L2/19-115

Proposal to ensure usability of fixed-width spaces

For consideration by Unicode Technical Committee

2019-03-31
Marcel Schneider (charupdate@orange.fr)

“We must always say what we see.
Above all we must always
— which is more difficult —
see what we see.”

Alain Finkielkraut quoting Charles Péguy

Problem

The digital representation of virtually all of the world’s writing systems is threatened by Unicode disrupting the traditionally non-breaking line break property of the major part of the whitespace characters. Unicode denied them the no-break ("GL" for “glue”) property, that should have been given to all spaces in the range starting at U+2002 and ending at U+200A. Today the only space in this range that is actually non-breaking, U+2007 FIGURE SPACE testifies that the correct Line_Break property value GL was on the table, but was restricted to the one space where it is least useful. FIGURE SPACE occurs at line start *before* numbers, not as a group separator *in* numbers — unlike what the relevant Unicode specification, UAX #14 *Unicode Line Breaking Algorithm,* alleges about U+2007; please refer to the Background section for details.

The following hint proves that the original design of the space range was accurate. We actually need breaking variants of the larger two (em and en) spaces, so that there is a point in having them both breaking and non-breaking. This duality is what the pairs U+2000 EN QUAD & U+2002 EN SPACE, and U+2001 EM QUAD & U+2003 EM SPACE were obviously designed for, given they were subsequently tailored that way in Donald Knuth’s TeX, along with all the others being non-breaking there.

The Unicode Standard (§ 23.2) suggests to compose non-breaking spaces with WORD JOINER. Beyond being uselessly counter-intuitive and in disconnect with real-world users expecting these spaces to be non-breaking, that plot is in turn threatened by the time-shifted dual encoding of the break preventer, as spotted in the preceding *Proposal to focus line break prevention design on end-user input*. It all resulted in lowering the incentive to implement those characters, notably in fonts. By contrast, since Unicode caught up in 1999 encoding U+202F NARROW NO-BREAK SPACE, this actually useful character has been added to fonts that support neither U+2009 THIN SPACE, U+2060 WORD JOINER, U+FEFF ZWNBSP, nor any other peculiar space (data is provided below).

While Unicode customarily cares about sparing implementers hardships, the counter-intuitive and counter-productive Unicode fixed-width space scheme is probably the most unstraightforward part of the Standard. It results in making the functional and interoperable representation of text extremely hard to implement. That issue raises the more concern as it was obviously well understood and would have been very easy to fix, but ended up being needlessly made up in spite of Unicode encoding principles calling for accurate representation of writing systems, as well as for interoperability, and in spite of Unicode encoding practice favoring most streamlined schemes.

Background

The “encoding error” that the em and en space pairs’ presence is claimed to be is not plausible, the less as the Xerox Character Code Standard (XCCS, that the Unicode Standard is reported to be based upon) has the following unambiguous rows in the table of spaces on page C-9 [1]:

Code(s)	Name	Also called	Width
357 | 55	em quad	em space	1.0 em fixed width
357 | 54	en quad	en space	½ em fixed width

With that table at hand, duplicate encoding of these spaces *by mistake* was absolutely impossible inside the Unicode project. Rather were they encoded so *by design,* making a clever use of aliases, for the purpose of providing both a breaking space and a non-breaking space of the respective width. These wide spaces are indeed the only ones of the series with use cases where they are expected to be breaking, beside of use cases where they should be non-breaking, TeX demonstrates.

Sources are lacking due to under-documentation of Unicode’s setup, but it is sufficiently obvious that the GL property value was removed or denied against design at a stage of the project where the code points were fixed, but properties could still be changed. Yet denying the Line_Break=GL property value to 8 space characters out of 9 is supposed to have been a source of conflict leading to one of those *many crises* that may have contributed to make working conditions hard to accept in the early Unicode team, that is consistently reported to have seen *authors dropping like flies.* After subsequently carrying these unintended duplicate pairs over a couple of years, Unicode ultimately (version 2.0) defined decomposition mappings so that in decomposed Unicode normalization forms NFD and NFKD, U+2000 EN QUAD and U+2001 EM QUAD are replaced with U+2002 EN SPACE and U+2003 EM SPACE respectively, in spite of the now-divergent but as-originally-intended line break behavior specified in TeX. Simultaneously released *Stability Policies* even locked these decomposition mappings. — in the Proposed actions section below we’ll see what to make of that.

