
i18n @ W3C
Richard Ishida

W3C Internationalization Initiative

W3C Internationalization Initiative

عربي

W3C's goal is a Web for All, regardless of language, script or culture. The Web community has made tremendous progress in internationalizing the Web over recent decades, but as Web penetration in language communities increases, as usage scenarios grow, and as new applications such as digital publishing emerge, there remains more to do.

For the Web to truly work for stakeholders all around the world, there must be a collaboration of language experts, Web site designers, developers, and vendors who are active in moving the Web forward. To ensure a rapid response to the growth of the Web, the W3C wants to marshal the resources of organizations and experts who care about these problems and enlist their help in strengthening internationalization support for the Web.

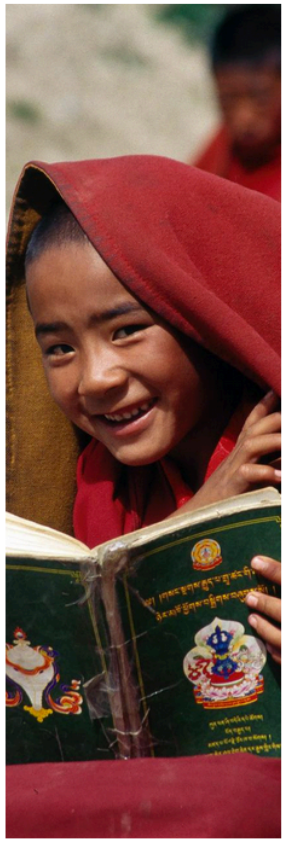
To accelerate progress in this area, the W3C is also looking to supplement the core funding it receives from W3C Member fees so that it can increase in-house resources dedicated to this work.

The internationalization initiative will provide participants and funding to address three main aspects of the internationalization continuum:

- **Language enablement** appeals most directly to stakeholders (e.g., governments, publishers, community groups, etc.) who utilize the language.
- **Developer support** appeals most directly to tech companies that are building the infrastructure for a global Web and supporting W3C standards groups.
- **Author support** appeals to people creating Web content in their own language, as well as to companies who build or localize Web sites in many languages.

The Web needs your help

Success in meeting these goals requires participation and funding from language, developer, and author communities, in order to expand the effort over and above what can be achieved with our core funding.



Language enablement

The W3C wants to ensure that local requirements for language support on the Web are identified and addressed. Text layout is an area of particular interest, and is concerned with things such as rules for line-breaking & justification, local approaches to expressing emphasis or decorating text, localizing counter styles, supporting bidirectional text in markup, initial-letter styling, hyphenation, page layout, and so on. These typographic conventions are often very different from the Western norm in languages that use writing systems such as Arabic, Devanagari, Thai, Mongolian, and so forth.

In Web pages and in digital publishing, needs can be addressed by improving W3C standards for rendering text (such as CSS, WebVTT, SVG, etc.), and markup (such as HTML). The goal here is to ensure that the Web supports the native typographic features that users around the world are used to, and enables users to interact with the Web in line with long-standing print traditions. The language matrix captures an overview of where work is needed.

To achieve this goal, the W3C needs to assess current support for the world's languages on the Web, identify gaps, prioritize them, develop requirements, and then take steps to close the gaps. To do so, it needs to establish a network of experts who can advise on language-related requirements, and increase resources available to facilitate the work in this area.

Developer support

A core focus for the W3C is to support

Advanced Publishing Laboratory

Monotype

THEPACIELLOGROUP

Alibaba Group

Web for All

- **Language enablement**

  understand where the gaps are for users of the global Web

- **Developer support**

  build standards & applications that support a global Web

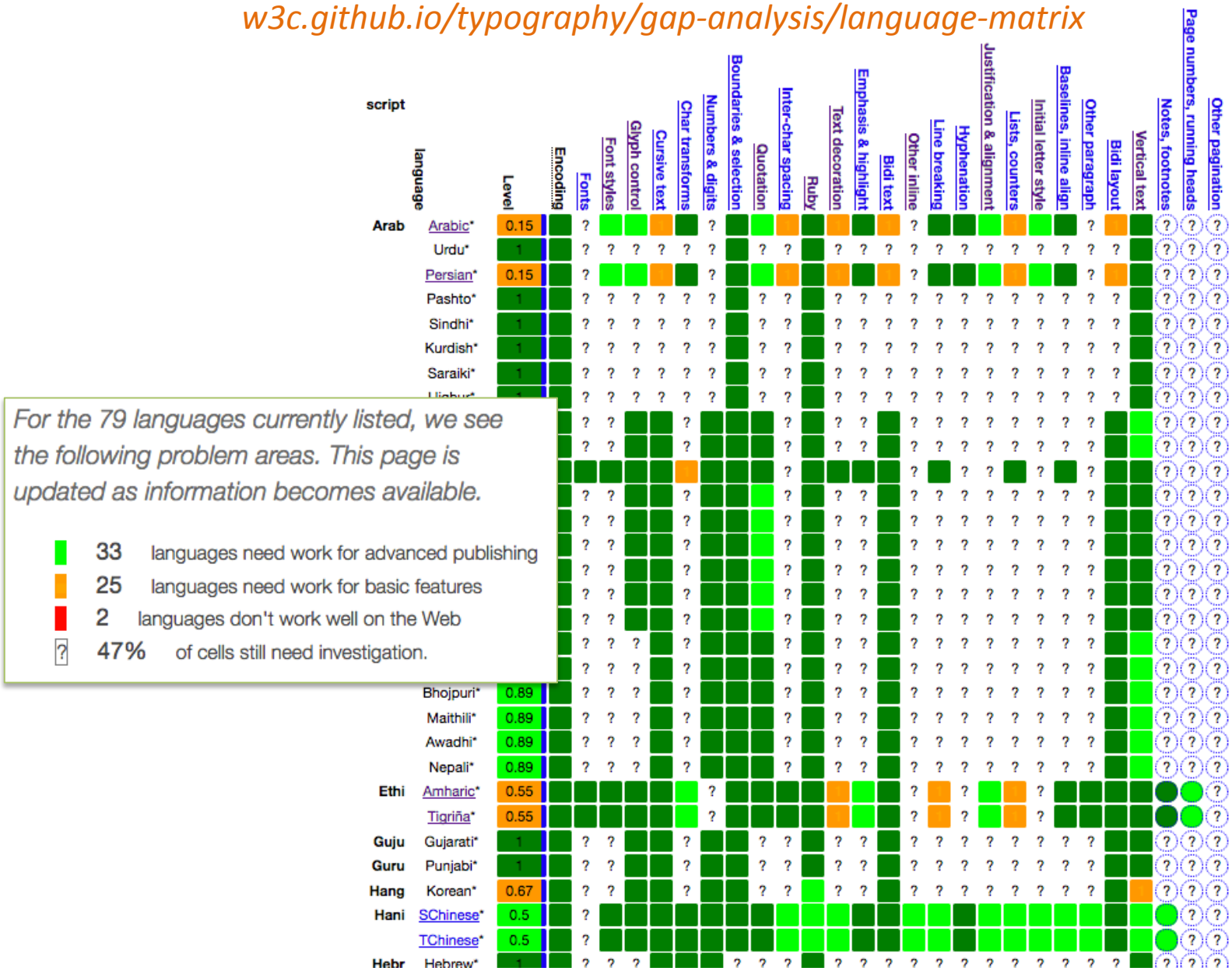
- **Author support**

  help people create content in their own language, or create content that will be localised

Language Enablement

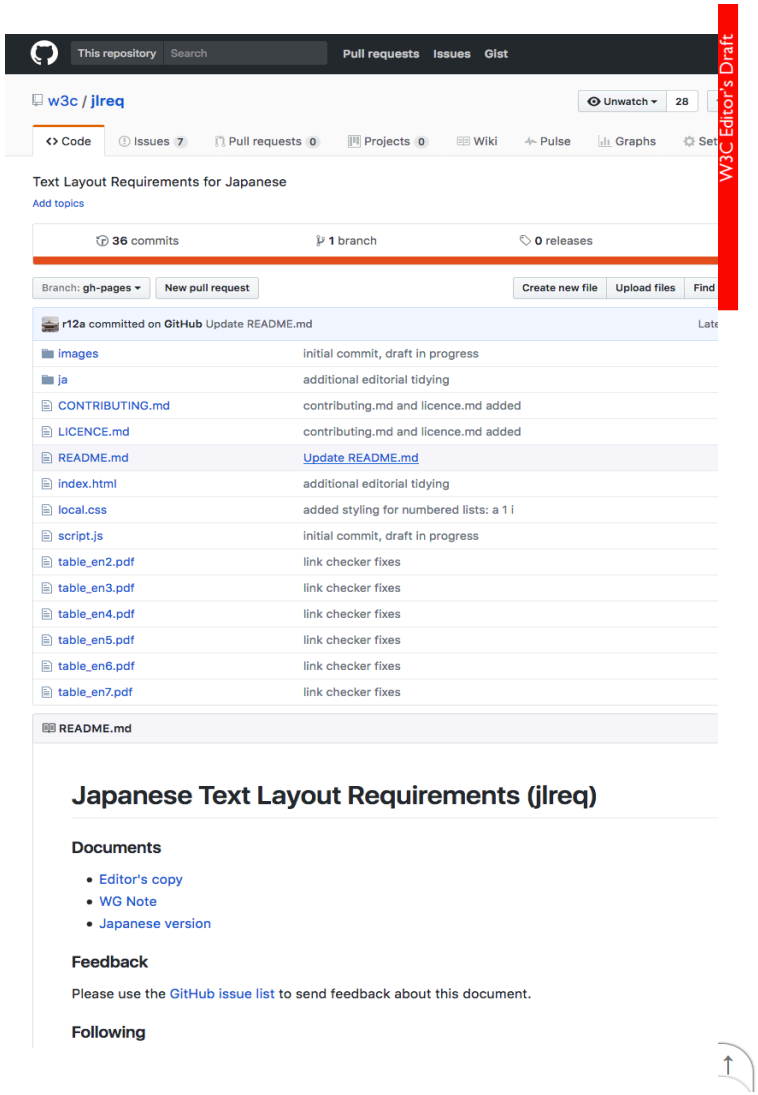
language matrix

w3c.github.io/typography/gap-analysis/language-matrix



phase 1 – jlreq

w3c.github.io/jlreq/



The following are the basic elements of a page format. Fig. 11 illustrates an example of a page format in vertical writing mode).

a. Trim size and binding side (vertically set Japanese documents are bound on the right-hand side, and horizontally set documents are bound on the left-hand side. See Fig. 12 .)

b. Principal text direction (vertical writing mode or horizontal writing mode).

c. Appearance of the kihon-hanmen and its position relative to the trim size.

d. Appearance of running heads and page numbers, and their positions relative to the trim size and kihon-hanmen.

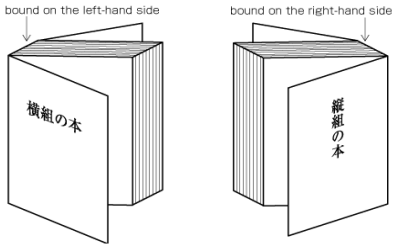


Fig. 12: Binding-side (bound on the right-hand side and bound on the left-hand side).

NOTE

Establishing a kihon-hanmen may be seen as defining not only a rectangular area on a page, but also within that area an underlying, logical grid, to guide the placement of such things as characters, headings, and illustrations. However, once a kihon-hanmen is established, there is no absolute requirement to align characters with the grid, especially when setting characters inside a line. The only factors that influence the placement of characters are strong gravitational forces that (i) attract the first and last characters on a line to align with the border of the kihon-hanmen, and (ii) attract each line position to the line positions on which the kihon-hanmen is based.

It may help in understanding the basic concepts of Japanese layout and kihon-hanmen to think in terms of a slit-based model, rather than a grid-based model. Each slit is the full length of the lines on which the kihon-hanmen is based.

2.2.4 Elements of Kihon-hanmen

The kihon-hanmen is the hanmen style designed as the basis of a book. The following are the basic elements of the kihon-hanmen (see Fig. 13).

a. Character size and typeface name

b. Text direction (vertical writing mode or horizontal writing mode)

klreq

clreq

W3C Editor's Draft

ko en

2.3 'Letter Face Position in Character Frame' Standard | '글자틀 내 글자면 위치' 표준

Standardization of 'letter face position in character frame' of fixed width Hangul fonts improves the compatibility of the space between Hangul font characters. (The relation between each side's spaces remains even when the Hangul font is changed. It prevents a paragraph's left outline being scattered when the opening quotation mark or parenthesis at the line head has an unexpected space).

고정폭 한글 폰트의 '글자틀 내 글자면 위치'를 표준화는 한글 폰트 간의 글자사이 비율 호환성을 향상하기 위함이다(한글 폰트를 변경해도 문장부호의 좌우 여백 관계가 그대로 유지되도록 한다. 글줄 시작에 위치한 열기 괄호·따옴표에 의도치 않은 여백이 생겨 단락의 왼쪽 외곽선이 흐트러지는 경우를 방지한다).

