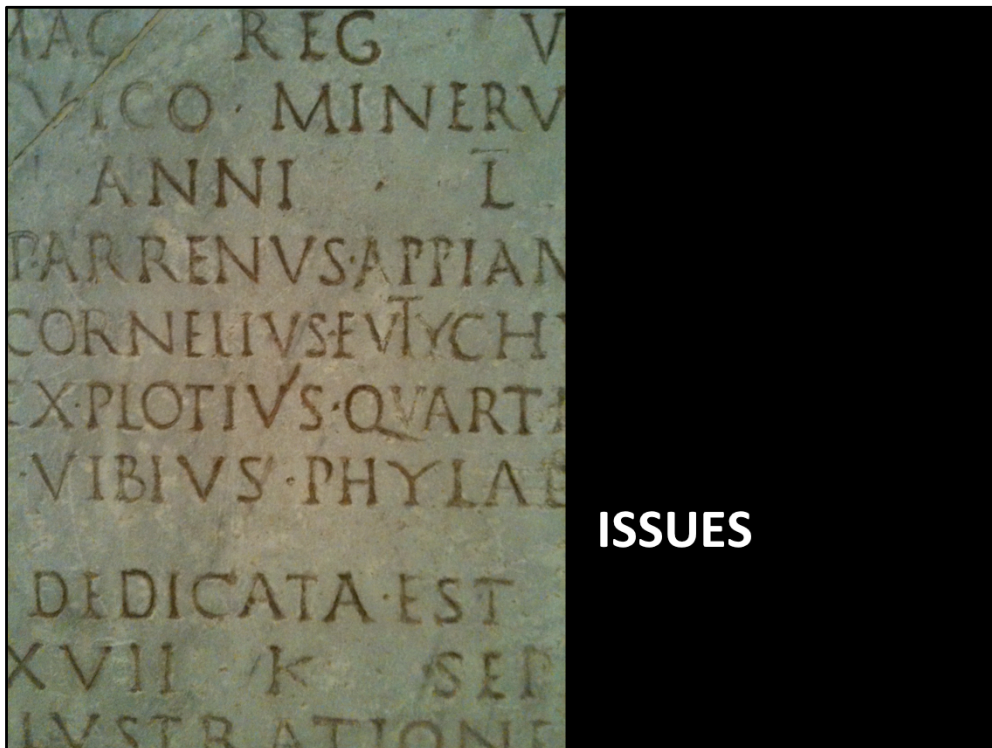


The background image shows a cityscape with a prominent mosque featuring a blue dome and minaret. The city is built on a hillside, with a large, arid mountain in the background. Several communication towers are visible on the mountain. The title text is overlaid on a semi-transparent grey band across the upper portion of the image.

# New Internationalization Developments at the W3C

Richard Ishida  
W3C



## ISSUES

The Internationalization Working Group at the W3C is involved in reviewing specifications for internationalization issues – specifications from inside and outside W3C. It also creates resources for content authors, specification developers and others related to internationalization features of the Open Web Platform, including educational articles, tests, etc.

This presentation will focus on one additional area that has been gathering pace recently – the development of documents that describe text layout requirements for a given language or script. We will start by giving a brief outline of a couple of issues that illustrate the need for these documents.

## CSS3 Text Module

W3C Editor's Draft

W3C

CSS Text Module

Editor's Draft, 20 May 2014

This version:  
<http://dev.w3.org/csswg/css3-text/>

Latest version:  
<http://www.w3.org/TR/css3-text/>

Editor's Draft:  
<http://dev.w3.org/csswg/css3-text/>

Previous Versions:  
<http://www.w3.org/TR/css3-text/>

Feedback:  
[www-style@w3.org](mailto:www-style@w3.org)

Test Suite:  
<http://test.csswg.org/suites/css3-text/>

Editors:  
fantasai (Invited Expert)  
Koji Ishii (Invited Expert)

Issue Tracking:  
<http://www.w3.org/Style/CSS/Tracking/>

Copyright © 2014 W3C® (MIT, ERCIM, Keio, Beihang, and others). All rights reserved. No warranty is given for the W3C trademarks and logos.

Abstract

This CSS3 module defines the text-align property, which covers line breaking, justification, and text alignment. CSS3 is a language for describing the presentation of documents on screen, on paper, in speech, etc.

7.3 Justification Method: the 'text-justify' property

Name:	<b>text-justify</b>
Value:	auto   none   inter-word   distribute
Initial:	auto
Applies to:	block containers and, optionally, inline elements
Inherited:	yes
Percentages:	N/A
Media:	visual
Computed value:	specified value
Animatable:	no
Canonical order:	N/A

This property selects the justification method used when a line's alignment is set to 'justify' (see 'text-align'). The property applies to block containers, but the UA may (but is not required to) also support it on inline elements. It takes the following values:

**auto**

The UA determines the justification algorithm to follow, based on a balance between performance and adequate presentation quality.

One possible algorithm is to choose the appropriate justification behavior based on the language of the paragraph e.g. following JLREQ for Japanese, using cursive elongation for Arabic, using 'inter-word' for English, etc. Another possibility is to use a justification method that is a simple universal compromise for all writing systems, such as primarily expanding word separators along with secondarily expanding between CJK and Southeast Asian letters.

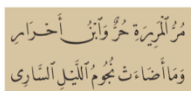


Figure 2. An example of curvily-justified Arabic text rendered by Tasmem.



Figure 3. Mixed-script text with text-justify: auto; this interpretation uses a universal-compromise justification method, expanding at spaces as well as between CJK and Southeast Asian letters.

The CSS3 Text module (<http://www.w3.org/TR/css3-text/>) will specify how to apply international typographic effects to web pages. One example is text justification, which produces straight lines down both the left and right margins.

## Arabic justification

والدهان والمخرج والتاجر  
والضابط والحي والميت ويومئ  
الينا فواز حديد عوسم اذ يهيئ  
حسن فكرت عشر مسرحيات ثم  
يومئ اذ تراود

Let's suppose that we want to justify this Arabic text, so that there are straight lines at both left and right margins.