Hailed as best source of information, Patrick Andries’ *Unicode 5.0 en pratique* (French, see full title in the References section) says about THIN SPACE in the range of fixed-width spaces, § 6.3.2, p. 144 [2]:

> One could believe that THIN SPACE could be used in French typography before the marks “;”, “?”, “!” and the footnote indicators; unfortunately, THIN SPACE is breaking. Processes not implementing UAX #14 (see § 4.2.7 *Line break classes*) will assume a break opportunity after, and before big punctuation marks. And all processes are allowed to break the line after a THIN SPACE occurring between digits as a group separator, or before a footnote indicator. Fortunately, other ways of representing this non-breaking thin space do exist; see § 6.3.4 *Thin spaces in French.* (*On pourrait croire que l’ESPACE FINE pourrait servir en typographie française devant les signes « ; », « ? », « ! » et les appels de note, malheureusement, l’ESPACE FINE est sécable. Les processus qui ne mettent pas en œuvre l’UAX n° 14 (voir § 4.2.7 Classes de coupure de ligne) penseront donc pouvoir couper la ligne après son apparition et devant les signes de ponctuation haute. Et tous les processus pourront légitimement couper la ligne après une espace fine insérée entre des chiffres comme séparateurs de milliers, par exemple, ou avant l’appel de note. Heureusement, il existe d’autres manières de coder cette espace fine insécable, voir § 6.3.4, Espaces fines en français.*)

We see the implicit disapproval of the THIN SPACE’s inconvenient line break property value on one hand, and on the other hand some concern about (mis)using the line break algorithm that is not surprisingly unable to

reliably palliate inconvenient line break property values. But as shown below, there is no clear dividing line between those processes that do implement UAX #14 and those that don't. — The other ways pointed here are workarounds with WORD JOINER, styling, or NARROW NO-BREAK SPACE which is the actual and best supported way of doing this, and is also acknowledged by the author since 2007 (L2/07-209R). Please refer to the upcoming *Proposal to clarify the purpose of U+202F NARROW NO-BREAK SPACE.*

Unicode Standard Annex (UAX) #14 ***Unicode Line Breaking Algorithm*** specifies an elaborate system of line break classes and line break rules that seem to be designed to make no-break spaces widely unnecessary. For instance, punctuation like exclamation/question mark and (semi)colon belong to classes EX or IS (infix numeric separator) respectively. These are subject to rule LB13 *Do not break before,* ***even after spaces*** [highlighting added], which has precedence over rule LB18 *Break after spaces.* However, this scheme has multiple issues. The class SP only encompasses U+0020 SPACE, while this space is inconvenient for the purpose of spacing off French punctuation because it is too wide and additionally expands in justification. Further, the use of SPACE as a group separator is not covered, as it may be hard to algorithmically assess whether a string of digits with interspersed SPACEs is meant to represent individual numbers or one large number. Above all, the line breaking algorithm does not even attempt to properly handle spaced quotation marks, as opposed to opening and closing punctuation, and even without SP it seems helpless when rule LB19 says: *Do not break* ***before or after*** *quotation marks, such as* '"'. Formally:

× QU
QU ×

That is because quotation marks are tricky, as they are either symmetric or locale dependent. Even angle quotation marks may be both opening and closing («» in French, »« in German de-DE, »» as alternate quotes in Swedish), additionally to being bidi-mirrored. Accurately removing break opportunities brought in by spaces («<SP>content<SP>») would require an even more elaborate algorithm; yet again, SP-wide spacing is inappropriate here as it is widely disliked with quotation marks, too. As a result, the *Unicode Line Breaking Algorithm* is unable to prevent big French punctuation with U+2009 THIN SPACE from breaking off. In this field it is proposing some impractical schemes that boil down to incomplete and therefore pointless patches. Consistently, while the useful parts of UAX #14 are up and running, its SPACE hacks are scarcely ever implemented; the Unicode implementation fully implementing UAX #14 is yet to be found.

