Technical Reports

{Proposed Draft} Unicode Technical Report #XX

CHARACTER FOLDINGS

L2/01-447

Version	0d3
Authors	Asmus Freytag (asmus@unicode.org)
Date	2000-11-06
This Version	(this document)
Previous Version	none
Latest Version	http://www.unicode.org/unicode/reports/trXX
Tracking Number	0

NOTE TO UTC:

In response to action item 84-04 I have updated the rationale section and updated the table in section 5.0.

Summary

This report identifies a set of character foldings, in other words, operations that ignore certain distinctions between similar characters.

Status of this document

<When approved, add formal status language for proposed draft>

Contents

- 1 Scope
- 2 Rationale
- 3 Definitions
- 4 Descriptions
- 5 Specifications
  - 5.1 Specification of foldings
  - 5.2 Folding algorithm [TBD]
- 6 Notes
- References
- Modifications

1.0 Scope

This technical reports defines a consistent set of folding operations useful for fuzzy searches, among other things. Each of the folding operations specified in this report has well understood properties, and is appropriate in specific

contexts. For example some identifiers needs case folding, some do not. Some text searches needs to preserve <super> form (trademark symbol), while others do not. Not all of these folding operations may be appropriate in the same contexts. See the description for some of the more problematic folding or expansion operations.

This report defines a single search term folding specification (STF) which combines canonical normalization with optional folding operations.

e.g. STF = NFx+<case folding>+<font>+<super>+<sup>

This allows implementers to decide which folding option is useful for a particular use.

2.0 Rationale

For the purpose of fuzzy text matching, including both defined identifier matching and general text searching it is often necessary to selectively ignore otherwise meaningful distinctions between related characters, for example case, wide vs. narrow, etc. This process can be called *search term folding*. Depending on the operation, different foldings need to be applied, and possible interactions must be carefully managed.

2.1 Relation to Normalization

Search term folding is somewhat similar to normalization in that it also reduces several alternative representations into one, but there are important differences. Normalization is often intended for permanent transformation of data, but search term folding is by nature transient. Where the normalization forms offer two different levels of distinctions they preserve (canonical and compatibility), the choice of what type of distinction can or even must be ignored for search term folding needs to be more specific and depends on the nature of the operation. One size does not fit all. Some compatibility mappings have the unfortunate side effect of affecting search term stability, preventing their use in general text searches.

Furthermore, normalization and case folding are defined as separate and independent operations, but case folding often occurs together with other foldings in search term folding. In order to avoid inconsistencies, search term folding needs to address their interaction.

Search term folding is best expressed as applied to text after canonical normalization, however, whether the normalized text is in the composed or decomposed form is of secondary importance in terms of defining the foldings. It will be relevant to distinguish between NFC and NFD whenever pre-folded search terms are to be interchanged. Otherwise, due to the transient nature of search term folding, the distinction is immaterial.

3.0 Definitions

Folding operation – a folding operation removes a distinction between related characters. For example, case folding removes the case distinction, by replacing upper and title case variants of a character with the lower case.

Compatibility mappings – compatibility mappings substitute characters with their compatibility decomposition. Many compatibility mappings are foldings, some are multigraph expansions.

Multigraph expansion – a multigraph expansion replaces a multigraph, such as *e.g.* double prime, by its expansion into an equivalent series of single characters, in this case, two single primes. Multigraph expansions are a subset of compatibility mappings.

4.0 Folding and Expansion Operations

4.1 General notes

4.1.1 Accent folding

this needs to go beyond the decomposition and removal of accents, umlauts, cedillas but should also include barred, slashed forms etc, as well as hooks, descenders, etc.

4.1.2 Letter forms

Letterforms need to have 'final sigma' and 'final Hebrew' added to them.

Greek letter forms should be folded for Greek text. They should not be folded for mathematical and scientific usage as doing so would conflate very distinct concepts (e.g. angle (THETA) and temperature (THETA SYMBOL) to give an examples of common usage in physics).