Fig. 3: Letter face position in the character frame | 글자틀 내 글자면 위치

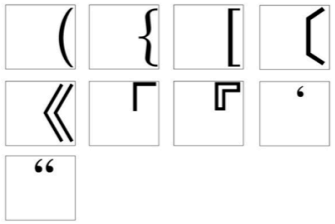
ISSUE 3

See https://github.com/w3c/klreq/issues/10

2.3.1 Arrangement of 'Letter Face Position in Character Frame' for Full Width Parentheses | 전각 괄호의 '글자틀 내 글자면 위치' 지정

In horizontal writing, the letter face of a full width opening parenthesis is placed on the right end of the character frame, and the left space is considered a user controlled area. In vertical writing, the letter face of a full width opening parenthesis is placed on the bottom end of character frame, and the space is considered a user controlled area.

가로짜기의 경우 전각 열기 괄호의 글자면은 글자틀의 오른쪽(가로짜기)·아랫쪽(세로짜기) 끝에 두고, 왼쪽 여백은 사용자 조정 영역으로 간주한다. 세로짜기의 경우 전각 열기 괄호의 글자면은 글자틀의 아랫쪽 끝에 두고, 여백은 사용자 조정 영역으로 간주한다.



W3C Editor's Draft

zh-hant zh-hans en

1. **Bilingual Annotations.**

中外文对照

中外文對照

Bilingual annotations aim to provide a Chinese translation of text in foreign languages or acronyms, or to offer the original text for words that have been translated into Chinese. This is mainly used for proper nouns, titles or those terms whose concepts are difficult to convey after translation. It is commonly found in translated works, mainly in light novels.

为外来语、首字母缩略词标注其中译，或对翻译名词标注其原文，多见于专有名词、作品名及译后概念较难传达的词汇。常见于译作，尤以轻小说为主。

為外來語、首字母縮略詞標注其中譯，或對翻譯名詞標注其原文，多見於專有名詞、作品名及譯後概念較難傳達的詞彙。常见于译作，尤以轻小说為主。

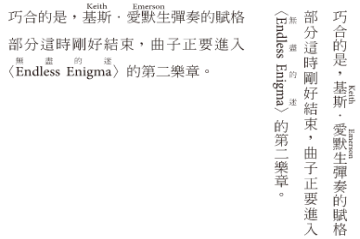


Fig. 7: An example of positioning for billingual annotations. | 中外文對照行間注的排版示例。 | 中外文对照行间注的排版示例。

2. **Interlinear Comments.**

行间批语

行間批語

Interlinear comments are ways to annotate the meaning of text fragments or a single word, and are so named for their interlinear positioning. They usually lie in the interlinear space and co-exist with the body text. Compared to other annotation methods, i.e. headnotes or footnotes, interlinear comments are more compact and stick better to the body. These kinds of comments are often found in ancient books, such as Rouge Inkstone, an early commentary of the novel Dream of the Red Chamber.

行间批语是为一段文本片段或单个词汇标注解释的排版方式，因共存于正文文本，显示于其行间而得名。行间批语比眉批、脚注等注释方法更具紧凑、依附性，多见于如《红楼梦》早期抄本的脂砚斋批语等古籍。

行間批語是為一段文本片段或單個詞彙標注解釋的排版方式，因共存於正文文本，顯示於其行間而得名。行間批語較眉批、腳注等注釋方法更具緊湊、依附性，多見於如《紅樓夢》早期抄本的脂硯齋批語等古籍。

alreq

elreq

2.6.3 Arabic Script and Typography

Arabic script has some characteristics that are challenging for typographers and font designers. Examples below show some characteristics worth to be considered carefully. How could typography, which came late to the Arabic world, then follow the tradition of the many authors and artists who manually shaped the Arabic writing over decades? even in it's simplest *Naskh* style?

1. **Multi-level baselines**

   Letters may join through a finely inclined line

   يستبشر

   or two, square-ended lines

   يستمع

   Multilevel baselines don't occur in all fonts. The above examples use the Arabic Typesetting font. Compare those examples to to more typical fonts:

   يستبشر
   يستمع

2. **Multi-context joining**

   Rendering of letters depends not only on their place in the word (initial, medial, final) but also on their neighboring letters, i.e. the letter they join with. Each letter has a different appearance in each combination.

   نم نمل نجر نسيم نبات

   Fig. 18: Initial letter noon, showing many different forms.

   Fonts don't always comply with or respect this kind of 'tuning'. To do so, fonts need many glyphs in order to adapt to each context. In more modern typefaces some of these connections are implemented by ligatures, but ligatures can't capture or cover all joining behavior.

   In the two left most words, the initial noon differs in that one raises a kind of stroke. This property of raising a stroke is common for a number of letters (beh, teh, noon, theh) which are taller than their connected letters in order to be distinguished in some contexts, such as بسل vs. سبل, or to resolve ambiguity. See also the section about teeth letters below.

3. **Words as groups of letters**

   A word shape is not (only) a "horizontal" connections of letters, but of groups of letters (syntagmes).

   Example two words in some nice Naskh font.

3.3.2 Justification When ETHIOPIC WORDSPACE is the Word Delimiter

Since the arrival of the printing press in Ethiopia in 1863 (Pankhurst, 1998), full justification of Ethiopic has been a common typesetting practice in Ethiopian, and later Eritrean, publishing houses. Earlier, Ethiopic justification rules are a feature of Hiob Ludolf's Historia Æthiopica, which is noted as the first use of movable type for Ethiopic script (Ludolf, 1681). Prior to letterpress typography, calligraphic manuscripts rendered on parchment also featured full, or approximately full, justification. Though the latter likely reflects the scribe's desire not to waste a millimeter of available lateral writing space.

The placement of Ethiopic wordspace presents a complication to the justification of Ethiopic text. Two placement styles developed in typeset literature which will be referred to here as "word bound" and "centered" styles. Additionally, the word spacing following an Ethiopic fullstop may (or may not) be governed by a special rule and in combination with the two wordspace spacing styles. These spacing rules are discussed in the following sections.

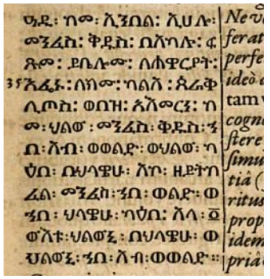
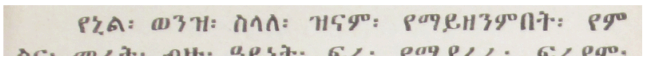


Fig. 25: Ethiopic Justification in Historia Æthiopica (Ludolf, 1681)

3.3.3 Justification with Word Bound Wordspace and Punctuation

In keeping with line justification for Latin script, the non-printed or "blank space" (space and gaps) between words is treated as stretchable. The width of the space symbol itself will be elongated to some aesthetic width value that may vary from space symbol to space symbol across a printed line. In Ethiopic justification, the blank space between the Ethiopic word separator and the words it separates is likewise allowed to stretch. This stretching of blank space may be either symmetrical ("centered") or asymmetrical but in the latter case space stretching is always between the right side of the separator and the following word –referred to here as "word bound".

In "word bound" justification the word separator, which may be either a punctuation symbol or U+1361 ETHIOPIC WORDSPACE [፡], appears to adhere to the word to the left as if it were its final character. Figures Fig. 26: Ethiopic justification in word bound style (Erikson, 1921 (1913 EC)) and Fig. 25: Ethiopic Justification in *Historia Æthiopica* (Ludolf, 1681) both illustrate the word bound style.



ilreq/IIP

tlreq

W3C Editor's Draft

3.3 Initial letter styling

Drop initial is a typographic effect emphasizing the initial letter(s) of a block element with a presentation similar to a 'floated' element.

3.3.1 Selecting initial letters

Initial letters in Indic scripts must be selected on the basis of orthographic syllables, rather than individual letter forms (see an example at the end of section 3, Text segmentation).A detailed definition of Indic syllables can be found in section 2, Indic Syllable boundaries.In Indian languages the size of the Initial Letter is determined by the number of the lines between top line of the syllable and lowest bit in the orthographic Indic syllable cluster where subjoined consonant and other diacritics appears.

3.3.2 Typical drop initial usage in Indic scripts

Most of the Indic drop initial letters in magazines and newspapers use 2 to 4 line drops. Some examples are shown below.

Fig. 4: Examples of Indic Initial letters

The Sunken and raised Initial letter are not preffered in Indian languages.In examples of this kind, reference points on the drop cap must align precisely with reference points in the text. In Indic scripts the top reference point is the hanging base line for those scripts that have one, and the bottom alignment point is the text after-edge.

Initial letter wrap property is not applicable for Indian languages.No contour-filling is required in Indian languages.

Alignment of the top line of the non-highlighted characters is at the top of the thicker top line of the initial letter is commonly used in India.In some examples top lines of the initial letter and the following letters don't touch. This is due to variable technology/formats used by the publishers. It is preferred that both the top lines of Initial letter and neighbouring text should touch.Here are some additional examples of initial highlighted letter and drop letter based on the Indic syllable definition.

W3C Editor's Draft

zh en

2.3 Tibetan Syllables | 藏文音节

Word boundaries within a section are not indicated, only 'syllables', known as tsheg-bar /tsek bar/. Syllable boundaries are usually separated by the tsek character, U+0F0B TIBETAN MARK INTER-SYLLABIC TSHEG ་.

The pronunciation of Tibetan words is typically much simpler than the orthography, which involves patterns of consonants. These patterns reduce ambiguity and can affect pronunciation and tone.

The following diagram shows characters in all of the syllabic positions, and lists the characters that can appear in each of the non-root locations. The two-syllable word in the example is འཁྲིམས་སྟོན་ 'grems-ston /d̪ɛm-ton/ (exhibition).

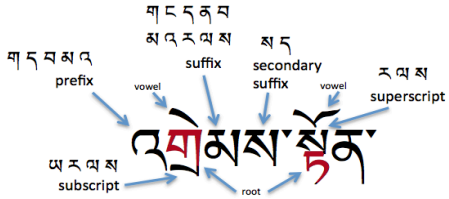


Fig. 6: Syllable composition in Tibetan

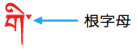
2.3.1 Structural Rules | 结构规则

The primary consonant in a syllable is called the root consonant (or radical) (མིང་གཞི་), and the other consonants in the syllable (normally up to 6 in total) annotate or modify it. The following rules help identify the root:

要分析藏文结构必须先得找出根字母，然后其他的部分根据结构规则就能找到。根字母的判断方法如下：

1. A consonant with a vowel is always the root, unless it is the phrase connector ལ་ , and letters with superscripts or subscripts are root consonants.

   一个辅音上有元音字母，那就是根字母，除非是 ལ་ 如下上面是元音字母下面是辅音字母，在此中是根字母。

2. In a 2-consonant syllable with no vowel, the first consonant is always the root.

   一个辅音上有上标字或者下标字那么这个辅音字母也是根字母。

breaking news: mlreq

w3c.github.io/mlreq/

W3C Editor's Draft

Figure 2: Alignment relationship between punctuation and Mongolian script text.

§ 3.4.2 Display rules for Mongolian space

Mongolian case supplementary component display rules and Mongolian space (0x202f) processing.

The inputting of Mongolian case supplementary component into the computer requires the Mongolian space 0x202F first, and then the corresponding Mongolian script. For example, the internal code of the word " ᠮᠣᠩᠭᠣᠯ " is: 0x202F 0x1824 0x1828, among which 0x202F is called Mongolian space, inputted before the Mongolian case supplementary component .

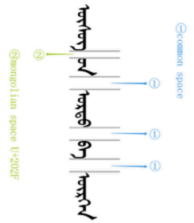
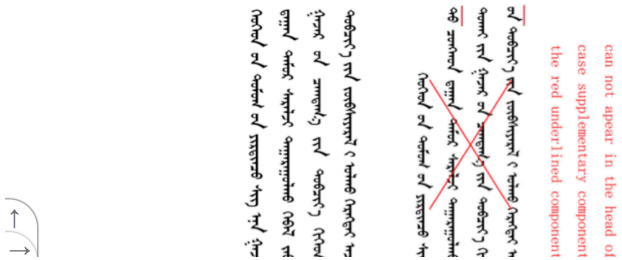


Figure 3: The difference between the common space and Mongolian space.

