W3C CSS Working Group

Leading the World Wide Web to its full potential

Type:	Outgoing Liaison Statement
Title:	Liaison Statement from W3C CSS WG to ISO/IEC JTC 1/SC29 WG11
Status:	In accordance with the resolution taken at the 8 January CSS Working Group meeting
Date:	Monday, 13 January 2020
Document:	CSSWG-SC29WG11-2020-01-13

W3C CSS Working Group

Liaison Statement to SC29 WG11 on Open Font Format

In accordance with the Class C liaison between W3C and ISO/IEC JTC 1, W3C CSS Working Group, which develops the Cascading Style Sheet language, wishes to draw to the attention of ISO/IEC JTC 1/SC29 WG11, which develops the Open Font Format, their use of concepts and terminology drawn from Open Font Format in the CSS specifications, dating back to CSS Level 1 in 1996.

CSS WG also draws attention to the more recent explicit use of Open Font Format features in CSS WG specifications such as CSS Fonts, CSS Text, and CSS Writing Modes.

In the course of developing these specifications, CSS WG has over the years noted a few points of specification ambiguity, of possible conflict with other specifications, or of limitations in the support of particular writing systems.

In the spirit of open cooperation, CSS WG desires to enter into technical dialog with SC29 WG11 to discuss and resolve any identified ambiguity and to encourage the development of new features, or extension of existing features, to enable the use of CSS and Open Font Format with a greater range of writing systems. This cooperation is expected to benefit future development of both CSS and Open Font Format.

CSS WG is chartered to perform all technical work in public, to allow the greatest opportunity for expert commentary and review in the course of specification development. CSS WG therefore requests that, if SC29 WG11 is agreeable to such cooperation, that the work be conducted in a public and archived manner. CSS WG is using GitHub for specification development and issue tracking, which meets this requirement. CSS WG is aware of the SC29 WG11 ad-hoc group method, whose open membership and archived, public mailing list also meets this requirement.

An annex to this liaison statement gives a non-exhaustive list of issues CSS WG has noted to date.

W3C CSS Working Group

Annex A: Issues with the use of Open Font Format in CSS

1. Lack of font metrics for writing systems other than Western & CJK

Many scripts have top (or bottom) boundaries that do not correspond to any of the existing Open Font Format metrics, e.g. Hebrew or Thai, whose top edge matches neither the cap height nor the ex height, nor the ideographic ink face top. In order for CSS to be able to position and align text appropriately, CSS WG requests reliable metrics; and CSS WG requests them for all writing systems CSS WG plans to support.

Specifically, CSS WG requests:

A. The top and bottom lines to match for drop caps.

B. If there are other values needed for visual alignment or positioning, whatever metrics those might be. These would be used to control the visual distance between the text and other objects, as an alternative to metrics A.

2. Italic / Oblique Default Angle

CSS WG has received a comment, archived at https://lists.w3.org/Archives/Public/www-style/2018Jul/0021.html

> “As far as we can tell, there's no field in OpenType that would let us know what the font designer's preferred oblique angle would be. We're therefore using 14 as the default slant. If OpenType later adds a field for "default angle for obliques", then we can consider using that instead of 14.”

CSS WG notices however, that an italicAngle value exists in the **post** table; should this be used as the preferred oblique angle?

3. Compatibility problem with 'vert'

The 'vert' feature is for typesetting characters vertically. Certain characters need to be rotated when typesetting vertically, e.g. brackets, and this is OK. However there are a number of characters which could reasonably be typeset upright (e.g. equals sign =, or arrows), but in CJK fonts tend to have 'vert' alternates which rotate them sideways. This means it's impossible to typeset them upright.

However, the fonts cannot be updated to have upright 'vert' glyphs because legacy documents rely on their current behavior.

A discussion thread is archived at
https://lists.w3.org/Archives/Public/www-archive/2019Dec/

CSS WG as yet has no consensus on what actions need to be taken or which specifications will be affected, to improve new documents while not adversely affecting legacy ones.

4. Clarifications for 'vert'

CSS WG has determined that Open Font Format might need to clarify expectations for font designers when deciding what the 'vert' glyph of a given character should represent.