There are also some frankly bad parts in UAX #14. Since its earliest available version (revision 4 from July 1998 https://www.unicode.org/reports/tr14-4/, as presented at *Thirteenth International Unicode Conference* in September 1998, Session B14), when it was still a draft Unicode Technical Report not endorsed by the Consortium, until its now-latest version Unicode 12.0.0 it invariably claims:

2007	FIGURE SPACE

This is the preferred space to use in numbers. It has the same width as a digit and keeps the number together for the purpose of line breaking.

The width of FIGURE SPACE disqualifies it as a group separator, as it is wider than the surrounding interword spaces (except in cases where justification requires justifying spaces to be significantly expanded):

This screenshot with FIGURE SPACE as a thousands separator shows that FIGURE SPACE is inappropriate in this role when occurring inline, and the following one shows how ugly it looks in isolation, too:

Consistently, other typesetting-related standards specify that a *thin space* is to be used for the purpose of forming triads of digits for legibility. Cf. the upcoming *Proposal to clarify the purpose of U+202F NARROW NO-BREAK SPACE.* The truth is that FIGURE SPACE is used *around* numbers, not *in* numbers. It is especially convenient as a leading space to improve the legibility of a column of numbers in the absence of tabulations, along with U+2008 PUNCTUATION SPACE filling in the advance width of the FULL STOP or COMMA decimal or group separators, like in this example:

1	<2007 2007 2007 2008 2007 2007 0031>
12	<2007 2007 2007 2008 2007 0031 0032>
123	<2007 2007 2007 2008 0031 0032 0033>
1,234	<2007 2007 0031 002C 0032 0033 0034>
12,345	<2007 0031 0032 002C 0033 0034 0035>
123,456	<0031 0032 0033 002C 0034 0035 0036>

Accordingly, the FIGURE SPACE is also known as *tabular space;* it translates to French *ESPACE TABULAIRE*, and *espace de lisibilité* (legibility space) is also found in UIs. Also,*The Unicode Standard,* although providing too scarce information on the topic, tells about these two space characters (version 12.0.0, § 6.2, p. 264):

> U+2007 FIGURE SPACE has a fixed width, known as *tabular width*, which is the same width as digits used in tables. U+2008 PUNCTUATION SPACE is a space defined to be the same width as a period.

Extrapolating from its use in such plain-text-style tables, we can reasonably assume that PUNCTUATION SPACE might also be used as a group separator at the condition that — being a breaking space (Line_Break=BA for Break After) carrying one single, trailing break opportunity by default — it is followed by either WORD JOINER or ZWNBSP. (About these and why to use either of them, please see the preceding *Proposal to focus break prevention design on end-user input*.) But neither TUS nor UAX #14 provide any hint about what to make of PUNCTUATION SPACE, while UAX #14 adds another misleading note in the *IS: Infix Numeric Separator* section:

> *Note:* FIGURE SPACE, not being a punctuation mark, has been given the line break class GL.

That is consistent with what UAX #14 states about FIGURE SPACE elsewhere (quoted above), but is based on a false premise. It assumes that users of the Unicode Standard are really typesetting FIGURE SPACE to represent the group separator in scripts using a space instead of another character such as listed here: COMMA, FULL STOP, ARMENIAN FULL STOP, ARABIC COMMA, NKO COMMA, just to give a few examples; and that readers of UAX #14 consequently expect FIGURE SPACE to be mentioned here, rather than PUNCTUATION SPACE. (Please see under Proposed actions below about correcting this note.)

The following thread from April 2015 on the Typophile forum [3] shows user reception of UAX #14 and awareness of Unicode's easily avoidable mistakes. (Some parts are skipped or boldened for diagonal reading.)