Mongolian space (0x202F) and case supplementary component cannot appear at the beginning of a line, for example, Figure 4 is the correct processing method and Figure 5 is the wrong processing method.



phase 2 – gap analysis

W3C Editor's Draft

TABLE OF CONTENTS

Japanese Gap Analysis

W3C Editor's Draft 12 June 2018

This version:
https://w3c.github.io/alreq/gap-analysis/
Latest published version:
https://www.w3.org/TR/alreq-gap/
Latest editor's draft:
https://w3c.github.io/alreq/gap-analysis/
Bug tracker:
File a bug (open bugs)
Editor:
Richard Ishida (W3C)
Github:
repository

Abstract

This document describes and prioritises gaps for the support of Japanese on the Web and in eBooks. In particular, it is concerned with text layout. It checks that needed features are supported in W3C specifications, in particular HTML and CSS and those relating to digital publications. It also checks whether the features have been implemented in browsers and ereaders. **This is a preliminary analysis.**

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

This document describes and prioritises gaps for the support of Japanese on the Web and in eBooks. In particular, it is concerned with text layout. It checks that needed features are supported in W3C specifications, in particular HTML and CSS and those relating to digital publications. It also checks whether the features have been implemented in browsers and ereaders. This document complements the document Text Layout Requirements for the Japanese Script, which describes the requirements for areas where gaps appear. It is linked to from the language matrix that tracks Web support for many languages.

This document is currently an individual contribution.

NOTE

Sending comments on this document

If you wish to make comments regarding this document, please raise them as github issues. Only send comments by email if you are unable to raise issues on github (see links below). All comments are welcome.

To make it easier to track comments, please raise separate issues or emails for each comment,

gap analysis

2.11.2 Double-sided ruby

In order to get double-sided ruby to appear on either side of the base text in HTML, you need the CSS `ruby-position` property to work with the value `under`. It currently works per the standard only with Firefox. It works with the proprietary `-webkit` prefix in Chrome and Safari, but doesn't work with Edge. See test results.

Double-sided ruby doesn't appear often in Japanese text, but it is used, and should be available to content authors.

2.11.3 Ruby aligment and other styling

For background information read Aligning annotations and bases.

Fine control in HTML over the placement of ruby annotations relative to the base require the availability of the CSS property `ruby-align`. This only works in Firefox, and sometimes in Edge with proprietary syntax. See results for major browsers.

The impact of this is probably more important for advanced typographic layout.

Needs work for basic styling support.

2.12 Emphasis & highlights

Bold and italic are not always appropriate for expressing emphasis, and some scripts have their o… ways of doing it, that are not in the Western tradition at all. Does this script require support for emp… or highlighting text that cannot be achieved currently? See available information or check for current… needed data.

For requirements, see JLReq Composition of Emphasis Dots.

2.12.1 Boten marks

There are tests for boten mark support that show that they are not supported by Chrome 63.0.3239.132 or Edge, but they are supported by Firefox 58.0 and Safari 11.0.3.

The expected behaviour for basic support of boten marks is specified in CSS.

Given that there are alternative ways of showing emphasis, and although there are already two implementations, I mark this as an advanced need.

Additional requirements arose while the text was being written for CSS. They relate to text that is annotated with ruby at the same time as annotation marks. The conclusions were:

1. Emphasis marks go outside ruby.
2. If ruby only spans part of the emphasised word, emphasis marks stay as close as possible to the base.
3. Where a ruby annotation is hidden or empty, the emphasis marks should continue at the same height.

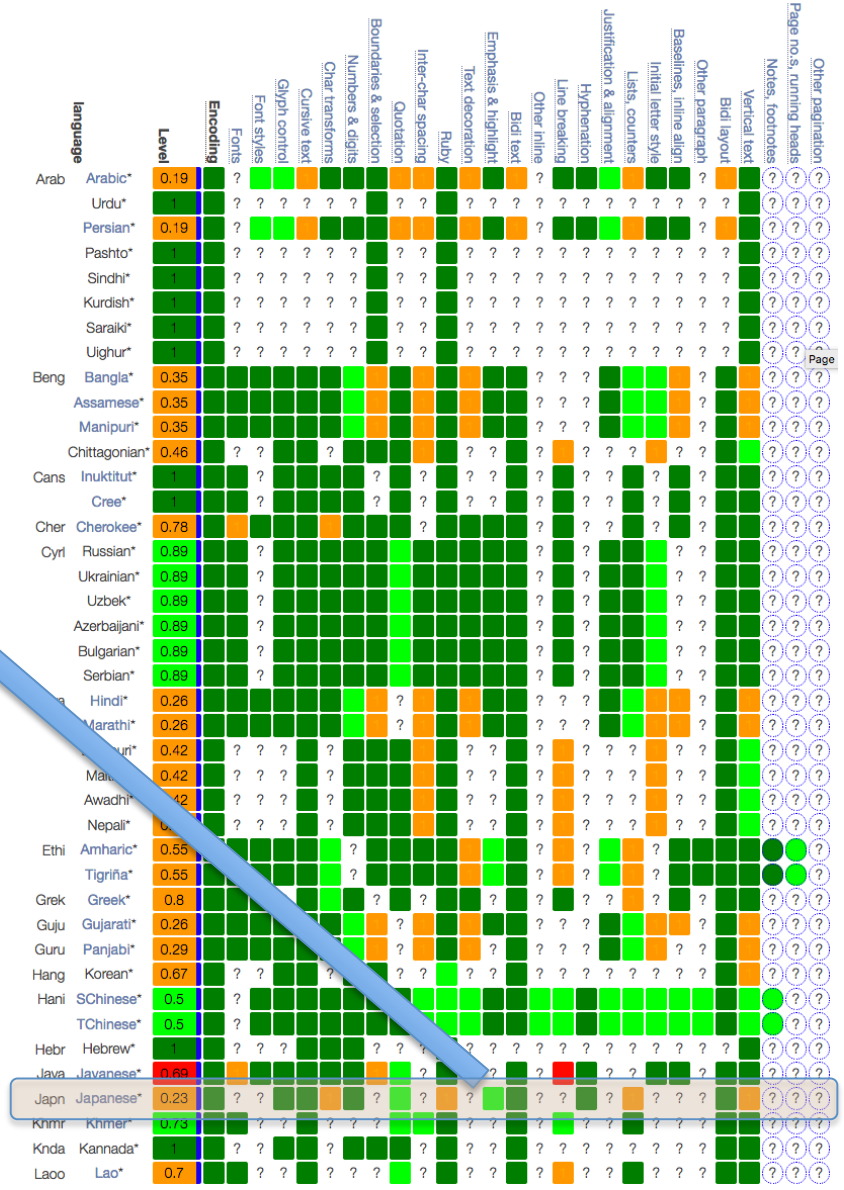
JLReq also requires that emphasis marks not appear over commas, full stops and brackets.

There are currently no tests for these behaviours, and no requirements in jlreq for the former (with ruby). **Need to take a closer look.**

JLReq calls out the usual practise of using sesame shaped boten for vertical text, and bullets for horizontal. Controls for this exist in CSS, and tests show that it is supported by the browsers that support boten.

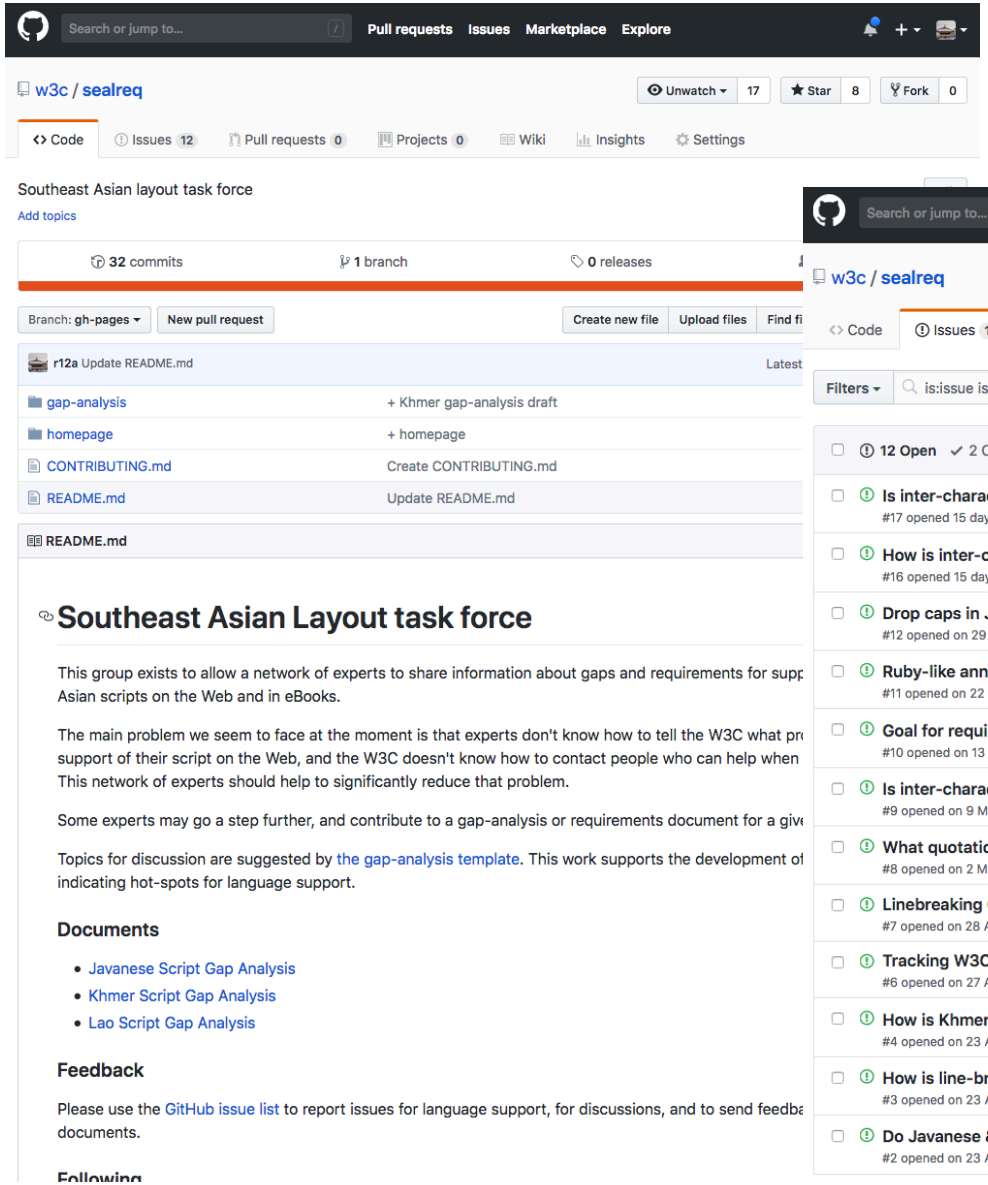
The default side for boten is to the right of vertical lines and above horizontal lines, according to JLReq. CSS controls allow this positioning to be set by the author, and in fact it is the default in both browsers that support boten for text where a lang tag identifies it as Japanese.

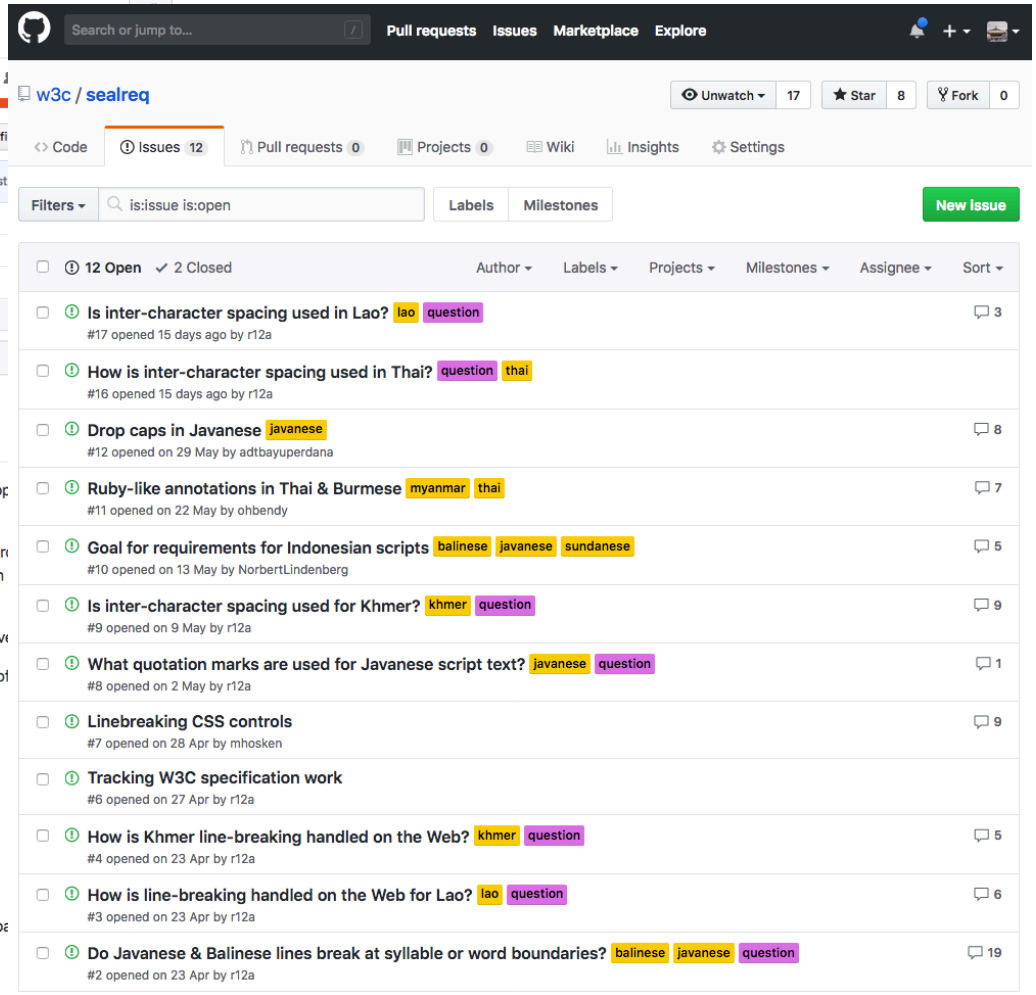
Needs work for advanced level support.



