



Web for All

- **Developer support**
 - Standards developers & user agent implementers should enable local needs
- **Language enablement**
 - In-country users and other experts tell us their requirements, to pass on to developers.
- **Author support**
 - Content authors and users learn how to use locally-adapted features.





Developer support

Build standards & applications that support a global Web

Guangzhou, China

RTL on the Web

- Arabic script (incl. Persian, Urdu, Uighur, Kashmiri, etc.)
- Hebrew
- Thaana
- Rohingya
- Syriac, Mandaic, Assyrian, etc.
- N'Ko, Adlam, Mende Kikakui
- Etc...

Bidi in HTML5

W3C Working Group Note

Additional Requirements for Bidi in HTML & CSS

W3C Working Group Note 21 July 2015

This version: <http://www.w3.org/TR/2015/NOTE-html-bidi-20150721/>
Latest published version: <http://www.w3.org/TR/html-bidi/>
Previous version: <http://www.w3.org/TR/2010/WD-html-bidi-20100304/>
Editor: Aharon Lanin, Google
Richard Ishida, W3C
Copyright © 2013-2015 W3C® MIT, ERCIM, Keio. Behaviors W3C liability trademark and document use rules apply.

Abstract

Authoring a web app that needs to support both right-to-left and left-to-right interfaces, or to take as input and display both left-to-right and right-to-left data, usually presents a number of challenges that make it an especially laborious and bug-prone task. Some of these are due to browser bugs, but some can be traced back to the specification of the bidirectional aspects of a given HTML, or CSS feature. And some of these challenges can be greatly simplified by adding a few strategically placed new HTML, and CSS features.

This document was used to work through and communicate recommendations made to the HTML and CSS Working Groups for some of the most repetitive pain points. It is being published now for the historical record, in order to capture some of the thinking that lay behind the evolution of the specifications and to help people in the future working on bidirectional issues understand the history of the decisions taken. Notes have been added to give a brief summary of what was actually implemented in the HTML or CSS specifications.

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](http://www.w3.org/TR/) at <http://www.w3.org/TR/>.

This document contains initial proposals for features to be added to HTML to support bidirectional text in languages such as Arabic, Hebrew, Persian, Thai, Urdu, etc., and describes the eventual solutions that were adopted by HTML5. This is a W3C Draft produced by the Internationalization Working Group, part of the [W3C Internationalization Activity](#). The Working Group expects to advance this Working Draft to Working Group Note. Please send comments on this document to public-i18n-bidi@w3.org (publicly archived).

This document was published by the [Internationalization Working Group](#) as a Working Group Note. If you wish to make comments regarding this document, please send them to public-i18n-bidi@w3.org (subscribe/archives). All comments are welcome.

Publication as a Working Group Note 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.

TOP RATED RESTAURANTS

Aroma - 3 reviews
פיזה סgoalah - 5 reviews

Aroma - 3 reviews
סgoalah פיזה - 5 reviews

<bdi dir="rtl">

<p dir="rtl">

Bidi in HTML5

W3C Working Group Note

Additional Requirements for Bidi in HTML & CSS

W3C Working Group Note 21 July 2015

This version: <http://www.w3.org/TR/2015/NOTE-html-bidi-20150721/>
Latest published version: <http://www.w3.org/TR/html-bidi/>
Previous version: <http://www.w3.org/TR/2010/WD-html-bidi-20100304/>
Editor: Aharon Lanin, Google
Richard Ishida, W3C
Copyright © 2013-2015 W3C® MIT, ERCIM, Keio. Behaviors W3C liability trademark and document use rules apply.

Abstract

Authoring a web app that needs to support both right-to-left and left-to-right interfaces, or to take as input and display both left-to-right and right-to-left data, usually presents a number of challenges that make it an especially laborious and bug-prone task. Some of these are due to browser bugs, but some can be traced back to the specification of the bidirectional aspects of a given HTML, or CSS feature. And some of these challenges can be greatly simplified by adding a few strategically placed new HTML, and CSS features.

This document was used to work through and communicate recommendations made to the HTML and CSS Working Groups for some of the most repetitive pain points. It is being published now for the historical record, in order to capture some of the thinking that lay behind the evolution of the specifications and to help people in the future working on bidirectional issues understand the history of the decisions taken. Notes have been added to give a brief summary of what was actually implemented in the HTML or CSS specifications.

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](http://www.w3.org/TR/) at <http://www.w3.org/TR/>.

This document contains initial proposals for features to be added to HTML to support bidirectional text in languages such as Arabic, Hebrew, Persian, Thai, Urdu, etc., and describes the eventual solutions that were adopted by HTML5. This is a W3C Draft produced by the Internationalization Working Group, part of the [W3C Internationalization Activity](#). The Working Group expects to advance this Working Draft to Working Group Note. Please send comments on this document to public-i18n-bidi@w3.org (publicly archived).

