
Safari HTML Reference

[Apple Applications](#) > Safari



2008-09-09



Apple Inc.
© 2008 Apple Inc.
All rights reserved.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, mechanical, electronic, photocopying, recording, or otherwise, without prior written permission of Apple Inc., with the following exceptions: Any person is hereby authorized to store documentation on a single computer for personal use only and to print copies of documentation for personal use provided that the documentation contains Apple's copyright notice.

The Apple logo is a trademark of Apple Inc.

Use of the "keyboard" Apple logo (Option-Shift-K) for commercial purposes without the prior written consent of Apple may constitute trademark infringement and unfair competition in violation of federal and state laws.

No licenses, express or implied, are granted with respect to any of the technology described in this document. Apple retains all intellectual property rights associated with the technology described in this document. This document is intended to assist application developers to develop applications only for Apple-labeled computers.

Every effort has been made to ensure that the information in this document is accurate. Apple is not responsible for typographical errors.

Apple Inc.
1 Infinite Loop
Cupertino, CA 95014
408-996-1010

Apple, the Apple logo, Mac, Mac OS, and Safari are trademarks of Apple Inc., registered in the United States and other countries.

iPhone is a trademark of Apple Inc.

Java and all Java-based trademarks are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

Simultaneously published in the United States and Canada.

Even though Apple has reviewed this document, APPLE MAKES NO WARRANTY OR REPRESENTATION, EITHER EXPRESS OR IMPLIED, WITH RESPECT TO THIS DOCUMENT, ITS QUALITY, ACCURACY, MERCHANTABILITY, OR FITNESS FOR A PARTICULAR PURPOSE. AS A RESULT, THIS DOCUMENT IS

PROVIDED "AS IS," AND YOU, THE READER, ARE ASSUMING THE ENTIRE RISK AS TO ITS QUALITY AND ACCURACY.

IN NO EVENT WILL APPLE BE LIABLE FOR DIRECT, INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT OR INACCURACY IN THIS DOCUMENT, even if advised of the possibility of such damages.

THE WARRANTY AND REMEDIES SET FORTH ABOVE ARE EXCLUSIVE AND IN LIEU OF ALL OTHERS, ORAL OR WRITTEN, EXPRESS OR IMPLIED. No Apple dealer, agent, or employee is authorized to make any modification, extension, or addition to this warranty.

Some states do not allow the exclusion or limitation of implied warranties or liability for incidental or consequential damages, so the above limitation or exclusion may not apply to you. This warranty gives you specific legal rights, and you may also have other rights which vary from state to state.

Contents

Introduction to Safari HTML Reference 13

Organization of This Document 13

Standard HTML 15

Standard HTML Tags 15

- a 15
- abbr 15
- acronym 15
- address 16
- applet 16
- area 16
- audio 16
- b 17
- base 17
- basefont 17
- bdo 18
- big 18
- blockquote 18
- body 18
- br 19
- button 19
- caption 19
- center 19
- cite 19
- code 20
- col 20
- colgroup 20
- dd 20
- del 21
- dfn 21
- dir 21
- div 21
- dl 22
- dt 22
- em 22
- fieldset 22
- font 23
- form 23
- frame 23
- frameset 23

h1 - h2 - ... - h6 24
head 24
hr 24
html 24
i 24
iframe 25
img (or image) 25
input 25
ins 25
isindex 26
kbd 26
label 26
legend 26
li 27
link 27
listing 27
map 27
menu 27
meta 28
noframes 28
noscript 28
object 29
ol 29
optgroup 29
option 29
p 30
param 30
pre 30
q 30
s 30
samp 31
script 31
select 31
small 31
source 32
span 32
strike 32
strong 33
style 33
sub 33
sup 33
table 34
tbody 34
td 34
textarea 34
tfoot 34

- th 35
- thead 35
- title 35
- tr 35
- tt 36
- u 36
- ul 36
- var 36
- video 36
- Supported Attributes 37
 - abbr 37
 - accept-charset 37
 - accept 37
 - accesskey 38
 - action 38
 - align 38
 - alink 38
 - alt 38
 - archive 38
 - autocapitalize 38
 - autocomplete 38
 - autocorrect 39
 - autosave 39
 - axis 39
 - background 39
 - behavior 39
 - bgcolor 39
 - bgproperties 40
 - border 40
 - bordercolor 40
 - cellpadding 40
 - cellspacing 40
 - char 40
 - challenge 40
 - charoff 40
 - charset 41
 - checked 41
 - cellborder 41
 - cite 41
 - class 41
 - classid 41
 - clear 41
 - code 41
 - codebase 42
 - codetype 42
 - color 42

cols 42
colspan 42
compact 42
composite 42
content 42
contenteditable 42
coords 43
data 43
datetime 43
declare 43
defer 43
dir 43
direction 43
disabled 43
enctype 44
face 44
for 44
frame 44
frameborder 44
headers 44
height 44
hidden 44
href 44
hreflang 45
hspace 45
html 45
http-equiv 45
id 45
incremental 45
ismap 45
keytype 45
label 45
lang 46
language 46
left 46
leftmargin 46
link 46
longdesc 46
loop 46
marginheight 46
marginwidth 47
max 47
maxlength 47
mayscript 47
media 47
method 47

min 47
multiple 48
name 48
nohref 48
noresize 48
nosave 48
noshade 48
nowrap 48
object 48
onabort 49
onbeforecopy 49
onbeforecut 49
onbeforepaste 49
onblur 49
onchange 49
onclick 49
oncontextmenu 50
oncopy 50
oncut 50
ondblclick 50
ondrag 50
ondragend 50
ondragenter 51
ondragleave 51
ondragover 51
ondragstart 51
ondrop 51
onerror 51
onfocus 52
oninput 52
onkeydown 52
onkeypress 52
onkeyup 52
onload 52
onmousedown 52
onmousemove 53
onmouseout 53
onmouseover 53
onmouseup 53
onpaste 53
onreset 53
onresize 53
onscroll 53
onsearch 54
onselect 54
onselectstart 54

onsubmit 54
onunload 54
oversrc 54
pagex 54
pagey 55
placeholder 55
plain 55
pluginpage 55
pluginspage 55
pluginurl 55
precision 56
profile 56
prompt 56
readonly 56
rel 56
results 56
rev 56
rows 56
rowspan 56
rules 57
scheme 57
scope 57
scrollamount 57
scrolldelay 57
scrolling 57
selected 57
shape 58
size 58
span 58
src 58
standby 58
start 58
style 58
summary 58
tabindex 58
tableborder 59
target 59
text 59
title 59
top 59
topmargin 59
truespeed 59
type 60
unknown 60
usemap 60
valign 60

- value 60
- valuetype 60
- version 60
- visibility 60
- vlink 61
- vspace 61
- width 61
- wrap 61
- z-index 61
- Standard Input Type Values 61
 - text 61
 - password 62
 - checkbox 62
 - radio 62
 - submit 62
 - reset 62
 - file 62
 - hidden 62
 - image 62
 - button 62
 - search 63
 - range 63

HTML Extensions 65

- Additional HTML Elements 65
 - canvas 65
 - embed 65
 - keygen 65
 - layer 66
 - marquee 66
 - nobr 66
 - noembed 66
 - nolayer 67
 - plaintext 67
 - wbr 67
 - xmp 67
- Additional meta Tag Keys 68
 - viewport 68

Document Revision History 71

Tables

HTML Extensions 65

Table 1	Viewport properties	68
Table 2	Special viewport property values	69

Introduction to Safari HTML Reference

You can use Hypertext Markup Language (HTML) in conjunction with other web content technologies to deploy cross-platform media-rich interactive content to a variety of sources. HTML documents—which can be anything from a textual product description to a photo library to an interactive form—can be read by web browsers on every common platform, displayed and interacted with on portable digital devices, and integrated into Web Kit-based applications on OS X, along with a variety of other technologies.