phase 3 – networks

w3c.github.io/sealreq/





notifications

w3c/sealreq (+0/-0/💬3)

2 issues received 3 new comments:

- #17 Is inter-character spacing used in Lao? (2 by laonux, jclark) lao question
- #2 Do Javanese & Balinese lines break at syllable or word boundaries? (1 by adtbayuperdana) balinese javanese question

w3c/csswg-drafts (+1/-0/💬9)

1 issues created:

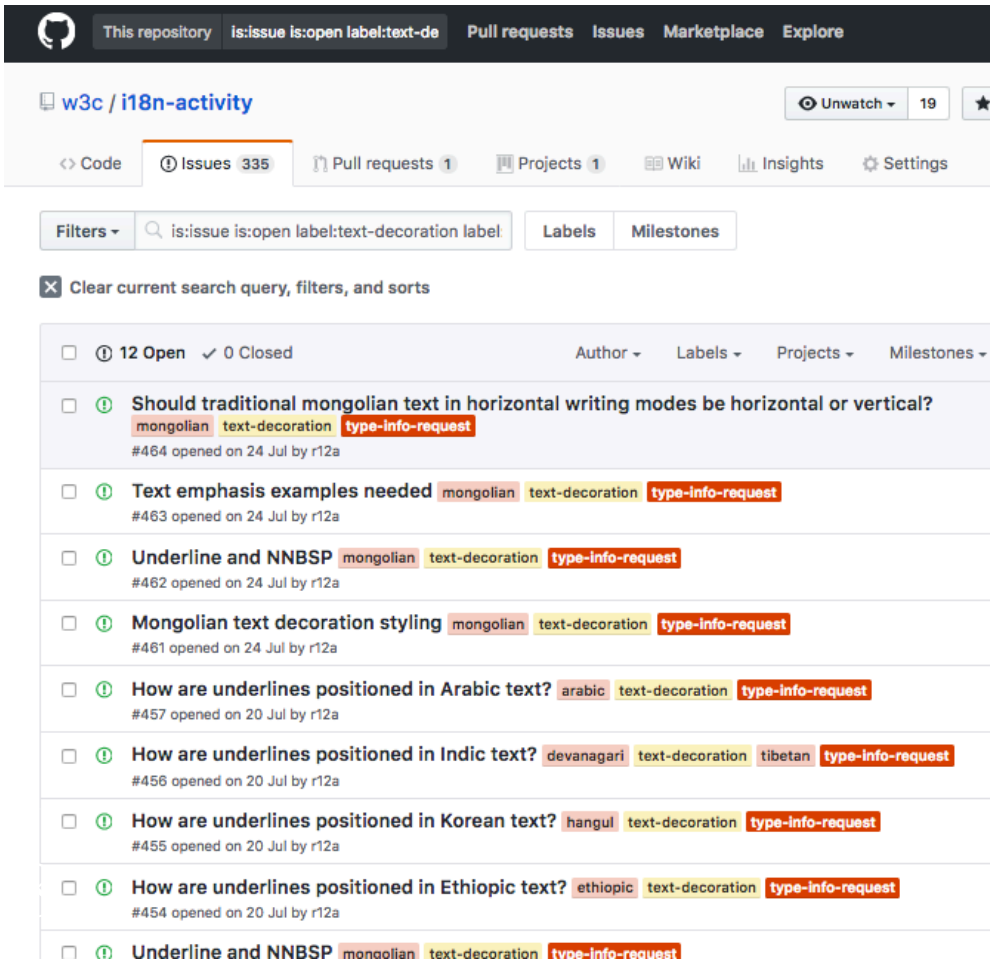
- #2975 [css-text-4] hyphenate-character doesn't just put hyphen at end of line (by r12a) i18n-sealreq

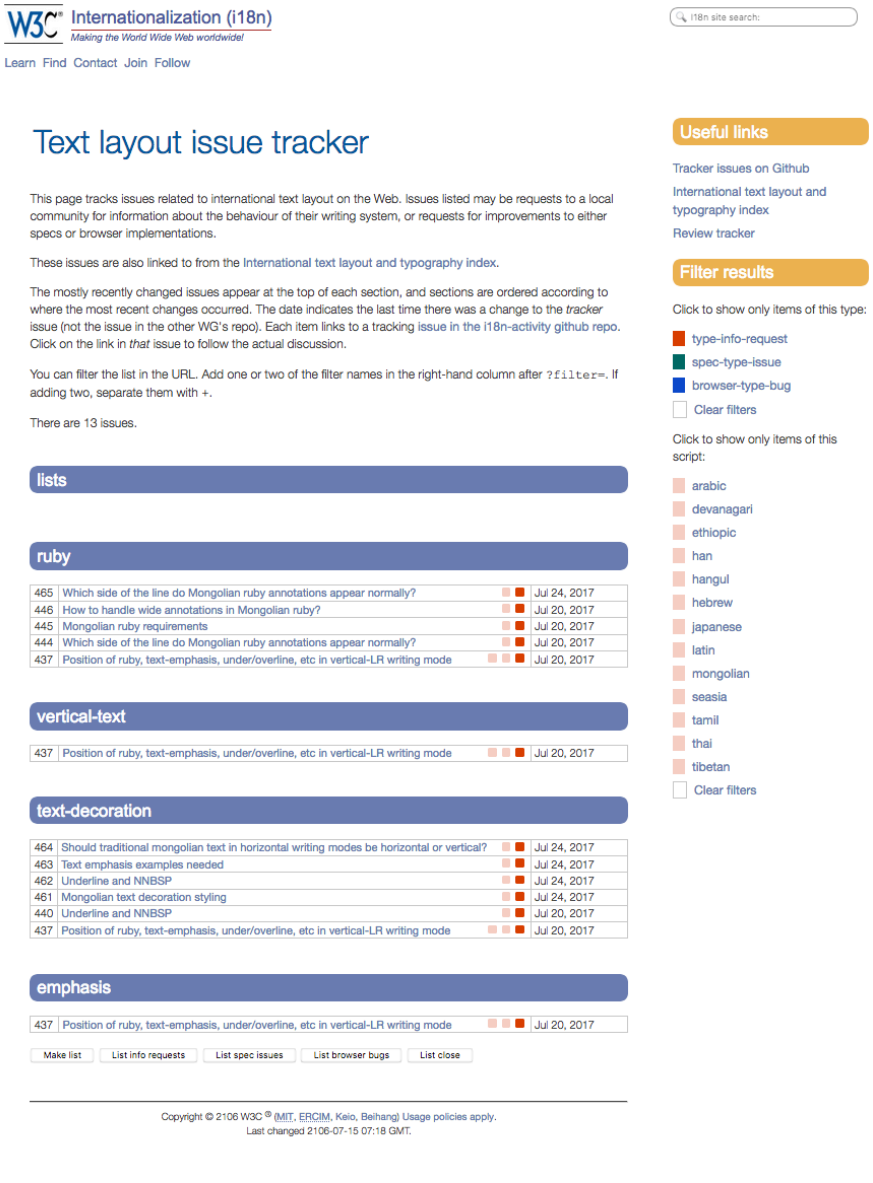
3 issues received 9 new comments:

- #2975 [css-text-4] hyphenate-character doesn't just put hyphen at end of line (7 by r12a, Crissov, kojiishi, litherum) i18n-tracking
- #2976 [css-text-4] Dealing with unusual line-break/hyphenation rules (1 by r12a) i18n-sealreq i18n-tlreq i18n-tracking
- #2809 [css-text-4] hyphenate-character doesn't accept just a character (1 by r12a) css-text-4 i18n-tracking

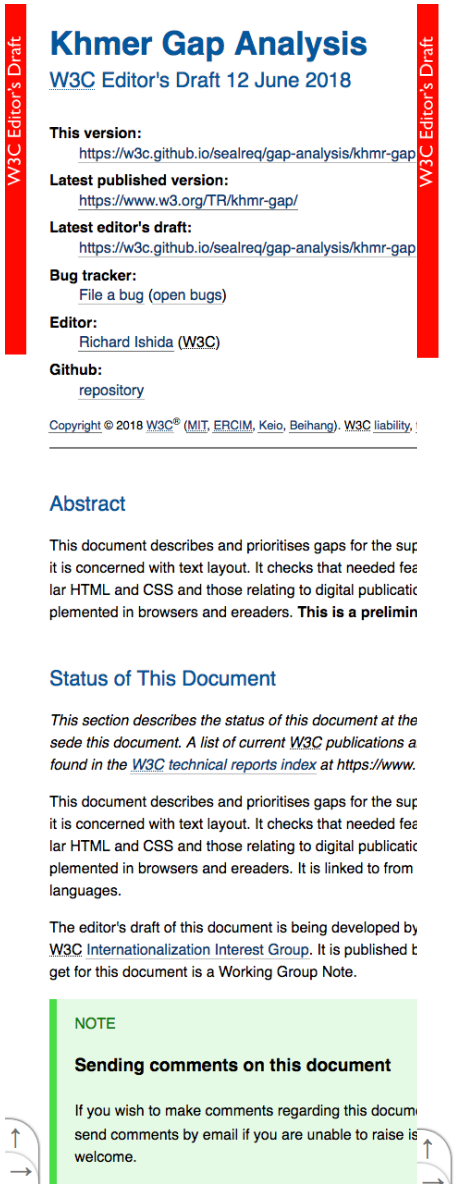
issue tracker

w3c.github.io/i18n-activity/textlayout/





sealreq docs



W3C Editor's Draft

Khmer Gap Analysis

W3C Editor's Draft 12 June 2018

This version:
https://w3c.github.io/sealreq/gap-analysis/khmr-gap

Latest published version:
https://www.w3.org/TR/khmr-gap/

Latest editor's draft:
https://w3c.github.io/sealreq/gap-analysis/khmr-gap

Bug tracker:
File a bug (open bugs)

Editor:
Richard Ishida (W3C)

Github:
repository

Abstract

This document describes and prioritises gaps for the sup
it is concerned with text layout. It checks that needed fea
lar HTML and CSS and those relating to digital publicatio
plemented in browsers and ereaders. **This is a prelimin**

Status of This Document

*This section describes the status of this document at the
sede this document. A list of current W3C publications a
found in the W3C technical reports index at https://www.*

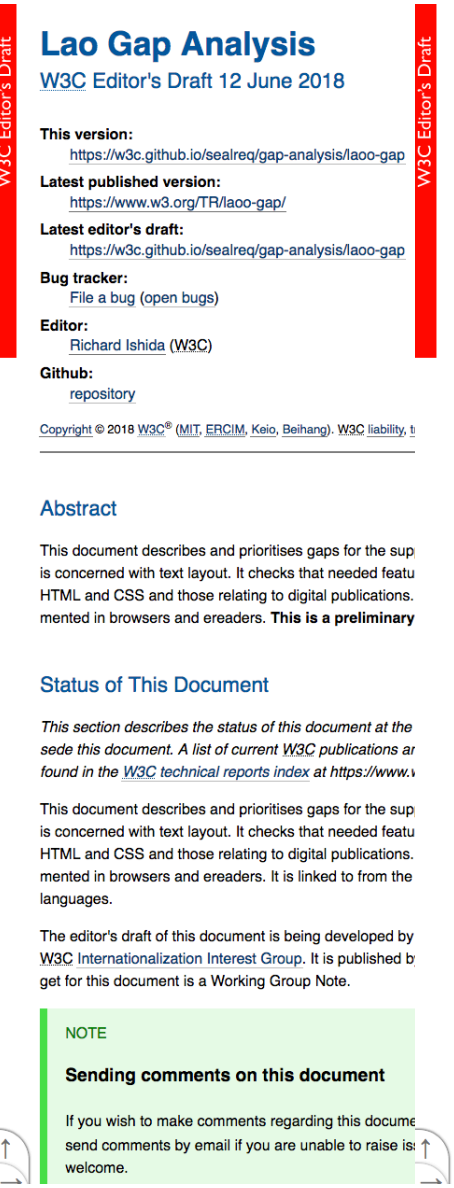
This document describes and prioritises gaps for the sup
it is concerned with text layout. It checks that needed fea
lar HTML and CSS and those relating to digital publicatio
plemented in browsers and ereaders. It is linked to from
languages.