Generally speaking, received wisdom says that Arabic does this by stretching the baseline inside words, rather than stretching the inter-word spacing (as would be the case in English text).



## Arabic justification

والدهان والمخرج والتاجر  
والضابط والحي والميت ويومئ

To make it simple, lets just focus on these two lines.

## Arabic justification

Baseline extension  
U+0640 ARABIC TATWEEL 'ʾ'



- Looks ugly
  - Complex rules for which baselines can be stretched
  - Fails when the text is dynamically resized or font is changed
- rules for
- Only between certain characters
  - Number of extensions per word/line
  - Length of word
  - Location in line
  - Font/style differences
  - etc

One way you may hear that this can be done is by using a special baseline extension character in Unicode, **U+0640 ARABIC TATWEEL**.

The slide shows some Arabic text from a newspaper where we have justified the first two lines using tatweels in exactly the same way it was done in the newspaper.

Apart from the fact that this looks ugly, one of the big problems with this approach is that there are complex rules for the placement of baseline extensions. These include:

1. extensions can only appear between certain characters, and are forbidden around other characters
2. the number of allowable extensions per word and per line is usually kept to a minimum
3. words vary in appropriateness for extension, depending on word length
4. different font styles have different rules
5. there are rules about where in the line extensions can appear – usually not at the beginning
6. etc...

An ordinary web author who is trying to add tatweels to manually justify the text may not know how to apply these rules.

## Arabic justification

Naskh font



والدهان والمخرج والتاجر  
والضابط والحلي والميت ويومئ الينا فو

Here we have changed to a font in the Naskh style. You can see that the tatweels applied to the word that was previously at the end of the first line now make the word too long to fit there. The word has wrapped to the beginning of the next line, and we have a large gap at the end of the first line.

## Arabic justification

Nastaliq font

والدهان والمخرج والتاجر  
والضابط والحى والميت ويومى الينا  
فواز حد يد عو شم اذ يهئ حسن فكرت  
عشر مسرجات تم يومى اذ تراود

Not only that, but each different style of Arabic font has different rules. For example, the rules for where and how words are elongated are different in this Nastaliq version of the same text. (All the characters are exactly the same, only the font has changed.)

## Arabic justification

Ruqah font

والدهان والمخرج والتاجر والضابط والحج  
والميت ويومئى البنا فواز صديق عوسم اذ بهي  
حسن فكلت عشر مريضات تم يومئى اذ تراود

And fonts in the Ruqah style never use elongation at all. (We'll come back to how you justify text using Ruqah-style fonts in a moment.)



## Arabic justification

Naskh font

والدهان والمخرج والتاجر والضابط  
والحي والميت ويومئ الينا فوز حديد  
عوسم اذيهيئ حسن فكرت عشر  
مسر حيات ثم يومئ اذ تراود

In this slide we have removed all the tatweel characters, and we are showing the text using a Naskh-style font. Note that this text has more ligatures on the first line, so it is able to fit in more of the text on that line than the first font we saw.

## Arabic justification

Naskh font

والدهان والمخرج والتاجر والضابط  
والحي والميت ويومئى الينا فوز حديد

Let's again focus on the first two lines, and consider how to justify them.

## Arabic justification

Naskh font  
kashida elongations



والدهان والمخرج والتاجر والضابط  
والحي والميت ويومئ اليها فواز حديد

The image shows two lines of Arabic text in a Naskh font. The text is right-aligned and justified. The first line is 'والدهان والمخرج والتاجر والضابط' and the second line is 'والحي والميت ويومئ اليها فواز حديد'. Two green arrows point to the elongated characters 'ي' and 'ا' in the second line, illustrating the kashida elongations used for justification.

High end systems have the ability to allow relevant characters to be elongated by working with the font glyphs themselves, rather than requiring additional baseline extension characters. In principle, if you are going to elongate words, this is a better solution for a dynamic environment. It means, however, that:

1. the rules for applying the right-sized elongations to the right characters has to be applied at runtime by the application and font working together, and as the user or author stretches the window, changes font size, adds text, etc, the location and size of elongations needs to be reconfigured
2. there needs to be some agreement about what those rules are, or at least a workable set of rules for an off-the-shelf, one-size-fits-all solution.

The latter is the fundamental issue we face. There is very little, high-quality information available about how to do this, and a lack of consensus about, not only what the rules are, but how justification should be done.

Some experts will tell you that text elongation is the primary method for justifying Arabic text, while...

## Arabic justification

Inter-word & intra-word  
spacing

والدهان والمخرج والتاجر والضابط  
والحي والميت ويومئ الينا فواز حديد

... others will tell you that inter-word and intra-word spacing (where there are gaps in the letter joining within a single word) should be the primary approach, and kashida elongation may or may not be used in addition where the space method is strained.

The space-based approach, of course, makes a lot of sense if you are dealing with fonts of the Ruqah style, which do not accept elongation. However, the fact that the rules for justification need to change according to the font that is used presents a new challenge for a browser that wants to implement justification for Arabic. How does the browser know the characteristics of the font being used and apply different rules as the font is changed? Fonts don't currently indicate this information.

By the way, for all the complexity so far described this is all quite a simplistic overview of what's involved in Arabic justification. For example, high end systems that justify Arabic text also allow the typesetter to adjust the length of a line of text by manual adjustments that tweak such things as alternate letter shapes, various joining styles, different lengths of elongation, and discretionary ligation forms.

## Arabic justification

We need an description of the text layout features to capture the script needs.

Then we need to figure out how to adapt Open Web Platform technologies to implement the requirements.

To do this, we need experts to provide information and develop consensus.

