L2/25-183

UTC #184 properties feedback & recommendations

Markus Scherer & Josh Hadley / Unicode properties & algorithms group, 2025-jul-16

Participants

The following people have contributed to this document:

Markus Scherer (chair), Josh Hadley (vice chair), Asmus Freytag, Elango Cheran, John Wilcock, Ken Whistler, Mark Davis, Ned Holbrook, Peter Constable, Robin Leroy, Roozbeh Pournader

1. Core Spec

1.1 Error in Permuting Combining Class Weights example [#413]

Recommended UTC actions

1. **Action Item** for Roozbeh Pournader, PAG: Propose informative changes for core spec 5.13.1 Canonical Equivalence Table 5-4. Permuting Combining Class Weights and associated text, taking into account feedback ID20250509132809. For Unicode 18.0. See L2/25-183 item 1.1.

Feedback (verbatim)

Date/Time: Fri May 9 13:28:09 PDT 2025
ReportID: ID20250509132809
Name: Ned Holbrook
Report Type: Report Error in Publication Data
Opt Subject: Error in Permuting Combining Class Weights example

https://www.unicode.org/versions/Unicode16.0.0/core-spec/chapter-5/#G36537 attempts to give an example of "permut[ing] weights to have U+0651 ARABIC SHADDA precede all vowel signs" that appears to be incorrect: assuming that the internal weights behave the same as combining classes such that lower values appear first, I believe a correct example would remap shadda (CCC 33) to a weight of 27 (rather than 32, as shown). If so, https://unicode.org/faq/normalization.html#9 also needs to be corrected.

Background information / discussion

Table 5-4. Permuting Combining Class Weights

Combining Class	Internal Weight
27	→ 33
28	→ 27
29	→ 28
30	→ 29
31	→ 30
32	→ 31

33	→ 32

UnicodeData.txt (Unicode 16.0)

```
None
064B;ARABIC FATHATAN;Mn;27;NSM;;;;;N;;;;;

064C;ARABIC DAMMATAN;Mn;28;NSM;;;;;N;;;;;

064D;ARABIC KASRATAN;Mn;29;NSM;;;;;N;;;;;

064E;ARABIC FATHA;Mn;30;NSM;;;;;N;ARABIC FATHAH;;;;

064F;ARABIC DAMMA;Mn;31;NSM;;;;;N;ARABIC DAMMAH;;;;

0650;ARABIC KASRA;Mn;32;NSM;;;;;N;ARABIC KASRAH;;;;

0651;ARABIC SHADDA;Mn;33;NSM;;;;;N;ARABIC SHADDAH;;;;

0652;ARABIC SUKUN;Mn;34;NSM;;;;;N;;;;;
```

2. UCD

2.1 U+00B8 ¸ CEDILLA is used as a letter [#400]

Recommended UTC actions

1. **Consensus**: Update the derivation of the Word_Break property to assign Word_Break=ALetter to U+00B8 ¸ CEDILLA. For Unicode Version 17.0. See L2/25-183 item 2.1.
2. **Action Item** for Robin Leroy, PAG: In UCD file WordBreakProperty.txt, assign Word_Break=ALetter to U+00B8 ¸ CEDILLA. For Unicode Version 17.0. See L2/25-183 item 2.1.
3. **Action Item** for Josh Hadley, PAG: In Unicode Standard Annex #29, Unicode Text Segmentation, update the derivation to assign Word_Break=ALetter to U+00B8 ¸ CEDILLA. For Unicode Version 17.0. See L2/25-183 item 2.1.

PAG input

From Kirk Miller:

Saanich orthography uses U+00B8 CEDILLA as a letter in all orthographic positions. This character does not however behave as a letter.

From Roozbeh Pournader, Apr 14, 2025,

I think it's possible to change the Word_Break property for U+00B8, especially since it probably won't be disrupting any existing use case. We should probably change it to ALetter.

From Robin Leroy, Apr 14, 2025,

I think if U+00B8 is being used as a letter, we should do to it what we did to U+02C7 in Unicode 4.0, that is change its General_Category from Sk to Lm (see decision 92-C27 and section 1D of L2/02-267R3).
The segmentation properties will follow, as well as others (for instance, you would be able to use it in programming language identifiers).
We would want to make it Identifier_Type=Uncommon_Use to keep it out of Identifier_Status=Allowed.

Background information / discussion

On SENĆOŦEN

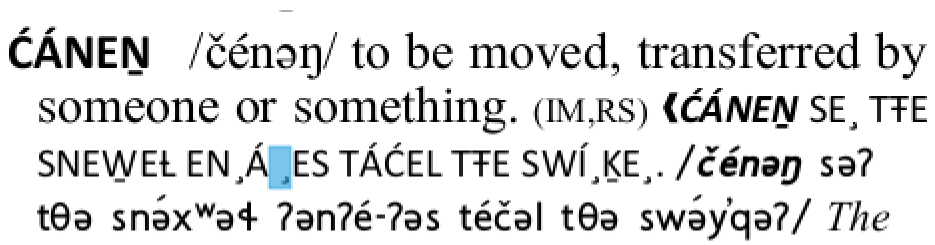
As a simple Saanich letter, U+00B8 stands for glottal stop. It also forms digraphs for glottalized resonants, namely ⟨L¸⟩, ⟨M¸⟩, ⟨N¸⟩, ⟨N̲¸⟩, ⟨U¸⟩, ⟨Y¸⟩. These digraphs are not considered to be single letters, and a combining diacritic is never used.

Originally Saanich orthography had used the ASCII comma, or omitted the mark altogether, and these approaches are still seen in informal contexts. However, the comma created problems because it was not recognized as text by search engines etc. It has been replaced with the cedilla in grammars and dictionaries, both print and online, as well as in the spell-checker, predictive text, automatic text/audio matching, and text-to-speech engines created by the Canadian National Research Council in collaboration with the W̱SÁNEĆ community (Timothy Montler, pc).

Examples of words from the SENĆOŦEN Classified Word List incorporating a cedilla in medial and final position:

SENĆOŦEN	Phonetic transcription	English translation
AXEN̲¸ÁL¸N̲EN	ʔex̣əŋ̓él̓ŋən	want to do it: 1739.1.
Á¸Á¸LȻEN̲	ʔeʔél̓kʷəŋ	leave so. all alone: 1437.
ÁĆENÁ	ʔéčəne	my goodness!: 1072.2.
Á¸ĆEX	ʔéʔčəx̣	gray crab: 236.
Á¸ȻEL¸	ʔéʔkʷəl̓	weave a basket: 1745.1.
Á¸EŁ; ŁEŁIȻ; N̲EXÁSET	ʔéʔəɬ; ɬəɬíkʷ; ŋəx̣ésət	hurry: 1403.
Á¸I¸	ʔéʔiʔ	be on one's way, going on: 1481.2.
Á¸IT; ȺYET; S¸Á¸IT	ʔéʔit; ʔéyət; sʔéʔit	lingcod: 268.
ÁJET	ʔéč̓ət	wipe: 1755.
Á¸LEN̲	ʔéʔləŋ	house: 699.
Á¸LEN̲¸ENEȻ	ʔéʔləŋənəkʷ	village: 696.
ÁLEW̲EL¸	ʔéləxʷəl̓	waste time: 1733.1.

Although word-initial glottal stop is not written, as seen above, the character can appear after a space: suffixes may be separated from the stem by spaces, and they may start with a glottal stop. In such cases word-medial glottal stops may appear in initial orthographic position, as here -


ĆÁNEṈ /čénəŋ/ to be moved, transferred by someone or something. (IM,RS) ❮***ĆÁNEṈ*** SE¸ TŦE SNEW̱EȽ EN¸Á ¸ES TÁĆEL TŦE SWÍ¸ḴE¸. /***čénəŋ*** səʔ tθə snə́xʷəɬ ʔənʔé-ʔəs téčəl tθə swə́y̓qəʔ/ *The*

Here the word /ʔənʔé-ʔəs/ is written EN¸Á [space]¸ES, with only the first glottal stop omitted.

[Note that the apparent mismatch in the second entry, Á¸Á¸LȻEṈ /ʔeʔél̓kʷəŋ/, is not a typo. A glottalized resonant may become preglottalized after a stressed vowel, and this is reflected in the orthography, thus Á¸L here for /él̓/]

On the use of the orthography

The orthography is used for Saanich which is listed on Glottlog as part of the Northern Straits languages, which are identified as "extinct," but with a further annotation "awakening" for some. Ethnologue lists it as a "dialect of Northern Straits" and gives the number of speakers as "10K", with the status as "endangered" and the digital support level as "emerging". Wikipedia gives details on the efforts to revive the Saanich dialect and to teach the associated orthography. The use of character in question is a result of a recent change to the orthography, which therefore cannot be considered "stable".

On the principles for adapting properties to changes in use

The issue here is a new orthography finding a new use for an existing character. This raises the question of what principles govern the range of possible changes.

- **properties defining the identity of a character** Sometimes, characters see massive and widespread changes in use which changes users perception of the primary purpose(s) of the character. In that case, we can say that the identity of the character has shifted. An example might be the # character which many more people now associate with a "hashtag" than with an alternate way of writing lbs. The center of gravity of users' understanding of what this character is has shifted. Sometimes a character is found to be misclassified. That tends to happen more for more recently encoded characters. Some properties, such a General_Category cannot capture all uses of a character and thus may conflict with some actual use.
- **properties relevant to identifiers** Changes to identifier-relevant properties should be based on how widespread and also on how stable any new use of a character happens to be. For a character in uncommon use, it is possible to also be classified as Technical or Obsolete, if such would be their status from the perspective of more commonly used orthographies.
- **properties relevant to default algorithms** Changes affecting default algorithms can be most easily accommodated if there's no contention between orthographies or any adverse effects on other orthographies, particularly those which are widespread. If there are conflicts, instead of updating the default algorithm, a better solution would be for implementations to provide a suitable tailoring.
- **properties referenced by external standards** Changes to such properties must be evaluated carefully to consider whether they negatively affect the dependent standard. This may make it

preferable to add an explicit exception, for example in specifying a Unicode algorithm, rather than modifying any of the properties the algorithm is based on.

On possible changes to properties

This section evaluates several options for accommodating the use of U+00B8 in SENĆOŦEN.

A relevant precedent is the change of General_Category for U+02C7 CARON from Modifier_Symbol to Modifier_Letter in Unicode 4.0, decided by UTC-92-C27, based on Section 1D of L2/02-267R3. That document motivated the change as follows: “These character [sic, referring to a list including U+02C7] are intermixed with letters as a part of words, and should be allowed in both of them; there is also no reason to exclude them from identifiers. Moving them into Lm would reflect that fact, and produce better default behavior for all processes dealing with words and identifiers.” The document also stated about a list of characters including U+00B8 that “The above are all special-case spacing symbols that are not used in the interior of words or identifiers in practice, any more that other symbols are. They should be left as Symbols to reflect this.” SENĆOŦEN has other ideas.