This document details every HTML tag and property supported by Web Kit and Safari. You should read this if you are developing web content that will be displayed in Safari or within a Web Kit-based application.

Organization of This Document

The following articles describe key aspects of Safari's HTML support:

- [“Standard HTML”](#) (page 15) describes all the standard HTML tags (as defined by the World Wide Web Consortium, or W3C) supported by Safari. It also lists all the supported properties and attributes.
- [“HTML Extensions”](#) (page 65) describes all the non-standard extensions supported by Safari, including some supported by other browsers and others that are unique to Safari.

Standard HTML

Safari and the Web Kit implement a large subset of the HTML 4.01 Specification defined by the World Wide Web Consortium (W3C). This reference defines every symbol in the specification that Safari implements. If a tag is not listed here or in “[HTML Extensions](#)” (page 65), it is not supported by Safari and the Web Kit. Some tags are also marked as deprecated, which means they are supported by Safari, but since they are no longer supported by the HTML specification are not guaranteed to be supported in the future.

Standard HTML Tags

a

Used to specify a hyperlink or a page anchor.

Example

```
<a href="URL">content</a>
```

When the `href` attribute is used with this tag, the text or image specified by `content` becomes a hyperlink, linked to the URL specified by `URL`. When the `name` attribute is used, the tag becomes an anchor which can be linked to by a hyperlink.

abbr

Used to specify an abbreviated form of a string.

Example

```
<abbr title = "fullstring"> abbrev </abbr>
```

In Safari, the string specified by `abbrev` is displayed on screen while the content of `fullstring` is revealed in tooltip form by holding the mouse over the abbreviated value. This tag is also useful for applications that read the underlying HTML code of a page, such as screen readers.

acronym

Used to specify the acronym form of a string.

Example

```
<acronym title = "fullstring"> acro </acronym>
```

In Safari, the string specified by `acro` is displayed on screen while the content of `fullstring` is revealed in tooltip form by holding the mouse over the acronym itself. This tag is also useful for applications that read the underlying HTML code of a page, such as screen readers.

address

Used to specify a street address.

Example

```
<address>
  streetaddress
</address>
```

This tag specifies a street address. The address enclosed within the tags will be italicized. Line breaks (such as ones between a street address and a city/state/zip) are not automatically inserted.

applet

Used to embed a Java applet within a page.

Example

```
<applet height="value" width="value" archive="URL">
<applet height="value" width="value" code="URL">
```

The applet will be displayed at the location of the tag in the page, with a height specified by `height` and a width specified by `width`. The location of the applet is given by the URL specified by `archive` if the applet is stored in a Java archive or zip file, or `code` if the applet is in a standard java class file.

Important: This tag has been deprecated in the HTML 4.01 standard. You should use the `<object>` tag to embed Java applets unless you have a specific reason to use this tag.

area

Used to specify a specific area within an image map.

Example

```
<area shape ="shapetype" coords ="coords" href ="URL">
```

This tag defines discrete areas within an image map (defined by an enclosing `<map>` tag). The area defined by this tag will act as a hyperlink, linked to the URL specified by `URL`, bounding shape specified by `shape` and coordinates specified by `coords`.

audio

Used to embed audio into a webpage.

Example

```
<audio src="url"  
  autoplay="true"  
  start="00:00:00.00"  
  loopstart="00:00:00.07" <!-- 7 seconds -->  
  loopend="00:00:00.19"  
  end="00:00:00.27"  
  playcount="4" <!-- play 4x -->  
  controls="true" >
```

The `audio` element may contain fallback content for browsers that do not support this element. Any content enclosed within the `audio` element is ignored by browsers that support the `audio` element (but it must be valid HTML).

The `audio` element supports inclusion of `source` elements to provide multiple versions of an audio clip encoded with different codecs, at different bit rates, and so on. These `source` elements must be the first elements inside the `audio` element before any fallback content. See [“source”](#) (page 32) for more information.

Availability: Available in Safari 3.1 and later.

Support Level: HTML 5

b

Used to display text in a bold style.

Example

```
<b> content </b>
```

The text specified by `content` will be displayed in the bold style but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

base

Used to define the base URL for all linked objects on page.

Example

```
<base href ="URL">
```

The URL specified by `href` will act as the base URL for any relatively-linked object—such as an image, hyperlink, or Java applet—on the page. If a URL is specified absolutely (with a fully-qualified URL), it will not be affected by this tag. This tag must be placed in the `head` section of a page.

basefont

Used to specify the base font for a page.

Example

```
<basefont color ="color" face ="face" size ="size">
```

The font is used as the default font for the page unless otherwise specified. The font is specified by `face`, its size is specified by `size`, and its color is specified by `color`. These attributes and their various options are defined in .

Important: This tag has been deprecated in the HTML 4.01 standard. You should use CSS styling to set this property for the enclosing elements unless you have a specific reason to use this tag.

bdo

Used to display text in a different direction.

Example

```
<bdo dir ="dir"> content </bdo>
```

The text specified by `content` will be displayed left-to-right if `dir` is set to "ltr"; it will be displayed right-to-left if it is set to "rtl."

big

Used to display text in a large size.

Example

```
<big> content </big>
```

The text specified by `content` will be displayed in a larger size but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

blockquote

Used to display text in an indented quotation style.

Example

```
<blockquote> content </blockquote>
```

The text specified by `content` will be indented (on both sides of the text block), but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

body

Defines the entirety of the document body.

Example

```
<body> content </body>
```

The content specified by `content` comprises most of the content of the page. Though you can specify style attributes within this tag, this behavior has been deprecated in HTML 4.01 and should be replaced with CSS style attributes.

br

Represents a single line break.

Example

```
<br>
```

button

Defines an interactive button on the page.

Example

```
<button> content </button>
```

The text specified by `content` will be displayed within the frame of the button. This differs from the “button” input type in that you can specify content within the button tags.

caption

Defines a caption for an HTML table.

Example

```
<caption> content </caption>
```

The text specified by `content` will be displayed as a caption for the table in which it is enclosed.

center

Defines a region of content to be centered.

Example

```
<center> content </center>
```

The content specified by `content` will be centered within its enclosing element.

Important: This tag has been deprecated in the HTML 4.01 standard. Styles should be more finely tuned using CSS instead of using HTML style tags.

cite

Used to specify a citation.

Example

```
<cite> content </cite>
```

This tag specifies a citation. The text enclosed within the tags will be italicized.

code

Used to specify text as computer code.

Example

```
<code> content </code>
```

This tag specifies a block of code. The text enclosed within the tags will use a “tele-type” monospaced character font.

col

Used to specify attributes of columns in a table.

Example

```
<col properties >
```

This tag allows you specify attributes for a given table column, with those attributes specified by `properties`. A series of `<col>` tags must be placed in order of the actual table columns. For example, to set center column alignment for a columns, you would use `<col align="center">`. These must be placed within a table or a `colgroup`.

colgroup

Used to specify attributes for multiple columns in a table.