والدهان والمخرج والتاجر والضابط  
والحي والميت ويومئ الينا فواز حديد  
عوسم اذيهيئ حسن فكريت عشر  
مسر حيات ثم يومئ اذ تراود

The key message:

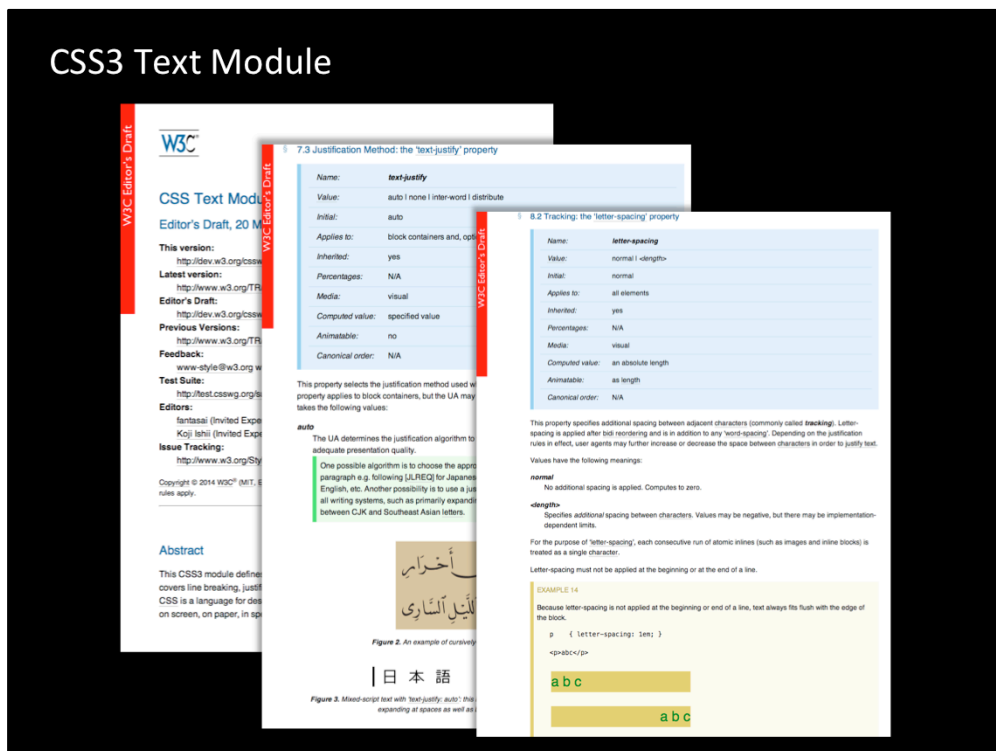
We need an Arabic Layout Requirements document to capture the script needs.

Then we need to figure out how to adapt Open Web Platform technologies to implement the requirements.

To do this, we need experts to provide information and develop consensus.



## CSS3 Text Module



The CSS3 Text module (<http://www.w3.org/TR/css3-text/>) also specifies how to do letter-spacing (also known as tracking). This can be used for aesthetic effects but sometimes also has semantic implications. For example, in German it produces a form of emphasis. We have come across some issues while trying to understand how to do letter-spacing on non-Latin text. These issues are actually relevant to other aspects of web page text layout, in addition to letter-spacing.

## Thai letter-spacing

### คำนำ

ภาษาเป็นเครื่องมือถ่ายทอดความคิดของมนุษยชาติแต่โบราณกาล บางชนชาติเท่านั้นที่มีทั้งภาษาพูดซึ่งใช้เสียงเป็นสื่อและภาษาเขียนซึ่งสื่อความด้วยอักษรหรืออักษร ชนชาติที่มีอักษรเป็นของตนเองบ่งบอกถึงความเจริญงอกงามทางอารยธรรมของชนกลุ่มนั้น อักษรไทยเป็นหนึ่งในสมบัติทางอารยธรรมของชาวไทยที่มีวิวัฒนาการมาอย่างต่อเนื่องนับร้อยปี เป็นเอกลักษณ์หนึ่งที่ควรค่าแก่การภูมิใจและใส่ใจดูแลหวงแหน

ตัวพิมพ์ไทยเป็นวิวัฒนาการหนึ่งของอักษรไทย จากการจดจารึกสู่การพิมพ์ จากสื่อธรรมชาติ เช่น แผ่นหิน กระดาษ กลายเป็น ฟิล์ม เฟลท และจอคอมพิวเตอร์ซึ่งนับวันจะมีบทบาทมากยิ่งขึ้น ดังนั้นการทำให้คอมพิวเตอร์สามารถรับและแสดงผลภาษาไทยได้อย่างสมบูรณ์ จึงเป็นสิ่งพึงกระทำ เป็นที่นำอันดีในช่วงเวลาที่ผ่านมามีหลายหน่วยงานได้ทำการพัฒนารูปแบบการแสดงผลภาษาไทย โดยมีความก้าวหน้าไปถึงระดับหนึ่ง อย่างไรก็ตามพบว่ายังขาดความเป็นเอกภาพในการใช้งาน ตัวอักษรไทยกับเทคโนโลยีคอมพิวเตอร์ ซึ่งส่วนหนึ่งอาจเป็นผลเนื่องมาจากพื้นฐานความรู้ความเข้าใจในเรื่องการแสดงผลภาษาไทยด้วยคอมพิวเตอร์ของผู้พัฒนา ยังไม่ได้เป็นไปในทิศทางเดียวกัน

The title in the top-left corner of the Thai text on this slide has been letter-spaced.