However, a change to General_Category in 2025 for a character that is part of Latin-1 has implications that are wider-ranging than a change to a Latin-2 character in 2002. In particular, it was pointed out that this would cause U+00B8 ¸ CEDILLA to become IDNA_2008 PVALID; such a change, while not without precedent, see, *e.g.*, https://www.rfc-editor.org/rfc/rfc9233.html#name-u111c9-sharada-sandhi-mark and UTC-152-C5, is generally viewed with suspicion in the IETF world.

The derived effects of a change to General_Category=Lm would be to set the Alphabetic property, the ID_Start, ID_Continue, XID_Start, and XID_Continue properties, and to change the segmentation properties Word_Break to ALetter and Sentence_Break to OLetter. In such an eventuality, it would be appropriate to set Identifier_Type=Uncommon_Use for this character, leaving it at Identifier_Status=Restricted.

Based on this analysis four possibilities were considered.

1. Change the General_Category to Lm, and set Identifier_Type=Uncommon_Use.
2. Do not change the General_Category, but effect all the changes that would derive from that change, by means of Other_Alphabetic and Other_ID_Start, and set Identifier_Type=Uncommon_Use.
3. Change the derivation of the Word_Break property to special-case this character. **[This is the recommendation.]**
4. Do nothing.

Concerns about downstream standards motivated avoiding **option 1**.

Precedent was found for **option 3**: the derivation of the Word_Break property already special-cases a host of Modifier_Symbols: the set `\p{Sk}&\p{Word_Break=ALetter}` is currently equal to [˂-˅ ˒-˗ ˞˟ ˥-˫ ˭ ˯-˿ ꜈-꜖ ꞉ ꞊ ꭛ ꭪ ꭫] (57 characters). Of those, 35 [˂-˅ ˒-˗ ˞˟ ˭ ˯-˿ ꞉ ꞊ ꭛ ꭪ ꭫] were made Word_Break=ALetter in Unicode 10 based on document L2/16-336 with action UTC-149-A51 (itself originating from feedback L2/16-205 [Thu Jun 2 08:46:28 CDT 2016] with action UTC-148-A41a). The motivation there was usage in IPA and UPA. The remaining 22, the tone letters [˥-˫ ꜈-꜖], were made Word_Break=ALetter in Unicode 13 (see PRI-355 [Mon Feb 12 20:20:51 CST 2018], UTC-155-A4, L2/19-188, UTC-159-C28, PRI-396 [Tue Jul 30 12:55:18 CDT 2019], UTC-161-A43, and UTC-162-A4).

It was noted that the (X)ID_Start and (X)ID_Continue properties are explicitly unconcerned with spoofing issues, and already allow for hosts of invisible characters and lookalikes of syntax and punctuation. As such there was no sustained objection to **option 2**. However, no clear need was evident for identifiers in SENĆOŦEN, an orthography used by a small language community; in addition, the use of the cedilla (replacing the comma) in this orthography is a somewhat recent development. Should the use of the cedilla prove stable and should a need for SENĆOŦEN identifiers arise, *e.g.*, if there were evidence for a desire to use SENĆOŦEN in programming language identifiers or usernames based on default identifiers, this could be revisited. The situation is somewhat different from the 57 varieties of Modifier_Symbols already Word_Break=ALetter, as those were assigned Word_Break=ALetter based on usage in phonetic notation rather than in an orthography. As pointed out in L2/16-336, if it were made Alphabetic, this would make it the only Alphabetic Modifier_Symbol.

Option 4 was dispreferred, since some of the other options would improve the behavior of default algorithms for SENĆOŦEN without harming default behavior in other orthographies, and are easy to implement.

In discussion, Xws7ámeshqen, an orthography of a closely related language which uses the ASCII digit 7 for the glottal stop, was mentioned. While the digit 7 poses no word breaking issues, this illustrates the limits of what orthographic choices can be accommodated by changes in properties: even if 7 were used word-initially, it cannot be made XID_Start without breaking parsing for nearly all programming languages: numbers whose leading digit is 7 would become identifiers. The same issue does not occur for the cedilla (it is not reserved for syntax, and is not commonly specially handled in parsing in a way that would conflict with making it an identifier character). Another Salish orthography that uses 7 is Sḵwx̱wú7mesh; one orthography for Aln8bak uses 8 as a vowel (thanks to Joshua Tsai for pointing these out).

2.2 Add Not_Applicable as an alias for InPC=NA [#401]

Recommended UTC actions

1. **Consensus**: For property value Indic_Positional_Category=NA, add a long alias Not_Applicable. For Unicode 18.0. See L2/25-183 item 2.2.
2. **Action Item** for Robin Leroy, PAG: For property value Indic_Positional_Category=NA, add a long alias Not_Applicable in both PropertyValueAliases.txt and IndicPositionalCategory.txt. For Unicode 18.0. See L2/25-183 item 2.2.

PAG input

From Roozbeh Pournader, PAG:

Indic_Positional_Category has a default value of NA which is relatively cryptic. It means Not_Applicable and could benefit from such an alias.

Note: The value NA (capitalized exactly that way) is also used for Hangul_Syllable_Type, which already has an alias of Not_Applicable, and for Age, which is aliased to Unassigned.

Background information / discussion

This came up during plenary discussion of the wording of UTC-183-C15.

The IndicPositionalCategory.txt header comments already document:

```
None
#  All code points not explicitly listed for Indic_Positional_Category

#  have the value NA (not applicable).
```

2.3 U+2800 Braille pattern blank is categorized as a symbol [#420]

Recommended UTC actions

1. **Consensus**: Change the Line_Break assignment of U+2800 BRAILLE PATTERN BLANK from Alphabetic (AL) to Break_After (BA). For [target, e.g. Unicode 17.0]. See L2/25-183 item 2.3.
2. **Action Item** for Robin Leroy, PAG: In UCD file LineBreak.txt, change the Line_Break assignment of U+2800 BRAILLE PATTERN BLANK from Alphabetic (AL) to Break_After (BA). For [target, e.g. Unicode 17.0]. See L2/25-183 item 2.3.

Feedback (verbatim)

Date/Time: Thur May 15 16:11:17 PDT 2025
ReportID: ID20250515161117
Name: President International Council on English Braille Judith Dixon
Report Type: Report Error in Publication Data
Opt Subject: U+2800 Braille pattern blank is categorized as Os

U+2800 is categorized as Os[sic], Other symbol so does not behave like a space. ICEB, the standard-setting body for English language braille publishes codebooks. U+2800 does not break words. We must use U+0020 which is not sized correctly. It would be better if U+2800, Braille Pattern Blank, which is always used as a space was categorized as Zs.

Background information / discussion

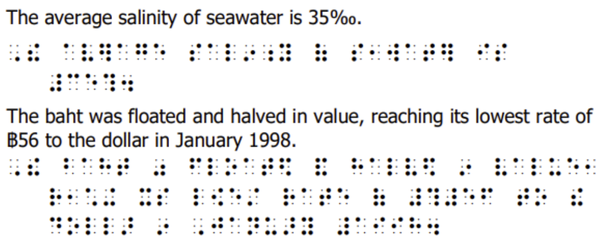
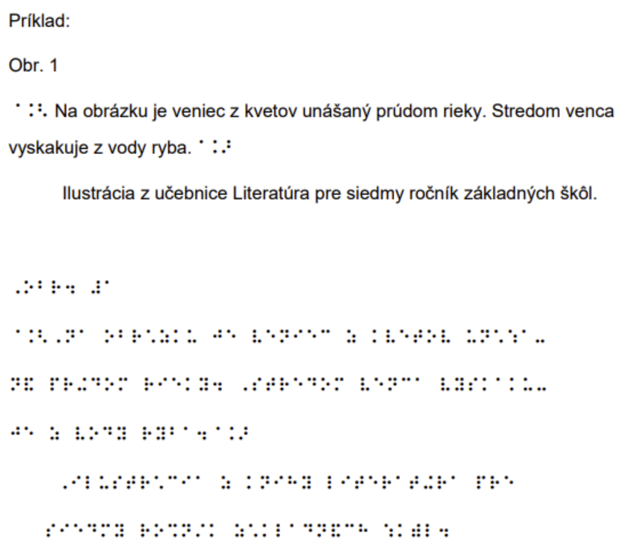
Typo: The General_Category is So=Other_Symbol.

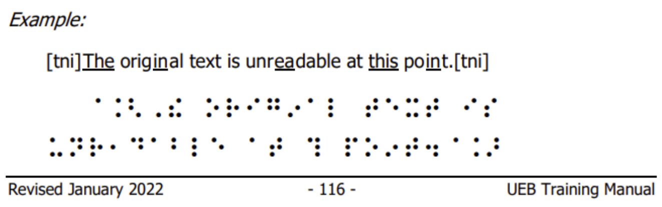
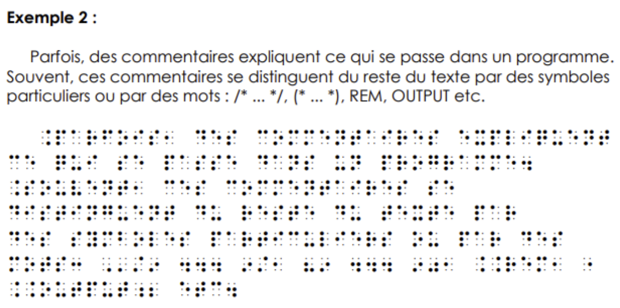
The practical problem described by the submitter, as stated, that « U+2800 does not break words », would not be resolved by changing the General_Category alone: U+202F NARROW NO-BREAK SPACE is a space and does not break words, nor does it create a line break opportunity.

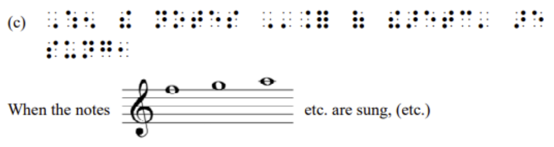
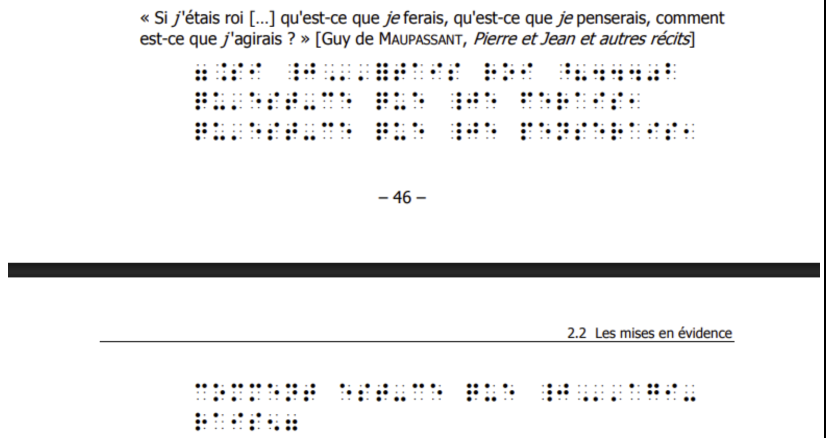
Strictly speaking the report is not about *word* segmentation of Braille in the sense of UAX #29; since all braille characters are Word_Break=Other, the default word breaking rules find a word break at every position, the opposite of the problem reported.
Instead the use of the phrase « break[ing] words » by the submitter likely refers to « word wrapping », also known in the Unicode Standard as line breaking. **We propose that the line breaking behaviour of U+2800 be corrected**, thereby addressing the practical issues faced by publishers as reported by the submitter.