Example

```
<colgroup properties ></colgroup>
```

This tag allows you specify attributes for multiple table columns, with those attributes specified by `properties`. For example, to set center column alignment for three different columns, you would use `<colgroup span="3" align="center">`. These must be placed within a table.

dd

Used to specify a definition for a term.

Example

```
<dd> content </dd>
```

This tag specifies a definition for a term within an HTML definition list. The text enclosed within the tags will be indented under the term specified by the enclosing `<dt>` block.

del

Used to specify a block of deleted text.

Example

```
<del> content </del>
```

This tag specifies a block of deleted text, which will be marked with a crossbar.

dfn

Used to specify a definition.

Example

```
<dfn> content </dfn>
```

This tag specifies a definition of any sort.

dir

Used to specify a directory list.

Example

```
<dir>  
  
<li> content </li>  
  
</dir>
```

This tag specifies a directory list, each element of which is specified by an `` tag..

Important: This tag has been deprecated in the HTML 4.01 standard. List styles should be more finely tuned using CSS instead of using HTML style tags, and the structure should be defined instead with the `` and `` tags.

div

Used to specify a styleless section in a document.

Example

```
<div> content </div>
```

This tag specifies a section in a document, as a block element. Multiple divs will stack vertically on the page. Use CSS styles to tune the style properties of this element.

dl

Used to specify a definition list.

Example

```
<dl> content </dl>
```

This tag specifies a definition list. Within the bounds of this block, terms to be defined should be marked using the `<dt>` tag, and their definitions should be marked using the `<dd>` tag,

dt

Used to specify a definition term.

Example

```
<dt> content </dt>
```

This tag specifies a definition term. It should be used to mark an actual term within the bounds of a definition list (`<dl>`). Definitions should follow each term, and be marked using the `<dd>` tag,

em

Used to specify emphasized text.

Example

```
<em> content </em>
```

This tag specifies a block of emphasized text. Styles should be more finely tuned using CSS instead of using HTML style tags.

fieldset

Used to specify a set of fields.

Example

```
<fieldset>  
  
  caption input  
  
  caption input  
  
</fieldset>
```

This tag encloses a set of input fields, and will draw a box around them. The fields themselves are made with input tags specified by `input` and the name of the field is plaintext specified by `caption` .

font

Defines a font style for the content the tag encloses.

Example

```
<font> content </font>
```

The content specified by `content` will be altered based on a variety of properties, such as `face`, `size`, and `color`.

Important: This tag has been deprecated in the HTML 4.01 standard. Styles should be more finely tuned using CSS instead of using HTML style tags.

form

Used to specify an HTML form.

Example

```
<form> formContent </form>
```

This tag specifies a form on a page. Each individual form (with its variety of inputs such as checkboxes, text fields, and password fields) should be enclosed in its own form tag set. If using the form for some kind of submission, the form's submit button should also be enclosed within this tag set.

frame

Used to specify an individual frame.

Example

```
<frame src ="URL">
```

This tag specifies an individual frame within a frameset. The URL for the frame is specified by `src`.

frameset

Used to specify a frameset.

Example

```
<frameset>
```

This tag specifies the overall frameset for a number of frames (each specified with the `<frame>` tag. The URL for the frame is specified by `src`. The size of each column should be specified by the `cols` and `rows` properties.

h1 - h2 - ... - h6

Used to specify various headers.

Example

```
<h#> content </h#>
```

This tag specifies a block of header text, with `<h1>` representing the largest font size and `<h6>` representing the smallest. Styles should be more finely tuned using CSS instead of using HTML style tags.

head

Used to specify meta-information about the HTML document.

Example

```
<head> content </head>
```

This tag can contain a number of informational tags, such as `<title>` for the page title or `<style>` for a CSS definition block.

hr

Used to specify a horizontal line.

Example

```
<hr>
```

This tag specifies a horizontal line. Styles should be more finely tuned using CSS instead of using HTML style tags.

html

Used to specify the HTML document.

Example

```
<html>
```

This tag specifies an HTML document, and should encompass all the content of the page.

i

Used to display text in an italic style.

Example

```
<i> content </i>
```

The text specified by `content` will be displayed in the italic style but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

iframe

Used to display a URL in an inline frame.

Example

```
<iframe src ="URL"></iframe>
```

The URL specified by `src` will load into an inline frame placed wherever the `iframe` is entered.

img (or image)

Used to display an inline image.

Example

```
<img src ="URL">
```

The image file specified by `src` will be displayed inline in the enclosing element.

input

Used to display an input for an HTML form.

Example

```
<input type ="type">
```

This tag specifies some kind of input mechanism in an HTML form. The type specified by `type` can be one of the following: `button` for a basic button; `checkbox` for a checkbox element; `file` for a file upload interface; `hidden` for an invisible input type; `password` for a shielded password field; `radio` for a radio button element; `reset` for a form-reset button; `submit` for a form-submit button; or `text` for a standard text field.

Safari on iPhone extends the `input` tag with two additional properties, `autocorrect` and `autocapitalize`, described in “[Standard Attributes](#)” (page 37).

ins

Used to specify a block of inserted text.

Example

```
<ins> content </ins>
```

This tag specifies a block of inserted text, which will be marked with an underline.

isindex

Used to specify an index field.

Example

```
<isindex prompt ="prompt">
```

This tag displays a prompt and a search field, which as a form will submit the value using a GET request. The default prompt in Safari is “This is a searchable index. Enter search keywords:”, but this can be overridden by specifying a string for the `prompt` attribute.

Important: This tag has been deprecated in the HTML 4.01 standard. You should use standard HTML input types to process queries instead of using this tag.

kbd

Used to specify text as keyboard text.

Example

```
<kbd> content </kbd>
```

This tag specifies a block of keyboard text. The text enclosed within the tags will use a “tele-type” monospaced character font.

label

Used to specify a label for input controls.

Example

```
<label for ="id"> content </label>
```

This tag specifies a label for the input control whose name is specified by `for` . The text specified by `content` makes up the body of the label.

legend

Used to specify the caption for a fieldset.

Example

```
<legend> content </legend>
```

This tag specifies the label for a fieldset (specified by the `<fieldset>` tag). The caption specified by `content` is merged with the box surrounding the fieldset.

li

Used to specify a list element.

Example

```
<li> content </li>
```

Within a list block (specified by `` for an unordered list, or `` for an ordered list), this tag specifies single list element, whose content is specified by `content`. List styles should be more finely tuned using CSS instead of using HTML style tags.

link

Used to specify a connection to an external file.

Example

```
<link href = "URL">
```

This tag specifies an external file which is related to the HTML document it is enclosed in. For example, you should use this tag in the head of an HTML document to specify an external CSS stylesheet.

listing

Deprecated, equivalent to “[pre](#)” (page 30).

Example

```
<listing>
```

This tag is supported for backwards compatibility only. You should not use it in new documents.

Support Level: Deprecated in HTML 3.2, not available in HTML 4 and later.

map

Used to specify a browser-processed image map.

Example

```
<map name = "id" id = "id">
```

This tag encloses the `area` elements that define the regions of an image map. The identifier specified by `id` and by `name` should be used by an `` element's `usemap` property.

menu

Used to specify a menu list.

