A Generalized Unified Character Code: Western European and CJK Sections

Status of This Memo

IESG Note

   This is not an IETF document. Readers should be aware of RFC 4690,
   "Review and Recommendations for Internationalized Domain Names
   (IDNs)", and its references.

   This document is not a candidate for any level of Internet Standard.
   The IETF disclaims any knowledge of the fitness of this document for
   any purpose, and in particular notes that it has not had IETF review
   for such things as security, congestion control, or inappropriate
   interaction with deployed protocols. The RFC Editor has chosen to
   publish this document at its discretion. Readers of this document
   should exercise caution in evaluating its value for implementation
   and deployment.

Abstract

   Many issues have been identified with the use of general-purpose
   character sets for internationalized domain names and similar
   purposes. This memo describes a fully unified coded character set
   for scripts based on Latin, Greek, Cyrillic, and Chinese (CJK)
   characters. It is not a complete specification of that character
   set.

Table of Contents

1. Introduction

   Many issues have been identified with the use of general-purpose
   character sets for internationalized domain names and similar
   purposes. This memo specifies a fully unified coded character set
   for scripts based on Latin, Greek, Cyrillic, and Chinese characters.

   There are four important principles in this work:

   1. If it looks alike, it is alike. The number of base characters
       and marks should be minimized. Glyphs are more important than
       character abstractions.

   2. If it is the same thing, it is the same thing. Two symbols that
       have the same semantic meaning in all contexts should be encoded
       in a way that allows their identity to be discovered by removing
       modifiers, rather than having to resort to external equivalence
       tables.

   3. For simplicity, when a character form can be evaluated on the
       basis of either serif or sanserif fonts, the sanserif font is
       always preferred.

   4. The use of combining characters and modifiers is preferred to
       adding more base characters.

   Based on these principles, it becomes obvious that:

   o Ligatures, digraphs, and final forms are constructed with special
      modifiers so that relationships to basic forms are obvious.

   o Symbols consisting of multiple marks are always constructed from
      combining characters and positional modifiers; thus, the "i"
      character is constructed from the vertical line symbol followed by
      a combining dot above. Similarly "f" is composed of a centered
      vertical line, a right hook in the top position, and an
      appropriately-positioned composing hyphen.

   This document draws strongly from the design and terminology of
   Unicode [Unicode] but represents a radically different approach.

1.1. Terminology

   All special-use terms in this document, including descriptions of
   behaviors and related relationships, are used with their common-sense
   meanings.

1.2. Discussion

   Questions to, and contributions for, this coding system should be
   addressed to the mailing list
   unified-ccs@xn--iwem3b1f.xn--90ase1a.bogus.domain.name.

2. Types of Characters

   This document defines several types of characters. Note that these
   definitions are not the same as the Unicode definitions for similar
   or identical terms.

2.1. Base Character

   Any character that is used as an atomic shape, rather than being
   assembled from such a character in combination with combining
   (overstriking) marks, symbols, or specially-designed base characters.
   When used alone, base characters always take up space. For example,
   a, c, l,...

2.2. Nonspacing Marks

   Marks, symbols, and character components that are used to form
   characters when used in combination with base characters. They do
   not occupy separate character positions when displayed.

   For example, the special combining symbols LeftUpperHook and
   RightLowerHook, described in Section 5, are nonspacing marks.

2.3. Case Indicators

   In scripts with case, only the lower-case characters are base
   characters. Upper-case forms are represented by using the UC
   modifier. So the traditional "A" character is represented by
   "a<UC>". Note that this means that case-independent comparisons are
   made simply by ignoring the <UC> modifiers rather than by complicated
   mapping operations.

   The initial set of case modifiers consists exclusively of:

   UC Upper-case, code value 1 (hexadecimal)

   The code values two through four are reserved for the impending
   encoding of scripts with more than two cases; five is reserved for
   expansion in case a script with more than four cases is identified.

2.4. Joining Indicators

   Zero-width joiners are used to build characters, not only to separate
   or join words. As compared to Unicode, a richer set of joiners is
   used to distinguish between the inter-word and ligature-forming
   (including half-character forming) cases. Unicode ZWJ and ZWNJ are
   supplemented by ZWCJ, OJ, and ONJ. ZWCJ is used to modify a spacing
   basic character into a nonspacing role. For example, there is no "w"
   character, but only "u<ZWCJ>u". Upper-case "W" is coded as
   u<ZWCJ>u<UC> -- the CWCJ binds more tightly than the UC modifier.