4.1.3 Multigraphs

Some multigraphs that are EAW ambiguous should potentially be treated differently when resolved to EAW wide than when resolved to narrow.

4.1.4 Semantically neutral foldings

Text that is subjected to a standard Unicode rendering process should display the same whether or not the semantically neutral foldings have been applied (in fact, in practice it is far more likely that un-folded data will display poorly on such a system).

Therefore these foldings are candidates for permanent data transformations.

The following foldings are semantically neutral

- Positional forms folding
- Vertical forms folding

4.2 Problematic foldings or expansions

4.2.1 Fraction expansion

Fraction expansion as defined in the compatibility decompositions can lead to a drastic change of the semantics of a string and can lead to term boundary issues for searching. For example: Expanding the fraction in this string: DIGIT 5 + VULGAR FRACTION ONE QUARTER turns it into DIGIT 5 + DIGIT 1 + FRACTION SLASH + DIGIT 4. This now will be found by a search for "51". Because of the semantics of FRACTION SLASH the expansion changed the numeric value from "5 and a quarter" into "51 over 4". Fraction expansion is therefore best avoided altogether.

4.2.2 Bullet expansions

If a circled bullet character is simply replaced by its contents, *e.g.* CIRCLED DIGIT 5 is replaced by DIGIT 5, the separation from the surrounding text is lost, and the DIGIT 5 could run together with adjacent numbers. For bullet characters using parenthesized or dotted letters or digit, this issue is somewhat mitigated by fact that the bullet itself contains punctuation. Bullet characters are commonly used like footnote marks to refer to other text, in other words, they do not just occur at the beginning of bulleted lines.

4.2.3 Spacing accents substitution

Spacing accents are mapped by compatibility decomposition to SPACE + non-spacing accent. This inappropriately introduces a space character into the term, as well as introducing non-spacing marks where none were in the data before.

4.2.4 Math folding

Form NFKC provides an aggressive folding of letter like mathematical symbols to their nearest ASCII or Hebrew equivalent. In particular the Hebrew characters used as letterlike symbols do not have RIGHT TO LEFT directionality and the set of such letters in mathematical usage is sufficiently restricted that such folding makes little sense, except in pure 'looks like' style searches.

4.2.5 Various "cluster" expansions

Unicode contains many clusters, e.g. square symbols, some of the letterlike characters that are made up of several characters. 'Decomposing' these may or may not be the right thing for search equivalence. Parenthesized characters and numbers would probably be immune to the term boundaries issues raised earlier, but the story is less clear for others.

5.0 Specifications

5.1 Basic algorithm

The basic algorithm for search term folding can be stated as

a. Apply canonical decomposition
b. Apply optional folding operations
c. Recurse 1 & 2 until stable (*)
d. Apply composition if necessary

(*) Step (c) might be removed by by enforcing certain foldings before decomposition.

5.2 Specification of folding operations

The following table summarizes the definition of the folding operations. Foldings that are *multigraph expansions* have been collected at the end of the table.

- The *description* column identifies the folding.
- The *source* column identifies the set of characters subject to the folding operation by referencing a set of code points, a set of general categories, or a compatibility mapping tag. All characters matching the source condition are subject to the given folding. Note that this column does *not* indicate the set of characters with which the source characters are equivalenced by the folding.
- The *target* column indicates the result of the folding, either by reference to an operation, or, in some cases, by providing the single Unicode character to which a whole set of source characters is folded.
- The *data file* column indicates which data file carries the character by character information to implement the operation referred to in the *target* column.