This document was published by the [Internationalization Working Group](#) as a Working Group Note. If you wish to make comments regarding this document, please send them to public-i18n-bidi@w3.org (subscribe/archives). All comments are welcome.

Publication as a Working Group Note 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.

The names of these states in Arabic **الكويت** and **مصر, البحرين** respectively.

The names of these states in Arabic **الكويت** and **مصر, البحرين** respectively.

string metadata



يتم تحقيق ذلك بإضافة العنصر المضمن `bdi`.

```
<textarea dir="auto"></textarea>
```



في HTML5 يتم تحقيق ذلك بإضافة العنصر المضمن `.bdi`.



string metadata



W3C @w3c · 6h

Call for Review: CSS Fonts Module Level 3 is a W3C Proposed Recommendation ift.tt/2vJzUld



1



5



✉



r12a @r12a · 6h

يتم تحقيق ذلك بإضافة العنصر المضمن `bdi`.



4



4



✉



W3C Developers @w3cd devs · 6h

The first #CSS #Houdini specification to reach #CandidateRecommendation: CSS Painting Level 1 w3.org/TR/2018/CR-css... - this is a major step in the evolution of the extensibility of Web browsers



2



3



✉



string metadata

W3C @w3c · 6h
Call for Review: CSS Fonts Module Level 3 is a W3C Proposed Recommendation [ift.tt/2vJzUld](#)

r12a @r12a · 6h
في HTML5 يتم تحقيق ذلك بإضافة العنصر المضمن .bdi

W3C Developers @w3cd devs · 6h
The first #CSS #Houdini specification to reach #CandidateRecommendation: CSS Painting Level 1 [w3.org/TR/2018/CR-css...](#) - this is a major step in the evolution of the extensibility of Web browsers

string metadata

 في HTML5 يتم تحقيق ذلك بإضافة العنصر المضمن .bdi



W3C @w3c · 6h
Call for Review: CSS Fonts Module Level 3 is a W3C Proposed Recommendation [ift.tt/2vJzUld](#)

r12a @r12a · 6h
يتم تحقيق ذلك بإضافة العنصر المضمن .bdi في HTML5.

W3C Developers @w3cd devs · 6h
The first #CSS #Houdini specification to reach #CandidateRecommendation: CSS Painting Level 1 [w3.org/TR/2018/CR-css...](#) - this is a major step in the evolution of the extensibility of Web browsers

Bidi support for strings

Strings on the Web: Language and Direction Metadata
W3C Editor's Draft 22 July 2020

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/>

Editor:
Addison Phillips (Amazon.com)
Richard Ishida (W3C)

Participate:
[GitHub w3c/string-meta](#)
[File a bug](#)
[Commit history](#)
[Pull requests](#)

Copyright © 2007-2020 W3C® (MIT, ERCIM, Keio, Behring). W3C liability, trademark and permissive document license rules apply.

Abstract
This document describes the best practices for identifying language and base direction for strings 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 of <https://www.w3.org/TR/>.

We welcome comments on this document, but to make it easier to track them, please raise separate issues 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.

GitHub Issues are preferred for discussion of this specification.

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. The group does not expect this document to become a W3C Recommendation. 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.

X تصميم و إنشاء موقع الويب: CSS و HTML

✓ تصميم و إنشاء موقع الويب: HTML و CSS

spec reviews

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

Early review opportunities	Review requested	In review	Awaiting comment resolution	Completed
DCAT-Rev > FPWD (2018-05-08) #57 opened by aphillips FPWD	CSS Writing Modes Level 4 > Ongoing #520 opened by aphillips css-writing-modes	Verifiable Credentials Data Model 1.0 > 2018-09-20 #578 opened by aphillips FPWD needs-review	HTML 5.3 > 2018-05-25 #539 opened by aphillips LC html	ORDL Information Model (+ more) > 2017-05-15 #397 opened by aphillips LC
CSS Logical Properties and Values Level 1 (16 May 2017) #427 opened by r12a FPWD	CSS Painting API Level 1 > 2018-10-09 #581 opened by aphillips LC css-paint-api	CSS Generated Content 2016-06-20 #283 opened by r12a FPWD	CSS Align > 2017-06-30 #422 opened by aphillips LC	Basic Card Payment, 2016-04-21 #269 opened by r12a WD
Resource Timing 2016-04-26 #284 opened by r12a FPWD			TTML2 > 2017-09-30 #428 opened by aphillips CR	Gamepad API > 2018-06-27 #569 opened by LWatson CR
CSS Object Model 2016-03-23 #285 opened by r12a FPWD			Accessibility (A11Y) Requirements for People with Low Vision > 2106-03-17 #285 opened by r12a FPWD	WebVTT #399 opened by r12a CR LC
CSS Ruby #264 opened by r12a WD			Web Authentication, > 2016-08-16 #268 opened by r12a WD	Device and Sensors Reviews (6 docs) > 2017-12-31 #512 opened by aphillips CR
Media Stream Track Content Hints > FPWD (2018-07-03) #576 opened by aphillips media-capture-and...			Browser Payment API > 2017-02-22 #347 opened by aphillips LC	DOM 4.1 > 2018-03-01 #527 opened by xlq CR
			Review request: changes to CR of DNT (tracking-dnt) #487 opened by r12a CR	Semantic Sensor Network Ontology > Urgent #421 opened by aphillips LC
			UTR #53, Unicode Arabic Mark Ordering Algorithm #494 opened by r12a FPWD	ARIA > 2017-04-30 #322 opened by aphillips LC
			Intersection Observer review #405 opened by LWatson FPWD	UI Events KeyboardEvent key Values, > 2016-11-27 #267 opened by r12a LC
				Screen Orientation