> **Note:** Correct *word* breaking of Braille (as opposed to *line* breaking) is out of scope for the Unicode Standard; such segmentation would be highly language-specific and system-specific. Even within a single system, since the same dot patterns are used for punctuation, digits, or letters or sequences of letters (e.g., in UEB, ⠛ is 7 and g, ⠶ is gg and part of various punctuation marks), this would require highly context-dependent rules or a dictionary-based approach. It is possible that the default word breaking behaviour could be improved, but this is a separate issue from the clearly suboptimal line breaking of Braille.

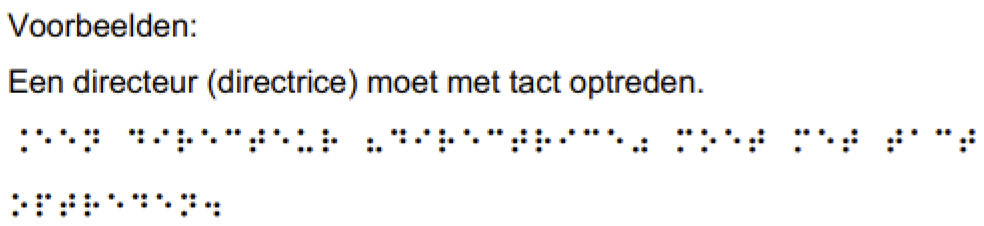
The issues faced by publishers are evident from the large number of publications with longer-form Braille examples that are laid out on multiple lines, as seen below; if represented using Unicode Braille, such examples currently need to be manually linebroken, or to use U+0020 SPACE.

[Hor24 p. 40] (uses U+0020)	**[SABP24, p. 99] (uses U+2800, manual linebreaking)**
 The average salinity of seawater is 35‰. [illegible] The baht was floated and halved in value, reaching its lowest rate of ฿56 to the dollar in January 1998. [illegible]	 Príklad: Obr. 1 [illegible] Na obrázku je veniec z kvetov unášaný prúdom rieky. Stredom venca vyskakuje z vody ryba. [illegible] Ilustrácia z učebnice Literatúra pre siedmy ročník základných škôl. [illegible]

[HRH22, p. 116 (uses U+2800, manual linebreaking)]	**[CFGG14, p. 8] (uses U+0020)**
 *Example:* [tni]The original text is unreadable at this point.[tni] [illegible] Revised January 2022 - 116 - UEB Training Manual	 **Exemple 2 :** Parfois, des commentaires expliquent ce qui se passe dans un programme. Souvent, ces commentaires se distinguent du reste du texte par des symboles particuliers ou par des mots : /* ... */, (* ... *), REM, OUTPUT etc. [illegible]

[BANA97, p. 4] (uses U+0020)	[CBFU08, pp. 46 sq.] (non-Unicode Braille)
 (c) When the notes etc. are sung, (etc.)	 « Si *j*'étais roi [...] qu'est-ce que *je* ferais, qu'est-ce que *je* penserais, comment est-ce que *j*'agirais ? » [Guy de MAUPASSANT, *Pierre et Jean et autres récits*] – 46 – 2.2 Les mises en évidence

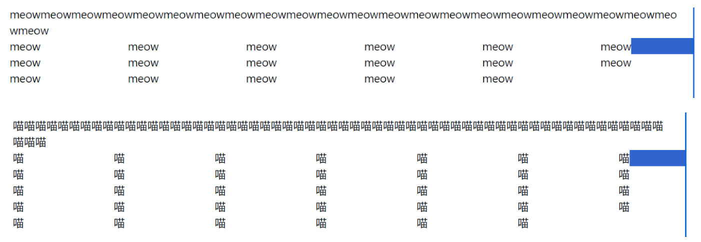
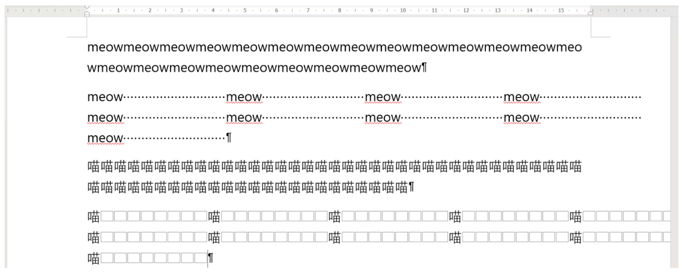
[BA20, p. 18] (uses U+2800, manual linebreaking)



Voorbeelden:

Een directeur (directrice) moet met tact optreden.

The PAG discussed at length the possibility and implications of categorizing U+2800 as a Space_Separator, and giving it the White_Space property. One notable property of spaces, as opposed to other Break_After characters, is that they disappear into the margins when the line is broken after them, as shown in the following figures, both with U+0020 and U+3000.

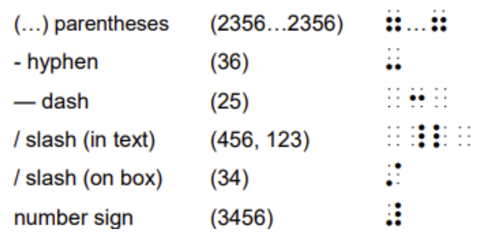
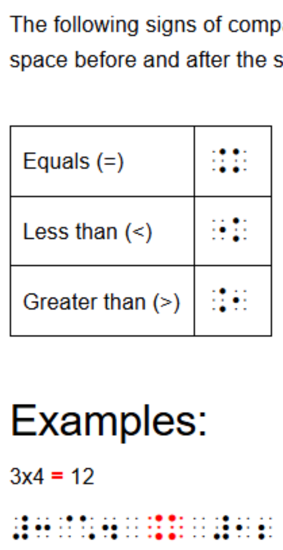
Chrome text field	Word
	

```
meowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeo
wmeow
meow        meow        meow        meow        meow        meow
meow        meow        meow        meow        meow        meow
meow        meow        meow        meow        meow

喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵
喵喵喵
喵        喵        喵        喵        喵        喵        喵
喵        喵        喵        喵        喵        喵        喵
喵        喵        喵        喵        喵        喵        喵
喵        喵        喵        喵        喵        喵        喵
喵        喵        喵        喵        喵        喵
```

```
meowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeowmeo
wmeowmeowmeowmeowmeowmeowmeowmeowmeow¶
meow··························meow··························meow··························meow··························
meow··························meow··························meow··························meow··························
meow··························¶
喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵
喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵喵¶
喵□□□□□□□□喵□□□□□□□□喵□□□□□□□□喵□□□□□□□□喵□□□□□□□
喵□□□□□□□□喵□□□□□□□□喵□□□□□□□□喵□□□□□□□□喵□□□□□□□
喵□□□□□□□□¶
```

It was found that many publications that use Unicode Braille characters use U+2800 with a marking glyph. A marking space is not without precedent, consider U+1680 ‒ OGHAM SPACE MARK. However, in some cases, U+2800 is clearly used as a symbol, to indicate the presence of a space when describing a Braille sequence corresponding to a symbol or punctuation, similar to the use of ␣ (U+2423 OPEN BOX) to indicate the presence of a space. Especially in cases where it is used in a trailing position, as in the [UNE13] example, it is clear that it would be inappropriate for this symbol to disappear.

[UNE13, p. 84]	[BB19, Nemeth Unit 2]
	

(…) parentheses	(2356…2356)	⠶…⠶
- hyphen	(36)	⠤
— dash	(25)	⠀⠒⠀
/ slash (in text)	(456, 123)	⠀⠸⠇⠀
/ slash (on box)	(34)	⠌
number sign	(3456)	⠼

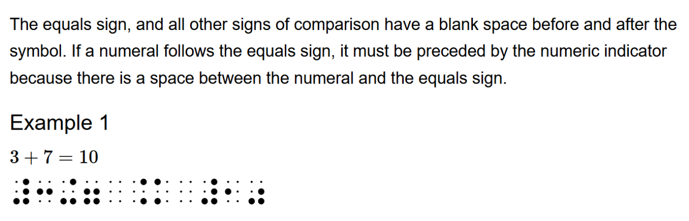
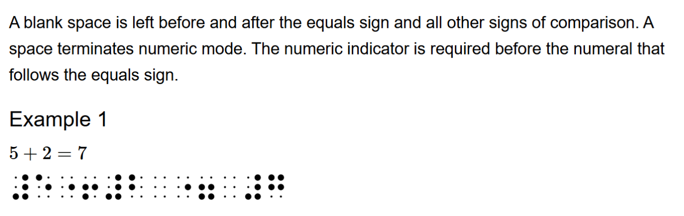
The following signs of comparison are 2-cell symbols. All signs of comparison require a space before and after the symbol.

