L2/25-046

Technical Reports

Proposed Draft Unicode® Technical Report #59

EAST ASIAN SPACING

Editors	Koji Ishii 石井宏治 (Google Inc.)
Date	2024-12-16
This Version	https://www.unicode.org/reports/tr59/tr59-1.html
Previous Version	n/a
Latest Version	https://www.unicode.org/reports/tr59/
Revision	1

Summary

East Asian established typography defines that a small amount of visible space between East Asian scripts and other scripts improves readability. This report describes the algorithm and the data which can be used to automatically add visible space.

Status

*This is a **draft** document which may be updated, replaced, or superseded by other documents at any time. Publication does not imply endorsement by the Unicode Consortium. This is not a stable document; it is inappropriate to cite this document as other than a work in progress.*

> ***A Unicode Technical Report (UTR)** contains informative material. Conformance to the Unicode Standard does not imply conformance to any UTR. Other specifications, however, are free to make normative references to a UTR.*

Please submit corrigenda and other comments with the online reporting form [Feedback]. Related information that is useful in understanding this annex is found in Unicode Standard Annex #41, "Common References for Unicode Standard Annexes." For the latest version of the Unicode Standard, see [Unicode]. For a list of current Unicode Technical Reports, see [Reports]. For more information about versions of the Unicode Standard, see [Versions].

Contents

1 Overview and Scope

East Asian text usually consists of multiple scripts, such as Han ideographs, Kana syllables, and Hangul syllables, along with Latin letters and numeric characters. East Asian established typography conventions define that a thin spacing between East Asian scripts and other scripts improves readability. This spacing should be represented by adjusting glyph spacings similar to kerning, rather than by using space characters.

While detailed rules of the spacing can vary across documents, it is important that the choice made by an author for a specific document be clearly established, so that a rendering system can display what the author intended. It is also important that this choice be established independently of the font resources, as the rendering systems may have to use other fonts than those intended or specified in the document. Finally, the expression of the author's choice should be relatively concise, to facilitate document authoring and minimize document size.

This report describes an algorithm, together with a character property, which can serve as a default for inserting spacing for the purpose of reliable document interchange.

The character property is independent of external information, such as provided in the tables of the selected font, so that implementations could reliably render a character stream based solely on the property values.

The intent is that an author should be able to specify where they want to override, and that in the absence of an explicit specification, the spacing is applied according to this specification.

The actual choice for the property values should result in a reasonable or legible default, but it may not be publishing-material quality, and it may not be a good choice if used in a specific style or context.

The property values are chosen first to match existing practice in East Asian contexts in their respective environments. For characters that are not generally used in such environments, similarity to existing characters has been taken into consideration. It also takes East Asian characters in non-East Asian texts into account.

2 Conformance

The property defined in this report is informative. The intent of this report is to provide, in the absence of other information, a reasonable way to determine the correct automatic spacing, but this behavior can be overridden by inserting space characters as described in Space Characters, or by a higher-level protocol, such as through markup or by the preferences of a layout application. This default determination is defined in the accompanying data file Data, but in no way implies that the spacing is inserted only by this rule.

For more information on the conformance implications, see [Unicode], Section 3.5, Properties, in particular the definition (D35) of an informative property.

3 The East_Asian_Spacing Property

The name of this specification and of this property reflect well-known East Asian typographic tradition. UTC members are not currently aware of similar traditions for other writing systems. It would be possible to widen the set of scripts considered by this algorithm in the future, and rename this specification accordingly, but once established, the name of the property would have to remain stable (at least as an alias). If there is a reasonable expectation that this may happen, then the property could be renamed without the "East Asian" reference.

There is currently no proposed short property name.

3.1 Definitions

UTR59-D1. *Auto spacing:* The addition of a small amount of visible space in rendered text as determined by an algorithm like the one defined in this document. Depending on the implementation, a visible space may be added as a glyph space in a text layout process, or by adding an appropriate Unicode space character.

UTR59-D2. *East Asian scripts:* Scripts in the following list:

- Bopomofo (Bopo)
- Han (Hani)
- Hangul (Hang)
- Hiragana (Hira)
- Katakana (Kana)
- Khitan_Small_Script (Kits)
- Nushu (Nshu)
- Tangut (Tang)
- Yi (Yiii)

UTR59-D3. *Language context:* The language of the input string, which may be detected, or determined by higher-level protocols, the preferences of the application or the operating system, or other appropriate methods.

3.2 Property Values

The possible property values are given in Table 1.