Current spec work for Arabic

 Internationalization (i18n)
Making the World Wide Web worldwide!

Learn Find Contact Participate Follow

W3C site search:

Language enablement issue tracker

This page tracks issues related to language support on the Web. Issues listed may be requests to a local community for information about the behaviour of their writing system, or requests for improvements to either specs or browser implementations.

These issues are also linked to from the Language enablement index.

There are 17 issues.

Bidirectional text direction

Should <ins> and have 'unicode-bidi: isolate'?		Jun 8, 2020	949
Possible compatibility problem in 14.3.5 Bidirectional text		Jun 1, 2020	948
input field placeholders should respect the dir attribute when set to 'auto'		Sep 24, 2019	765
Block elements converted to inline should retain isolation		Feb 25, 2019	638
UAs may have somewhat more liberal handling of the dir attribute than the spec does		Oct 13, 2017	506

Text decoration

Add auto value for text-decoration-skip		Jul 20, 2017	434
Minimum width for unskipped lines?		Apr 24, 2017	395

Styling initials

[css-initial] Cursive drop-caps		Apr 30, 2018	543
---------------------------------	--	--------------	-----

Useful links

[Tracker issues on Github](#)
[International text layout and typography index](#)
[Review tracker](#)

Filter results

Click to filter items by type:

- type-info-request
- spec-type-issue
- browser-type-bug
- Clear filters

Click to filter items by layout group:

- afrmreq (Africa)
- alreq (Arabic/Persian) 
- clreq (Chinese)
- elreq (Ethiopic)
- eurreq (Europe)
- hreq (Hebrew)
- lreq (Indic)
- jreq (Japanese)
- kreq (Korean)

Current spec work for Arabic

Line breaking

[css-text-3] Bidi and pre-wrap end of line spaces		Oct 25, 2019	789
[css-text-3] Define break at space to remove a space explicitly		Sep 16, 2019	763

Fonts

[css-fonts] Specify what generic font family maps to nastaliq		Oct 17, 2019	784
---	--	--------------	-----

Lists, counters, etc

[css-lists-3] marker-side: match-parent vs list-style-position: outside match-parent		Aug 22, 2019	730
--	--	--------------	-----

Cursive text

[css-text] Should zero width space break Arabic shaping?		Apr 25, 2019	686
[css-text] Cursive shaping breaks (near an image representing a letter)		Jan 21, 2019	629

Forms & user interaction

input field placeholders should respect the dir attribute when set to "auto"		Sep 24, 2019	765
--	--	--------------	-----

'specdev' guidelines

'specdev' guidelines

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](#)

Copyright © 2014-2018 W3C® (MIT, ERCIM, Keio, Beihang). W3C liability, trademark and permissive document license rules apply.

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 latest 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 though it were 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

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.

[Author, Issue, 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 occurring 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.

ASCII	UTF-8	UTF-16	UTF-32
A9E62F74E6B52E6C5F763C9373D27A6F776E2B6F62E615C6C2C27A76E55D726E746E	49E62F74E6B52E6C5F763C9373D27A6F776E2B6F62E615C6C2C27A76E55D726E746E	49E62F74E6B52E6C5F763C9373D27A6F776E2B6F62E615C6C2C27A76E55D726E746E	49E62F74E6B52E6C5F763C9373D27A6F776E2B6F62E615C6C2C27A76E55D726E746E
Cyrillic	Сириллица	Сириллица	Сириллица
Han	한국어	한국어	한국어
Emoj	Emoji	Emoji	Emoji

First, as the number of bytes per character goes up, the number of characters available inside the byte count limit goes down. ASCII has four times the available characters as emoji, three times as many as Chinese such as Chinese, and roughly twice as many as the Cyrillic example.