   The initial set of joining indicators consists exclusively of:

   ZWCJ Character joiner (also known as "ligature joiner"), code value
      6 (hexadecimal).

   OJ Overlay joiner (permits use of a subsequent character that would
      normally be spacing as nonspacing), code value 7 (hexadecimal).

   ONJ Overlay non-joiner (turns a nonspacing mark into a standalone
      character), code value 8 (hexadecimal). This joiner should not be
      necessary, and is normally prohibited by the "shortest string"
      rule. But there may be unanticipated cases.

   ZWJ Zero-width joiner for words or word-like constructions, code
      value 9 (hexadecimal).

   ZWNJ Zero-width non-joiner for words or word-like constructions,
      code value A (hexadecimal).

2.5. Character-Matrix Positioning Indicators

   Many characters are defined by constructed glyphs using nonspacing
   marks. For example, the characters "b" and "d" are coded as
   o<VerticalLine><PositionLeft> and o<VerticalLine><PositionRight>,
   respectively. The Catalan ligature that has caused some difficulties
   in Internationalizing Domain Names in Applications (IDNA) [RFC3490]
   is coded as l<ZWCJ><.><PositionVMiddle><ZWCJ>l

The initial table of positioning indicators is:

```
+-------------------+-----------+
| Name              | Hex value |
+-------------------+-----------+
| PositionLeft      |        20 |
| PositionCenter    |        21 |
| PositionRight     |        22 |
| PositionTop       |        30 |
| PositionVMiddle   |        31 |
| PositionBottom    |        32 |
| PositionDescender |        33 |
+-------------------+-----------+
```

2.6. Position Shaping Controls

   These controls designate character form changes for initial or final-
   form characters. Where the distinction is important, medial-form
   characters are the default when no qualification occurs. As with
   case comparisons, comparisons are performed by ignoring these control
   functions.

```
+-------------+-----------+
| Name        | Hex value |
+-------------+-----------+
| InitialForm |        71 |
| FinalForm   |        72 |
+-------------+-----------+
```

2.7. Repetition Indicators

   For compactness of coding, two repetition indicators are introduced
   for double (Repeat2) and triple (Repeat3) characters that may be
   treated as ligatures or special cases. Two consecutive uses of a
   character compare equal to the character followed by <Repeat2>. The
   interpretation of u<ZWCJ>u<Repeat3> is left as an exercise for the
   reader.

   The initial table of repetition indicators is:

```
+---------+-----------+
| Name    | Hex value |
+---------+-----------+
| Repeat2 |        50 |
| Repeat3 |        51 |
| Repeat1 |        52 |
+---------+-----------+
```

For larger repeats, these repeats can be combined; the sequence
<Repeat2><Repeat3> represents six repeats, while the
<Repeat3><Repeat2> represents five repeats. Following the "shortest
string" principle (see Section 4), Repeat1 must not ever appear
except in combination with Repeat2 and/or Repeat3. The generation of
other numbers is left as an exercise for the reader.

2.8. Control Characters

   Because it is intended primarily for domain names, this specification
   has no provision for control or spacing characters.

3. Code Assigment Groupings

   Following the reasoning used in Unicode [Unicode], every character
   occupies exactly 23 bits (conventionally stored as three octets, with
   the leading bit always zero). This value is chosen because both 3
   and 23 are prime numbers, unlike 42.

   The code point value zero is permanently reserved and will not be
   used unless it is necessary to expand the code space.

   Code values between 1 and 255 (decimal) are reserved for the special
   character formation codes described in Section 2.3 through
   Section 2.7.

   Code values between 256 and 511 (decimal) are reserved for character
   formation marks for non-ideographic characters. Most, but not all,
   of these are nonspacing (combining) characters.

   Code values between 512 and 1023 are reserved on general principles
   and in case it is necessary to invent new rules and make them
   retroactive.

   Code values of 1024 and above are to be allocated for characters,
   glyphs, and other character elements.

4. Canonical Form

   When glyphs are constructed using the mechanisms described here,
   there is a single canonical form for representing any given glyph.
   There are no exceptions to that form, and any sequence of characters
   and qualifiers that is not consistent with the form is invalid. If
   there are two possible ways to represent a given character, the
   shorter one (in octet count) is the only permitted form. If there
   are two possible ways that are of the same length, the only permitted
   form is the one that has the smaller value when the numeric values of
   all of the octets in each are summed.