The editor's draft of this document is being developed by
W3C Internationalization Interest Group. It is published b
get for this document is a Working Group Note.

NOTE

Sending comments on this document

If you wish to make comments regarding this docum
send comments by email if you are unable to raise is
welcome.



W3C Editor's Draft

Lao Gap Analysis

W3C Editor's Draft 12 June 2018

This version:
https://w3c.github.io/sealreq/gap-analysis/laoo-gap

Latest published version:
https://www.w3.org/TR/laoo-gap/

Latest editor's draft:
https://w3c.github.io/sealreq/gap-analysis/laoo-gap

Bug tracker:
File a bug (open bugs)

Editor:
Richard Ishida (W3C)

Github:
repository

Abstract

This document describes and prioritises gaps for the sup
is concerned with text layout. It checks that needed featu
HTML and CSS and those relating to digital publications.
mented in browsers and ereaders. **This is a preliminary**

Status of This Document

*This section describes the status of this document at the
sede this document. A list of current W3C publications ar
found in the W3C technical reports index at https://www.*

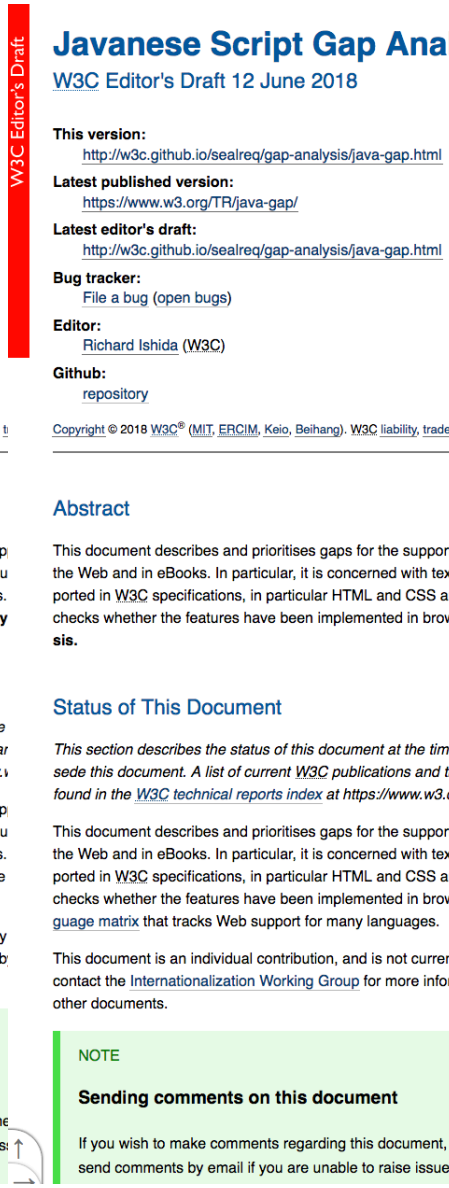
This document describes and prioritises gaps for the sup
is concerned with text layout. It checks that needed featu
HTML and CSS and those relating to digital publications.
mented in browsers and ereaders. It is linked to from the
languages.

The editor's draft of this document is being developed by
W3C Internationalization Interest Group. It is published b
get for this document is a Working Group Note.

NOTE

Sending comments on this document

If you wish to make comments regarding this docume
send comments by email if you are unable to raise iss
welcome.



W3C Editor's Draft

Javanese Script Gap Analysis

W3C Editor's Draft 12 June 2018

This version:
http://w3c.github.io/sealreq/gap-analysis/java-gap.html

Latest published version:
https://www.w3.org/TR/java-gap/

Latest editor's draft:
http://w3c.github.io/sealreq/gap-analysis/java-gap.html

Bug tracker:
File a bug (open bugs)

Editor:
Richard Ishida (W3C)

Github:
repository

Abstract

This document describes and prioritises gaps for the support of Javanese written with the Javanese script on the Web and in eBooks. In particular, it is concerned with text layout. It checks that needed features are supported in W3C specifications, in particular HTML and CSS and those relating to digital publications. It also checks whether the features have been implemented in browsers and ereaders. **This is a preliminary analysis.**

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

This document describes and prioritises gaps for the support of Javanese written with the Javanese script on the Web and in eBooks. In particular, it is concerned with text layout. It checks that needed features are supported in W3C specifications, in particular HTML and CSS and those relating to digital publications. It also checks whether the features have been implemented in browsers and ereaders. It is linked to from the language matrix that tracks Web support for many languages.

This document is an individual contribution, and is not currently a work item in any group, however, you can contact the Internationalization Working Group for more information. We welcome contributions to this and/or other documents.

NOTE

Sending comments on this document

If you wish to make comments regarding this document, please raise them as github issues. Only send comments by email if you are unable to raise issues on github (see links below). All comments are

text layout index

w3c.github.io/typography/

W3C Editor's Draft

TABLE OF CONTENTS

International text layout and typography index

W3C Editor's Draft 03 August 2018

This version:
https://w3c.github.io/typography/
Latest published version:
https://www.w3.org/TR/typography/
Latest editor's draft:
https://w3c.github.io/typography/
Bug tracker:
File a bug (open bugs)
Editor:
Richard Ishida (W3C)
Github:
repository

Abstract

This document points browser implementers and specification de
port typographic features of scripts or writing systems from arou
formation in specifications, to tests, and to useful articles and pap
to from time to time.

Status of This Document

*This section describes the status of this document at the time of i
persede this document. A list of current W3C publications and the
be found in the W3C technical reports index at https://www.w3.org*

The information in this document helps to link users and develope
pographic needs around the world. It is expected that this docume
terial becomes available or comes to our attention.

NOTE

Sending comments on this document

If you wish to make comments regarding this document, pleas
send comments by email if you are unable to raise issues on
are welcome.

To make it easier to track comments, please raise separate is
point to the section you are commenting on using a URL for t

This document was published by the Internationalization Working

Comments regarding this document are welcome. Please send th
chives).

Publication as an Editor's Draft does not imply endorsement by th
ment and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document was produced by a group operating under the W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual

2.11 Ruby annotation

Ruby is used for phonetic and semantic annotations of East Asian text, including furigana, pinyin and zhuyin fuhao systems. In addition to positioning annotations along the correct side of the base text, there are many fine adjustments of the annotation and base text to support.

Requirements

- *Chinese Layout Requirements*: Interlinear annotations
- (Chinese) *Bopomofo on the Web*
- (Chinese) *The Manual of the Phonetic Symbols of the Mandarin Script* english chinese
- *Japanese Layout Requirements*: Ruby and Emphasis Dots • Positioning of Jukugo-ruby
- (Japanese) *Use Cases & Exploratory Approaches for Ruby Markup*
- *Implementing Japanese Subtitles on Netflix*: *Rubies*

GitHub resources

- Requests for information
- Spec issues
- Browser bugs
- Type samples

Spec links

- *HTML5*: The ruby element
- *CSS3 Ruby*

Tests

- *HTML5*, the ruby element and its children

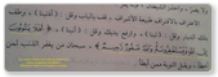
type samples repo

w3c.github.io/type-samples/

Type samples

This page lists pictures in the type-samples github repository. There are 62 items.

justification

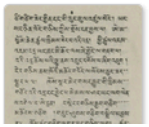


arab ar book justification
justification Two justification mechanisms. Kashida applied to a span of text inside a paragraph justified by inter-word spaces.

hebr he book justification
justification Stretched characters in Hebrew.

arab ar newsprint justification
justification Justification using simple baseline extensions.

arab ar signage justification text-decoration
text-decoration Colour change between linked cursive characters.
justification Arabic and latin text stretched to same size.



tibt bo book justification
justification Line end padding with tsek marks.

Select a feature

All
abbreviation (1)
bidi-text (1)
emphasis (3)
font-style (1)
fonts (4)
glyphs-diacritics (4)
hyphenation (2)
initial-letter (5)
justification (7)
lists (5)
notes-footnotes (1)
numbers (5)
punctuation (4)
quotations (2)
ruby (3)
text-decoration (6)
vertical-text (8)

Filter by script

All
arab (10)
bopo (1)
deva (2)
ethi (3)
grek (1)
hang (2)
hani (2)
hebr (17)
jpan (2)

Filter by medium

All
book (23)

breaking news: jlreq

w3c.github.io/jlreq/charter/

- Japanese gap analysis
- JLReq errata & improved usability
- Produce new, informative documents
- Issue list discussions

Language Enablement

Language matrix

Expert networks

Gap-analysis

Layout requirements

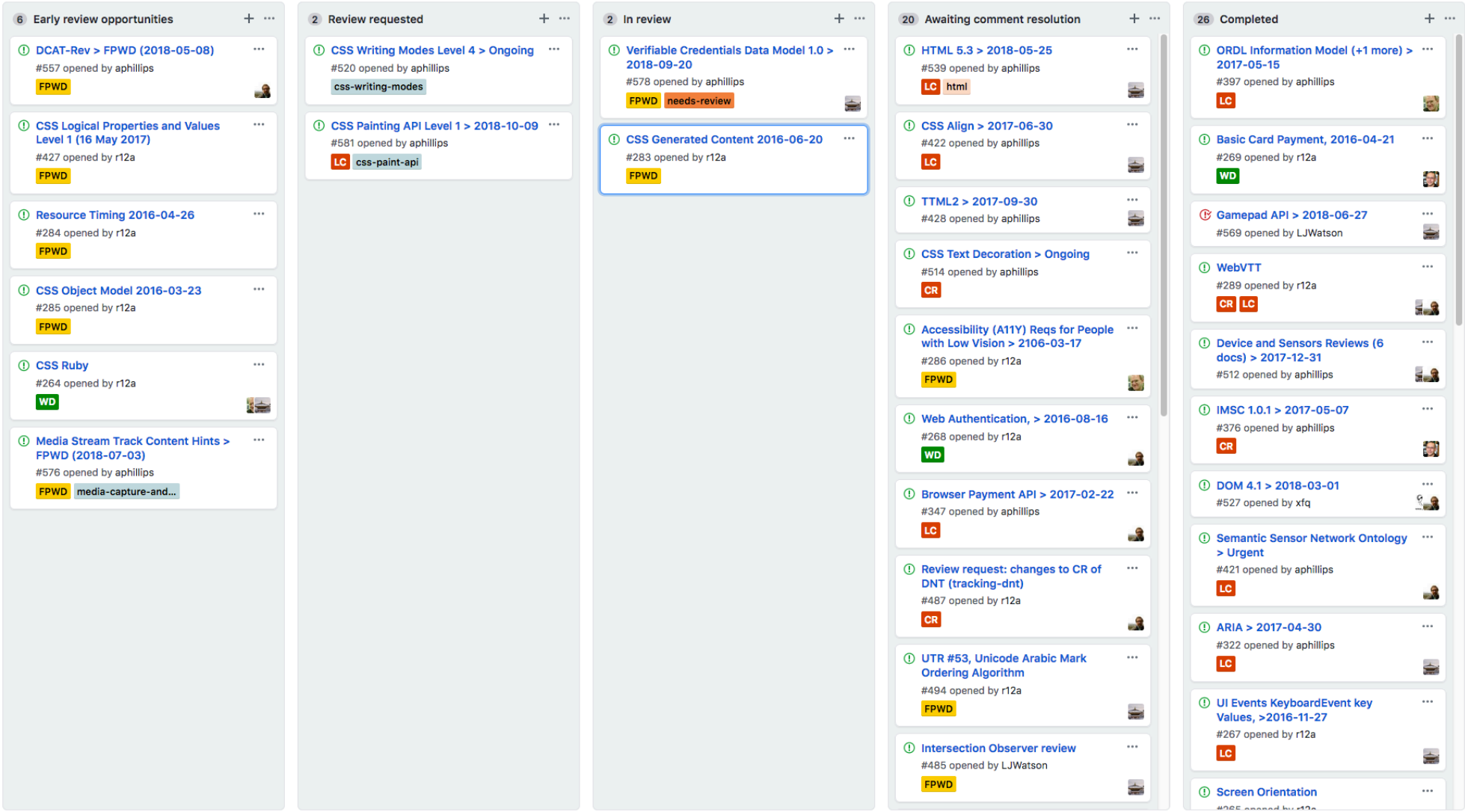
Text layout index

Tracker tools & notifications

Developer support

spec reviews

github.com/w3c/i18n-activity/projects/1



6 Early review opportunities

- DCAT-Rev > FPWD (2018-05-08)
  #557 opened by aphillips
  FPWD
- CSS Logical Properties and Values Level 1 (16 May 2017)
  #427 opened by r12a
  FPWD
- Resource Timing 2016-04-26
  #284 opened by r12a
  FPWD
- CSS Object Model 2016-03-23
  #285 opened by r12a
  FPWD
- CSS Ruby
  #264 opened by r12a
  WD
- Media Stream Track Content Hints > FPWD (2018-07-03)
  #576 opened by aphillips
  FPWD media-capture-and...