CSS WG notes that early drafts of Unicode UAX50 included codepoint-by-codepoint data on which characters should be drawn upright vs rotated in upright typesetting modes, but this extra data set was abandoned. See
https://www.unicode.org/reports/tr50/tr50-6.html#w1aac15b1 and
http://www.unicode.org/reports/tr50/tr50-6.Orientation.txt

CSS WG envisages guidelines such as the following:

In mixed mode (UTR50), Tu, Tr, and U characters are typeset upright using vert alternates; R characters are typeset sideways (rotated 90deg clockwise) with vrtr alternates.

Note that many typesetting systems can additionally control characters to explicitly typeset them upright or rotated.

Fonts interoperating with UTR50 implementations therefore

- MUST have vert alternates for Tu
- MUST have vert alternates for Tr
- MAY have vert alternates that are appropriate for upright typesetting for other characters as well
- SHOULD provide appropriate “rotated” glyphs as vert alternates for characters with SVO=R in this obsolete data file http://www.unicode.org/reports/tr50/tr50-6.Orientation.txt
- MAY have vrtr alternates for any characters (and might particularly want to for [typical reasons for vrtr, e.g. glyph alignment with vertical central axis, brush stroke variation, etc.])

5. Clarifications for Bopomofo

Some special positioning behavior, for tone marks in particular, is necessary for Bopomofo to be rendered correctly. CSS WG has had requests to make this special positioning part of ruby layout in CSS; but it is not a ruby feature, it is fundamental to the script and happens even in plaintext environments (such as writing in pencil on lined paper in school).

Thus, this seems more like a type of complex script rendering problem than a layout one, and it seems that support for it needs to be built into the fonts. CSS WG hopes that Open Font Format has sufficient technology to enable bopomofo rendering, and suggests that SC29 WG11 may wish to clarify technical expectations for bopomofo fonts, which historically have not supported such rendering.

6. Cursive elongation

Cursive writing in general, and in cursive writing systems especially, has a lot of facilities for justification when written by hand. Due to the technical difficulty, these facilities are rarely invoked in computer typesetting. CSS allows User Agents (Uas) to make use of such facilities under its 'auto' justification rules; however CSS WG is unaware of any UAs who have done so, as it would require support from the text rendering subsystem. If the Open Font Format community is interested in pursuing such possibilities, however, CSS can then enable it for the Web.

7. Fidelity of Font Metrics

This is less a problem with the Open Font Format spec per se, but perhaps the members of SC29 WG 11 can help: CSS is increasingly making use of font metrics, to allow typesetting to respond more closely to the desires of the font designer. Examples of this include making use of the superscript and subscript position / size metrics, the use of underline position and thickness metrics, various glyph measures such as cap height / ex height, and alternate baselines. One problem CSS WG runs into, and the reason why in some cases CSS has not made using font data the default approach, is that many fonts have *incorrect* metrics.

Whether the metrics are wrong because they were erroneously copied from another font and never updated, or because a tool puts essentially random default values into its tables when the author does not bother to set it themselves, CSS WG observes that the result of automatic layout with such metrics is bad.

In the past, when designing for print, the designer was able to adjust the results visually until everything lined up as intended, but on the Web this is not possible: the automatic output of the layout engine is the end result.

Perhaps a concerted effort by both SC29 WG11 and CSS WG can make the case to tool providers and font designers that these metrics matter, and help the Open Font Format ecosystem move towards high-fidelity metrics.

8. Required vs Optional Ligatures

This issue may also relate more to tooling or authoring advice, but perhaps a clarification to the Open Font Format specification would help. CSS WG has observed that ligatures that are required for correct text shaping need to be in rlig, not liga/hlig/clig.

This will avoid problems such as those documented at https://github.com/w3c/csswg-drafts/issues/2644

9. Overline Metrics

CSS Text Decoration Level 3 allows UAs to use font metrics for the positioning of underlines; Level 4 allows authors to even request this explicitly. However there is no equivalent metric for overlines.

If SC29 WG 11 were to add overline capabilities to Open Font FormatCSS WG could then reference them from the CSS specifications.