Equals (=)	⠀⠨⠅⠀
Less than (<)	⠀⠐⠅⠀
Greater than (>)	⠀⠨⠂⠀

Examples:

3x4 = 12

⠼⠒⠈⠡⠲⠀⠨⠅⠀⠼⠂⠆

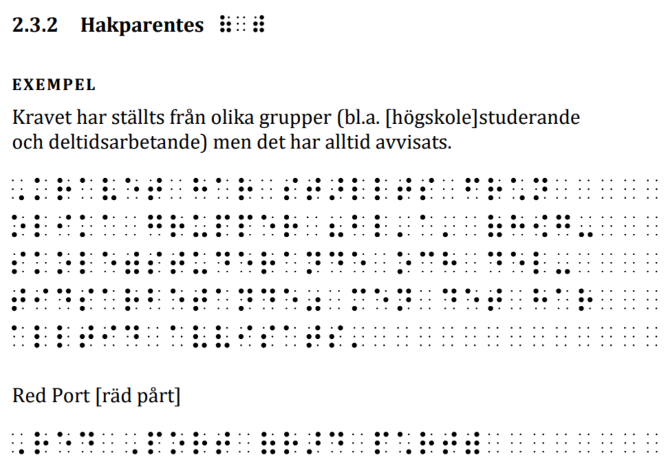
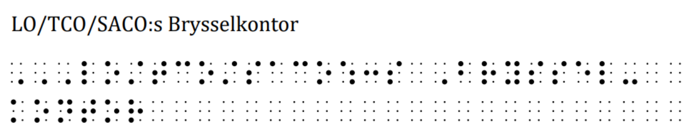
[APH16, Lesson 1.5]	[APH18, Lesson 1.4]
 The equals sign, and all other signs of comparison have a blank space before and after the symbol. If a numeral follows the equals sign, it must be preceded by the numeric indicator because there is a space between the numeral and the equals sign. Example 1 $3 + 7 = 10$	 A blank space is left before and after the equals sign and all other signs of comparison. A space terminates numeric mode. The numeric indicator is required before the numeral that follows the equals sign. Example 1 $5 + 2 = 7$

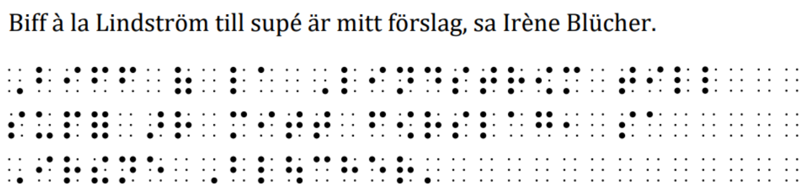
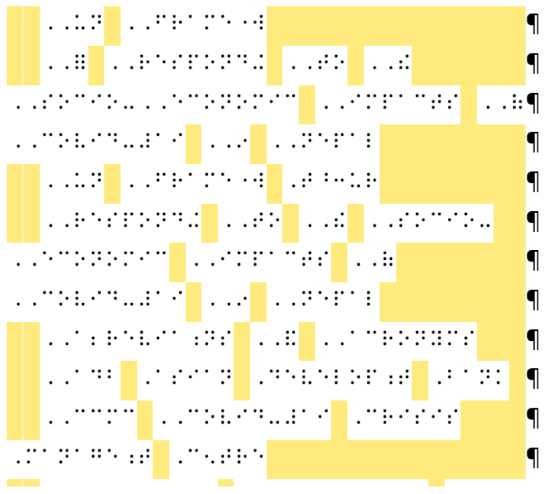
Considering this usage as a marking symbol, together with concerns about changing the relatively stable—and often hardcoded—White_Space property (a property of only 25 code points, last changed in Unicode 6.3, when U+180E MONGOLIAN VOWEL SEPARATOR ceased to be a space), the PAG preferred a more targeted change.

This naturally led to two questions:

1. In fonts where the U+2800 is marking, is it acceptable to retain the trailing U+2800 before a break, as would happen with a non-White_Space lb=BA U+2800?
2. Are there cases where a line break after U+2800 would be infelicitous, for instance, are there symbols whose Braille transcription in some system includes a medial space?

On marking U+2800 in line-wrapped text and trailing marking U+2800, examples were found of publications that included them.

[Pun09 p. 23]	[Pun09 p. 31]
 2.3.2 Hakparentes EXEMPEL Kravet har ställts från olika grupper (bl.a. [högskole]studerande och deltidsarbetande) men det har alltid avvisats. Red Port [räd pårt]	 LO/TCO/SACO:s Brysselkontor

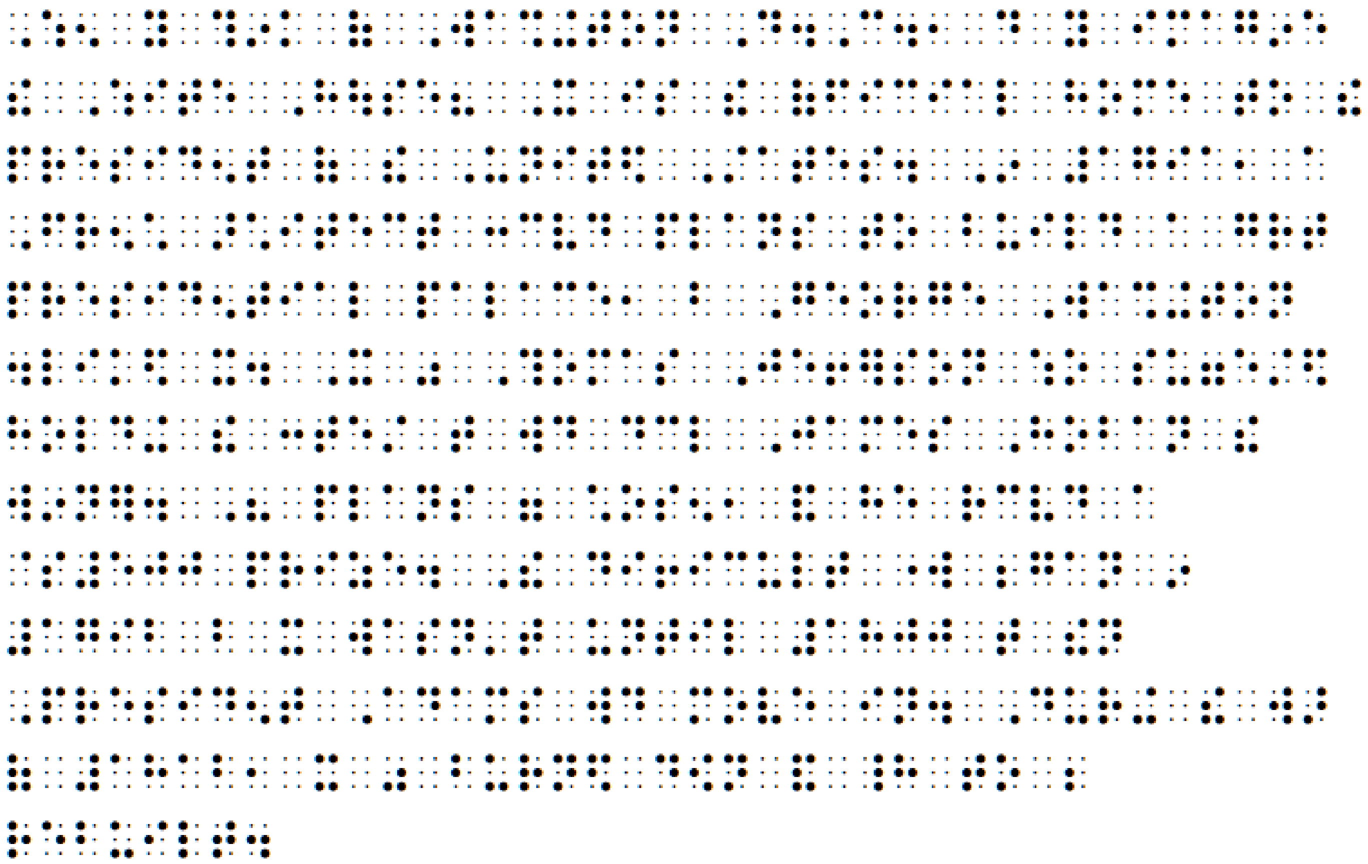
[Pun09 p. 39]	[UN21]
 Biff à la Lindström till supé är mitt förslag, sa Irène Blücher.	

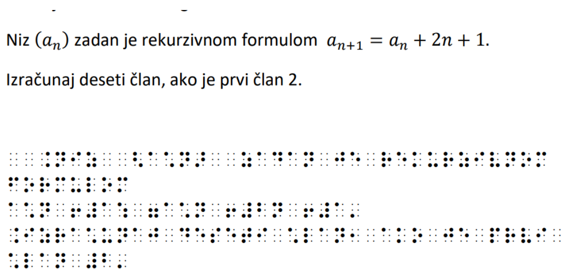
Text in the [UN21] Word document indicates that it was produced using Sensus Braille; each line is a paragraph, and the lines are padded to 32 characters with U+2800 (highlighted by text search in the figure). Note, in the [Pun09 p. 23] example, contrast of U+2800 with U+0020 in the heading.

While these examples illustrate that trailing U+2800s are not intrinsically a problem, they all consist of text padded with U+2800, which property assignments will not achieve. It was noted that such manual line breaking and padding would work the same after a change to Line_Break=BA, so the publishers of these documents would be unaffected.

However, publications were found where marking U+2800 was used in multiline examples that were not padded to a fixed width, including cases with trailing U+2800; showing that the effect of Line_Break=BA with a marking U+2800 would be acceptable to some publishers.

[BB19, Unit 15 Reading Workout]



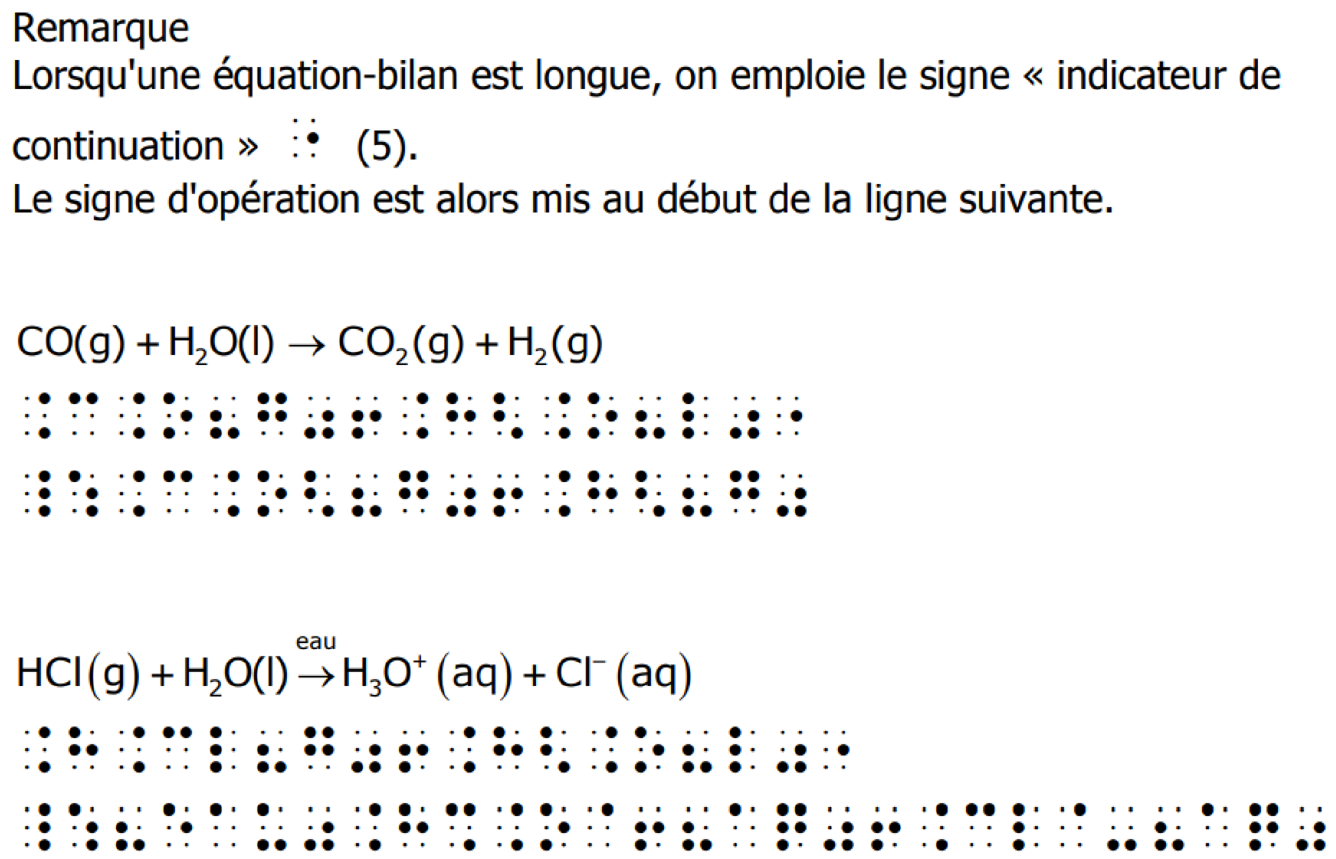
[Pug21, p. 10]	**Underlying text**
 Niz (a_n) zadan je rekurzivnom formulom $a_{n+1} = a_n + 2n + 1$. Izračunaj deseti član, ako je prvi član 2.	Niz (*a*$_n$) zadan je rekurzivnom formulom *a*$_{n+1}$ = *a*$_n$ + 2*n* + 1. Izračunaj deseti član, ako je prvi član 2. ⠨⠝⠊⠵ ⠣⠁⠡⠝⠜ ⠵⠁⠙⠁⠝ ⠚⠑ ⠗⠑⠅⠥⠗⠵⠊⠧⠝⠕⠍ ⠋⠕⠗⠍⠥⠇⠕⠍ ⠁⠡⠝ ⠖⠼⠁⠱ ⠶⠁⠡⠝ ⠖⠼⠃⠝ ⠖⠼⠁⠄ ⠨⠊⠵⠗⠁⠡⠥⠝⠁⠚ ⠙⠑⠎⠑⠞⠊ ⠡⠇⠁⠝⠂ ⠁⠅⠕ ⠚⠑ ⠏⠗⠧⠊ ⠡⠇⠁⠝ ⠼⠃⠄