> A: **Are there any uses for thin or hair space where it wouldn't make sense for them to be non-breaking?** I ask because I can't think of any, and **the fact they aren't hard spaces makes them pretty unusable** in reflowable text such as web. Why did Unicode not at least go with two equal sets of spaces, one of which would be non-breaking, instead of just a single non-breaking word space?
>
> B: The intent of those were originally to do fine-tuning by hand on metal justified type and should not be breakable. Even in today's world of digital type they are still useful.
>
> A: Can you give some examples for digital type?
>
> C: They are great for any sort of **groups of figures or letters, which are meant to be shown as a unit. Abbreviations, dates, large numbers, bank accounts, measurement units,** mathematical equations …
> You basically encode that the unit belongs together, but is just visually separated a little bit. And that's why you don't want it to break at the end of the line. Power Supply: 110 V
>
> D: Also, they're **needed in French** to set some punctuation. Anyway, I agree with inktrap: **their default behaviour should be non-breaking. And that's how they work in InDesign, for example.**
> I don't know how Unicode encoded spaces are supposed to act.
>
> E: Agree wholeheartedly with all the responses. **They are very valuable as non-breaking. I cannot think of a non non-breaking use for them.**
>
> F: According to the Unicode Standard Annex #14, Line Breaking Properties, the following characters are in the class BA, which means there is an opportunity to break after, but not before.

2000	EN	QUAD
2001	EM	QUAD
2002	EN	SPACE
2003	EM	SPACE
2004	THREE-PER-EM	SPACE
2005	FOUR-PER-EM	SPACE
2006	SIX-PER-EM	SPACE
2008	PUNCTUATION	SPACE
2009	THIN	SPACE
200A	HAIR	SPACE

> 205F MEDIUM MATHEMATICAL SPACE
>
> That being said, the breaking algorithm (cf Example Pair Table) specifies that you may not break after a character in BA if it is followed by a character in one of the classes CL, EX, SY, IS, ZW and WJ.
>
> **EX contains the exclamation mark and the question mark; IS contains the semicolon. It appears that French typography is saved and does not need the narrow no-break space (U+202F; cf**

> **Unicode Notes on French). A consequence of the algorithm is that those characters may not be "unusable in reflowable text such as web" as you say.**

Note: This is partly untrue, partly outdated. The question and exclamation marks and semicolon precisely must not be typeset with a normal-width space, even less with a justifying space, while according to the new school, the colon (equally IS) is likewise typeset with a thin space. Unicode notes about French that the thin space can be represented by NARROW NO-BREAK SPACE. Further, that part of the algorithm has no incidence since its implementation is currently skipped. — Later the thread continues:

> G: In the case of characters with a place in Unicode, **a hairline space indeed should not be breakable,** since it would be used for letterspacing - or for adjusting kerning (but that is not a "legitimate" use, because the font is responsible for getting that right). […]
>
> C: So it looks like there is still no simple, recommendable and bullet-proof solution for this on the web. :-(
>
> **U+202F would be the way to go in theory, but most fonts will not have it and the browser's fallback results might be unacceptable.**
> The SPAN solutions do work, but they are just styling, not encoding. […]
>
> A: Reasonable way of doing it would be to **use normal non-breaking spaces and style their width in CSS.** That way, when copying, or if CSS fails, people would still get the non-breaking space which they already use and expect in those situations. […]

Note: Copying NBSP may fail, too, as this space is replaced with SP at copy-pasting in some applications.

Unicode's unilateral decision to deny most fixed-width spaces the GL Line_Break property value is thus assessed as failing to meet user expectations. Another example is found on the Francophone MacGeneration forum in a thread about diacriticized uppercase letters [4]:

> I've read here and there in this thread that we're talking about "no-break thin space." That is a pleonasm as the thin is always so. ([…] *j'ai lu çà et là dans ce fil qu'on parlait de « fine insécable ». C'est un pléonasme car la fine l'est toujours.*)

Among the **fonts shipped with Microsoft's Windows** operating system, **Verdana** is surely a textbook example of how the THIN SPACE failed to become widespread, due to its wrong Line_Break property value BA for "break after", whereas the NARROW NO-BREAK SPACE with its correct Line_Break property value GL for "glue" grew popular. On Windows 10, version 1607 (October 2017), the ever-in-Unicode THIN SPACE is not supported by Verdana's version 5.05, while the teenager NARROW NO-BREAK SPACE is there. It is not for support of Mongolian, since Verdana as per version 5.31 does not include Mongolian. The whole range U+2000..U+2011 is skipped, all those badly encoded spaces along with the useless duplicate HYPHEN (and the useful NON-BREAKING HYPHEN), but U+202F NNBSP is present. **Mongolian Baiti** follows a similar policy, not supporting the breaking fixed-width spaces, but of course U+202F.