Description	Source characters	Target Characters	Data file specifying the mapping

Accent removal	Latin/Greek/Cyrillic characters with canonical decomposition	base characters of canonical decomposition	[UnicodeData]
Accent folding (includes stroke, hook, descender)	Latin/Greek/Cyrillic characters with accents	related base character	AccentFolding.txt [TBD]
Case folding	Lu and Lt	case fold according to CaseFolding.txt	[CaseFolding]
Canonical duplicates folding (e.g. Ohm - Omega)	0374, 037E, 0387, 1FBE, 1FEF, 1FFD, 2000, 2001, 2126, 212A, 212B, 2329..232A	canonical decomposition	[UnicodeData]
Dashes folding	Pd	U+002D	
Greek letterforms folding	03D0..03D2, 03D5..03D6, 03F0..03F2, 03F4..03F5	compatibility decomposition	[UnicodeData]
Han Radical folding	2F00..2F5D, 2EF3, 2E9F	compatibility decomposition	[UnicodeData]
Hangzhou Numbers folding	3038..303A	compatibility decomposition	[UnicodeData]
Hebrew Alternates folding	FB20..FB28	compatibility decomposition	[UnicodeData]
Jamo folding	3131..3183	compatibility decomposition	[UnicodeData]
Kana folding	Hiragana	Katakana	[KanaFolding]
Ligature expansion Misc.	0587, 0675..0678, 0E33, 0EB3, 0EDC..0EDD, 0F77, 0F79, FB00..FB06, FB13..FB17, FB4F	compatibility decomposition	[UnicodeData]
Letterforms folding	Variants of letter forms 017F (long s)	related base form 0073	LetterFolding.txt [TBD]
Math symbol folding	<font>	compatibility decomposition	[UnicodeData]
Native digit folding	Nd	substitute ASCII digit of same numeric property	[UnicodeData]
Non-break folding	<no-break>	compatibility decomposition	[UnicodeData]
Overline folding	FE49..FE4B	203E (*)	
Positional Forms folding - includes Arabic ligatures	<initial>, <medial>, <final>, <isolate>	compatibility decomposition	[UnicodeData]
Small forms folding	<small>	compatibility decomposition	[UnicodeData]
Space folding	Zs	U+0020	
Spacing Accents <+>	00AF,00B4,00B8,02D8..02DD, 037A,0384,1FBD,1FBE..1FC0, 1FFE,2017,203E,309B..309C	compatibility decomposition	[UnicodeData]
Subscript folding	<sub>	compatibility decomposition	[UnicodeData]
Superscript folding	<super>	compatibility decomposition	[UnicodeData]
Symbol folding <+>	00B5, 2107,2135..2138	compatibility decomposition	[UnicodeData]
Underline folding	2017, FE4D..FE4F	005E	
Vertical forms folding	<vertical>	compatibility decomposition	[UnicodeData]
Width folding	<wide>, <narrow>	compatibility decomposition	[UnicodeData]
Multigraph expansions			
- Fraction expansion	<fraction>	compatibility decomposition	[UnicodeData]
- Circled	<circled>	compatibility decomposition	[UnicodeData]
- Parenthesized	2474..2487,249C..24B5, 3200..3243	compatibility decomposition	[UnicodeData]

- Dotted	2488..249B	compatibility decomposition	[UnicodeData]
- Ellipsis expansion	2024..2026	compatibility decomposition	[UnicodeData]
- Integral expansion	222D..222C,222F..2230	compatibility decomposition	[UnicodeData]
- Prime expansion	2033..2034,2036..2037	compatibility decomposition	[UnicodeData]
- Roman numerals	2160..2183	compatibility decomposition	[UnicodeData]
- Squared	<square>	compatibility decomposition	[UnicodeData]
- Squared (unmarked)	3358..3370, 33E0..33FE, 32C0..32CB	compatibility decomposition	[UnicodeData]
- Digraphs	0132..0133, 013F..0140. 0149, 01C4..01CC, 01F1..01F3, 1E9A	compatibility decomposition	[UnicodeData]
-Other multigraph, e.g. c/o, TEL	203C, 2047..2049, 20A8, 2100..2101, 2103, 2105..2106, 2109, 2116, 2121	compatibility decomposition	[UnicodeData]

Notation:

- .. indicates an inclusive range
- , indicates an alternative
- <xxxx> refers to a compatibility mapping tag as defined in [CompatibilityTags]
- Xx refers to a particular value for the General Category property defined in [UnicodeData]
- <+> means a folding is contained in the Unicode data files, but is not recommended
- (*) indicates a target that is subject to another folding, other than case folding

6.0 Notes on existing sources for folding and expansion data

6.1 Case folding

The [CaseFolding] data file provides pure case folding information.