The trailing U+2800 in the [Pug21] example corresponds to the space between *prvi* and *član*.

On the question of potential usage of U+2800 BRAILLE BLANK in contexts where it should be non-breaking, a survey of the references which cover orthographic and technical Braille systems for many languages (the documents cited above, as well as [Ald16] and [CEBF08]), revealed no such usage. Some symbols are distinguished by spacing, e.g., in Nemeth, ⟨⠨⠅⟩ κ vs. ⟨⠀⠨⠅⠀⟩ =, see [Sar21], but that spacing is never internal to the symbol.

Some technical notations use special conventions for line breaking; for instance, the chemical notation described in [CEBF08] (which does not include any spaces) uses U+2810 ⠐ BRAILLE PATTERN DOTS-5 for line continuation. Laying out these specialized notations, including inserting the appropriate line continuation characters, lies far outside the scope of the Unicode standard; just like breaking ordinary equations across multiple lines in *TeX, this most likely needs to be done by hand to obtain acceptable results.

[CEBF08, p. 8]



Remarque
Lorsqu'une équation-bilan est longue, on emploie le signe « indicateur de continuation » ⠐ (5).
Le signe d'opération est alors mis au début de la ligne suivante.

$CO(g) + H_2O(l) \rightarrow CO_2(g) + H_2(g)$

$HCl(g) + H_2O(l) \xrightarrow{\text{eau}} H_3O^+(aq) + Cl^-(aq)$

References

[APH16] American Printing House, *Nemeth Tutorial*, 2016. https://nemeth.aphtech.org/
[APH18] American Printing House, *UEB Math Tutorial*, 2018. https://uebmath.aphtech.org/
[Ald16] V. Aldridge, *Elements of German Braille Code (incorporating the revisions of 1998–2011)*, 2016. http://www.braille.ch/download/germ-brl.pdf.
[BA20] Braille Autoriteit, *Braillestandaard voor algemeen gebruik in het Nederlandse taalgebied*, 24 april 2020. https://braille-autoriteit.org/documenten/braillestandaard_voor_algemeen_gebruik_in_het_Nederlandse_taalgebied_versie_2019.pdf
[BANA97] Braille Authority of North America, *Music Braille Code*, 1997 https://www.loc.gov/nls/wp-content/uploads/2016/03/music_braille_code.pdf
[BB19] Braille Brain https://braillebrain.aphtech.org/
[CBFU08] *Code braille français uniformisé pour la transcription des textes imprimés (CBFU)*, deuxième édition septembre 2008. https://liblouis.io/braille-specs/french/cbfu_edition_internationale_1.pdf
[CEBF08] Commission pour l'évolution du braille français, *Notation braille dans le domaine de la chimie*, première édition juin 2008. https://liblouis.io/braille-specs/french/notation_braille_chimie2008.pdf
[CFGG14] Jean-Eudes Cayouette, Pierre Ferland,. Russell Gagnon, and Isabelle Grant, *Code Braille québécois pour la transcription de la notation informatique (CBI)*, 2014. https://www2.banq.qc.ca/documents/services/services_specialises/services_adaptes/CBI.pdf
[Hor24] Matthew Horspool, ed., *The Rules of Unified English Braille*, third edition 2024. https://iceb.org/Rules%20of%20Unified%20English%20Braille%202024.pdf
[HRH22] Josie Howse, Kathy Riessen, and Leona Holloway, eds., *Unified English Braille: Australian Training Manual*, revised January 2022. https://brailleaustralia.org/wp-content/uploads/2013/10/UEB-Australian-Training-Manual-January-2022-Unicode.pdf
[Pun09] Punktskriftsnämnden, *Svenska skrivregler för punktskrift*, andra upplagan 2009. https://www.mtm.se/app/uploads/2024/06/2024-05-30_2009_svenska-skrivregler-for-punktskrift_unicode-monster_for_punktskrift.pdf
[Pug21] Robert Pugar, *Brajična matematička notačija*, 2021. https://savez-slijepih.hr/app/uploads/2021/07/3-Brajicna-matematicka-notacija-Pugar_travanj_2021.pdf
[SABP24] Slovenská autorita pre Braillovo písmo, *Pravidlá písania a používania Braillovho písma v Slovenskej republike*, 2024. https://www.skn.sk/swift_data/source/2024/txt/pravidla_BP_graficka_uprava_brailovych_publikacii.pdf
[Sar21] Murray Sargent, *Unicode Math Braille Sequences*, 15th August 2021. https://devblogs.microsoft.com/math-in-office/unicode-math-braille-sequences/
[UN21] UN Framework for responding to the socio-economic impacts of COVID-19 in Nepal, 2021. https://nepal.un.org/en/download/67849/122902
[UNE13] UNESCO, *World Braille Usage*, third edition 2013. https://www.perkins.org/wp-content/uploads/2021/07/world-braille-usage-third-edition.pdf.

2.4 Misleading text in SpecialCasing.txt about iota subscript [#436]

Recommended UTC actions

1. **Action Item** for Markus Scherer, PAG: For the note in SpecialCasing.txt about iota-subscript moving to the end, propose replacement text with a discussion of the issue and possible approaches in core spec 3.13.2 Default Case Conversion. For Unicode 18.0. See L2/25-183 item 2.4.
2. **Action Item** for Markus Scherer, PAG: Clarify the SpecialCasing.txt heading “Unconditional mappings”, saying that the mappings are not language-sensitive nor context-sensitive. For Unicode 17.0. See L2/25-183 item 2.4.

Feedback (verbatim)

Date/Time: Fri June 13 11:31:46 PDT 2025
ReportID: ID20250613113146
Name: Jeff Davis
Report Type: Report Error in Publication Data
Opt Subject: Misleading text in SpecialCasing.txt

Regarding a question I asked about ICU here:
https://groups.google.com/a/unicode.org/g/icu-support/c/n09HXzVZCWk"

The maintainers suggested that the text in SpecialCasing.txt about the treatment of U+0345 when uppercasing or titlecasing is misleading, and that I should file a report. I don't claim expertise in this area, but I am confused about where, how, and under what conditions the rule in SpecialCasing.txt should be applied.

Specifically:

(a) It's listed under "Unconditional mappings", but ICU only seems to make use of the rule when using a Greek locale; and
(b) Moving characters around doesn't seem to be permitted by toUppercase(X) as defined in 16.0 3.13.2 R1. More generally, moving characters around doesn't seem to be a "mapping" as I understand it.

Background information / discussion

https://www.unicode.org/Public/16.0.0/ucd/SpecialCasing.txt

```
None
# IMPORTANT-when iota-subscript (0345) is uppercased or titlecased,

#  the result will be incorrect unless the iota-subscript is moved to the end

#  of any sequence of combining marks. Otherwise, the accents will go on the capital iota.

#  This process can be achieved by first transforming the text to NFC before casing.

#  E.g. <alpha><iota_subscript><acute> is uppercased to <ALPHA><acute><IOTA>
```

Normally, the ICU uppercase functions do what the core spec says:
https://www.unicode.org/versions/Unicode16.0.0/core-spec/chapter-3/#G34078

R1 toUppercase(X): Map each character C in X to Uppercase_Mapping(C).

So in order to get Greek right when there is an implicit or explicit iota subscript (ypogegrammeni) followed by another combining mark, you need to normalize first.

Markus: I don't actually think that NFC will do the trick, and might make it worse, because it pulls the iota subscript into a composite letter despite a following lower-ccc combining mark.
NFD would work, or (with ICU), FCD.

SpecialCasing.txt provides mappings, but the comment can be read as a modified specification of the uppercasing and titlecasing operations, different from the core spec.

We could clarify the heading “Unconditional mappings”, saying that the mappings are not language-sensitive nor context-sensitive.

The discussion of Greek iota handling in case mappings should probably be *moved* from the data file into the core spec.

3. Proposed new scripts & characters

PAG members reviewed the following proposals, provided feedback to SAH, and the feedback has been addressed.

No further recommended actions from our side.

- L2/25-077 Unicode request for w with palatal hook [SEW #622]
    - Another Latin letter with palatal hook, propertywise like existing ones, *e.g.*, ᶀ.
- L2/25-112 Unicode request for Cyrillic letter jeru with connecting bar [SEW #633]
    - A lowercase-only Cyrillic letter for phonetics, propertywise like the existing U+1D2B ᴫ CYRILLIC LETTER SMALL CAPITAL EL encoded for the UPA in the Phonetic Extensions block.
- L2/25-114 Unicode request for letters with double tilde overlay [SEW #398]
    - Two more lowercase-only Latin letters, propertywise like U+AB38 ꬸ.
- L2/22-260 and L2/25-113 Proposal to encode three characters in Tulu-Tigalari [SEW #641]
    - Another reflex (descendant) of Brahmi U+11008 𑀈, compare Malayalam U+0D5F ൟ.
      Propertywise like 𑀈 (in particular, lb=Aksara) except for Indic_Conjunct_Break. Propertywise like ൟ except for Line_Break (because Malayalam uses spaces instead of breaking at orthographic syllable boundaries) and Indic_Conjunct_Break. The difference in Indic_Conjunct_Break stems from the fact that Tulu-Tigalari is an invisible stacker script rather than a virama script. It therefore follows from the derivation that the Tulu-Tigalari archaic II gets Indic_Conjunct_Break=Consonant rather than Indic_Conjunct_Break=None like 𑀈 and ൟ.
- WG2/N5303 Proposal to encode one Tangut ideograph [SEW #645]
    - Another Tangut ideograph, propertywise like the others.