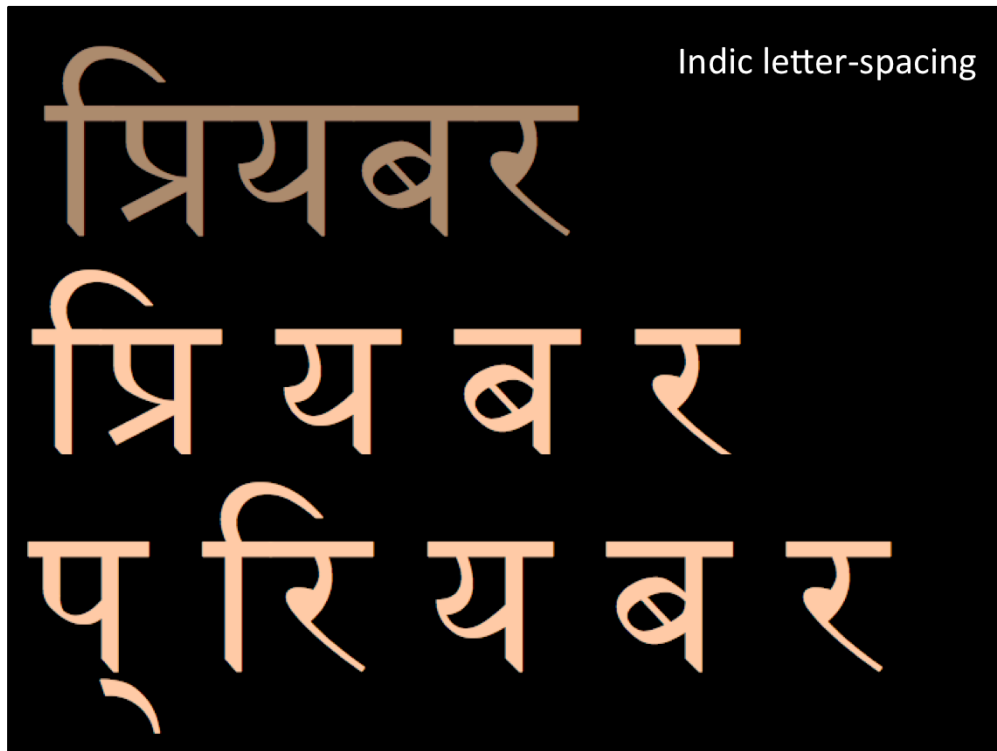
## Thai letter-spacing

The diagram illustrates the conversion of Thai text for letter-spacing. It shows a word 'คำนำ' (Kham Na) being converted from a standard character set to a set with specific Unicode characters for letter-spacing. A green arrow labeled 'convert' points from the left list to the right list.

Standard Thai Characters	Unicode Characters for Letter-spacing
ค	U+0E04 THAI CHARACTER KHO KHWAI
ำ	U+0E33 THAI CHARACTER SARA AM
น	U+0E19 THAI CHARACTER NO NU
ำ	U+0E33 THAI CHARACTER SARA AM

This slide shows how the normal set of characters used to produce this word have to be converted to a different set of characters in order to allow the letter-spaces to appear where expected. The list of characters on the right should only be used internally for this purpose – content authors should actually write and store the text as shown in the left-hand list.

We became aware of this issue thanks to an email from an expert on a W3C list.



This slide shows Hindi text. Hindi can be letter-spaced, in which case you would expect it to look like the text on the second line.

However, the representation of the first syllable as a single unit depends on the availability of glyphs in the font. Change the font for a less sophisticated one and you may actually see the text as displayed on the bottom line of the slide.

Indic letter-spacing

**User-perceived character**

Character	Unicode	Script
प	U+092A	DEVANAGARI
LETTER PA		
र्	U+094D	DEVANAGARI
SIGN VIRAMA		
र	U+0930	DEVANAGARI
LETTER RA		
ि	U+093F	DEVANAGARI
VOWEL SIGN I		
य	U+092F	DEVANAGARI
LETTER YA		
व	U+092C	DEVANAGARI
LETTER BA		
र	U+0930	DEVANAGARI
LETTER RA		

The devanagari script, which is used to write Hindi, is based on syllabic structures. The list to the right shows the characters used to write the word, but note how the first four characters are combined visually to make one user-perceived unit. The characters in this unit should not be separated by letter-spacing (nor by first-letter styling, justification, line-wrapping, vertical text layout, etc).



Indic letter-spacing

प्रियबर

प्रि य र

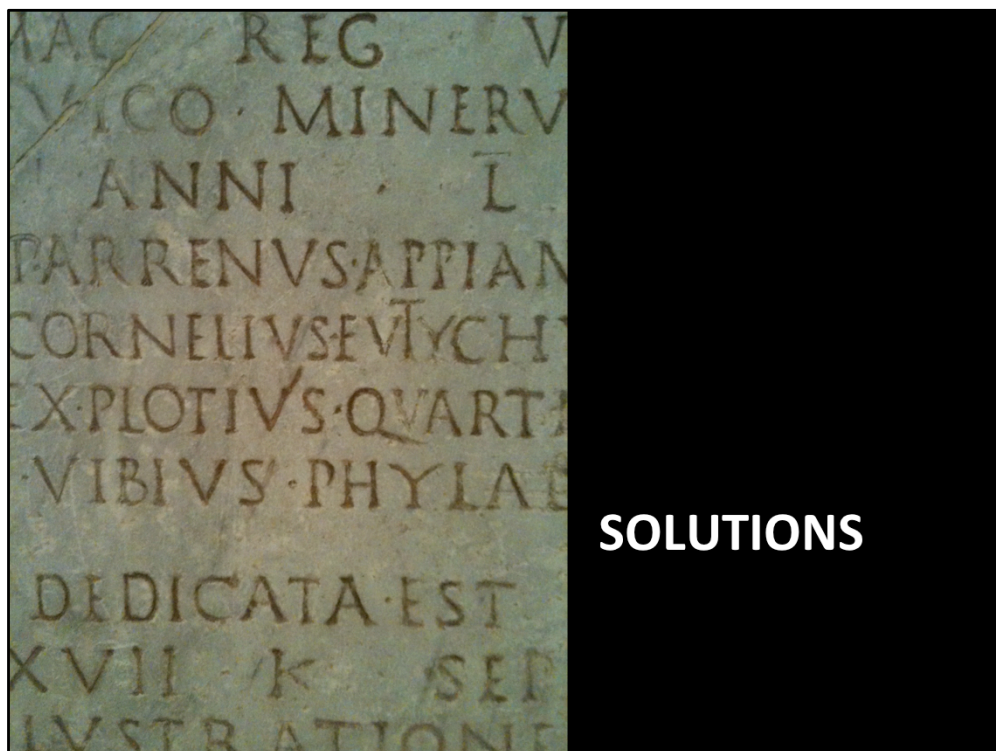
प रि य

Grapheme clusters

प	U+092A DEVANAGARI
LETTER PA	
्	U+094D DEVANAGARI
SIGN VIRAMA	
र	U+0930 DEVANAGARI
LETTER RA	
ि	U+093F DEVANAGARI
VOWEL SIGN I	
य	U+092F DEVANAGARI
LETTER YA	
व	U+092C DEVANAGARI
LETTER BA	
र	U+0930 DEVANAGARI
LETTER RA	