The ordering rules are as follows:

1. A base character or composite character (see below) must come
    first.

2. The base character may be followed by ZWCJ or OJ, but not both,
    followed by a base or nonspacing character or mark.

3. If ZWCJ appears, the next character must be a base character or
    nonspacing mark.

4. If OJ appears, the next character must be a base character, since
    the function of OJ is to make a spacing base character into a
    nonspacing (overlay) character.

5. That character can be followed by positional qualifiers that
    apply to it. Vertical positional qualifiers precede horizontal
    positional qualifiers.

6. That sequence of characters may be followed by a case qualifier.

7. That entire sequence of characters forms a composite character.
    When the composite character is non-trivial, the rules may be
    applied to it recursively. If grouping is needed to distinguish
    between one composite character and the next, ZWNCJ may be used
    at the beginning of a composite character to identify a group
    boundary.

5. Examples of Graphic Element Codes

    The initial lists of positioning and combining controls appear above.
    This section shows codes for some base characters. Names in upper
    case are the Unicode names for the characters. These are followed,
    for information, by the Unicode code point designations. The code
    point list is informative, not normative, and may not be complete
    (especially since additional matching code points may be added to
    Unicode over time). Note that several Unicode characters that are
    considered different by Unicode are assigned the same code sequence
    in the system specified here.

Name	Hex value	Comment
FULL STOP (U+002E)	110	Used as both base character (in bottom center position) and as movable dot with OJ and positional qualifiers.
HYPHEN-MINUS (U+002D)	108	Used as a spacing base character (in horizontally and vertically centered position) and as a movable half-width horizontal line with OJ and positional qualifiers. In the context of this specification, should be known as Half Horizontal Line.
LOW LINE (U+005F)	109	Used as a spacing base character (in bottom position) and as a movable full-width horizontal line with OJ and positional qualifiers. In the context of this specification, should be known as Horizontal Line.
VERTICAL LINE (U+007C)	102	As with the horizontal lines, normally a spacing base character (in the middle position between left and right), but can be used as a right to left movable full-height vertical line with OJ and/or positional qualifiers.
HalfHeightVerticalLine	105	Similar to VERTICAL LINE, but only half height.
SOLIDUS (U+002F)	103	Used only for character formation; forward slash
REVERSE SOLIDUS (U+005C)	104	Used only for character formation; reverse slash
RightUpperHook	131	Used only for character formation; nonspacing mark.
LeftUpperHook	132	Used only for character formation; nonspacing mark.
LeftLowerHook	133	Used only for character formation; nonspacing mark.
RightLowerHook	134	Used only for character formation; nonspacing mark.
HalfHeightHoop	140	Used only for character formation; nonspacing mark.

```
| HalfHeightInvertedHoop |   141 | Used only for character       |
|                        |       | formation; nonspacing mark.   |
| DIGIT ZERO (U+0030)    |   400 |                               |
| DIGIT ONE  (U+0031)    |   401 |                               |
| DIGIT TWO  (U+0032)    |   402 |                               |
| DIGIT NINE (U+0039)    |   409 |                               |
| LATIN SMALL LETTER A   |   40A |                               |
| (U+0061)               |       |                               |
| LATIN SMALL LETTER O   |   418 | Unify with Greek Omicron      |
| (U+006F, U+03BF)       |       |                               |
| LATIN SMALL LETTER C   |   40C | Unifying C with Cyrillic ES   |
| (U+0063, U+0441)       |       |                               |
| GREEK SMALL LETTER     |   491 |                               |
| SIGMA (U+03C3)         |       |                               |
+------------------------+-------+-------------------------------+
```

6. Composite Characters and Unicode Equivalences

    This section provides examples of characters that are derived from or
    based on others, known as "composite characters".

```
+------------------+-------------+-------------------------------+
| Name             | Hex value   | Comment                       |
+------------------+-------------+-------------------------------+
| LATIN SMALL      | 418 007 102 |                               |
| LETTER B         |         020 |                               |
| (U+0062)         |             |                               |
| LATIN SMALL      | 418 007 102 |                               |
| LETTER D         |         022 |                               |
| (U+0064)         |             |                               |
| LATIN SMALL      | 40C 007 108 |                               |
| LETTER E         |         031 |                               |
| (U+0065)         |             |                               |
| LATIN SMALL      | 40A 006 40C |                               |
| LETTER AE        | 007 108 031 |                               |
| (U+00E6)         |             |                               |
| LATIN SMALL      | 102 131 030 | Note that 007 is not needed   |
| LETTER F         |     007 108 | before 131 because hooks are  |
| (U+0066)         |             | exclusively nonspacing        |
|                  |             | (combining).                  |
| LATIN SMALL      | 102 020 141 |                               |
| LETTER H         |     021 032 |                               |
| (U+0068)         |             |                               |
| LATIN SMALL      | 105 007 110 |                               |
| LETTER I         |     021 030 |                               |
| (U+0069)         |             |                               |
```