Example

```
<menu> content </menu>
```

This tag specifies a definition list. Within the bounds of this block, terms to be defined should be marked using the `<dt>` tag, and their definitions should be marked using the `<dd>` tag,

Important: This tag has been deprecated in the HTML 4.01 standard. List styles should be more finely tuned using CSS instead of using HTML style tags, and the structure should be defined instead with the `` and `` tags..

meta

Used to specify meta-information about an HTML page.

Example

```
<meta name ="title" content ="content">
```

This tag specifies a list of meta-information about a page, such as keywords for a search engine to index. The title specified by `name` defines what meta-information you are displaying. The text specified by `content` is the actual meta-information.

In addition to the standard meta values, Safari on iPhone also extends the meta tag with the following additional keys:

- `viewport`—defines the desired width for content rendering, as described in [Configuring the Viewport and “viewport”](#) (page 68).

- `format-detection`—enables or disables automatic detection of possible phone numbers in a webpage, as described in [Using iPhone Application Links](#).

For more information about these extensions, see *Safari Web Content Guide for iPhone*.

noframes

Used to specify content to display to non-frames-compliant browsers.

Example

```
<noframes> content </noframes>
```

This tag specifies a block of content that will display to browsers who do not support frames or have them deactivated.

noscript

Used to specify content to display to a browser that doesn't run scripts.

Example

```
<noscript> content </noscript>
```

This tag specifies a block of content that will display to browsers who do not support execution of scripts or have them deactivated.

object

Used to embed an object within a page.

Example

```
<object height="value" width="value" archive="URL" data="URL">content</object>  
<object height="value" width="value" data="URL" codebase="URL">content</object>
```

The object will be displayed at the location of the tag in the page, with a height specified by `height` and a width specified by `width`.

The location of the object is given by the URL specified by `archive` for a Java archive, `data` for some arbitrary embedded data (an image, for example), or `codebase` for object code of any other type.

ol

Used to specify an ordered list.

Example

```
<ol> content </ol>
```

This tag specifies an ordered, numbered list. Within the bounds of this block, list items should be defined using the `` tag.

optgroup

Used to specify a group of options.

Example

```
<optgroup label ="label"> options </optgroup>
```

Within a `select` input type, this tag specifies a subgroup of options. Within the bounds of this block, individual options are specified using the `<option>` tag. The title of the subgroup is specified by `label`, and in Safari is displayed as bold grey text, with its associated options indented under it.

option

Used to specify a list option.

Example

```
<option value ="value"> title </option>
```

Within a `select` input type, this tag specifies a single selectable option. The form value of the option is specified by `value`, and its visible name is specified by `title`. These can be placed directly within a `select` input type or within an `optgroup` within it.

p

Used to display a paragraph.

Example

```
<p> content </p>
```

This tag is used to indicate a paragraph in the document.

param

Represents a parameter for an `object` declaration.

Example

```
<param name ="name" value ="value">
```

This tag represents a specific parameter for an embedded `object` element. You can place any number of these , but they must be enclosed within the `<object>` block. The parameter's name/key is specified by `name` and its value is specified by `value` .

pre

Represents a block of pre-formatted text.

Example

```
<pre> content </pre>
```

This tag preserves the formatting of the block of text specified by `content` , specifically line breaks and multiple spaces (normal text operation in Safari displays no difference between a single space and multiple consecutive spaces). In Safari, text enclosed in this element is also rendered in a monospace "tele-type" font.

q

Used to display an inline quotation.

Example

```
<q> content </q>
```

The text specified by `content` will be displayed in quotes but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags. Para

S

Defines a block of text in strikethrough style.

Example

```
<s> content </s>
```

The content specified by `content` will be rendered with a crossbar.

Important: This tag has been deprecated in the HTML 4.01 standard. The `` tag is more appropriate for this function. Styles should be more finely tuned using CSS instead of using HTML style tags.

samp

Used to specify text as sample code.

Example

```
<samp> content </samp>
```

This tag specifies a block of code. The text enclosed within the tags will use a “tele-type” monospaced character font.

script

Used to embed and execute script code.

Example

```
<script type = "mimetype"> code </script>
```

This tag specifies a block of script code, such as JavaScript. The code specified by `code` will be invisible onscreen, but will be visible in the page source. Code embedded within script tags (unless defined inside functions) is executed immediately on page load. The MIME type of the script should be specified by `type`.

select

Used to specify a selection input type.

Example

```
<select> options </select>
```

This tag specifies a selection menu. This block must contain a set of `option` elements or `optgroup` elements containing options. In Safari, if the `size` property is explicitly set for this tag, the input box will resemble a Mac OS X combo box, otherwise it will resemble a pop-up menu.

small

Used to display text in a small size.

Example

```
<small> content </small>
```

The text specified by `content` will be displayed in a smaller size but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

source

Provides a resource URI for a multimedia element such as audio or video.

Example

```
<video poster="bananas.png" ... >
  <source
    src="bananas.mp4"
    type="video/mp4; codecs=&quot;avc1.42E01E, mp4a.40.2&quot;;"
    media="screen"
    pixelration="1.78"  <!-- 16:9 -->
  >
  </source>
  <source ...></source>
  <source ...></source>

  <!-- Fallback content for browsers that do not support the video tag goes
here. -->

</video>
```

Web developers should take care to specify type and codec information appropriately. Browsers use this information to choose the media that is most appropriate according to available codecs, screen resolution, and so on.

Availability: Available in Safari 3.1 and later.

Support Level: HTML 5

span

Used to specify an inline styleless section in a document.

Example

```
<span> content </span>
```

This tag specifies a section in a document. Multiple consecutive spans will be placed horizontal on the page by default. Use CSS styles to tune the style properties of this element.

strike

Defines a block of text in strikethrough style.

Example

```
<strike> content </strike>
```

The content specified by `content` will be rendered with a crossbar.

Important: This tag has been deprecated in the HTML 4.01 standard. The `` tag is more appropriate for this function. Styles should be more finely tuned using CSS instead of using HTML style tags.

strong

Used to specify text as “strong” emphasized text.

Example

```
<strong> content </strong>
```

This tag specifies a block of emphasized text. Styles should be more finely tuned using CSS instead of using HTML style tags.

style

Used to define an inline stylesheet.

Example

```
<style type =“mimetype”> css_declarations </style>
```

This tag specifies a CSS stylesheet within the page. All CSS declarations should be placed within this block. This tag should be placed in the `head` section of a page. If you are linking to an external stylesheet, use the `link` element instead.

sub

Used to specify text as subscript.

Example

```
<sub> content </sub>
```

The text specified by `content` will be displayed in a smaller size and will be subscripted, but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

sup

Used to specify text as superscript.

Example

```
<sup> content </sup>
```

The text specified by `content` will be displayed in a smaller size and will be superscripted, but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

table

Used to define a data table.

Example

```
<table> content </table>
```

This tag defines a table structure for a page. The HTML specified by `content` should contain the other structural elements such as table rows (`<tr>`) and table cells (`<td>`).

tbody

Used to define a table's body.

Example

```
<tbody> content </tbody>
```

This tag defines the body for a table. It is only a structural definition and by default does not render anything unique, so the HTML specified by `content` should contain the other structural elements such as table rows (`<tr>`) and table cells (`<td>`).

td

Used to define a table cell.