2 Review requested

- CSS Writing Modes Level 4 > Ongoing
  #520 opened by aphillips
  css-writing-modes
- CSS Painting API Level 1 > 2018-10-09
  #581 opened by aphillips
  LC css-paint-api

2 In review

- Verifiable Credentials Data Model 1.0 > 2018-09-20
  #578 opened by aphillips
  FPWD needs-review
- CSS Generated Content 2016-06-20
  #283 opened by r12a
  FPWD

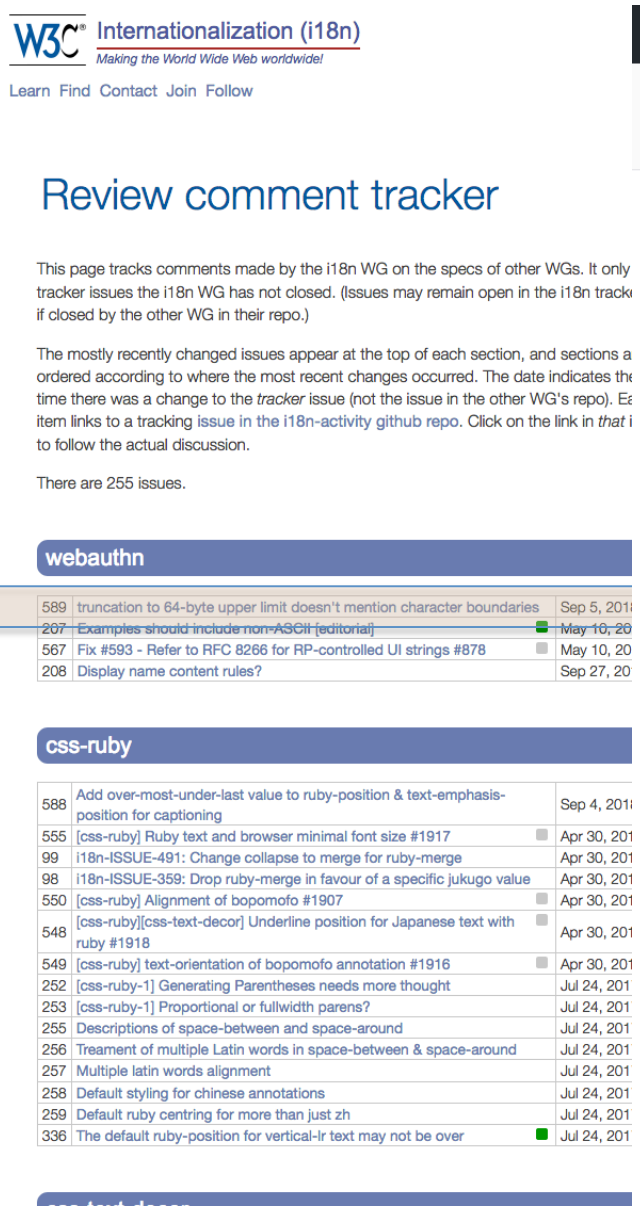
20 Awaiting comment resolution

- HTML 5.3 > 2018-05-25
  #539 opened by aphillips
  LC html
- CSS Align > 2017-06-30
  #422 opened by aphillips
  LC
- TTML2 > 2017-09-30
  #428 opened by aphillips
- CSS Text Decoration > Ongoing
  #514 opened by aphillips
  CR
- Accessibility (A11Y) Reqs for People with Low Vision > 2106-03-17
  #286 opened by r12a
  FPWD
- Web Authentication, > 2016-08-16
  #268 opened by r12a
  WD
- Browser Payment API > 2017-02-22
  #347 opened by aphillips
  LC
- Review request: changes to CR of DNT (tracking-dnt)
  #487 opened by r12a
  CR
- UTR #53, Unicode Arabic Mark Ordering Algorithm
  #494 opened by r12a
  FPWD
- Intersection Observer review
  #485 opened by LJWatson
  FPWD

26 Completed

- ORDL Information Model (+1 more) > 2017-05-15
  #397 opened by aphillips
  LC
- Basic Card Payment, 2016-04-21
  #269 opened by r12a
  WD
- Gamepad API > 2018-06-27
  #569 opened by LJWatson
- WebVTT
  #289 opened by r12a
  CR LC
- Device and Sensors Reviews (6 docs) > 2017-12-31
  #512 opened by aphillips
- IMSC 1.0.1 > 2017-05-07
  #376 opened by aphillips
  CR
- DOM 4.1 > 2018-03-01
  #527 opened by xfq
- Semantic Sensor Network Ontology > Urgent
  #421 opened by aphillips
  LC
- ARIA > 2017-04-30
  #322 opened by aphillips
  LC
- UI Events KeyboardEvent key Values, >2016-11-27
  #267 opened by r12a
  LC
- Screen Orientation

review comments



Internationalization (i18n)

Making the World Wide Web worldwide!

Learn Find Contact Join Follow

Review comment tracker

This page tracks comments made by the i18n WG on the specs of other WGs. It only
tracker issues the i18n WG has not closed. (Issues may remain open in the i18n track
if closed by the other WG in their repo.)

The mostly recently changed issues appear at the top of each section, and sections a
ordered according to where the most recent changes occurred. The date indicates the
time there was a change to the *tracker* issue (not the issue in the other WG's repo). Ea
item links to a tracking issue in the i18n-activity github repo. Click on the link in *that* i
to follow the actual discussion.

There are 255 issues.

webauthn

589	truncation to 64-byte upper limit doesn't mention character boundaries	Sep 5, 201
207	~~Examples should include non-ASCII [editorial]~~	~~May 10, 20~~
567	Fix #593 - Refer to RFC 8266 for RP-controlled UI strings #878	May 10, 20
208	Display name content rules?	Sep 27, 20

css-ruby

588	Add over-most-under-last value to ruby-position & text-emphasis-position for captioning	Sep 4, 201
555	[css-ruby] Ruby text and browser minimal font size #1917	Apr 30, 201
99	i18n-ISSUE-491: Change collapse to merge for ruby-merge	Apr 30, 201
98	i18n-ISSUE-359: Drop ruby-merge in favour of a specific jukugo value	Apr 30, 201
550	[css-ruby] Alignment of bopomofo #1907	Apr 30, 201
548	[css-ruby][css-text-decor] Underline position for Japanese text with ruby #1918	Apr 30, 201
549	[css-ruby] text-orientation of bopomofo annotation #1916	Apr 30, 201
252	[css-ruby-1] Generating Parentheses needs more thought	Jul 24, 201
253	[css-ruby-1] Proportional or fullwidth parens?	Jul 24, 201
255	Descriptions of space-between and space-around	Jul 24, 201
256	Treament of multiple Latin words in space-between & space-around	Jul 24, 201
257	Multiple latin words alignment	Jul 24, 201
258	Default styling for chinese annotations	Jul 24, 201
259	Default ruby centring for more than just zh	Jul 24, 201
336	The default ruby-position for vertical-lr text may not be over	Jul 24, 201

Search or jump to... / Pull requests Issues Marketplace Explore

w3c / webauthn — Watch 87 — Star 303 — Fork 55

Code | Issues 56 | Pull requests 18 | Projects 0 | Wiki | Insights | Settings

truncation to 64-byte upper limit doesn't mention character boundaries #973

Edit New issue

Open **aphillips** opened this issue on 27 Jun · 20 comments

aphillips commented on 27 Jun — Member

https://w3c.github.io/webauthn/#dictionary-pkcredentialentity

When referring to the `name` the spec says:

> Authenticators MUST accept and store a 64-byte minimum length for a name member's value. Authenticators MAY truncate a name member's value to a length equal to or greater than 64 bytes.

Note that the specification does not require truncation on a Unicode character boundary. Arbitrary truncation at a 64-byte limit on a multibyte encoding such as UTF-8 can corrupt the last character in the string. The spec should require that the truncation occur on a character boundary (is there a reason you didn't use character count instead of byte count in the first place?)

aphillips commented on 27 Jun • edited — Member

Hmm... also I think it doesn't mean to say `64-byte *minimum* length`. I suspect it means to say "maximum" there. PS> Please add the i18n-comment label.

equalsJeffH added i18n-comment type:technical labels on 27 Jun

equalsJeffH commented on 28 Jun • edited — Member

[this issue is related to issue #593 and PR #951]

@aphillips wrote:

> Note that the specification does not require truncation on a Unicode character boundary

I was wondering whether/when you'd bring this up.

I've done some modest research on this topic of "unicode string truncation" (due to the text you cite above) and apparently it is more complex than simply performing truncation on a Unicode character boundary -- it apparently ought to properly be done on extended grapheme cluster (EGC) boundaries.

I found detailed analysis here: https://hoytech.github.io/truncate-presentation/
..and a library: https://github.com/hoytech/Unicode-Truncate, but nothing regarding "unicode string

Assignees: No one—assign yourself

Labels: i18n-comment, type:technical

Projects: None yet

Milestone: L2-WD-00

Notifications: Subscribe — You're not receiving notifications from this thread.

8 participants

Lock conversation

'specdev' guidelines

W3C Editor's Draft

Internationalization Best Practices for Spec Developers

W3C Editor's Draft 07 August 2018

This version:
https://w3c.github.io/bp-i18n-specdev/

Latest published version:
https://www.w3.org/TR/international-specs/

Latest editor's draft:
https://w3c.github.io/bp-i18n-specdev/

Bug tracker:
File a bug (open bugs)

Editor:
Richard Ishida (W3C)

Github:
repository

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

This document provides advice to specification developers about how to incorporate requirements for international use. What is currently available here is expected to be useful immediately, but is still an early draft and the document is in flux, and will grow over time as knowledge applied in reviews and discussions can be crystallized into guidelines.

NOTE

Sending comments on this document

If you wish to make comments regarding this document, please raise them as github issues. Only send comments by email if you are unable to raise issues on github (see links below).

To make it easier to track comments, please raise separate issues or emails for each comment, and point to the section you are commenting on using a URL for the dated version of the document. All comments are welcome.

This document was published by the Internationalization Working Group as an Editor's Draft.

Comments regarding this document are welcome. Please send them to www-international@w3.org (archives).

Publication as an Editor's Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than work in progress.

This document was produced by a group operating under the W3C Patent Policy. W3C maintains a public list of any patent disclosures made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains Essential Claim(s) must disclose the information in accordance with section 6 of the W3C Patent Policy

W3C Editor's Draft

4.13 Truncating or Limiting the Length of Strings

Some specifications, formats, or protocols or their implementations need to specify limits for the size of a given data structure or text field. This could be due to many reasons, such as limits on processing, memory, data structure size, and so forth. When selecting or specifying limits on the length of a given string, specifications or implementations need to ensure that they do not cause corruption in the text.

#char_trunc_no_limit

Specifications *SHOULD NOT* limit the size of data fields unless there is a specific practical or technical limitation.

There are many reasons why a length limit might be needed in a specification or format. Generally length limits correspond to underlying limits in the implementation, such as the use of fixed-size fields in a database or data store, the desire to fit into practical boundaries such as packet size, or some other implementation detail related to storage allocation or efficiency.

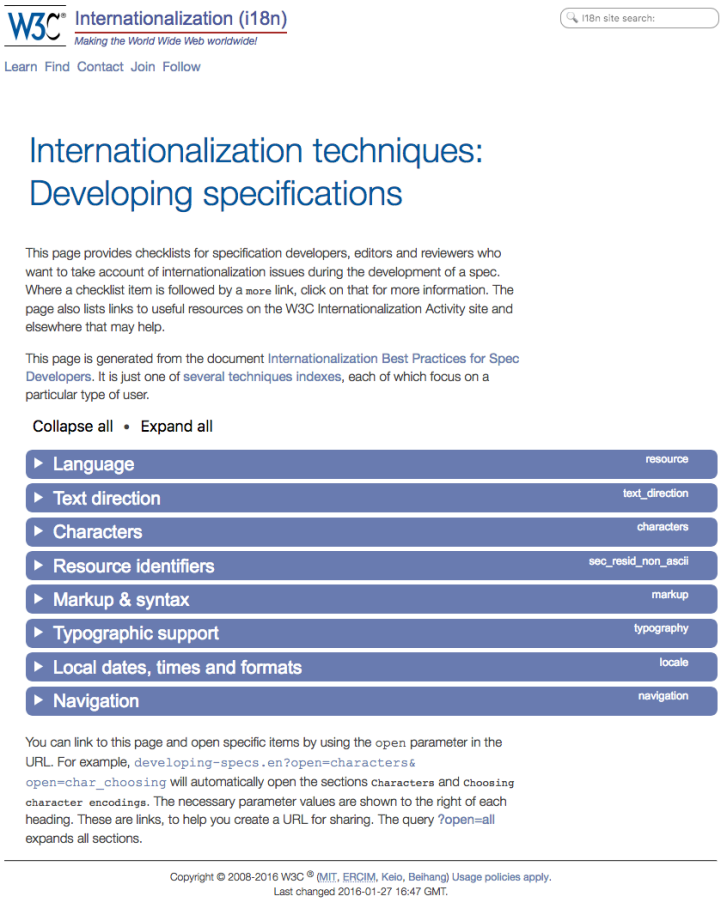
When truncating strings, it's necessary to decide what units to use when counting the size of the string. In many cases this is beyond the control of the specification, since the truncation is occuring for some preordained reason, such as the size of a fixed-length field in a file format or database. However, when the choice is available, some general guidelines can be applied.