4. Line Break

4.1 Consistency of symbols in Line_Break [#321]

Recommended UTC actions

1. **Note**: PAG recommends a conservative approach for Line_Break values of existing characters, including symbols, especially ones that have been encoded for several years. Line_Break value changes should mostly be motivated by reports of actual problems in behavior of specific characters, rather than consistency arguments about broad sets of characters. See L2/25-183 item 4.1.

PAG input

PAG investigated

- [162-A68] Action Item for Roozbeh Pournader: Investigate whether further Linebreak property updates need to be made in the Legacy Computing block, for Unicode version 13.0.
- [172-A67] Action Item for Mark Davis, PAG: Check Line_Break values of symbols (excluding Creative Commons) for inconsistencies with other similar characters; emoji should have lb=ID; for a future version of the Unicode Standard. See L2/22-124 item UCD11.

There are thousands of symbols, encoded over many years based on different origins and use cases. Some have Line_Break Alphabetic (e.g., for chess notation), some Ideographic (e.g., newer emoji, or symbols that originate in east asian character sets), some Ambiguous (occur in both western & eastern character sets). The identity of some symbols changed via late emojification.

In short, Line_Break values across symbols differ, sometimes for good reason, and sometimes probably by accident.

We think that these action items were too broad. PAG is concerned about destabilizing effects of making broad Line_Break property changes based on consistency arguments rather than reports of actual problems. The set of all symbols is too large to be treated uniformly.

However, for characters that were added in a recent version of Unicode, it is appropriate to adjust Line_Break values as we and our users gain experience with using the new characters.

4.2 Is U+2E17 a hyphen and a break opportunity? (Yes.) [#408]

Recommended UTC actions

1. **No Action**: PAG recommends no action. The problematic statement has been corrected in the proposed update; it is not normative, so no decision is needed. The property assignment Unambiguous_Hyphen for U+2E17 is appropriate.

PAG input

From Robin Leroy, PAG.

UAX #14, Version 16.0, has these sentences in Section 5.5, Use of Double Hyphen (the earlier mention of U+2E17 in the preceding paragraph is about its glyph, not about the character, and is irrelevant here).

> Certain linguistic notations make use of a double-stroke, oblique hyphen to indicate specific features. The U+2E17 DOUBLE OBLIQUE HYPHEN character used in this case is not a hyphen and does not represent a line break opportunity. Automatic hyphenation or SHY would result in the display of an ordinary hyphen.

But the character is used analogously to a hyphen (albeit contrastively with it) in these notations, and it does represent a line break opportunity, since it has lb=BA. That character indeed appears in the description of lb=BA under *Other Terminating Punctuation*, which is strange, since it is not Terminal_Punctuation.

In the meantime, UTC-181-C53 made this character lb=Unambiguous_Hyphen.

Background information / discussion

Alright, what *is* this U+2E17 ⸗ DOUBLE OBLIQUE HYPHEN, and how is it used in *certain linguistic notations*? In character names list, it appears under the subheading *Ancient Near-Eastern linguistic symbol*, with the annotation « • used in ancient Near-Eastern linguistics » for those who were not paying attention to he subheading.
Document L2/03-347 provides a useful survey of such usage. Ignoring the hyphen-at-a-linebreak situation, the usage is:

- In Coptic, word-finally after the form of a verb before suffixes (contrasting with the use of an ordinary hyphen after the form of a verb before an incorporated nominal object).
- In Demotic, to mark a suffixed pronoun. The examples show contrasts with hyphens.
- In Hittite, to mark a clitic boundary within words cited in broad transcriptions in citations that also transliterate some words, where transliterations separate signs with hyphens (or dots). The first Hittite example in the document ends with $^{\text{GIŠ}}$KIRI$_6$.TU[R-*az* …] *mu-ta-a-i n⸗an⸗mu*, from a citation of KBU 12.57, fragment 1, obv i 7′–8′; the word nanmu is in broad transcription; the transliteration would be *na-an-mu* (the corresponding text is 𒄑𒊬⸢𒌉⸣[𒊍 …] 𒈬𒋫𒀀𒄿 𒈾𒀭𒈬, see also photographs).

These usages are analogous to usages of hyphens in the same notations, with the use of U+2E17 expressing some contrast with the ordinary hyphen.

For a more recent example in another language (but staying within the realm of ancient Near Eastern linguistics, with Akkadian), consider the use of ⸗ in a morpheme-by-morpheme gloss to separate the enclitic conjunction *-ma* (as opposed to - used for other morpheme boundaries) in this gloss from [Kam23]:

erm-um	*ša*	*ṭupp-i*	*ḫepi⸗**ma***
envelope-NOM	of	tablet-GEN	break.STAT.3.SG.M⸗CONJ

ṭuppa-ša	*i-šrum-ū⸗ma*
tablet.CSTR-ACC.3.SG.F	3-peel(off).PRET-PL.M⸗CONJ

"The envelope of the (plaintiff's) tablet happened to be broken, so they (= the judiciary) broke open her (the sued party's) tablet" (RA 9, 22:19–25)

In some conventions for normalized Akkadian transcriptions, such as those used by [Hue11] or by the eBL project (see, e.g., https://www.ebl.lmu.de/corpus/L/1/4/SB/I#5), the conjunction *-ma* is separated from the preceding word with a hyphen, so this usage can be seen as distinguishing an object language hyphen from a morpheme-separating hyphen in the gloss.

Note, as noted already in L2/03-347, that the Leipzig glossing rules recommend an equals sign for clitics instead, as found also in L2/03-347, and in other modern works in ancient Near Eastern linguistics (see, e.g., [Zól16]).

Correct line breaking of morpheme-by-morpheme glosses clearly is a matter for higher-level protocols. However, these examples all illustrate that the usage of the double oblique hyphen is « some kind of hyphen, but the hyphen already means something different ». It has always introduced a break opportunity, being lb=BA since its encoding in Unicode 4.1. It should continue to do so, now as lb=HH, so that it benefits from the rule against breaking after word-initial hyphens. The paragraph should be corrected as follows:

> Certain linguistic notations make use of a double-stroke, oblique hyphen to indicate specific features, often contrasting with the ordinary hyphen. The U+2E17 ⸗ DOUBLE OBLIQUE HYPHEN character is used in this case is not a hyphen and does not represent a line break opportunity. Automatic hyphenation or SHY would result in the display of an ordinary hyphen.

References

[Hue11] John Huehnergard, *A Grammar of Akkadian*. Third Edition, 2011.

[Kam23] Iris Kamil, "*t*-Forms of the Akkadian Stative". In: *Brill's Journal of Afroasiatic Languages and Linguistics* 15.1, 262–290. 1 May 2023. https://doi.org/10.1163/18776930-01501008.

[Zól16] Gábor Zólyomi, *An introduction to the grammar of Sumerian*. 21 April 2016. https://core.ac.uk/download/pdf/51326989.pdf.

Addendum: U+2E40 ⹀ DOUBLE HYPHEN

It is curious that a section on the use of double hyphen does not mention this character. The explanation seems to be that the section was last looked at in Unicode 5.2, and the character encoded in Unicode 7.0. Since its use (as described in the encoding proposal L2/11-038 and alluded to in the character names list) is relevant both to the title of the section and the discussion of the Fraktur hyphen, the editor has added some text about it.

5. Collation

5.1 ISO/IEC 14651 CTT_V17_0 [#441]

Recommended UTC actions

1. **No Action**: The draft ctt.txt file has been updated according to the suggested changes.

Feedback (verbatim)

PRI-526 Unicode 17.0.0 Beta

Date/Time: Sun June 29 12:42:18 PDT 2025
ReportID: ID20250629124218
Name: Marc Lodewijck
Report Type: Public Review Issue
Opt Subject: PRI #526: ISO/IEC 14651 CTT_V17_0 [PAG]

A few corrections are needed in the draft ISO/IEC 14651 Common Template Table (CTT_V17_0), specifically in the section describing how to compute
implicit weights for siniform ideographic characters or any code point not explicitly mentioned in the CTT.

The following subsections require updates.

1. Han unified ideographs
   The 12 unified Han characters in the CJK compatibility area should be explicitly listed, as currently only mentioned but not covered by any code range.

There is also an inconsistent wording in the comment for WEIGHT_BASE: it currently states “Extension A through Extension I”, but the ranges actually
include Extension J as well.

```None
% Weights for unified Han characters follow the Unified Repertoire and

%   Ordering, which is a language-neutral, traditional radical-stroke order.



%   The original URO and Extensions A through J, plus the 12 unified Han characters

%   in the CJK compatibility area are weighted implicitly as defined here.



%   WEIGHT_BASE = 0xFB40 for original URO and 12 unified Han from CJK compat area.
```

```
  %      cp >= 0x04E00 && cp <= 0x09FFF % URO
+ %      cp == 0x0FA0E                  % CJK compatibility character
+ %      cp == 0x0FA0F                  % CJK compatibility character
+ %      cp == 0x0FA11                  % CJK compatibility character
+ %      cp == 0x0FA13                  % CJK compatibility character
+ %      cp == 0x0FA14                  % CJK compatibility character
+ %      cp == 0x0FA1F                  % CJK compatibility character
+ %      cp == 0x0FA21                  % CJK compatibility character
+ %      cp == 0x0FA23                  % CJK compatibility character
+ %      cp == 0x0FA24                  % CJK compatibility character
+ %      cp >= 0x0FA27 && cp <= 0x0FA29 % CJK compatibility characters
- %   WEIGHT_BASE = 0xFB80 for Extension A through Extension I Han characters.
+ %   WEIGHT_BASE = 0xFB80 for Extension A through Extension J Han characters.
  %      cp >= 0x03400 && cp <= 0x04DBF % Ext. A
  %      cp >= 0x20000 && cp <= 0x2A6DF % Ext. B
  %      cp >= 0x2A700 && cp <= 0x2B73F % Ext. C
  %      cp >= 0x2B740 && cp <= 0x2B81D % Ext. D
  %      cp >= 0x2B820 && cp <= 0x2CEAD % Ext. E
  %      cp >= 0x2CEB0 && cp <= 0x2EBE0 % Ext. F
  %      cp >= 0x2EBF0 && cp <= 0x2EE5D % Ext. I
  %      cp >= 0x30000 && cp <= 0x3134A % Ext. G
  %      cp >= 0x31350 && cp <= 0x323AF % Ext. H
  %      cp >= 0x323B0 && cp <= 0x33479 % Ext. J
  %   For a given Han character at code point cp:
  %   base1 = WEIGHT_BASE + ( cp >> 15 )
  %   base2 = ( cp & 0x7FFF )  0x8000
  %   Then weight the character as:  "";;;
```