Second, in each of the non-ASCII examples the byte boundary for truncation falls in the middle of a character. The resulting "dangling bytes" are rendered as `\FFFD` and the byte sequence itself is not valid UTF-8. If the byte sequence were then serialized into a file format, such as JSON or CSV, the surrounding file might not be wholly valid or the file processor might generate errors because the byte sequence itself is invalid. Unlike many legacy character encodings, UTF-8 is highly patterned, so the longest broken character sequence that can result from mid-character truncation is one character. By contrast, in many legacy encodings, a file or document containing a mid-character truncated string can be wholly changed or rendered unintelligible after that point.

[Author, Issue, Unit](#)

Specifications that limit the length of a string **MUST** specify which type of unit (extended grapheme clusters, Unicode code points, or code units) the length limit uses.

[Author, Issue, Byte Boundary](#)

Specifications that limit the length of a string **SHOULD** specify the length in terms of Unicode code points.

If a specification sets a length limit in code units (such as bytes), it **MUST** specify that truncation can only occur on code point boundaries.

[Author, Issue, Indicator](#)

Developer support



Much work done to support RTL & bidi

Guidelines for developers

In-depth research and advice

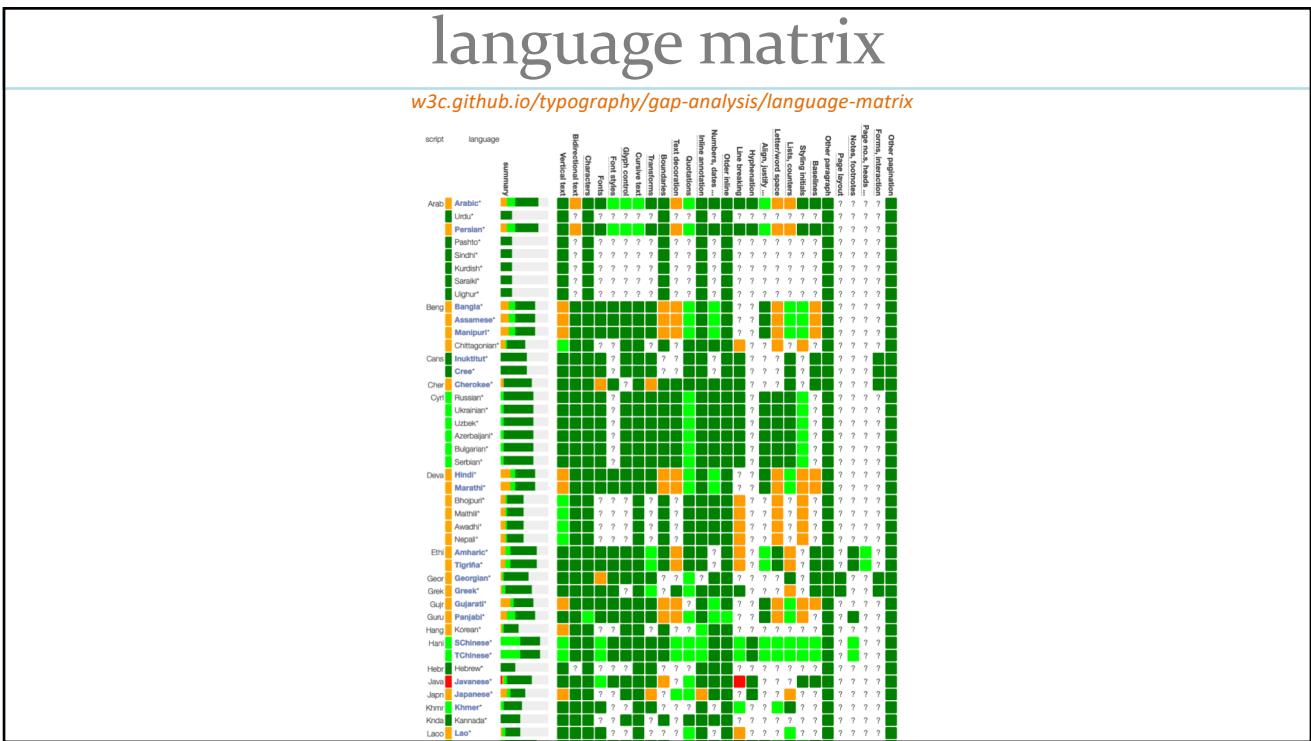
Work still very much in progress



Language Enablement

Understand where the gaps are for users of the Web.