Example

```
<td> content </td>
```

This tag defines a cell within a table. Cells are usually enclosed by table row (`<tr>`) definitions. Consecutive table cells will be placed horizontally onscreen.

textarea

Used to specify a text area input type.

Example

```
<textarea rows ="value" cols ="value"> content </select>
```

This tag specifies a scrollable, multi-line text input block. You can specify its size onscreen by specifying values for `rows` and `cols` .

tfoot

Used to define a table's footer.

Example

```
<tfoot> content </tfoot>
```

This tag defines a footer for a table. It is only a structural definition and by default does not render anything unique, so the HTML specified by `content` should contain the other structural elements such as table rows (`<tr>`) and table cells (`<td>`).

th

Used to define header text for a table column.

Example

```
<th> content </th>
```

Table headers should be placed in their own table row, usually the first in a table definition, and should correspond in number to table cell definitions in later rows. In Safari, the text specified by `content` will be displayed in a bold face.

thead

Used to define a table's header.

Example

```
<thead> content </thead>
```

This tag defines a header for a table. It is only a structural definition and by default does not render anything unique, so the HTML specified by `content` should contain the other structural elements such as table rows (`<tr>`) and table cells (`<td>`).

title

Used to define the visible window title for the page.

Example

```
<title> content </title>
```

The text specified by `content` will be displayed at the top of a browser window, but will remain invisible in embedded Web Kit web views unless requested programatically. This tag must be placed in the `head` section of a page.

tr

Used to define a table row.

Example

```
<tr> content </tr>
```

This tag defines a row within a table. Table cells are usually enclosed by these rows. Consecutive table rows will be placed vertically onscreen.

tt

Used to display text in a “tele-type” style.

Example

```
<tt> content </tt>
```

The text specified by `content` will be displayed in a monospaced “tele-type” style, but otherwise will match the style of the enclosing element. Styles should be more finely tuned using CSS instead of using HTML style tags.

u

Defines a block of underlined text.

Example

```
<u> content </u>
```

The content specified by `content` will be underlined.

Important: This tag has been deprecated in the HTML 4.01 standard. The `<ins>` tag is more appropriate for this function. Styles should be more finely tuned using CSS instead of using HTML style tags. Additionally, underlined text should not be used as it may be confused with actual hyperlinks.

ul

Used to specify an unordered list.

Example

```
<ul> content </ul>
```

This tag specifies an unordered, bulleted list. Within the bounds of this block, list items should be defined using the `` tag.

var

Used to specify a variable.

Example

```
<var> content </var>
```

This tag specifies a variable. The text enclosed within the tags will be italicized.

video

Used to embed video into a webpage.

Example

```

<video src="url"
  poster="freezeimage.png"
  autoplay="true"
  start="00:00:00.00"
  loopstart="00:00:00.07" <!-- 7 seconds -->
  loopend="00:00:00.19"
  end="00:00:00.27"
  playcount="4" <!-- play 4x -->
  controls="true"
  width="640"
  height="480"
>

```

The `video` element may contain fallback content for browsers that do not support this element. Any content enclosed within the `video` element is ignored by browsers that support the `audio` element (but it must be valid HTML).

The `video` element supports inclusion of `source` elements to provide multiple versions of a video clip encoded with different codecs, at different bit rates, and so on. These `source` elements must be the first elements inside the `video` element before any fallback content. See [“source”](#) (page 32) for more information.

Availability: Available in Safari 3.1 and later.

Support Level: HTML 5

Supported Attributes

The following table defines all the attributes that are supported by Safari and the Web Kit. This reference defines every symbol in the specification that Safari implements. If an attribute is not listed here, it is not supported by Safari and the Web Kit. Some attributes are also marked as deprecated, which means they are supported by Safari, but since they are no longer supported by the HTML specification are not guaranteed to be supported in the future.

abbr

Abbreviation for a table header cell

accept-charset

Supported character sets for a form

accept

Supported MIME types for a form

accesskey

Access key for accessibility/screen reading

action

URL of a form processor (for example, a CGI script)

align

Used to align inline elements within an element. Deprecated in HTML 4.01 (use CSS styling instead).

alink

Color of currently active hyperlink. Deprecated in HTML 4.01 (use CSS styling instead).

alt

Short description of an image, area, or input type.

archive

Comma-separated list of Java archive URLs.

autocapitalize

Can be set to "off" or "on", to turn form automatic capitalization off and on for that element, respectively.

Note: Not part of the HTML 4.01 standard.

Availability

Available in Safari on iPhone 1.1.1 and later.

autocomplete

Can be set to "off" or "on", to turn form autocompletion off and on for that element, respectively.

Note: Not part of the HTML 4.01 standard.

autocorrect

Can be set to “off” or “on”; to turn form automatic correction off and on for that element, respectively.

Note: Not part of the HTML 4.01 standard.

Availability

Available in Safari on iPhone 1.1.1 and later.

autosave

Used for the “search” input type. Use this specify an autosave name for the search field, so prior searches can be saved.

Note: Apple extension.

axis

Comma-separated list of category names for a table cell or table header cell.

background

Specifies a background image. Deprecated in HTML 4.01 (use CSS styling instead).

behavior

Used for the marquee element. Can be set to “scroll” to continually scroll, “slide” to slide it to the edge and start again at the beginning, or “alternate” to scroll back and forth.

Note: Not part of the HTML 4.01 standard.

bgcolor

Sets the background color of an element. Deprecated in HTML 4.01 (use CSS styling instead).

bgproperties

Sets the background style if a background image has been set. The only value available is “fixed”, which will lock the background in place as the page scrolls.

Note: Not part of the HTML 4.01 standard.

border

Sets the visible border width for a table.

bordercolor

Defines the color of a table border. Has no effect in Safari.

Note: Not part of the HTML 4.01 standard.

cellpadding

Defines the pixel spacing within table cells.

cellspacing

Defines the pixel spacing between cells.

char

Defines the axis of alignment for a block of text (for example, the decimal point in an English monetary value). The default is a decimal point.

challenge

An arbitrary string which acts as the challenge for a keygen.

Note: Not part of the HTML 4.01 standard.

charoff

Used to specify the offset to the alignment character specified by the “char” attribute.

charset

Defines the character encoding style of a given linked resource.

checked

This attribute is placed within the checkbox element, it will show up as checked on the HTML form. It does not require a value: its presence indicates the box is checked, its absence indicates it is not.

cellborder

Sets the width of the border of an individual table cell.

Note: Not part of the HTML 4.01 standard.

cite

Specifies the URL for the source of a citation, or a string explaining the reason for a change.

class

Space-separated list of CSS classes defining the style for an element.

classid

Specifies the URL of the implementation for an embedded object.

clear

Specifies the flow direction of text around a line break. Deprecated in HTML 4.01 (use CSS styling instead).

code

Defines the class file for an applet specified with the applet tag. Deprecated in HTML 4.01 along with the applet tag.

codebase

Defines the base URL for a classid, data file, or archive specified with the object tag. Deprecated in HTML 4.01 for the applet tag, but not for the object tag.

codetype

Defines the content-type for the code embedded by the object tag.

color

Defines the text color for the font elements. Deprecated in HTML 4.01 (use CSS styling instead).

cols

Defines the number of columns in a textarea or frameset.

colspan

Defines the number of columns spanned by an individual column definition.

compact

Compresses the space between elements in a list or menu. Deprecated in HTML 4.01 (use CSS styling instead).

composite

Specifies how an image gets composited onto a Dashboard widget.