Table 1. Property Values

Value	Short Name	Description	Examples
Wide	W	Characters that are considered as East Asian script.	Han ideographic characters and Kana syllables.
Narrow	N	Characters that need auto-spacing with adjacent “W” characters.	Latin letters and digits.
Other	O	Characters that don’t need auto-spacing.	Most symbols and punctuation characters such as COMMA and FULL STOP.
Conditional	C	Characters that are “N” for Chinese languages, and “O” for other languages or when the language is unknown.	Characters that can appear as prefix or suffix of Latin letters or digits, such as U+0025 PERCENT SIGN.

Characters that have the property value “N” are similar to “East_Asian_Width=Narrow” characters in [UAX11], but most punctuation characters and symbols are excluded. Similarly, characters that have the property value “W” are similar to “East_Asian_Width=Wide” characters, but most punctuation characters and symbols are excluded. Also, to follow existing practice, circled characters, square characters, and Emoji are defined as “O”.

For the value “C”, please refer to Symbols and Punctuation Characters for more details.

3.3 Spacing Algorithm

The East Asian Spacing should be inserted between “W” and “N”, and between “N” and “W”, after resolving the conditional value “C” to “N” or “O”.

The exact amount of the spacing can vary across documents. This property doesn’t define the exact amount. Instead, it should be defined by higher-level protocols or applications such as through markup

or by the preferences of a layout application.

There are two ways to represent a space: As a Unicode space character, or as a glyph space (similar to kerning, adjusting the metrics of adjacent glyphs on the device). Higher-level protocols or applications should use glyph spaces where possible.

3.3.1 Algorithm Steps

Inputs:

1. An input string.
2. Whether the language context is a Chinese language or not.
    1. If the language is unknown, it is not a Chinese language.
    2. It is a Chinese language if the language subtag is the macrolanguage "zh" (Chinese, widely used for tagging Mandarin) or one of the language subtags that are encompassed by "zh"; for example "cmn" (Mandarin), "yue" (Cantonese), "hak" (Hakka). (See “Macrolanguage: zh” in the IANA language-subtag-registry.)
3. If the string is to be displayed as vertical text, then this algorithm also needs information for each grapheme cluster for whether it is to be displayed upright or rotated; this can be determined by performing the algorithm in [UAX50], or by a higher-level protocol.

Process the input string as follows.

1. Iterate over each grapheme cluster in the input string according to [UAX29].
2. In each grapheme cluster, fetch the first code point.
3. Look up the East_Asian_Spacing property value for this code point.
4. If the string is displayed in vertical text layout, then determine if the character is displayed upright: The character is displayed upright if [UAX50] resolves the character's Vertical_Orientation property value to U, Tu, or Tr; or if a higher-level protocol overrides it to be displayed upright.
5. If the string is displayed in vertical text layout, and if the character is displayed upright, and if the property value is Narrow (N), then change the property value to Other (O).
6. If the property value is Conditional (C): If the language context is a Chinese language, then change the property value to Narrow (N); otherwise change the property value to Other (O).
7. Apply the property value to the entire grapheme cluster.
8. If a grapheme cluster G1 is followed by a grapheme cluster G2, and the property value of G1 is Wide (W) and the property value of G2 is Narrow (N), or the value of G1 is Narrow (N) and the value of G2 is Wide (W), then insert a visible space between G1 and G2.

3.4 Scope of the Property

3.4.1 Grapheme Cluster

As in all matters of typography, the interesting unit of text is not a character, but a grapheme cluster: it does not make sense to insert the auto spacing between a base character and a combining mark attached to it. Implementations should insert the auto-spacing before or after each grapheme cluster.

A possible choice for the notion of grapheme cluster is either that of legacy grapheme cluster or that of extended grapheme cluster, as defined in [UAX29].

Note that UAX #29 by default specifies extended grapheme clusters. We should probably settle on those, without option.

The property value for a grapheme cluster as a whole is then determined by taking the property value of the first character in the cluster, with the following exception:

- If the cluster contains an enclosing combining mark (General_Category=Me), then the whole cluster has the East_Asian_Spacing property value “O”.

3.4.2 Space Characters

The property values for space characters (General_Category=Zs) are “O”. This is to avoid inserting the auto-spacing around space characters, which can lead to undesirable double spacing.

It also allows authors to override the algorithm when higher-level protocols or applications don’t provide a way for authors to express their intent to override this algorithm, such as plain text files.