6.2 Collation

The weight tables from the Unicode Collation Algorithm provide the source information for a number of foldings. These are not provided explicitly, but fall out when some of the sort weight differences are selectively ignored.

- Accent folding
- Case folding
- Final forms folding
- Kana folding
- Width folding

The Collation data tables provides intrinsic multigraph expansions which cannot be separated from the foldings.

[Note: unidata.txt in the collation dir has annotation]

6.3 Compatibility decompositions

Compatibility decomposition provides several foldings and expansions. They are in fact the source of most of the

foldings in the table in section 5.0. There are two ways to subdivide compatibility decompositions

1. by compatibility tag (the value in <> in the date file, e.g. <super>)
2. by explicitly limiting the range of *source* characters

The specifications in section 5.0 use the first method, whenever the compatibility tag is well defined and meaningful. Where it is too broad (e.g. <compat>) foldings are further subdivided by defining specific ranges of source characters.

7.0 Suggested Future Additions

Additional foldings that have been suggested

- Slash folding (eg. 2044 FRACTION SLASH to 002F SOLIDUS)
- Additional look-alike foldings

Locale based foldings:

Locale based foldings could match

oe - o umlaut o slash - o umlaut ss to sharp s y to u and v to w (Swedish)

etc. In other words, they would be different for some user groups using the same script (in this case Latin). This folding could be generated from sorting tables.

References

Delete unneeded ones

[AccentFolding]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/AccentFolding.txt>
[CaseFolding]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/CaseFolding.txt>
[Case Mapping]
Mark Davis, Unicode Technical Report #21: Case Mapping, <http://www.unicode.org/unicode/reports/tr21>
[Character Mapping Tables]
Mark Davis, Unicode Technical Report #22: Character Mapping Tables, <http://www.unicode.org/unicode/reports/tr22>
[Collation]
Mark Davis, Unicode Technical Report #10: Collation, <http://www.unicode.org/unicode/reports/tr10>
[CompatibilityTags]
[EastAsianWidth]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/EastAsianWidth.txt>
[East Asian Width]
Asmus Freytag, *Unicode Standard Annex #11, East Asian Width*, <http://www.unicode.org/unicode/reports/tr11>
[KanaFolding]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/KanaFolding.txt>
[SpecialCasing]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/SpecialCasing.txt>
[Unicode]

The Unicode Standard, Version 3.0, Addison Wesley Longman, 2000.
[UnicodeCharacterDatabase]
Readme file, <ftp://ftp.unicode.org/Public/UNIDATA/UnicodeCharacterDatabase.html>
[UnicodeData]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/UnicodeData.txt>
[UnicodeData-Format]
Readme file <ftp://ftp.unicode.org/Public/UNIDATA/UnicodeData.html>

These may not be needed.
[Bidirectional Algorithm]
Mark Davis, Unicode Standard Annex #9: The Bidirectional Algorithm,
<http://www.unicode.org/unicode/reports/tr9>
[LineBreak]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/LineBreak.txt>
[Line Breaking]
Asmus Freytag, Unicode Standard Annex #14: Line Breaking Properties,
<http://www.unicode.org/unicode/reports/tr14>
[NamesList]
Data file <ftp://ftp.unicode.org/Public/UNIDATA/NamesList.txt>
[NamesList-Format]
Readme file <ftp://ftp.unicode.org/Public/UNIDATA/NamesList.html>

Changes from previous drafts

Improved the Rationale section