Note: Apple extension.

content

Specifies the content for a given meta key.

contenteditable

Set to true or false, defines whether or not a given element can be edited on the fly.

Note: Not part of the HTML 4.01 standard.

coords

A comma-separated list of lengths for the area element or for client-side image maps.

data

URL to the data required by an embedded object element.

datetime

Used for the del and ins elements, specifies the date and time of the change. Uses the ISO date format.

declare

Added alone to an object element's definition. Tells the object to declare but not instantiate itself.

defer

Added alone to a script element's definition. Tells the script to defer execution.

dir

Specifies the direction (ltr/rtl) of text within an element.

direction

Specifies the scrolling direction for the marquee element. Can be set to "left" or "right".

Note: Not part of the HTML 4.01 standard.

disabled

Added alone to an input type's definition. Disables (greys out) the element.

enctype

Defines the MIME content encoding type for a form.

face

Comma-separated list of font names. Deprecated in HTML 4.01 (use CSS styling instead).

for

Specifies the identifier of the control the label is relevant to.

frame

Specifies which sides of a table frame should be shown. Possible values are: `void` (none), `above`, `below`, `hsides` (top and bottom), `vsides` (right and left), `rhs` (right hand side), `lhs` (left hand side), `box` (all four sides), and `border` (all four sides).

frameborder

Specifies whether or not to display a frameborder. Can be set to "1" or "0".

headers

Space-separated list of identifiers of table header cells.

height

Specifies the height of the element. Deprecated in HTML 4.01 for applets and table elements.

hidden

Used for objects embedded with the deprecated `embed` tag. If set to "true" the object will be hidden onscreen. Its default value is "false".

href

Specifies the URL for any kind of link or area definition.

hreflang

Specifies the language of the resource linked to by the element's href attribute.

hspace

Specifies the pixel size of the horizontal spacing surrounding the element. Deprecated in HTML 4.01 (use CSS styling instead).

html

http-equiv

Used in the `meta` element. Contains some kind of information about the header of the page.

id

Specifies a unique identifier for an element.

incremental

Specifies whether or not the "search" input type should perform the search as the user is typing.

Note: Apple extension.

ismap

Specifies whether or not an image or image-type input corresponds to a server-side image map.

keytype

Specifies the key type for the `keygen` element. Can be set to "RSA" or "DSA".

Note: Not part of the HTML 4.01 standard.

label

Specifies the label for options or option groups within a pull-down menu.

lang

Specifies the base language for the element's attributes and textual content. Values are standard two-letter language codes.

language

Specifies the language of a `script` element. Deprecated in HTML 4.01.

left

Specifies the left coordinate of a `layer` element.

Note: Not part of the HTML 4.01 standard.

leftmargin

Placed in the `body` element, specifies the left margin of the page content.

Note: Not part of the HTML 4.01 standard.

link

Color of an unvisited hyperlink. Deprecated in HTML 4.01 (use CSS styling instead).

longdesc

Long textual description for images and frames. Augments the `alt` attribute for an element.

loop

Specifies whether or not the embedded object (movie or sound) will loop. Also specifies if a marquee element will loop.

Note: Not part of the HTML 4.01 standard.

marginheight

Specifies the height of the margins in a frame.

marginwidth

Specifies the width of the margins in a frame.

max

Same as the `maxlength` attribute.

Note: Not part of the HTML 4.01 standard.

maxlength

Specifies the maximum input length for a text input field in characters.

mayscript

Added alone to an applet element's definition. Tells the script to allow the applet access to the JavaScript scripting objects.

Note: Not part of the HTML 4.01 standard.

media

For the `style` and `link` elements, defines the media type that the element is designed for. Its default value is "screen", but can also be set to "tty" for a fixed-pitch device, "tv" for low-resolution televisions, "projection" for projectors, "handheld" for handheld devices, "print" for printed material, "braille" for braille devices, "aural" for speech synthesizers, or "all" for all devices.

method

Specifies the form submission method, either POST or GET.

min

Specifies the minimum input length for a text field.

Note: Not part of the HTML 4.01 standard.

multiple

Added alone to a `select` element's definition. If present, the element will permit multiple selections.

name

Specifies the name of the element, which can be used in a variety of ways.

nohref

Added alone to an `area` element's definition. If present, the area has no particular action assigned to it.

noresize

Added alone to a `frame` element's definition. If present, the frames cannot be resized.

nosave

Legacy attribute. Has no effect in Safari.

noshade

Added alone to an `hr` element's definition. If present, the rule is drawn as a single line and not as a "groove." Deprecated in HTML 4.01 (use CSS styling instead).

nowrap

Added alone to a table cell's definition. Suppresses word wrap if present. Deprecated in HTML 4.01 (use CSS styling instead).

object

Specifies the URL of a serialized applet in an `applet` element. Deprecated in HTML 4.01 along with the `applet` element.

onabort

JavaScript delegate. The code specified by this attribute is called when the image element it is assigned to is aborted during load.

Note: Not part of the HTML 4.01 standard.

onbeforecopy

JavaScript delegate. The code specified by this attribute is called before the associated element is copied.

Note: Not part of the HTML 4.01 standard.

onbeforecut

JavaScript delegate. The code specified by this attribute is called before the associated element is cut.

Note: Not part of the HTML 4.01 standard.

onbeforepaste

JavaScript delegate. The code specified by this attribute is called before the associated element has something pasted into it.

Note: Not part of the HTML 4.01 standard.

onblur

JavaScript delegate. The code specified by this attribute is called when the associated element loses focus.

onchange

JavaScript delegate. The code specified by this attribute is called when the associated element changes its value.

onclick

JavaScript delegate. The code specified by this attribute is called when the associated element is clicked.

oncontextmenu

JavaScript delegate. The code specified by this attribute is called when the associated element is right-clicked or when the mouse button is held down long enough to generate a contextual menu.

Note: Not part of the HTML 4.01 standard.

oncopy

JavaScript delegate. The code specified by this attribute is called when the associated element is copied.

Note: Not part of the HTML 4.01 standard.

oncut

JavaScript delegate. The code specified by this attribute is called when the associated element is cut.

Note: Not part of the HTML 4.01 standard.

ondblclick

JavaScript delegate. The code specified by this attribute is called when the associated element is double-clicked.

ondrag

JavaScript delegate. The code specified by this attribute is called when the associated element is dragged.

Note: Not part of the HTML 4.01 standard.

ondragend

JavaScript delegate. The code specified by this attribute is called when the associated element is done being dragged.

Note: Not part of the HTML 4.01 standard.

ondragenter

JavaScript delegate. The code specified by this attribute is called when a drag has entered the associated element.

Note: Not part of the HTML 4.01 standard.

ondragleave

JavaScript delegate. The code specified by this attribute is called when a drag has left the associated element.

Note: Not part of the HTML 4.01 standard.

ondragover

JavaScript delegate. The code specified by this attribute is called when a drag is over the associated element.

Note: Not part of the HTML 4.01 standard.

ondragstart

JavaScript delegate. The code specified by this attribute is called when the associated element has started to be dragged.

Note: Not part of the HTML 4.01 standard.

ondrop

JavaScript delegate. The code specified by this attribute is called when the associated element is dropped.