Muscat, Oman



language enablement requirements

TABLE OF CONTENTS

W3C Editor's Draft

1. Introduction
 - 1.1 Gap analysis
 - 1.2 Other related resources
 - 1.3 Language scopes
 - 1.3.1 Standard Arabic language
 - 1.3.2 Persian language
2. Characteristics of the Arabic script
 - 2.1 Fundamental principles
 - 2.1.1 Multi-level baselines
 - 2.1.2 Multi-context joining
 - 2.1.3 Words as groups of letters
 - 2.1.4 Vertical joining
 - 2.1.5 The so-called teeth letters
 - 2.1.6 The Islamic manuscript tradition
 - 2.2 Origins
 - 2.2.1 Writing Styles
3. Characters and phrases
 - 3.1 Characters & encoding
 - 3.2 Fonts
 - 3.2.1 Ligatures
 - 3.2.2 Diacritics
 - 3.2.3 Styling diacritics relative to base characters
 - 3.3 Direction
 - 3.3.1 Bidirectional text
 - 3.3.2 Vertical text
 - 3.3.3 Arabic embedded in vertically orientated text
 - 3.3.4 Upright vertical Arabic text
 - 3.4 Joining
 - 3.4.1 Joining Forms
 - 3.4.2 Joining Categories
 - 3.4.3 Joining Rules
 - 3.4.4 Joining Control
 - 3.4.4.1 Disjoining Enforcement
 - 3.4.4.2 Joining Enforcement
 - 3.4.4.3 Joining-Disjoining Enforcement
 - 3.4.4.4 Context-Based Joining
 - 3.4.5 Joining Segments
 - 3.4.5.1 Closed Joining Segments
 - 3.4.5.2 Open Joining Segments
 - 3.4.6 Non-Joining Characters
 - 3.4.7 Special requirements when dealing with multiple scripts
 - 3.4.7.1 Joining and Intra-Word Spaces
 - 3.4.7.2 Transparency
 - 3.4.7.3 Text border
 - 3.4.7.4 Styling individual letters
 - 3.4.8 Text Segmentation

Text Layout Requirements for the Arabic Script  

W3C Editor's Draft 08 September 2020

This version: <https://wic.github.io/aleq/>

Latest published version: <https://www.w3.org/TR/aleq/>

Latest editor's draft: <https://wic.github.io/aleq/>

Editors: Behnam Esfahani (Quora/Virgule Typeworks)
Moataz Hajzaden
Najla Toumi (Ecole Mohammadia d'Ingénieurs)
Richard Ishida (W3C)
Shevin Afshar (Netflix)
Taha Nakhleh (University of Reading)

Participants: GitHub wic/aleq
File a bug
Commit history
Pull requests

Copyright © 2020-2022 W3C™ MIT ERIC Heiko Behnig W3C liability trademark and permissive document license rules apply.

Abstract

This document describes requirements for the layout and presentation of text in languages that use the Arabic script when they are used by Web standards and technologies, such as HTML, CSS, Mobile Web, Digital Publications, and Unicode.

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](#) or <https://www.w3.org/TR/>.

This document describes the basic requirements for Arabic script layout and text support on the Web and in mobile. These requirements provide information for Web technologies such as CSS, HTML, and digital publications about how to support users of Arabic scripts. Currently the document focuses on Standard Arabic and Persian.

The editor's draft of this document is being developed by the Arabic Layout Task Force, part of the W3C Internationalization Interest Group. It is published by the Internationalization Working Group. The end target for this document is a Working Group Note.

To make it easier to track comments, please raise a separate issue for each comment, and at the start of the issue add a URL pointing to the section you are commenting on.

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

GITHub Issues are preferred for discussion of this specification.

language enablement requirements

§ 3.4.7.2 Transparency

Arabic fonts achieve joining by overlapping letters. A left-joining letter extends out of its bounding box from the left side and a right-joining letter extends out of its bounding box from the right side. Making each letter transparent can expose these overlapping joinings, which should be avoided. Joining the paths of the joined letter into a single shape can remove the overlaps and create the good results.



Figure 35: Applying transparency to Arabic letters should not expose their joining overlaps.

§ 3.4.7.4 Styling individual letters

For educational, technical, or even aesthetic reasons, users might want to apply a specific style to a single letter (or a few letters) in a word. For example, Figure 37 is the logo of the largest telecommunications provider in Oman.



Figure 37: Colour changes across joining characters in the logo for Omantel.

This should not break the letter's joining with its neighbors, as shown in Figure 38.



Figure 38: Applying style to a single letter should not interfere with its joining properties.

text layout index

w3c.github.io/typography/

TABLE OF CONTENTS

- Introduction
- Text direction
 - Vertical text
 - Bidirectional text direction
- Characters & Phrases
 - Fonts
 - Font styles, weight, etc.
 - Glyph shaping & positioning
 - Cursive text
 - Transforming characters
 - Text segmentation & selection
 - Punctuation
 - Text decoration
 - Quotations
 - Inline notes & annotations
 - Numbers & digits
 - Other inline features
- Lines & paragraphs
 - Line breaking
 - Hyphenation
 - Text alignment & justification
 - Word & letter spacing
 - Lists, counters, etc.
 - Styling initials
 - Baselines & inline alignment
- Layout & styling
 - General page layout and progression
 - Grids & tables
 - Notes, footnotes, etc
 - Page headers, footers, etc
 - Forms & user interaction
- Changes Since the Last Published Version

Language enablement index

W3C Editor's Draft 22 November 2019

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/>

Editor: Richard Ishida (W3C)

Participate:

- GitHub w3c/typography
- File a bug
- Commit history
- Put requests

Copyright © 2019 W3C™. MIT. ECRIM, Keio, Beihang. W3C liability, trademark and permissive document license rules apply.