2. Tangut ideographs
   Minor clarification: the current comment labels (“Tangut ideographs” and “Tangut ideograph supplement”) do not match the Unicode block names. These
   should be corrected to “Tangut” and “Tangut Supplement”.

```None
  % Tangut ideographic characters are weighted implicitly as defined here.


  %   WEIGHT_BASE = 0xFB00
- %      cp >= 0x17000 && cp <= 0x187FF % Tangut ideographs
- %      cp >= 0x18D00 && cp <= 0x18D1E % Tangut ideograph supplement
+ %      cp >= 0x17000 && cp <= 0x187FF % Tangut
+ %      cp >= 0x18D00 && cp =< 0x18D1E % Tangut Supplement
  %   For a given Tangut character at code point cp:
  %   base1 = WEIGHT_BASE
  %   base2 = ( cp - 0x17000 )  0x8000
  %   Then weight the character as:  "";;;
```

3. Tangut component characters
   Minor comment label adjustment: the current comment labels (“Tangut components” and “Tangut component supplement”) differ slightly from the Unicode
   block names, notably in casing. These should be updated to match exactly: “Tangut Components” and “Tangut Components Supplement”.

Range boundary update: the upper bound for “Tangut Components Supplement” should be 0x18DF2, as the remaining code points in the block are currently
unassigned.

Offset correction: the expression for base2 mistakenly uses 0x17000 as the starting point. This should be corrected to 0x18800, which is the beginning
of the “Tangut Components” block.

```
None
  % Tangut component characters are weighted implicitly as defined here.


  %   WEIGHT_BASE = 0xFB01
- %      cp >= 0x18800 && cp <= 0x18AFF % Tangut components
- %      cp >= 0x18D80 && cp <= 0x18DFF % Tangut component supplement
+ %      cp >= 0x18800 && cp <= 0x18AFF % Tangut Components
+ %      cp >= 0x18D80 && cp <= 0x18DF2 % Tangut Components Supplement
  %   For a given Tangut character at code point cp:
  %   base1 = WEIGHT_BASE
- %   base2 = ( cp - 0x17000 )  0x8000
+ %   base2 = ( cp - 0x18800 )  0x8000
  %   Then weight the character as:  "";;;


  % Nushu ideographic characters are weighted implicitly as defined here.
```

4. Khitan Small Script ideographs
   A separate line should be introduced for the character U+18CFF, which is part of the “Khitan Small Script” block but lies outside the main contiguous
   range. This ensures that it is explicitly included in the weighting logic.

```
None
  % Khitan Small Script ideographic characters are weighted implicitly as defined here.


  %   WEIGHT_BASE = 0xFB03
  %      cp >= 0x18B00 && cp <= 0x18CD5 % Khitan Small Script
+ %      cp == 0x18CFF                  % Khitan Small Script
  %   For a given Khitan Small Script character at code point cp:
  %   base1 = WEIGHT_BASE
  %   base2 = ( cp - 0x18B00 )  0x8000
%   Then weight the character as:  "";;;
```

No changes are required for the Nüshu ideographic characters section; the existing definitions are accurate and complete.

6. Security

6.1 review ID_Type=Recommended from Unicode 13, 14, 15, 15.1 [#290]

Recommended UTC actions

1. **Consensus**: Change the Identifier_Type of U+08C9 ARABIC SMALL FARSI YEH from Recommended to Technical, and of U+16FF0 and U+16FF1 VIETNAMESE ALTERNATE READING MARK CA and NHAY from Recommended to Obsolete. For Unicode 17.0. See L2/25-183 item 6.1.
2. **Action Item** for Josh Hadley, PAG: Change the Identifier_Type of U+08C9 ARABIC SMALL FARSI YEH from Recommended to Technical, and of U+16FF0 and U+16FF1 VIETNAMESE ALTERNATE READING MARK CA and NHAY from Recommended to Obsolete. For Unicode 17.0. See L2/25-183 item 6.1.

PAG input

From Markus Scherer, PAG

I believe that we have not closely scrutinized characters that were added since Unicode 13 and which got Identifier_Type=Recommended just because they are in UAX #31 Recommended scripts.

There are 142 such characters, excluding Unified_Ideograph characters.
`[[[:Identifier_Type=Recommended:]&[[:Age=15.1:]-[:Age=12.1:]]]-[:Unified_Ideograph:]]`

A number of them probably should be set to Uncommon_Use or Technical.

Related:
UTC-172-A64 Action Item for Asmus Freytag, PAG: For UTS #39, look at publicly available data from the ICANN root zone LGR project to look at adjustments to the Uncommon_Use property and further clarifications of its definition.

Background information / discussion

PAG has reviewed all of the relevant characters. Many have been removed from Recommended in previous actions. Only three more should be removed from Recommended.

6.2 UAX #31 does not classify all scripts [#406]

Recommended UTC actions

1. **Consensus**: Add the seven scripts that were added in Unicode 16 to UAX #31 table 4 (excluded). For Unicode 17.0. See L2/25-183 item 6.2.
2. **Action Item** for Markus Scherer, PAG: Add the seven scripts that were added in Unicode 16 to UAX #31 table 4 (excluded). Also change “Characters may also be moved from one table to another” to refer to scripts instead. For Unicode 17.0. See L2/25-183 item 6.2.
3. **Action Item** for Markus Scherer, RMG: Add a Unicode release task for adding new scripts to appropriate UAX #31 tables 4, 5, and 7.

PAG input

From Markus Scherer, PAG

I was just checking for the classification of a script in UAX #31 tables 4 (Excluded), 5 (Recommended), and 7 (Limited Use). I noticed that Garay is not listed in any of these. On further inspection, exactly the seven scripts added in Unicode 16 are missing: Garay, Gurung Khema, Kirat Rai, Ol Onal, Sunuwar, Todhri, Tulu-Tigalari (These are all Excluded scripts.)

The text under table 7 says:

> This is the recommendation as of the current version of Unicode; as new scripts are added to future versions of Unicode, characters and scripts may be added to Tables 4, 5, and 7.

This suggests that the tables want to be comprehensive, which means that we should make an effort to list each script.

However, it makes me wonder whether this is necessary. Most newly encoded scripts will be Excluded. Should we just remove table 4 and say that any script not Recommended or in Limited Use is Excluded?

Note that the CLDR data file scriptMetadata.txt includes this script classification in the "ID Usage" field.

Also, the UTS #39 Identifier_Type is partially derived from which UAX #31 table lists which script, but that makes the text of one spec partially redundant with data files for another. We seem to spend more time on the data files, and because they are machine-readable, they seem more user-friendly. Should we just remove the script classifications from UAX #31 tables altogether and refer to the CLDR script metadata and to the Identifier_Type data?

Basically, looking at Identifier_Type data, if any character with a certain Script value has ID_Type=Recommended, then the script as a whole is. Otherwise, if any character has ID_Type=Limited_Use, then that is the script status.

Related: The text under table 7 continues with this:

> Characters may also be moved from one table to another as more information becomes available.

We don't move *characters* between tables, but scripts. And then within a script with a certain status, for certain characters we assign Identifier_Type values like Technical and Uncommon_Use. If we keep the UAX #31 table structure, then we should change this sentence to say "Scripts".

6.3 tr31-41 standard math profile seems incorrect [#407]

Recommended UTC actions

1. **Consensus**: Correct the profile for UAX31-R3b associated with the Mathematical Compatibility Notation Profile defined in Section 7.1 of Unicode Standard Annex #31, based on feedback ID20250426211321 in L2/25-163, as in PU-UAX #31 for Unicode Version 17.0. See L2/25-183 item 6.3.

Feedback (verbatim)

Date/Time: Sat April 26 21:13:21 PDT 2025
ReportID: ID20250426211321
Name: Crystal Durham
Report Type: Report Error in Publication Data
Opt Subject: tr31-41 standard math profile seems incorrect

Quoting the relevant text from UAX#31§7.1:

The Mathematical Compatibility Notation Profile for default identifiers [...] is associated with a profile for UAX31-R3b, which consists of removing the
characters in `[[:Pattern_Syntax:] - [:ID_Compat_Math_Continue:]]` from the set of characters with syntactic use (these are the characters ∂, ∇, and ∞).

This seems incorrect. The - here is a set difference operation, meaning that this standard profile as described asks to remove all syntax characters
except those in `[:ID_Compat_Math_Continue:]`. Given that the parenthetical identifies that the removed characters are ∂, ∇, and ∞, I suspect that the
set union operator & was supposed to be used instead. Thus the text should read

[...] removing the characters in `[[:Pattern_Syntax:]&[:ID_Compat_Math_Continue:]]` from the set [...]

or

[...] excluding the characters in `[:ID_Compat_Math_Continue:]` from the set [...]

instead.

Background information / discussion

The error has been corrected in the Proposed Update.

6.4 Gunjala Gondi: UAX31=Excluded vs. ID_Type=Limited_Use [#411]

Recommended UTC actions

1. **Consensus**: Change the Identifier_Type values for Gunjala Gondi characters (sc=Gong) from Limited_Use to Excluded, to match the UAX #31 classification of the script. For Unicode 17.0. See L2/25-183 item 6.4.
2. **Action Item** for Markus Scherer, PAG: Work with CLDR-TC to align the CLDR script metadata UAX #31-derived “ID Usage” with a contemporary version of UAX #31, especially for Gunjala Gondi. See L2/25-183 item 6.4.
3. **Action Item** for Josh Hadley, PAG: Derive the Identifier_Type values for Gunjala Gondi characters from the UAX #31 classification of the script as specified. For Unicode 17.0. See L2/25-183 item 6.4.

PAG input

From Markus Scherer

UAX #31 lists Gong=Gunjala Gondi in Table 4. Excluded Scripts

However, the Unicode 16 version of IdentifierType.txt (and the Unicode 17 beta version) gives its letters ID_Type=Limited_Use.

This is due to this script being listed as Limited_Use in the CLDR 47 script metadata.

It is the only script where these two sources disagree. They are supposed to agree: The CLDR script metadata “ID Usage” is “The usage for IDs (tables 4-7) according to UAX #31.”

Which is the most appropriate classification?

According to the script encoding proposal L2/15-235, “It is not the primary script for Southern Gondi” (ggo); that language subtag has been deprecated, and the script metadata lists the default language as Adilabad Gondi (wsg).

The proposal also says that Southern Gondi is “presently spoken by 100,000 people” and “The script is being actively taught. Documents in the script are being published.”