Note: Not part of the HTML 4.01 standard.

onerror

JavaScript delegate. The code specified by this attribute is called when the associated element has an error in loading.

Note: Not part of the HTML 4.01 standard.

onfocus

JavaScript delegate. The code specified by this attribute is called when the associated element gets focus.

oninput

JavaScript delegate. The code specified by this attribute is called when text is entered into the associated element.

Note: Not part of the HTML 4.01 standard.

onkeydown

JavaScript delegate. The code specified by this attribute is called when a key is pressed over the associated element.

onkeypress

JavaScript delegate. The code specified by this attribute is called when a key is pressed and released over the associated element.

onkeyup

JavaScript delegate. The code specified by this attribute is called when a key is released over the associated element.

onload

JavaScript delegate. The code specified by this attribute is called when the associated element finishes loading.

onmousedown

JavaScript delegate. The code specified by this attribute is called when the mouse button is pressed over the associated element.

onmousemove

JavaScript delegate. The code specified by this attribute is called when a key is moved within the associated element.

onmouseout

JavaScript delegate. The code specified by this attribute is called when the mouse leaves the associated element.

onmouseover

JavaScript delegate. The code specified by this attribute is called when the mouse is over the associated element.

onmouseup

JavaScript delegate. The code specified by this attribute is called when the mouse button is released over the associated element.

onpaste

JavaScript delegate. The code specified by this attribute is called when the associated element is pasted.

Note: Not part of the HTML 4.01 standard.

onreset

JavaScript delegate. The code specified by this attribute is called when the associated form element is reset.

onresize

JavaScript delegate. The code specified by this attribute is called when the associated element is resized.

Note: Not part of the HTML 4.01 standard.

onscroll

JavaScript delegate. The code specified by this attribute is called when the associated element is scrolled (a text box would use this, for example).

Note: Not part of the HTML 4.01 standard.

onsearch

JavaScript delegate. The code specified by this attribute is called when the associated element is copied.

Note: Apple extension.

onselect

JavaScript delegate. The code specified by this attribute is called when text within the associated element is selected.

onselectstart

JavaScript delegate. The code specified by this attribute is called when the associated element begins to be selected. You can use this to prevent selections.

Note: Not part of the HTML 4.01 standard.

onsubmit

JavaScript delegate. The code specified by this attribute is called when the associated form element is submitted.

onunload

JavaScript delegate. The code specified by this attribute is called when the associated element is unloaded from the page.

oversrc

Specifies the source of an image to be displayed when the mouse is over an element. Useful for rollovers.

pagex

Specifies the x-coordinate of the location of a layer on the page.

Note: Not part of the HTML 4.01 standard.

pagey

Specifies the y-coordinate of the location of a layer on the page.

Note: Not part of the HTML 4.01 standard.

placeholder

Specifies the placeholder text displayed in light grey when the search input field is not currently in use.

Note: Apple extension.

plain

Added alone to an unordered list element's definition. If present, the list element (or all the list elements, if placed in the enclosing `ul` element) will not have a bullet in front of it.

Note: Not part of the HTML 4.01 standard.

pluginpage

Specifies the URL of the page where visitors can find the plug-in required to display content embedded with the `embed` element. This has been deprecated along with the `embed` element.

Note: Not part of the HTML 4.01 standard.

pluginspage

Same as the `pluginpage` attribute.

pluginurl

Same as the `pluginspage` attribute, except it links directly to the plug-in itself (in Java archive format).

precision

profile

Specifies the URL to a file of meta data or a list of said files.

prompt

Specifies the textual prompt for the `isindex` element. Deprecated in HTML 4.01.

readonly

Added alone to a textarea's definition. If present, the textarea will not be editable.

rel

Defines a relationship to another document. The URL specified by this property relates to this document by rel. In other words, it is the next-order relation.

results

Specifies how many results should be returned by the search input type.

Note: Apple extension.

rev

Defines a relationship to another document. This document relates to the URL specified by this property as rev. In other words, it is the reverse-order relation.

ROWS

Defines the number of rows in a textarea or frameset.

rowspan

Defines the number of rows spanned by an individual row definition.

rules

Specifies which rulings to show for a table element. Its potential values are “none”, “groups”, “rows”, “cols”, and “all.”

scheme

Defines the scheme to be used to interpret a meta value.

scope

Specifies the scope handled by a table’s header cells. Possible values are “row”, “col”, “rowgroup”, and “colgroup”.

scrollamount

Specifies the number of pixels that a marquee element will scroll between successive redraws of its onscreen view.

Note: Not part of the HTML 4.01 standard.

scrolldelay

Specifies the number of milliseconds that a marquee element will delay between successive redraws of its onscreen view.

Note: Not part of the HTML 4.01 standard.

scrolling

Specifies whether or not a frame or iframe element should have scrollbars. Setting this property to “yes” will always show the scrollbars, “no” will never show the scrollbars, and “auto” will show the scrollbars only if the content needs to scroll (this is the default).

selected

Added alone to an `option` definition. If present, the option with this property will be selected in the list.

shape

Defines the shape of an area element or a client-side image map. Can take the form of “default”, “rect”, “circle”, or “poly”.

size

Defines the physical size of a variety of inputs and fonts. Deprecated in HTML 4.01 (use CSS styling instead).

span

Specifies the number of columns that a given `col` or `colgroup` definition will span.

src

Specifies a URL for an external file or resource.

standby

Defines a message to show while an object is being loaded within an `object` definition.

start

Defines the starting sequence number for an ordered list. Deprecated in HTML 4.01 (use CSS styling instead).

style

Specifies CSS style information for the element. The style definitions are placed inline with this property. To specify external styles (defined in external files or in a `style` element), use the `class` property.

summary

Specifies a textual summary for a table of data.

tabindex

Specifies the tab index for a hyperlink or input element. Can range from 0 to 32767. As a user tabs through a page, the focus will follow this tabbing order.

tableborder

Specifies the width of a table's border. You should use CSS styling to define this property instead.

Note: Not part of the HTML 4.01 standard.

target

Defines the target window for a hyperlink. You can specify any target definition, but the following values are built-in: "_blank" will load the clicked URL into a new, unnamed window; "_self" will load in the same frame that was clicked; "_parent" will load into the parent frame of the frame that was clicked; "_top" will load the document into the original window, eliminating any existing frameset.

text

Defines the page-wide text color in the body element. Deprecated in HTML 4.01 (use CSS styling instead).

title

Defines a title for the associated element. In Safari, this will display a tool tip for the element with this value.

top

Defines the top coordinate of a `layer` element.

Note: Not part of the HTML 4.01 standard.

topmargin

Placed in the `body` element, specifies the top margin of the page content.

Note: Not part of the HTML 4.01 standard.

truespeed

Specifies whether or not a marquee operates at the true speed specified by its parameters. The default behavior will constrain the speed to certain minimum values of scroll delay and scroll amount.