Abstract

This document points browser implementers and specification developers to information about how to support typographic features of scripts or writing systems from around the world, and also points to relevant information in specifications, to tests, and to useful articles and papers. It is not exhaustive, and will be added to from time to time.

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/>.

The information in this document helps to link users and developers so that browsers can better support typographic needs around the world. It is expected that this document will be constantly updated, as new material becomes available or comes to our attention.

NOTE

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 from the dated version of the document.

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

GitHub Issues are preferred for discussion of this specification.

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. The group does not

4.2 Hyphenation

Some scripts don't use hyphenation, those that do have particular rules about how it should be applied that are typically language-specific.

Requirements

- Indic Layout Requirements: [Hyphenation](#)
- Latin Layout Requirements: [Hyphenation-The Classical Rules of Hyphenation and Pagination](#)

GitHub resources

- All issues
- Requests for information
- Spec issues
- Useful discussions
- Type samples

Spec links

- CSS3 Text: [Hyphenation: the hyphens property](#)

Tests

- CSS Text: [Hyphens](#)

Gap analysis

- Dutch
- Hungarian
- Javanese

questions about Arabic

Internationalization (i18n)

Learn Find Contact Participate Follow

Language enablement issue tracker

This page tracks issues related to language support on the Web. Issues listed may be requests to a local community for information about the behaviour of their writing system, or requests for improvements to either specs or browser implementations.

These issues are also linked to the [Language enablement index](#).

There are 11 issues.

Text decoration

How are underlines positioned in Arabic text? Jul 20, 2017 | 467

Vertical text

Vertical text handling in Unicode Sep 7, 2018 | 596

Numbers, etc

Hijri calendar abbreviation for Urdu. Apr 21, 2020 | 886
Why are bidi categories of Arabic-Indic & Eastern Arabic numbers different? Mar 8, 2019 | 647

Hyphenation

Hyphenation in Arabic script writing systems Mar 8, 2019 | 645

Cursive text

Are vertically-misaligned characters joined? Nov 19, 2019 | 82
Is it-test! Should zero width space break Arabic shaping? Apr 25, 2019 | 685
Correct Spacing for LAM before Non-Arabic and BEH Before Number Sep 7, 2018 | 595

Useful links

Tracker issues on GitHub
International text layout and typography index
Review tracker

Filter results

Click to filter items by type:

- type-info-request
- spec-type-issue
- browser-type-bug
- Clear filters

Click to filter items by layout group:

- afrreq (African)
- arreq (Arabic/Persian)
- cireq (Chinese)
- elreq (Ethiopic)
- eurreq (European)
- heireq (Hebrew)
- irreq (Indic)
- jreq (Japanese)
- kreq (Korean)
- lreq (Latin)
- mreq (Mongolian)
- sealreq (Southeast Asia)
- treq (Tibetan)
- Clear filters

questions about Arabic

The screenshot shows the W3C Internationalization (i18n) Language enablement issue tracker. The main navigation bar includes links for Learn, Find, Contact, Participate, and Follow. Below the navigation is a search bar labeled "i18n site search". The main content area is divided into several sections:

- Text decoration:** Issues related to underlines in Arabic text.
- Vertical text:** Issues related to vertical text handling in Unicode.
- Numbers, etc:** Issues related to calendar abbreviations for Urdu and bidi categories of Arabic-Indic & Eastern Arabic numbers.
- Hyphenation:** Issues related to hyphenation in Arabic script writing systems.
- Cursive text:** Issues related to vertically-misaligned characters joined, zero width space break Arabic shaping, and correct spacing for LAM before Non-Arabic and BEH before Number.

On the right side, there are two sections: "Useful links" and "Filter results". The "Useful links" section lists links to Tracker issues on GitHub, International text layout and typography index, and a Review tracker. The "Filter results" section allows users to filter items by type (type-info-request, spec-type-issue, browser-type-bug) and layout group (afrREQ, alegreq, clineq, elreq, eureq, hreq, ireq, jreq, kreq, lreq, mreq, sealreq, treq). There is also a "Clear filters" button.

networks

w3c.github.io/sealreq/

The screenshot shows two GitHub repositories related to Arabic text support:

- w3c/alreq**: This repository contains issues such as "gh-pages", "charter", "gap-analysis", "guidelines", "homepage", "images", "misc", "ignore", "pr-preview.json", "trans.yml", "CODE_OF_CONDUCT.md", "CONTRIBUTING.md", "LICENCE.md", "README.md", "echina", "index.html", "local.css", "shadow.html", and "w3c.json". Many of these issues are labeled as "Draft" or "Open".
- w3c/sealreq**: This repository contains issues such as "gh-pages", "About", "Releases", "Packages", "Contributors", "Environments", and "Languages". Notable issues include "Recommended symbols/signs for Paragraph and Section & general mirroring gaps", "Presentation of Flags with RTL text", "Section on Ellipsis and other Translations", "Section on Abjad", "Add Western Arabic Numerals (ASCII) to A.3 Numerical characters", "Document styling fixes", "Using variable font axis for Arabic joining segment elongation", "Arabic Hamza below with Katra break fonts.", "Vertical text handling in Unicode", "Section on Text Authoring/Editing UX", "Section on Numerical Separators and their Bidi behavior", "Section on Background of Arabic Numerals", "Section on Line Alignment", and "Characters from Presentation Form blocks legitimate to use in text".

Language Enablement



Language matrix

Gap-analysis

Layout requirements

Expert networks

Need more experts!

Author support

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

Wakan, Oman



articles

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

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.

Getting Started

- Getting Started with the W3C i18n site
- Introducing Character Sets and Encodings
- Languages 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

Internationalization Best Practices

On this page

- Getting Started
- Characters
- Language
- Markup & text
- Text direction
- Styling & layout
- Forms
- Navigation
- Web addresses
- Cultural issues
- Other

Getting started

Characters

Language

Markup & text

Text direction

Styling & layout

Forms

Navigation

Web addresses

Cultural issues

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 2011-01-12 13:03:53

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 list of implementation-specific information. This article gives the basic information that common users need to create the more common features of vertical text. It also compares the CSS and practice about what is possible.

Each section explains how you would mark 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 cases 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 should be achievable by applying the `writing-mode` property to the content that you want to be vertical.

In Japanese, the text runs from the right side of the figure box and progress to the left. Latin script has typically run from the top to the bottom clockwise, while the Han characters remain upright. Any graphic also remains upright.

About this article

This article has been reviewed by W3C Internationalization Working Group and has gone through public review. It is a work in progress. If you have comments, please bring them to the [bottom of this page](#).

Figure

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>
```

CSS

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

Text orientation

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](#)

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

Full-width transforms

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

Figure

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>
```

CSS

```
.upright {
  text-transform: full-width;
}
```

Text orientation

Test in your browser: [standard syntax + proprietary syntax](#)

This is appropriate for initials, but is not necessarily useful for all types of upright text, and note especially that this technique only works for Latin characters without accents!

Using full-width characters

Another way to achieve this is to just use fullwidth characters, such as W 3 C. These will automatically be displayed upright by default. You don't need any markup in this case.

Conclusion

Of course, this also only works for Latin script text that doesn't include accents (since those are the only letters for which full-width variants exist).

i18n test suite

w3.org/international/tests

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-rl` 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 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-rl-001` or `writing-mode-rl-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-rl-001	writing-mode:vertical-rl will display a line of text vertically.	pass	pass	pass	fail	pass	fail	fail	fail

vertical-rl

Basics

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

interactive tests

w3.org/international/tests

Interactive test: Vanilla

Set the language

Set font size

Set the font to

Adjust the width of the test box.

Adjust the height of the test box.

Go

مسـتـقـيم

* No properties are set by default.
 * More tests: [List of interactive tests & results](#) · [Character & phrase test repo](#)
 * Browser: Mozilla/5.0 (Macintosh; Intel Mac OS X 10.14; rv:83.0) Gecko/20100101 Firefox/83.0
 * Take a snapshot:

w3c / character_phrase_tests

ZWSP will not interrupt cursive joining behaviour in Arabic text. #26

[Edit](#) [New issue](#)

[r12a](#) opened this issue on 6 Apr · 0 comments

[r12a](#) commented on 6 Apr · edited · Member · [...
ZWPSP](#)

Tests pass if all letters in the word are joined. Try various fonts:

- Gecko: `#` Firefox/74.0 Mac OS X 10.14.8 Fonts: Geozero Pro `X`, Scherhazeze `X`, Amit `X`, Noto Naskh Arabic `X`, Al Bayan `X`, Mishali `X`, Noto Sans Arabic `X`, Arial `X`, Tahoma `X`, Myriad Arabic `X`, Damascus `X`
- Blitz: `<div>ا</div>` Chrome/80.0.3987.149 Mac OS X 10.14.8 Fonts: Geozero Pro `X`, Scherhazeze `X`, Amit `X`, Noto Naskh Arabic `X`, Al Bayan `X`, Mishali `X`, Noto Sans Arabic `X`, Arial `X`, Tahoma `X`, Myriad Arabic `X`, Damascus `X`
- WebKit: `<div>ا</div>` Safari/905.1.5 Mac OS X 10.14.8 Fonts: Geozero `X`, Scherhazeze `X`, Amit `X`, Noto Naskh Arabic `X`, Al Bayan `X`, Mishali `X`, Noto Sans Arabic `X`, Arial `X`, Tahoma `X`, Myriad Arabic `X`, Damascus `X`

Notes:

- `rlm` means 'straight'
- Some fonts don't work on the system that are not user fonts, so this test cannot be run with several fonts (identified in the results using "T"). The Scherhazeze font tested on Safari was a web font.