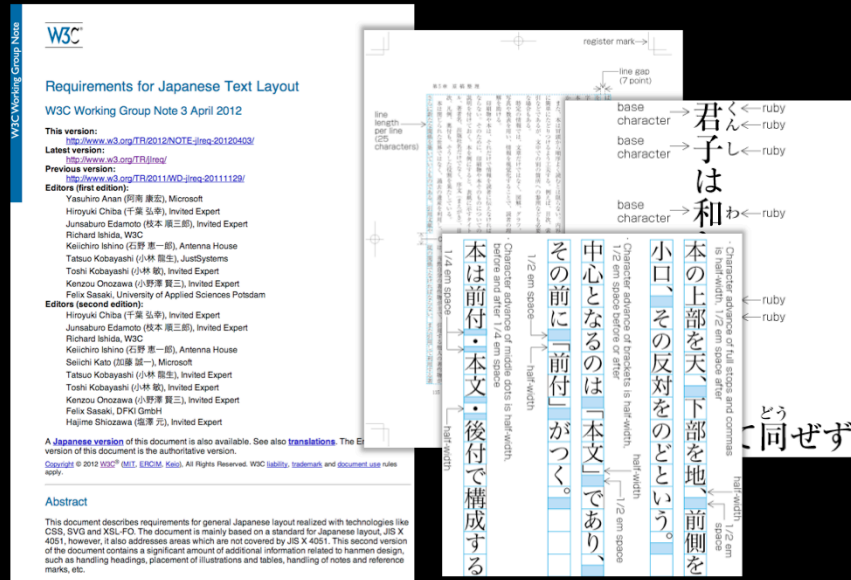
The current wording in the CSS3 Text spec says that you should base unit boundaries for letter-spacing on 'grapheme clusters'. This is a concept defined by the Unicode Standard, which typically associates base characters with combining characters that follow it. Unfortunately, the current definition of grapheme-clusters leads to a segmentation of the Hindi word as shown on the bottom line of the slide and does not allow for the combination of all four initial characters into a single unit.

Unfortunately, it is not so straightforward to fix this by extending the definition of a grapheme cluster because the representation of the first syllable as a single unit depends on the availability of glyphs in the font. It is not clear how to automatically detect what kind of font is being used for display (ie. does it support the larger syllable or not) and thereby where to break the word during letter-spacing.



Let's take a look at some of the initiatives currently under way that help to identify issues such as those just described, and provide information about what specifically needs to be supported.

## Requirements for Japanese Layout



The flagship document for this work is Requirements for Japanese Text Layout (<http://www.w3.org/TR/jlreq>). This large document provides detailed information about Japanese-specific requirements for text layout, including such things as justification, phonetic and semantic annotations, vertical text, and many more.

# Ruby annotation

鬼門きもん  
の方角ほうかくを凝視する

```
<ruby>鬼<rt>き<rb>門<rt>もん</ruby>
```

```
<ruby>
<rb>東<rb>南
<rt>とう<rt>なん
<rtc>たつみ</rtc>
</ruby>
```

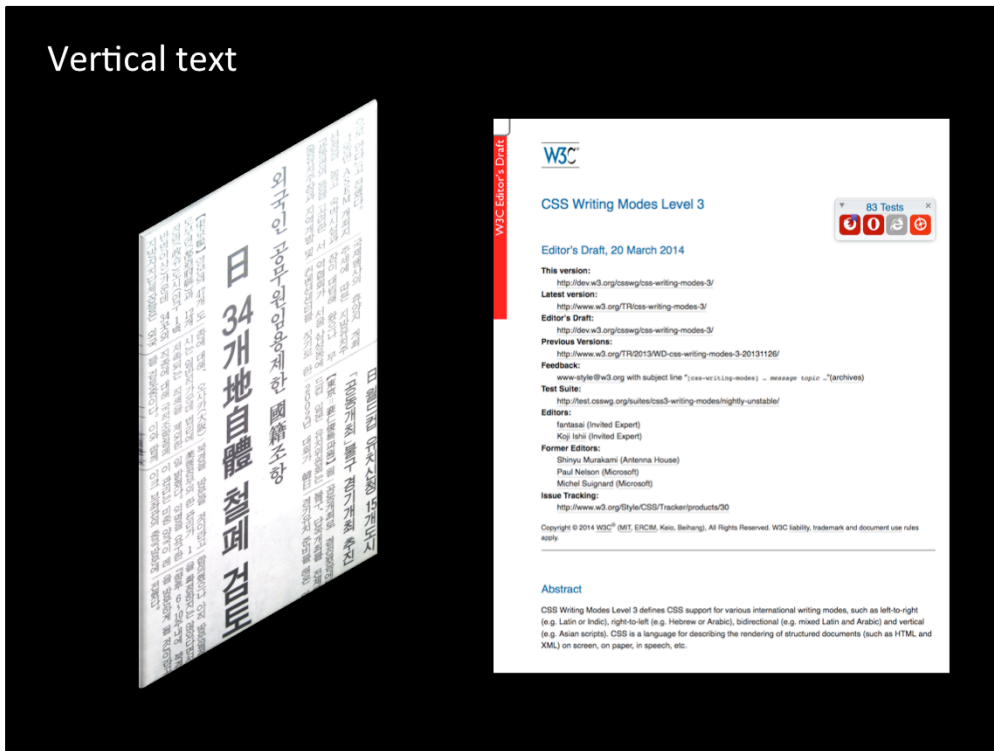
東南とうなん  
の方角ほうかく

W3C Editor's Draft
HTML5
A vocabulary and associated APIs for HTML and XHTML
Editor's Draft 4 May 2010

W3C Editor's Draft
CSS3 Ruby Module
W3C Working Draft 04 March 2010 (Editor's copy)
This version: <http://www.w3.org/2010/03/css3-ruby/>
Latest version: <http://www.w3.org/TR/css3-ruby/>
Previous version: <http://www.w3.org/TR/2009/CR-css3-ruby-20090514/>
Editors: Richard Ishida (W3C)
Former editors: Paul Nelson (Microsoft), Michael Stutzard (Microsoft), Martin Szwed (Microsoft)
Copyright © 2010 W3C®. All Rights Reserved. W3C, W3C logo, W3C seal, W3C document logo and software licensing rules apply.