Below you can see the effect of truncating strings in different scripts on an arbitrary code unit boundary. In this case, each string is encoded in UTF-8 and truncated after the 38th byte. There are several things to notice here.

| | | |

self-review checklist

w3.org/international/techniques/developing-specs



W3C Internationalization (i18n)
Making the World Wide Web worldwide!

i18n site search:

Learn Find Contact Join Follow

Internationalization techniques: Developing specifications

This page provides checklists for specification developers, editors and reviewers who want to take account of internationalization issues during the development of a spec. Where a checklist item is followed by a `more` link, click on that for more information. The page also lists links to useful resources on the W3C Internationalization Activity site and elsewhere that may help.

This page is generated from the document Internationalization Best Practices for Spec Developers. It is just one of several techniques indexes, each of which focus on a particular type of user.

Collapse all • Expand all

- ▶ Language — resource
- ▶ Text direction — text_direction
- ▶ Characters — characters
- ▶ Resource identifiers — sec_resid_non_ascii
- ▶ Markup & syntax — markup
- ▶ Typographic support — typography
- ▶ Local dates, times and formats — locale
- ▶ Navigation — navigation

You can link to this page and open specific items by using the `open` parameter in the URL. For example, `developing-specs.en?open=characters&open=char_choosing` will automatically open the sections `Characters` and `Choosing character encodings`. The necessary parameter values are shown to the right of each heading. These are links, to help you create a URL for sharing. The query ?open=all expands all sections.

in development

W3C Editor's Draft

Requirements for Language and Direction Metadata in Data Formats

W3C Editor's Draft 25 July 2018

This version:
https://w3c.github.io/string-meta/
Latest published version:
https://www.w3.org/TR/string-meta/
Latest editor's draft:
https://w3c.github.io/string-meta/
Bug tracker:
File a bug (open bugs)
Editors:
Addison Phillips (Invited Expert)
Richard Ishida (W3C)
Github:
repository

Abstract

This document describes the best practices for identifying language and base direction in data formats used on the Web.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

NOTE

Sending comments on this document

If you wish to make comments regarding this document, please raise them as github issues. Only send comments by email if you are unable to raise issues on github (see links below). All comments are welcome.

To make it easier to track comments, please raise separate issues or emails for each comment, and point to the section you are commenting on using a URL for the dated version of the document.

This document was published by the Internationalization Working Group as an Editor's Draft.

Comments regarding this document are welcome. Please send them to www-international@w3.org (archives).

Publication as an Editor's Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this doc-

W3C Editor's Draft

Character Model for the World Wide Web: String Matching

W3C Editor's Draft 27 July 2018

This version:
https://w3c.github.io/charmod-norm/
Latest published version:
https://www.w3.org/TR/charmod-norm/
Latest editor's draft:
https://w3c.github.io/charmod-norm/
Bug tracker:
File a bug (open bugs)
Editor:
Addison Phillips (Invited Expert)
Github:
repository

Abstract

This document builds upon on *Character Model for the World Wide Web 1.0: Fundamentals* [CHARMOD] to provide authors of specifications, software developers, and content developers a common reference on string identity matching on the World Wide Web and thereby increase interoperability.

Status of This Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at https://www.w3.org/TR/.

NOTE

This version of the document represents a significant change from the earlier editions. Much of the content is changed and the recommendations are significantly altered. This fact is reflected in a change to the name of the document from "Character Model: Normalization".

NOTE

Sending comments on this document

If you wish to make comments regarding this document, please raise them as github issues. When reviewing the document, please refer to the latest editor's copy. Only send comments by email if you are unable to raise issues on github (see links below). All comments are welcome.

To make it easier to track comments, please raise separate issues or emails for each comment, and point to the section you are commenting on using a URL.

This document was published by the Internationalization Working Group as an Editor's Draft.

Developer support

String metadata (bidi & language)

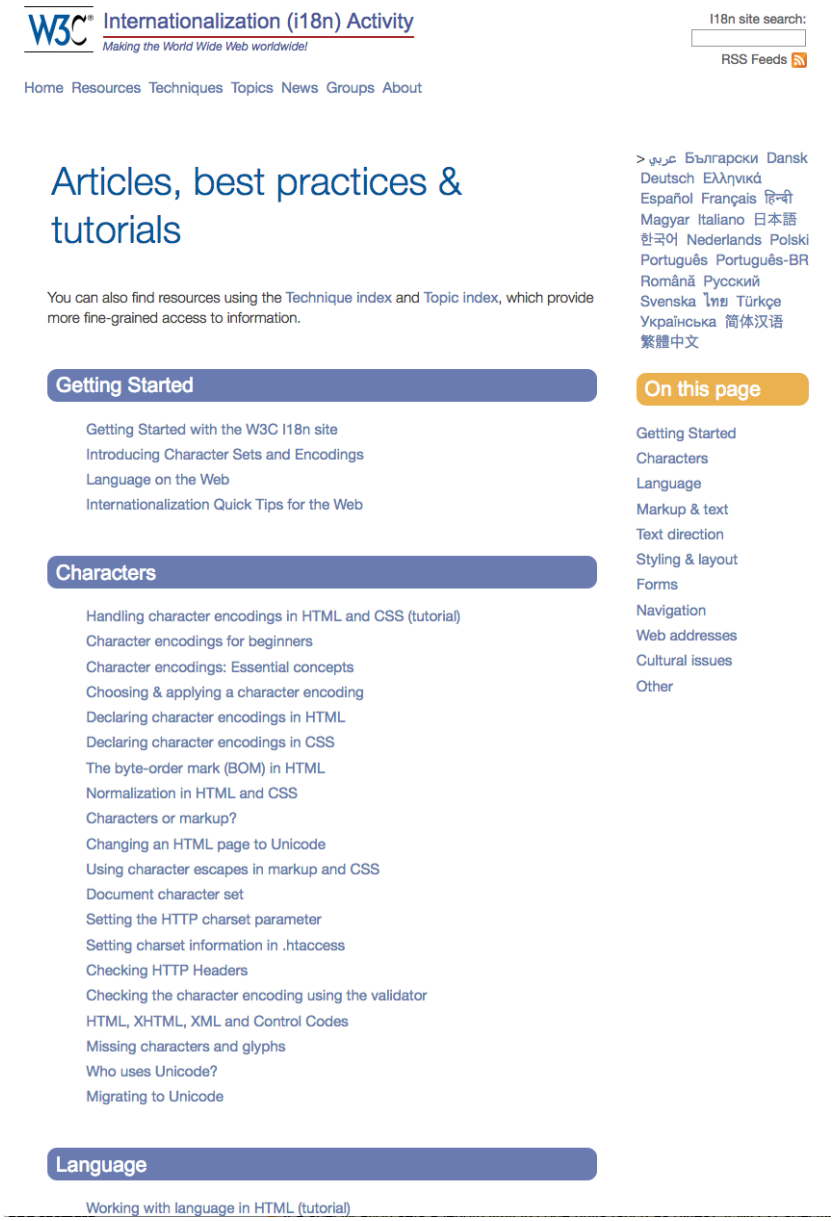
Review radar

Self-review checklist

Guidelines for developers

Author support

articles



Internationalization (i18n) Activity

Making the World Wide Web worldwide!

I18n site search:

RSS Feeds

Home Resources Techniques Topics News Groups About

Articles, best practices & tutorials

You can also find resources using the Technique index and Topic index, which provide more fine-grained access to information.

> عربي Български Dansk Deutsch Ελληνικά Español Français हिन्दी Magyar Italiano 日本語 한국어 Nederlands Polski Português Português-BR Română Русский Svenska ไทย Türkçe Українська 简体汉语 繁體中文

Getting Started

- Getting Started with the W3C I18n site
- Introducing Character Sets and Encodings
- Language on the Web
- Internationalization Quick Tips for the Web

Characters

- Handling character encodings in HTML and CSS (tutorial)
- Character encodings for beginners
- Character encodings: Essential concepts
- Choosing & applying a character encoding
- Declaring character encodings in HTML
- Declaring character encodings in CSS
- The byte-order mark (BOM) in HTML
- Normalization in HTML and CSS
- Characters or markup?
- Changing an HTML page to Unicode
- Using character escapes in markup and CSS
- Document character set
- Setting the HTTP charset parameter
- Setting charset information in .htaccess
- Checking HTTP Headers
- Checking the character encoding using the validator
- HTML, XHTML, XML and Control Codes
- Missing characters and glyphs
- Who uses Unicode?
- Migrating to Unicode

Language

- Working with language in HTML (tutorial)

On this page

vertical text guidelines

W3C Internationalization

Home Resources Techniques Topics News Groups About

Styling vertical Chinese, Japanese, Korean and Mongolian text

intended audience: CSS developers, and anyone who needs guidance on how to produce vertically-oriented text for Chinese, Japanese and Korean using CSS.

Updated 2017-03-13 13:53

This article explains how to use CSS to produce vertical text for languages such as Chinese, Japanese, Korean, and Mongolian. The CSS specification contains a lot of implementation-specific information. This article draws out the basic information that content authors need to create the more common features of vertical text. It also compares the theory and practice about what is possible.

Each section explains how you would mark up up your content according to the CSS spec (which is usually simple and straightforward), but then looks at what you currently have to do to achieve the same result in browsers that don't implement the standard property and value names. For more information, see Browser support.

Basic setup

Where it is supported, most of what you need should be achievable by applying the `writing-mode` property to the content that you want to be set vertically.

In Japanese, Chinese and Korean, lines start at the right side of the figure box and progress to the left. Latin script text typically runs down the page, with the letters rotated clockwise, while the Han characters remain upright. Any graphic also remains upright.

国際化活動 W3C。
ワールド・ワイド・ウェッブを世界中に広げましょう。

Fig. 1. Chinese and Japanese vertical text lines run right to left.

i18n site search:

Home > Resources > Articles

On this page

Basic setup

Other values of `writing-mode`

Changing the glyph orientation

『i』は、浅葱の双子の兄であり、共犯者だ。

CSS

```
figure {
        writing-mode: vertical-rl;
        }
.upright {
    text-orientation: upright;
    }
```

HTML

```
<figure>
  <p>『<span class="upright">i</span>』は、浅葱の双子の兄であり、共犯者だ。</p>
  </figure>
```

```
<p>『<span class="upright">i</span>』は、浅葱の双子の兄であり、共犯者だ。</p>
```

To make the letter 'i' upright, you would then use this CSS declaration:

```
.upright {
    text-orientation: upright;
    }
```

Test in your browser: standard syntax • proprietary syntax

For Webkit-based browsers you need to use the proprietary `-webkit-text-orientation` property. However, Webkit seems to have a problem centering the letter in the vertical line.

doesn't convert any characters to fullwidth. If you need to do so, you could nal CSS for that, if the browser supports it (see the next subsection).

ransforms. If you transform the text into fullwidth characters, that may actually on its own, since fullwidth characters are displayed upright by default.

国际化活动W3C万维网联盟

Fig. 7. Using a fullwidth transform to make Latin letters stand upright.

V3C' in the above example initially ran down the page, but applying the fullwidth ng the following CSS makes them stand upright.

```
 {
-transform: full-width;
```

Test in your browser: standard syntax • proprietary syntax

`text-transform: full-width` currently only works in Gecko.

Text in other scripts will not have full-width equivalents, and so will need to use the next approach.

priate for initialisms, but is not necessarily useful for all types of upright text, and ly that *this technique only works for Latin characters without accents*!

idth characters. Another way to achieve this is to just use fullwidth characters, C. These will automatically be displayed upright by default. You don't need any s case.

is also only works for Latin script text that doesn't include accents (since those etters for which full-width variants exist).

国际化
网联盟

i18n test suite

w3.org/international/tests

W3C Internationalization

Home Resources Techniques Topics News Groups About

Summarized test results: CSS3 Writing Modes, vertical text

intended audience: users, HTML coders, script developers, CSS coders, Web project managers, and anyone who wants to know whether browsers support the CSS Ruby spec.

Updated 2016-12-02 12:07

These tests check whether user agents correctly apply the `writing-mode` property per the CSS3 spec for the `vertical-lr` and `vertical-rl` values. They are just essential tests. More detailed tests for edge cases and finer aspects of rendering can be found in the CSS test suite.

To see the test, click on the link in the left-most column. To see detailed results for a single test, click on a row and look just above the table. The detailed results show the date(s) the test result was recorded, and the version of the browser tested.

Any dependencies are shown in notes above the table, and notes below the table will usually provide any additional useful information, including an explanation of why a result was marked as 'partially successful'.