[r12a](#) added [character](#), [font](#), [font-size](#), [font-family](#), [rlm](#), [zwsp](#) labels on 6 Apr

[r12a](#) mentioned this issue on 6 Apr
 [css-text] Should zero width space break Arabic shaping? w3c/csswg-drafts#3861

[Write](#) [Preview](#)

H B I [...](#) [...](#) [...](#) [...](#) [...](#) [...](#) [...](#) [...](#) [...](#) [...](#) [...](#)

Leave a comment

techniques index

w3.org/International/techniques/authoring-html

W3C Internationalization (i18n)
Making the World Wide Web accessible

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
- ▼ Language
 - ▶ Getting started
 - ▶ Declaring the overall language of a page
 - ▶ Identifying in-document language changes
 - ▼ Choosing language tags
 - 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 xmtlang="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 BCP47.
 - 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

Detailed report

✖ Conflicting character encoding declarations

Explanation

The following character encoding declarations are inconsistent:

a. <meta charset="iso-8859-1"/>
b. <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>

Browsers will apply precedence rules to determine the character encoding to use for the page, but this may not be the encoding you intended.

What to do

Change the character encoding declarations so that they match. Ensure that your document is actually saved in the encoding you choose.

Further reading

Character encodings explained
Choosing a character encoding
Changing the encoding of a document

↑ TOP

- ▶ ✖ Multiple encoding declarations using the `meta` tag
- ▶ ✖ Content-Language `meta` element used
- ▶ ✖ A language attribute value was incorrectly formed
- ▶ ✖ A language subtag is invalid
- ▶ ✖ A `lang` attribute value did not match an `xml:lang` value when they appeared together on the same tag.
- ▶ ⚠ Non-UTF-8 character encoding declared
- ▶ ⚠ Non-preferred name used for legacy character encoding
- ▶ ⚠ Found Unicode code points for directional controls
- ▶ ⚠ Unpaired directional controls found
- ▶ ⚡ `b` tags found with no class attribute

The W3C validators are hosted on server technology donated by HP, and supported by community donations.
Donate and help us build better tools for a better web.



Home About GitHub Feedback Version 2.0.3

Internationalization (I18n) Activity
Making the World Wide Web accessible

COPYRIGHT © 2013-2018 W3C® (MIT, ERCIM, Keio, Beihang). ALL RIGHTS RESERVED.
W3C LIABILITY, TRADEMARK, DOCUMENT USE AND SOFTWARE LICENSING RULES APPLY.
YOUR INTERACTIONS WITH THIS SITE ARE IN ACCORDANCE WITH OUR PUBLIC AND MEMBER PRIVACY STATEMENTS.



Author support



[Internationalisation articles](#)

[Techniques index](#)

[Internationalisation test suite](#)

[Internationalization checker](#)

important message

The W3C isn't a Genie in a lamp that solves all your problems. It brings people together & facilitates work, but this is **your** Web.

To produce change we need people **like you** to step up and provide guidance and work through issues.



Punakha, Bhutan

Get involved and help us ensure that it meets local needs around the world.

How to help?

If you know of experts who can help us improve the Web for RTL support and for Arabic typography, please let us know.

If you are able, please consider supporting the work through the Internationalization Sponsorship program.

Use and tell people about the work we're doing, and the educational materials we produce to help RTL content development.



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 the last decade, but as the number of users from non-Latin-script countries 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 concerted effort by governments, industry, 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 assist the W3C in its efforts to support the internationalization of 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 met. This is particularly true for the Arabic script, which is an area of particular interest, and for other scripts such as Chinese. It also wants to ensure that the Web is accessible for the breaking & justification, logical approaches to expressing emphasis or styling, and other features of character styles, supporting behavioral text in multiple directions, and other features such as page layout, and so on. These typographic conventions are often very different from the conventions used in print media that use writing systems such as Arabic, Devanagari, Thai, Mongolian and so forth.

In W3C's work on digital publishing, it also wants to achieve better integration with W3C standards for rendering text (such as CSS) and for the presentation of text in print (such as HTML). The goal here is to ensure that the Web supports the needs of the print industry, and vice versa, around the world as we use to, and encourage the use of traditional printing techniques alongside modern digital printing.

The "language enablement" captures an overall set of goals and activities for this purpose. To achieve this goal, the W3C needs to assess current support for various languages, identify gaps, prioritize them, develop requirements, and then work with the community to close them. To do so, it needs to establish a network of experts, define a set of specific 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 creators of specifications, of system-