The information provided related to phonetic and semantic annotation (called ruby) has led to a revision of the markup proposed to support this feature in the HTML5 specification (<http://www.w3.org/TR/html5/text-level-semantics.html#the-ruby-element>), and is now being applied to the CSS3 Ruby module (<http://www.w3.org/TR/css-ruby-1/>) that specifies how to do the needed positioning of the ruby.

## Vertical text



Other information has been used in other specifications: for example, the CSS3 Writing Modes spec (<http://www.w3.org/TR/css-writing-modes-3/>), which provides support for vertical text. This has been useful for Chinese and Korean text (shown here) and not just Japanese.



## Korean layout requirements

W3C Working Draft

W3C

### Requirements for Hangul Text Layout and Typography

W3C First Public Working Draft 14 May 2013

**This version:**  
<http://www.w3.org/TR/2013/WD-klreq-20130514/>

**Latest published version:**  
<http://www.w3.org/TR/klreq/>

**Latest editor's draft:**  
<http://www.w3.org/International/docs/korean-layout/>

**Authors:**  
Soon-Bum Lim, Sookmyung Women's University  
Wu-Jin Sim, Invited Expert  
Yong-Je Lee, Kaywon School of Art & Design  
Dongsun Nam, Hancore Inc.  
HyunYoung Kim, DAOUINCUBE, Inc.  
Yongho Lim, School of Visual Art

**Editor:**  
Richard Ishida, W3C

This document is also available in this non-normative format: [Korean version](#)

Copyright © 2013 W3C® (MIT, ERCIM, Keio, Beihang). All Rights Reserved. W3C liability apply.

#### Abstract

This document describes requirements for general Korean language/Hangul text layout and typography. The document is mainly based on the international standard for Korean text layout.

#### Status of This Document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this document is available at <http://www.w3.org/TR/>.

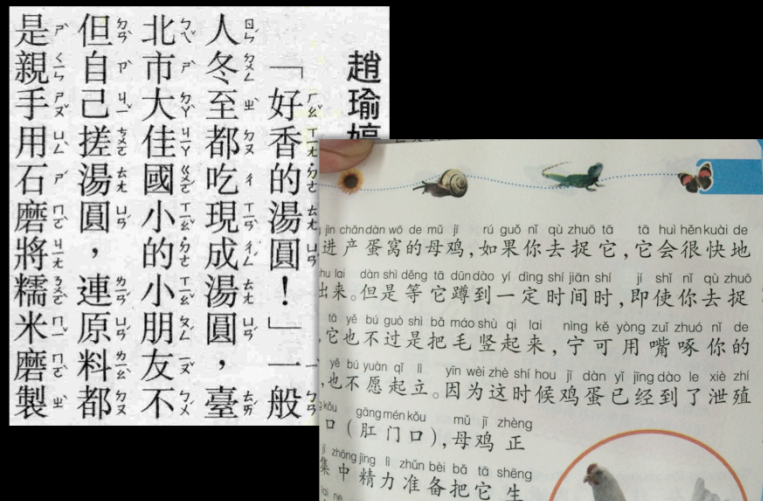
This document describes requirements for general Korean language/Hangul text layout and typography. The document is mainly based on the international standard for Korean text layout, and was originally developed by Korean typographers. The document is also available, but the English version is the authoritative version of the document to become a Working Group Note.

#### Apply Kerning Group for Hangul Fonts (unit: em/1000)

분다	group1 + 6	함기	group1 + 4	걸녀이	group1 + 8-1 + 6
논개	group2 + 4	속담	group2 + 5	섬나기	group3 + 8-2 + 6
견설	group3 + 5	일주	group3 + 7	해장소	group9-3 + 8-3 + 7
송강	group2 + 4	연결	group3 + 4	넘시다	group3 + 8-4 + 6
바더	group8-2 + 5	엄살	group3 + 5	이날지	group3 + 9-1 + 6

Working is now proceeding on a Korean layout requirements document (<http://www.w3.org/TR/klreq/>). One of the new topics that came up in this document, that wasn't in the previous Japanese requirements, was the need to kern adjacent characters to make the text look nicer.

## Chinese layout requirements



Work also recently got under way on a Chinese layout requirements document. So far work has focused on pinyin and bopomofo annotations, which work like the ruby in Japanese text.

## Chinese layout requirements

品韻好是不  
碧天端那北  
催兒正的梯小  
學凝彩名雲喜  
十  
南解三麗

掛北斗  
椿壽  
巨古  
鏡新  
桂香  
南桂花偏南枝  
天明愛稍酬  
珠光恁暗投

雞公仔  
雞公仔 尾 彎 彎， 做人 新抱 甚 艱 難，  
2 2 | 2 3 . | 3 6 5 | 1 - | 1 6 | 5 2 | 6 6 | 2 - |  
早 早 起 身 都 話 晏， 眼 泪 唔 干 就 落 下 間。  
6 3 | 2 1 . | 3 3 | 2 - | 6 1 | 2 5 | 2 6 | 2 - | 3 5 |  
下 間 有 个 冬 瓜 仔， 問 下 安 人 煮 定 蒸， 安 人  
6 6 | 2 - | 1 5 | 6 6 | 2 - | 3 3 | 2 2 3 | 5 2 6 | 1 - |  
又 話 煮， 老 爷 又 話 蒸， 蒸 蒸 煮 煮 都 唔 中 意，  
1. 2 | 7 6 | 5 3 | 2 - | 3 2 | 2 6 2 5 | 1 6 2 | 1 - | 1 2 |  
拍 起 台 頭 兩 三 朝， 三 朝 打 爛 三 條 夾 木 棍， 四 朝  
5 6 | 2 1 6 | 5 - | 5 0 ||  
跪 爛 九 條 裙。

During the course of the research we came across two types of musical notation, which it may also be worth documenting at some point.