Glottolog Adilabad Gondi: “not endangered ... 4 (Educational)”

Background information / discussion

Gunjala Gondi is a historic script with some recent revival efforts via teaching in some schools. According to Omniglot, the number of people currently able to read and write the script is 80. The relevant languages are mostly written in other scripts.

At this time, the UAX classification of “Excluded” from default identifiers is appropriate.

6.5 Changes in confusables data

PAG has reviewed documents, feedback, and recent character additions, and has made and is making appropriate changes to the data in confusables.txt for UTS #39, for Unicode version 17.0.

6.6 UTS #39: remove special confusables suggestion forms [#6]

Recommended UTC actions

1. **Action Item** for Mark Davis, PAG: Change UTS #39 to request all feedback via the standard UTC reporting forms. Remove non-standard mechanisms from the Feedback reference and from other relevant parts of the spec. Change the formerly linked pages to point to the standard mechanism. Change internal documentation to no longer refer to the special confusables suggestion mechanism. For Unicode 17.0. See L2/25-183 item 9.1.

PAG input

UTS #39 has been inviting suggestions for additional data about characters that are confusable with each other via special web pages and web forms. We have received only a relatively small amount of data this way which has been taken into account. The special pages and forms do not adhere to current standard UTC practice. PAG does not think that it is worth updating and maintaining them, nor that it is worth retaining a non-standard feedback mechanism for this data.

7. Emoji

7.1 The RGI set is not covered by the fully-qualified emoji [#409]

Recommended UTC actions

1. **Action Item** for Ned Holbrook, PAG: In UTS #51 data file emoji-test.txt, correct the statement about the relationship between the RGI_Emoji_Qualification property and the RGI set. For Unicode Version 17.0. See L2/25-183 item 7.1.

PAG input

From Robin Leroy, PAG: Since Unicode 12.0, the file emoji-test.txt has stated that

• The RGI set is covered by the listed fully-qualified emoji.

This has always been incorrect. Instead, the RGI set is equal to the set of sequences whose status is given as either *fully-qualified* or *component*. That is, in Unicode Versions 12.0..16.0, the following sets are equal, where the first one is the derivation of the RGI set given in ED-27:

```None
[\p{Basic_Emoji}

 \p{RGI_Emoji_Keycap_Sequence}

 \p{RGI_Emoji_Modifier_Sequence}

 \p{RGI_Emoji_Flag_Sequence}

 \p{RGI_Emoji_Tag_Sequence}

 \p{RGI_Emoji_ZWJ_Sequence}]
```

1.

```None
[\p{RGI_Emoji_Qualification=Fully_Qualified}

 \p{RGI_Emoji_Qualification=Standalone_Component}]
```

Background information / discussion

In Unicode 11, this would have been correct, as emoji-test only defined fully-qualified and non-fully-qualified, with the standalone components being fully-qualified. However, at that time, UTS #51 did not define an RGI set, only the RGI sequence set ED-26, and accordingly emoji-test did not make a statement about covering the RGI set. This statement in emoji-test was therefore never correct.

Note that the property RGI_Emoji, defined by a derivation in UTS #51, is never directly provided in a data file; the derivation 1. above relies on multiple data files, which is inconvenient for the Unicode tools. The characterization 2. has the advantage of depending only on one file, and of allowing for a retrojection of the RGI set prior to Unicode 11. The equality of these sets is checked by an invariant test.

8. Other

8.1 IBM PC character set mapping table [#405]

Recommended UTC actions

1. **Action Item** for Ken Whistler, PAG: Update the two readmes at /Public/MAPPINGS/ to note that all of the MAPPINGS are obsolete, only for historical interest, and that the UTC will not add more data files there. See L2/25-183 item 8.1.

Document + Feedback (verbatim)

Unicode Document Register# L2/25-142

Feedback submitted through Unicode Contact form:

Date: Sun, 27 Apr 2025 19:09:09 -0400

Author: Michael Walden

Subject: IBM PC character set mapping table

Hello UTC,

I have created a mapping table to map the legacy IBM PC character set to Unicode.

I would like to submit my IBMPCCP437.TXT mapping table to the following directory
on the Unicode.org web site:

https://www.Unicode.org/Public/MAPPINGS/VENDORS/MISC/

My IBMPCCP437.TXT mapping table can be downloaded from this URL:

https://MW.Rat.bz/cp437map/IBMPCCP437.TXT

I propose that my mapping table should obsolete the following two mapping tables
that, when used together, cover the full IBM PC character set but with errors
in the included mappings.

https://www.Unicode.org/Public/MAPPINGS/VENDORS/MISC/IBMGRAPH.TXT
https://www.Unicode.org/Public/MAPPINGS/VENDORS/MICSFT/PC/CP437.TXT

My mapping table contains all characters in one and has, I believe, no errors.

For more details see the comments in my IBMPCCP437.TXT file.

The following web page briefly says a little more about my mapping table.

IBM PC Code Page 437 to Unicode Mapping Table
https://MW.Rat.bz/cp437map

Lastly, the following four web pages relate to my IBMPCCP437.TXT mapping table
and discuss the reasons I had for choosing the correct characters and also
demonstrate the mapping against the IBM PC MDA ROM font character table.

Dr. David J. Bradley on IBM PC's Character Set and More
https://MW.Rat.bz/djb

The IBM PC Character Set Confusion Clarified
https://MW.Rat.bz/confusion

IBM PC Technical Reference Character Set (00-FF) Quick Reference
https://MW.Rat.bz/ibmpctr

IBM PC MDA ROM Font Character Table
https://MW.Rat.bz/mdarom
So, the question is is this the correct email address to send such a request
to? If not, where then?

Lastly, feel free to pass on my five links from above to anyone who might
appreciate this sort of subject matter.

Thanks,

- Michael Walden

Background information / discussion

The mapping tables that are published on the Unicode website have helped in the early days of Unicode to establish its repertoire, and have helped seed conversion tables for implementation.

For almost every encoding/charset/codepage there are multiple ways that they have been implemented across platforms, across libraries, and across time.

For a long time now, tracking and providing conversion tables is out of scope for the UTC. Changing existing mapping tables could actually create confusion about the provenance of the repertoire and of implementation mappings.

Note that the ICU-TC has published a charset repository (beyond the mapping tables built into ICU by default), based on actual behavior of several implementations. Even that has seen little activity since nearly all text that has been created in the last 15 years or so is in a Unicode encoding.

8.2 PD-UTR #59 East Asian Spacing questions & suggestions [#433]

Recommended UTC actions

1. **Action Item** for Koji Ishii, PAG: Consider feedback ID20250614171421 and make appropriate changes to proposed draft UTR #59. See L2/25-183 item 8.2.

Feedback (verbatim)

PRI-510

Date/Time: Wed June 04 17:14:21 PDT 2025
ReportID: ID20250614171421
Name: Allen Watkins Smith
Report Type: Public Review Issue
Opt Subject: Proposed Draft UTR #59, East Asian Spacing

1: A paragraph like the one at the end of section 1 in UAX #11 (East Asian Width), noting that the new property almost certainly does not preserve canonical equivalence (due to the use of East_Asian_Width in deriving property values for it), is probably needed. (I don't know how being "informative" versus East_Asian_Width's "normative" will affect the necessity for this.)

3.4.3:

3rd paragraph: Does typography professionally used for East Asian scripts use parentheses to highlight - as in to emphasize or make stand out from the rest of the text - words? If so, "parentheses do." → "parentheses do in typographic practice for East Asian scripts." or "parentheses do in typographic practice for some East Asian scripts." If not, then "parentheses do" → "underlining or bold-facing does".
(Please add or substitute any common or otherwise well-known means of emphasis used in practice with East Asian scripts. There is at least one interlinear annotation method used for emphasis in Japanese writing, if I recall correctly, and I've seen underlining - and, at least with hand-drawn characters, what might be called bold-facing - used with East Asian scripts, but I have no idea how common, well-known, or formally acceptable any particular method is.)

3.4.4:

Unless I completely misunderstand it, Tr would not cause a character to be displayed upright (as in similar to or in the same way as in horizontal text). If Tr being treated in the auto-spacing algorithm as if displayed upright is what is meant, why?

4.1.1: "scripts." → "scripts, or if all Script_Extensions scripts are East Asian scripts." (Some characters have a Script of Common or Inherited but a geographically or otherwise restricted set of Script_Extensions scripts listed, being customarily used only in those scripts.)

4.2.1A: From the examples given, other characters frequently found around numbers should also be included; for instance, one example is "$20", but $ has a General_Category property of "Currency_Symbol (Sc)", not "Other_Punctuation (Po)". I suggest adding a rule of '"Include if the Line_Break property is "Postfix_Numeric (PO)" or "Prefix_Numeric (PR)".' As well as some other number-associated code points, this will take in all currency symbols - unless excluded by a later rule (such as for East_Asian_Width) - including the entire Currency_Symbols block.

Copyediting (I may do some more later if it's needed/wanted):

3.4.2:

2nd paragraph: "such as plain text files" → "such as in plain text files"
4th paragraph: "the code point to where the algorithm doesn't insert" → "this code point where the algorithm fails to insert"
5th paragraph: "SPACE to where the algorithm inserts" → "SPACE where the algorithm mistakenly inserts"

3.4.3:

1st paragraph: "characters insert" → "characters cause the insertion of"
2nd paragraph: "to both sides of them, considering that not doing so look unbalanced." → "to both sides of the word, from the viewpoint that not doing so looks unbalanced."
3rd paragraph: "and numeral digits, rather" → "and digits, rather"

8.3 UCDXML kEH_FVal entries replace Unikemet.txt pipe separators with spaces [#435]

Recommended UTC actions

1. **No Action**: The UCDXML data has been regenerated and the pipe symbols in kEH_FVal values have been restored.

Feedback (verbatim)

PRI-526 Unicode 17.0.0 Beta

Date/Time: Tue May 20 16:29:16 PDT 2025
ReportID: ID20250520162916
Name: Andrew West
Report Type: Public Review Issue
Opt Subject: PRI 526 : kEH_FVal in XML data files [PAG]

There are 15 kEH_FVal entries in Unikemet.txt that include a vertical bar () as a delimiter. In the XML data files for Unicode 17.0 beta all instances of "" in the kEH_FVal text string have been replaced by a space " ". I believe that the vertical bar should be preserved.

Background information / discussion

The feedback above is exactly as received, but the () and the "" should probably read (|) and "|".

Unicode 17 beta ucd/Unikemet.txt

```
None
U+13055    kEH_FVal    ? | mnẖ.t
```

Unicode 17 beta UCDXML (nounihan grouped) data:

```
None
<char cp="13055" na="EGYPTIAN HIEROGLYPH B005A" [...] kEH_FVal="?   mnẖ.t" kEH_UniK="B005A"
kEH_JSesh="B104"/>
```