest text font from version 1.86 of the Quest text font. These symbols can be useful in preparing a eutofont format font within a plain text file.

U+EAB0 
U+EAB1 
U+EAB2 
U+EAB4 

U+EAB8 
U+EABC 
U+EABD 
U+EABE 

U+EABF