## Possible additions

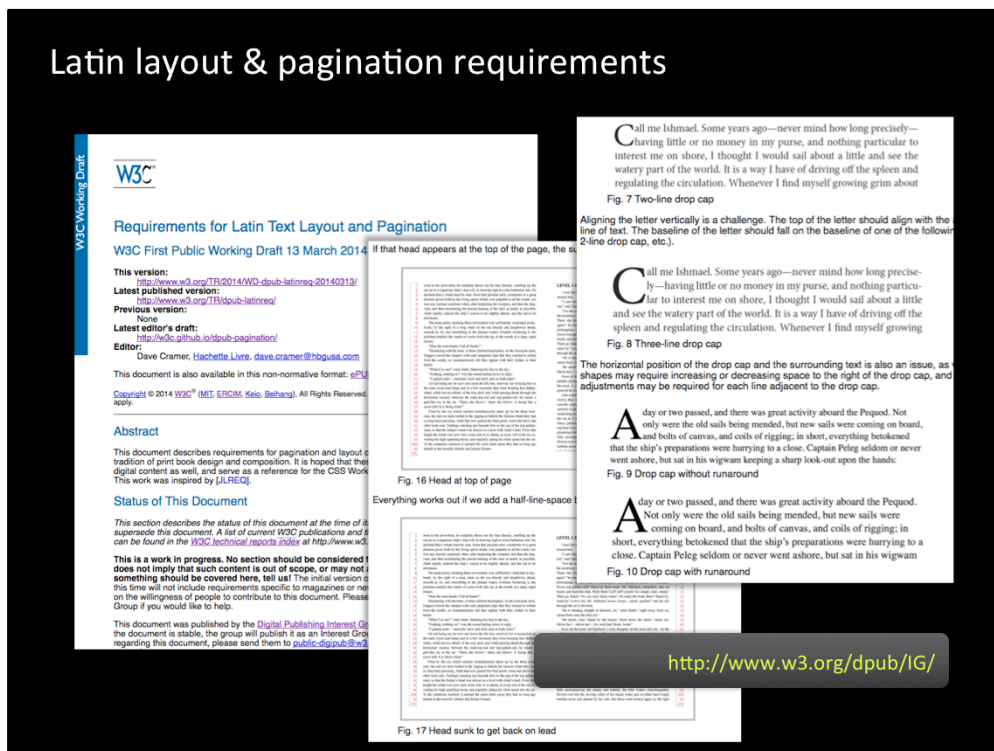
ھەممە ئادەم تۇغۇلۇش ئىدىيەسىنى ئىزدەش، ئىزدەش،  
ھۆرمەت ۋە ھوقۇقتا باب-باراۋەر بولۇپ تۇغۇلغان.  
ئۇلار ئەقىلگە ۋە ۋىجدانغا ئىگە ھەممە بىر-  
بىرىگە قېرىنداشلىق مۇناسىۋىتىگە خاس روھ بىلەن  
مۇئامىلە قىلىشى كېرەك.

འགྲོ་བ་མིའི་རིགས་རྒྱུད་ཡོངས་ལ་སྐྱེས་ཙམ་ཉིད་  
ནས་ཆེ་མཐོངས་དང་། ཐོབ་ཐང་གི་རང་དབང་འདྲ་  
མཉམ་དུ་ཡོད་ལ། ཁོང་ཚོར་རང་བྱུང་གི་སྒོ་རྩལ་  
དང་བསམ་ཚུལ་ལཟང་པོ་འདོན་པའི་འོས་བབས་  
ཀྱང་ཡོད། དེ་བཞིན་ཕན་ཚུན་གཅིག་གིས་གཅིག་ལ་  
བུ་སྙན་གྱི་འདྲ་ཐེས་འཛིན་པའི་བྱ་སྤྱོད་ ཀྱང་ལག་  
ལེན་བསྟར་དགོས་པ་ཡིན།།

[illegible]

Some interest has been expressed for the idea of developing Uighur, Tibetan and traditional Mongolian layout requirements.

## Latin layout & pagination requirements



The Japanese layout requirements document also inspired the new Digital Publishing Interest Group to start work on Requirements for Latin Text Layout and Pagination (<http://w3c.github.io/dpub-pagination/>).

Currently this document both documents the requirements and points out gaps in the Open Web Platform where these features are not supported. At some point these will be separated, leaving one perennial document describing the requirements, and another that will hopefully become soon out of date as the requirements are implemented in the OPW.

# Indic layout requirements

**W3C भारत**

## Indian Languages layout requirements (Hindi as a case study)

W3C India First Proposed Working Draft

This version:  
Previous version:  
[http://w3cindia.in/Indic\\_Akshara.html](http://w3cindia.in/Indic_Akshara.html)

Editors:  
Swaran Lata, W3C India  
Somnath Chandra, W3C India  
Prashant Verma, W3C India

Copyright © 2013 W3C India

### Abstract

This document describes the requirements for Indian Languages layout to be read on the screen. It addresses some of the major issues in CSS segmentation to overcome the & Horizontal arrangements of text. This document also describes the format in E-publishing. This standardize format of text layout repeated head, line breaking etc. Based on this study, definition contains various test results in Hindi.

### Objective

The main objective of this document is to provide examples for the desired solution.

### Table of Contents

- 1. Introduction
- 1.1 What is CSS
- 2. Basic composition of Indian Languages
- 2.1 UNICODE & CLDR
- 2.2 Devanagari code chart
- 2.3 Extended Devanagari script

### 3.1.1 Examples for Hindi language

If some styling feature is to be applied to the text, the conjunct character, a syllable or a Grapheme cluster should be used.

स्थिति (Position)	चरित्र (Character)
प्रस्थान (Departure)	अक्षर (Character)
हिंदी (Hindi)	।

### 4.2.1 Hyphenation

Hyphens are used when a word remains incomplete at the end of a line while writing or when specifying a range. There are different cases of hyphenation, some of the cases are given below:

**Case 1:** Hyphens are commonly used in Copulative compounds words in Hindi language. Hindi has both prefixes and suffixes which are joined to words with a hyphen.

examples:  
नर-मारी, लाभ-हानि, माता-पिता, ऊंच-नीच

**Case 2:** Single word can break at the end of the line at akshara level using hyphen

Test result of Hyphenation:

### 3.4 Underlining of the characters

There are some examples of Indian languages in which Matra's are not readable due to underlining of characters. When we see these pages on internet, the information is not clearly readable because if we hyperlink the text in Indian languages some modifiers (matras) are it can create problem in reading the information correctly. Therefore some changes may be required to be implemented in CSS standards developed by W3C with respect to Indian languages.