For completeness, the following summary is based on an overview of all fonts shipped with Windows 10, version 1607. A rough count based on roman-style normal-weight results in 24 font families (out of 42) having non-Latin-1 spaces, among which (following Table 6-2. Unicode Space Characters, p. 264 of TUS v. 12.0) 13 support EN SPACE and EM SPACE, along with EN QUAD and EM QUAD except for 2 of them, and along with the 3-, 4- and 6-PER-EM except for 1 family. Both FIGURE and PUNCTUATION SPACEs are supported by 12 families (the same). Support for THIN SPACE aligns on the same basis, while HAIR SPACE is present in 5 more

families. NARROW NO-BREAK SPACE is in 11 families, 2 of which have neither of the preceding characters; on the other hand it is lacking in 2 families that have a complete set of first Unicode spaces (without OGHAM SPACE, added for version 3.0, and present in 1 font only), plus in 1 that skips the EN/EM QUADs.

Surprisingly or not, the breaker U+200B ZERO WIDTH SPACE is present in 17 families, whereas the break-preventers U+FEFF ZERO WIDTH NO-BREAK SPACE and U+2060 WORD JOINER are included only in 6 families for the one, and 4 families for the other, resulting in 9 families with either of them, and only 1 having them both (cf. the preceding *Proposal to focus break prevention design on end-user input*).

The supposed fault of Unicode was to not transparently place an order for the *Code Charts* layout tool at one of the best skilled member companies, as developing such a piece of software from scratch is too long a process for one single staffer. It supposedly resulted in opaque relationships aiming at injecting pre-existing code into the project, along with an offer to team up for customization within a useful delay, so that the first version of the *Code Charts* would meet the deadline. Due to a lack of funding and the inability to do fundraising at that stage of the Unicode project, the supposed counterpart was immaterial: a commitment to erode the usability of the upcoming Standard outside of publishing software such as WYSIWYG DTP programs. Messing with the whitespaces' Line_Break property values was then a part of the plan aiming at undermining the Unicode Standard, that the original sin is since then deeply embedded in. This proposal is part of an ongoing effort to free the Standard and restore it in its original purity and splendor.

Proposed actions

1. Add a clause to the stability policies, stipulating that decomposition mappings defined in contradiction with character identity may be canceled.
2. Cancel the decomposition mappings of U+2000 and U+2001.
3. Change the Line_Break property value of U+2002..U+2006 and U+2008..U+200A from BA to GL, and update TUS and UAX #14 accordingly.
4. In UAX #14, correct the note in the *IS: Infix Numeric Separator* section: "*Note:* NARROW NO-BREAK SPACE and THIN SPACE as well as PUNCTUATION SPACE, being used as a group separator in numbers, have the Line_Break property value GL."

References

UAX #14: *Unicode Line Breaking Algorithm* < https://unicode.org/reports/tr14/>

ANDRIES Patrick, *Unicode 5.0 en pratique : codage des caractères et internationalisation des logiciels et des documents*, Dunod, Paris, 2008

[1] http://unicode.org/mail-arch/unicode-ml/Archives-Old/UML023/0481.html

[2] https://books.google.fr/books?id=GgbWZNTRncsC&pg=PA144&lpg=PA144&dq=ESPACE+FINE+Patrick+Andries&source=bl&ots=27i00LQVxs&sig=ACfU3U00eXhmCX1rk3DiPIuqXAY39prJ4g&hl=fr&sa=X&ved=2ahUKEwiq4qG0uaLhAhWlxYUKHbZaBgwQ6AEwAXoECGIQAQ#v=onepage&q=ESPACE%20FINE%20Patrick%20Andries&f=false

[3] TPHL, “When would thin and hair spaces need to be breakable”, < http://www.typophile.com/node/124323>

[4] https://forums.macg.co/threads/typo-caracteres-capitales-accentues.25739/page-5#post-3151460

Acknowledgments

Thanks to everyone who directly or indirectly helped put this paper together.

Thanks to Google for Google Search.

Thanks to Microsoft for Word Online and OneDrive.