Key:

pass | fail | partially successful

The proprietary test results are for either prefixed implementations, using `-webkit` or `-ms`, or for the nightly version of Firefox, or for non-standard writing-mode values in Internet Explorer.

If **writing-mode-vrl-001** or **writing-mode-vlr-001** fails, or either of the corresponding **-prop** tests, the remaining tests for the section can be ignored.

vertical-rl

Basics

Test link	Assertion	Firefox	Chrome	Opera	Safari	Edge	IE	Android	UC
writing mode:vertical-rl, vertical lines (writing-mode-vrl-001.html)	writing-mode:vertical-rl will display a line of text vertically.	pass	pass	pass	fail	pass	fail	fail	fail

i18n site search:

Home > Resources > Tests

On this page

About this article

This article was published without public review. If you have comments, please send them using the link near the bottom of this page.

vertical-rl

Basics

Test link	Assertion	Firefox	Chrome	Opera	Safari	Edge	IE	Android	UC
writing mode:vertical-rl, vertical lines (writing-mode-vrl-001.html)	writing-mode:vertical-rl will display a line of text vertically.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, line wrap (writing-mode-vrl-002.html)	writing-mode:vertical-rl will wrap lines from right to left.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, alignment (writing-mode-vrl-003.html)	writing-mode:vertical-lr will cause lines to display from the right side of the enclosing box.	pass	pass	pass	fail	pass	fail	fail	fail

Links: Section 3.1 • Related tests

Proprietary syntax

Test link	Assertion	Firefox	Chrome	Opera	Safari	Edge	IE	Android	UC
writing mode:vertical-rl, vertical lines (writing-mode-vrl-001-prop.html)	writing-mode:vertical-rl will display a line of text vertically.	pass	pass	pass	pass	pass	pass	pass	pass
writing mode:vertical-rl, line wrap (writing-mode-vrl-002-prop.html)	writing-mode:vertical-rl will wrap lines from right to left.	pass	pass	pass	pass	pass	pass	pass	pass
writing mode:vertical-rl, alignment (writing-mode-vrl-003-prop.html)	writing-mode:vertical-lr will cause lines to display from the right side of the enclosing box.	pass	pass	pass	pass	pass	pass	pass	pass

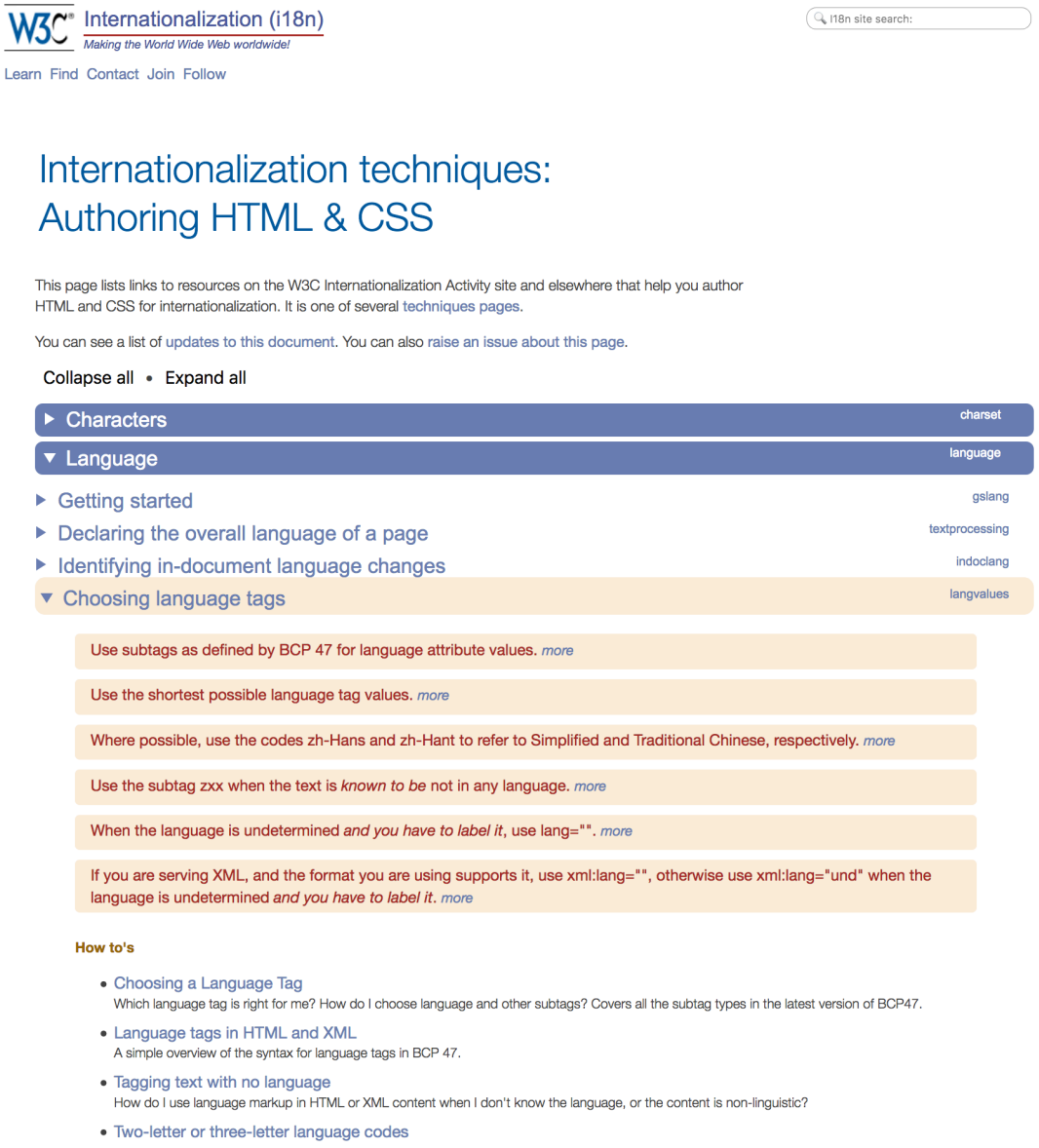
Links: Section 3.1 • Related tests

Glyphs

Test link	Assertion	Firefox	Chrome	Opera	Safari	Edge	IE	Android	UC
writing mode:vertical-rl, default Han orientation (writing-mode-vrl-005.html)	By default, writing-mode:vertical-rl will display chinese characters upright.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, default Latin orientation (writing-mode-vrl-007.html)	writing-mode:vertical-lr will display Latin characters rotated 90° right by default.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, default Arabic orientation (writing-mode-vrl-008.html)	By default, writing-mode:vertical-lr will display Arabic characters rotated 90° right.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, default Arabic direction (writing-mode-vrl-009.html)	By default, writing-mode:vertical-lr will display Arabic characters progressing up the page.	pass	pass	pass	fail	pass	fail	fail	fail
writing mode:vertical-rl, Arabic joining	By default, writing-mode:vertical-lr will display Arabic characters using cursive joining.	pass	pass	pass	fail	pass	fail	fail	fail

techniques index

w3.org/International/techniques/authoring-html



Internationalization (i18n)
Making the World Wide Web worldwide!
i18n site search:
Learn Find Contact Join Follow

Internationalization techniques: Authoring HTML & CSS

This page lists links to resources on the W3C Internationalization Activity site and elsewhere that help you author HTML and CSS for internationalization. It is one of several techniques pages.

You can see a list of updates to this document. You can also raise an issue about this page.

Collapse all • Expand all

- ▶ Characters — charset
- ▼ Language — language
  - ▶ Getting started — gslang
  - ▶ Declaring the overall language of a page — textprocessing
  - ▶ Identifying in-document language changes — indoclang
  - ▼ Choosing language tags — langvalues

Use subtags as defined by BCP 47 for language attribute values. *more*

Use the shortest possible language tag values. *more*

Where possible, use the codes zh-Hans and zh-Hant to refer to Simplified and Traditional Chinese, respectively. *more*

Use the subtag zxx when the text is *known to be* not in any language. *more*

When the language is undetermined *and you have to label it*, use lang="". *more*

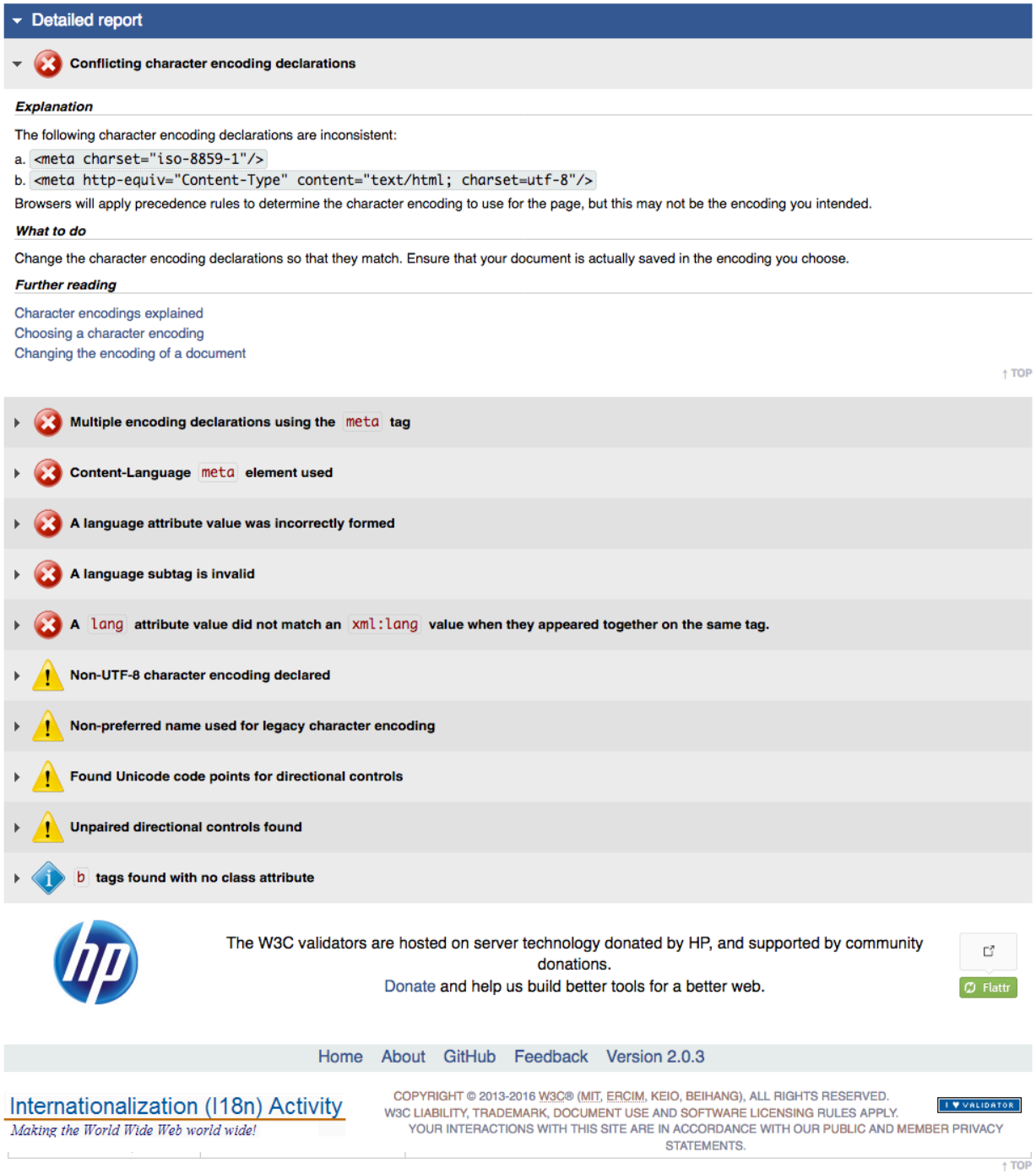
If you are serving XML, and the format you are using supports it, use xml:lang="", otherwise use xml:lang="und" when the language is undetermined *and you have to label it*. *more*

How to's

- Choosing a Language Tag
  Which language tag is right for me? How do I choose language and other subtags? Covers all the subtag types in the latest version of BCP47.
- Language tags in HTML and XML
  A simple overview of the syntax for language tags in BCP 47.
- Tagging text with no language
  How do I use language markup in HTML or XML content when I don't know the language, or the content is non-linguistic?
- Two-letter or three-letter language codes

i18n checker

validator.w3.org/i18n-checker



Author support

Internationalisation articles

Techniques index

Internationalisation test suite

Internationalization checker

Next steps...

next steps

- Widen participation of expert networks in lreq groups
- Increase scope and documented output of language enablement work
- Significantly improve specdev guidelines and the self-review checklist for developers
- Investigate ways to extend the i18n test framework to support tests and results for paged media generators

what you can do

- **Join** a layout network as follower or contributor.
- **Contribute** to creating gap-analysis docs.
- **Learn** about strings and internationalization.
- **Use** the information in our articles and test results.
- **Check** your pages with the i18n checker.
- **Support** the sponsorship program.