राष्ट्रमंडल खेलों के लिए स्टेडियम सौंपने को तैयार नहीं है....

Matra is not readable

Elsewhere, work is slowly proceeding on development of Indic layout requirements, which aims to address the needs of several major Indian scripts (<http://www.w3cindia.in/Indic-req-draft/Indic-layout-requirements.html>).

## Predefined counter styles

W3C Working Draft

**W3C**

**Predefined Counter Styles**

W3C First Public Working Draft

This version:  
<http://www.w3.org/TR/2013/WD-css3-counter-20130916/>

Latest published version:  
<http://www.w3.org/TR/css3-counter/>

Latest editor's draft:  
<http://www.w3.org/International/voice/>

Editor:  
Richard Ishida, W3C

Authors:  
Ian Hickson  
Tantek Çelik  
Tab Atkins

Copyright © 2013 W3C® (MIT, ERCIM, Keio, Beihai)

**Abstract**

This document describes numbering system for those wishing to create user-defined counter styles.

**Status of This Document**

This section describes the status of this document. A list of current W3C publications and technical reports is available at <http://www.w3.org/publications/>.

This document describes numbering system for those wishing to define user-defined counter styles. The Working Group expects that this document will be updated, replaced or document as other than work in progress.

**Table of Contents**

1. Introduction
  - 1.1 Purpose of the document
  - 1.2 How to use this information
2. Arabic
3. Armenian
4. Bengali
5. Cyrillic
6. Devanagari
7. Ethiopic
8. Georgian
9. Greek
10. Gujarati
11. Gurmukhi
12. Hebrew
13. Ideographic
14. Japanese
15. Kannada
16. Khmer
17. Korean (Hangeul)
18. Lao
19. Latin
20. Lepcha
21. Malayalam
22. Mongolian
23. Myanmar (Burmese)
24. Oriya
25. Tamil
26. Telugu
27. Thai
28. Tibetan
29. European
- A. References
- B. Revision Log
- C. Acknowledgements

```
@counter-style cambodian {
  system: numeric;
  symbols: '\17E0' '\17E1' '\17E2' '\17E3' '\17E4' '\17E5' '\17E6' '\17E7' '\17E8' '\17E9';
  /* symbols: '0' '1' '2' '3' '4' '5' '6' '7' '8' '9'; */
}

@counter-style cambodian-consonant {
  system: alphabetic;
  symbols: '\1780' '\1781' '\1782' '\1783' '\1784' '\1785' '\1786' '\1787' '\1788' '\1789'
    '\178A' '\178B' '\178C' '\178D' '\178E' '\178F' '\1790' '\1791' '\1792' '\1793' '\1794'
    '\1795' '\1796' '\1797' '\1798' '\1799' '\179A' '\179B' '\179C' '\179D' '\179E' '\179F'
    '\17A0' '\17A1' '\17A2';
  /* symbols: 'a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' 'k' 'l' 'm' 'n' 'o' 'p' 'q' 'r' 's' 't' 'u' 'v' 'w' 'x' 'y' 'z'; */
}

@counter-style khmer {
  system: numeric;
  symbols: '\17E0' '\17E1' '\17E2' '\17E3' '\17E4' '\17E5' '\17E6' '\17E7' '\17E8' '\17E9';
  /* symbols: '0' '1' '2' '3' '4' '5' '6' '7' '8' '9'; */
}

@counter-style khmer-consonant {
  system: alphabetic;
  symbols: '\1780' '\1781' '\1782' '\1783' '\1784' '\1785' '\1786' '\1787' '\1788' '\1789'
    '\178A' '\178B' '\178C' '\178D' '\178E' '\178F' '\1790' '\1791' '\1792' '\1793' '\1794'
    '\1795' '\1796' '\1797' '\1798' '\1799' '\179A' '\179B' '\179C' '\179D' '\179E' '\179F'
    '\17A0' '\17A1' '\17A2';
  /* symbols: 'a' 'b' 'c' 'd' 'e' 'f' 'g' 'h' 'i' 'j' 'k' 'l' 'm' 'n' 'o' 'p' 'q' 'r' 's' 't' 'u' 'v' 'w' 'x' 'y' 'z'; */
}
```

The CSS Counter Styles specification allows a content author to specify many different, user-defined, systems for numbering things such as lists, chapter headings, etc. To support that specification the Internationalization Working Group has published a set of ready-made counter-style definitions, currently just over 120 for around 28 different scripts. Authors can simply copy and paste the definitions in the Predefined Counter Styles document (<http://www.w3cindia.in/Indic-req-draft/Indic-layout-requirements.html>) into their CSS code.

## Arabic layout requirements

< insert information here >

So much for what is currently in progress, but as we have seen, there is still a need for more initiatives. We saw, for example, that we are lacking information about Arabic justification, but there are surely many other aspects of Arabic typography for the Web that need to be documented.



## South East Asian layout requirements

< insert information here >

As indeed there are for South East Asian scripts, and scripts and languages from other parts of the world.

<insert name here> layout requirements

< insert information here >

## How can you help?

- Participate in the development of current layout requirements documents: review, write, suggest, tweet about it, etc., or get your friends involved.
- Propose an expert team to work on a new set of requirements.
- Help map the requirements to the Open Web Technologies.
- Lobby the browser implementers and use the new features.

The W3C needs help from people around the world to make this happen. You can help in the ways such as the following:

- Participate in the development of current layout requirements documents: review, write, suggest points, tweet about it, etc., or get your friends involved.
- Propose an expert team to work on a new set of requirements.
- Help map the requirements to the Open Web Technologies.
- Lobby the browser implementers and use the new features.