LATIN SMALL LETTER N (U+006E)	105 020 141 021 032	
LATIN SMALL LETTER P (U+0070, U+03C1, U+0440)	418 007 102 033 020 033	Unified P, Greek Rho, Cyrillic ER
LATIN CAPITAL LETTER A (U+0041)	40A 001	
LATIN CAPITAL LETTER B (U+0042)	418 007 102 020 001	
LATIN CAPITAL LETTER C (U+0043)	40C 001	
LATIN CAPITAL LETTER D (U+0044)	418 007 102 022 001	
GREEK SMALL LETTER FINAL SIGMA (U+03C2)	491 072	

7. Ideographic Characters

    Because of the traditional model of forming characters using selected
    radicals and strokes in combination, Han-derived ("CJK") characters
    are even more naturally represented, with less ambiguity, in the
    system specified here than European ones. The mechanisms used in
    this specification and represented in the tables (see Section 8) are
    similar to those described as "Radicals" and "Strokes" in Section 5.1
    and in Section 5.2 ("Ideographic Description Characters") of The
    Unicode Standard [Unicode]. Of course, following the same principles
    outlined above for European characters, only radicals, stroke, and
    description controls would be treated as base characters; no distinct
    compound precomposed ideographic characters are registered.

8. IANA Considerations

    IANA is requested to keep the actual registry of characters and code
    tables. The registry entries consist of a character name (preferably
    matching the Unicode character name when one is available), the code
    sequence used to represent the character and optional descriptive
    information. The characters and codes identified in Section 2,
    Section 5, and Section 6 above should be used to initialize the
    table. Since the coding system is user-extensible, registrations
    should be accepted for new characters as long as they don't look like

old ones. A designated expert with a background in calligraphy or
abstract art, and considerable experience in evaluating claims about
the count of angels on heads of pins, should be selected to advise
IANA on "looks like".

9. Security Considerations

The representation of characters in this format should be a
significant boon for security. It eliminates many possibilities of
phishing attacks, since Principle 1 prevents the existence of two
characters that look alike but are different.

By detaching the encoding of characters for domain names from the
encoding of characters for other purposes, it also guarantees that
reasonable-looking names will have been encoded by competent
entities, thereby providing a significant degree of safety by
obscurity.

Because of the method by which upper-case forms are encoded and
because similarity is sometimes in the mind of the beholder, this
specification will not completely eliminate opportunities for visual
confusion. For example, because the lower-case characters are quite
different, LATIN CAPITAL LETTER A and GREEK CAPITAL LETTER ALPHA will
never compare equal, even though they look alike.

10. Acknowledgments

The authors would like to acknowledge the many contributions of
J.F.C. Morphin for pointing out the inadequacies of trying to address
the challenges of internationalization within the context of existing
engineering principles. His comments and related ones, in
combination with issues encountered in trying to internationalize
domain names based on Unicode, have contributed greatly to the frame
of mind underlying large parts of the proposal documented here. The
theoretical framework for this coding system is based, in part, on
Unicode and its collection of names and sample glyphs but represents
a very different approach to the coding system itself.

11. References

11.1. Normative References

   [Unicode] The Unicode Consortium, "The Unicode Standard, Version
              5.0", 2007.
              Boston, MA, USA: Addison-Wesley. ISBN 0-321-48091-0

11.2. Informative References

   [RFC3490] Faltstrom, P., Hoffman, P., and A. Costello,
              "Internationalizing Domain Names in Applications (IDNA)",
              RFC 3490, March 2003.

Authors' Addresses

   John C Klensin
   1770 Massachusetts Ave, #322
   Cambridge, MA 02140
   USA

   Phone: +1 617 491 5735
   EMail: john+ietf@jck.com

   Harald Tveit Alvestrand
   Google
   Beddingen 10
   Trondheim,   7014
   Norway

   EMail: harald@alvestrand.no