U+0020 SPACE indicates a semantic boundary, which is stronger than the spacing for readability. Using the code point for the East Asian Spacing purpose can make distinguishing semantic boundaries from the spacing for readability difficult, and therefore it’s discouraged.

U+2009 THIN SPACE should usually represent a thin space, which is suitable to represent the auto-spacing for readability. Inserting the code point to where the algorithm doesn’t insert the auto-spacing should indicate that the auto-spacing is desired there.

Likewise, inserting U+200B ZERO WIDTH SPACE to where the algorithm inserts the auto-spacing should prevent the auto-spacing from being inserted by rendering systems.

3.4.3 Symbols and Punctuation Characters

In some existing practices, symbols and punctuation characters insert the spacing, while they don’t in other existing practices.

For example, when one side of a word is a letter or a digit and the other side is a punctuation character, such as “20%”, “$20”, or “C#”, and they appear adjacent to “W” characters, some existing practices prefer inserting the spacing to both sides of them, considering that not doing so look unbalanced.

On the other hand, some other existing practices prefer not inserting the spacing between punctuation characters and “W” characters even in such cases. They view the spacing as a way to secure legibility by preventing East Asian letters from being too close to other letters and numeral digits, rather than to highlight words as parentheses do.

The conditional value “C” is assigned to such characters. Chinese languages content often prefer the spacing in such cases, and thus they should resolve “C” to “N”, while other content should resolve “C” to “O”.

If the author is uncertain whether their content is used in a Chinese language context or not, and they want to express their intentions, they can override the algorithm as described in Space Characters.

3.4.4 Vertical Text Layout

In vertical text layout, a character may be displayed upright or sideways rotated. A character is displayed upright if the resolved Vertical_Orientation property value by performing the algorithm in [UAX50] is U, Tu, or Tr, or if higher-level protocols override to display it upright.

If a character that has the East_Asian_Spacing property value “N” is displayed upright, the rendering system should handle it as if it has the property value “O” instead.

3.4.5 Right-to-Left Scripts

This specification has a current limitation in that the handling of right-to-left scripts is not specified. This includes scripts that are predominantly written right to left, such as Arabic, along with right-to-left scripts that are meant to be written vertically, such as Chorasmian.

4 Data File

This property is derived by the following algorithm:

1. Assign the property value “W” if it’s in the following set:

    1. Include if the Script property is one of the East Asian scripts.
    2. Include if the Script_Extensions property is one of the East Asian scripts, except when the East_Asian_Width property is “Neutral (N)” or “Narrow (Na)”.
    3. Exclude if the East_Asian_Width property is “East Asian Halfwidth (H)”.
    4. Exclude if the General_Category property is “Punctuation (P)” or “Other_Number (No)”.
    5. Exclude if the General_Category property is “Symbol (S)” except “Modifier_Symbol (Sk)”.
    6. Include the following code point: U+3013 GETA MARK.
2. Otherwise, assign the property value “C” if it’s in the following set:
    1. Include if the General_Category property is “Other_Punctuation (Po)”.
    2. Exclude if the East_Asian_Width property is “East Asian Fullwidth (F)”, “East Asian Halfwidth (H)”, or “East Asian Wide (W)”.
    3. Exclude the following code points: U+0022 QUOTATION MARK, U+0027 APOSTROPHE, U+002A ASTERISK, U+002F SOLIDUS, U+00B7 MIDDLE DOT, U+2020 DAGGER, U+2021 DOUBLE DAGGER, U+2026 HORIZONTAL ELLIPSIS.
3. Otherwise, assign the property value “N” if it’s in the following set:
    1. Include if the General_Category property is “Letter (L)”, “Mark (M)”, or “Decimal_Number (Nd)”.
    2. Exclude if the East_Asian_Width property is “East Asian Fullwidth (F)", “East Asian Halfwidth (H)”, or “East Asian Wide (W)”.
4. Otherwise, assign the property value “O”.

The derived data file is available for reference.

Acknowledgments

The initial version is based on a proposal authored by Koji Ishii, Yasuo Kida, and Fuqiao Xue. The proposal is originated from a discussion at W3C primarily driven by Elika Etemad and Florian Rivoal. Markus Scherer assisted in the documentation. Peter Constable, Asmus Freytag, Ken Lunde, Nat McCully, Taro Yamamoto, and many members of the UTC have participated in the review of the data and the documentation.

References

For references for this report, see Unicode Standard Annex #41, "Common References for Unicode Standard Annexes."

Modifications

The following summarizes modifications from previous revisions of this report.

Revision 1

- First working draft.