Note: Not part of the HTML 4.01 standard.

type

Defines a textual content type for elements like scripts and objects, and input types for the input element. See the input types table below for possible values.

unknown

usemap

Specifies the URL for an image map on a document. Usually this is an anchor (for example, “#myMap”) defined as a map element on the page.

valign

Specifies vertical alignment within a a column or table element definition.

value

Specifies the current value for an input type. For those elements that can display their values (such as text fields), they will display this value onscreen. Otherwise the values are all available as form values when submitted.

valuetype

Specifies the value type for a `param` element within an object definition. Can be either “data” if the value will be evaluated and passed as a string, “ref” if the value is a URL and will be unevaluated when passed, or “object” if the value is an identifier that refers to an object definition.

version

Specifies the version of the HTML DTD used to verify the document. Deprecated in HTML 4.01, use DOCTYPE declarations instead.

visibility

Defines whether or not a `layer` element is visible or not.

Note: Not part of the HTML 4.01 standard.

vlink

Color of an already-visited hyperlink. Deprecated in HTML 4.01 (use CSS styling instead).

vspace

Specifies the pixel size of the vertical spacing surrounding the element. Deprecated in HTML 4.01 (use CSS styling instead).

width

Specifies the height of the element. Deprecated in HTML 4.01 for applets and table elements.

wrap

Defines the wrap style for a textarea. Can be set to “soft” to wrap without outputting carriage returns to the field, “hard” to wrap with outputting carriage returns, and “off” to not wrap at all.

Note: Not part of the HTML 4.01 standard.

z-index

Defines the z-coordinate position of a `layer` element. The higher this value, the higher the layer will be relative to the others on the page.

Note: Not part of the HTML 4.01 standard.

Standard Input Type Values

Safari supports many different input types. They can be specified using the `type` attribute of the `input` element. The following table shows these input types.

text

A standard text field.

password

A visually-shielded password field.

checkbox

A standard checkbox.

radio

A radio button.

submit

A submission button for a form.

reset

A reset button for a form.

file

A file upload interface.

hidden

A hidden input type (to store values without showing them on the page). Note that the input can still be seen in the page source.

image

An image that acts as an input.

button

A button input type. More versatile than a submit button.

search

Provides a search field. Uses the `incremental` , `placeholder` , `autosave` , and `results` attributes from the table above in addition to standard HTML attributes.

Note: Apple extension.

range

Provides a slider. Its minimum value should be set with the `min` attribute, its maximum value should be set with `max` , and its discrete step size should be set with `step` .

Note: Apple extension.

HTML Extensions

Safari and the Web Kit implement a large subset of the HTML 4.01 Specification defined by the World Wide Web Consortium (W3C). There are other tags that are not defined by the Specification, but are extensions to it. These are listed here. If a tag is not listed here or in ["Standard HTML"](#) (page 15), it is not implemented by Safari and the Web Kit.

Additional HTML Elements

canvas

Used to specify an advanced drawing region.

Example

```
<canvas id="identifer" height="value" width="value">
```

This tag specifies the location of an advanced drawing region. The `canvas` tag supports the same attributes as the `` tag with the exception of the `src` attribute, which is ignored. You can specify any of the other attributes you would normally specify for an image. The identifier specified by `id` is required for Dashboard widgets, as are the height and width specified by `height` and `width` respectively.

embed

Used to embed an object within a page.

Example

```
<embed height = "value" width = "value" src/code="URL" > content </em>
```

The object, if visible, will be displayed at the location of the tag in the page, with a height specified by `height` and a width specified by `width`. The location of the object is given by the URL specified by `src`, or `code` if the applet is in a standard java class file.

Important: This tag has been deprecated in the HTML 4.01 standard. You should use the `<object>` tag to embed objects unless you have a specific reason to use this tag.

keygen

Used to provide for public key generation for forms.

Example

```
<keygen name="name" challenge="challenge_value">
```

This tag places a form element on the page which will generate a 512, 1024, or 2048-bit public key as its value. The challenge specified by `challenge` and the public key are DER encoded and digitally signed with a private key (stored in a local database). The result is then encoded in base64 and is returned as the value of this field.

layer

Used to specify individual layers on a webpage.

Example

```
<layer>content</layer>
```

This tag specifies an independent layer of content on a webpage. This tag is not well-supported and should be replaced with `<iframe>` frames using CSS styling techniques in HTML 4.01 Transitional documents. In HTML 4.01 Strict documents, this tag should be replaced with `<object>` or `<div>`.

marquee

Used to specify a horizontally-scrolling block of content.

Example

```
<marquee>content</marquee>
```

This tag specifies a block of content that scrolls horizontally (by default, across 100% of the enclosing element). The content specified by `content` can be arbitrary—it is not limited to text alone. You can specify a custom width using CSS styling techniques.

nobr

Used to specify a region of content with no embedded line breaks.

Example

```
<nobr>content</nobr>
```

The content specified by `content` will be displayed with no line breaks. It is intended for blocks that must remain on one line.

noembed

Used to specify content to display to browsers that do not support embedded objects.

Example

```
<noembed>content</noembed>
```

nolayer

Used to specify content to display to browsers that do not support layers.

Example

```
<nolayer>content</nolayer>
```

plaintext

Represents a block of pre-formatted text.

Example

```
<plaintext>content</plaintext>
```

This tag preserves the formatting of the block of text specified by `content`, specifically line breaks and multiple spaces (normal text operation in Safari displays no difference between a single space and multiple consecutive spaces). In Safari, text enclosed in this element is also rendered in a monospace “tele-type” font. This also stops the interpretation of HTML tags, so they are rendered onscreen. This tag is not well-supported and may cause unintended behavior in Safari, consider using the `<pre>` tag instead.

wbr

Used to specify a block in which line breaks are permitted.

Example

```
<wbr>content</wbr>
```

Within a `<noabr>` block (in which line breaks are disabled), any content specified by `content` will be permitted to use line breaks. The line breaks themselves must still be requested using the `
` tag.

xmp

Represents a block of literal text.

Example

```
<xmp>content</xmp>
```

This tag preserves the formatting of the block of text specified by `content`, specifically line breaks, multiple spaces, and the greater-than and less-than symbols that accompany HTML tags. This block is also prefaced with a newline. In Safari, text enclosed in this element is also rendered in a monospace “tele-type” font. This tag is not well-supported and may cause unintended behavior in Safari, consider using the `<pre>` tag instead.

Additional meta Tag Keys

viewport

Changes the logical window size used when displaying a page on iPhone.

Example

```
<meta name = "viewport" content = "width = 320,
    initial-scale = 2.3, user-scalable = no">
```

Use the viewport meta key to improve the presentation of your web content on iPhone. Typically, you use the viewport meta tag to set the width and initial scale of the viewport.

For example, if your webpage is narrower than 980 pixels, then you should set the width of the viewport to fit your web content. If you are designing an iPhone-specific web application, you should set the width to the width of the device.

[Table 1](#) (page 68) describes the properties supported by the viewport meta key and their default values. When providing multiple properties for the viewport meta key, you should use a comma-delimited list of assignment statements.

Table 1 Viewport properties

Property	Description
width	The width of the viewport in pixels. The default is 980. The range is from 200 to 10,000. You can also set this property to the constants described in Table 2 (page 69).
height	The height of the viewport in pixels. The default is calculated based on the value of the width property and the aspect ratio of the device. The range is from 223 to 10,000 pixels. You can also set this property to the constants described in Table 2 (page 69).
initial-scale	The initial scale of the viewport as a multiplier. The default is calculated to fit the webpage in the visible area. The range is determined by the <code>minimum-scale</code> and <code>maximum-scale</code> properties. You can set only the initial scale of the viewport—the scale of the viewport the first time the webpage is displayed. Thereafter, the user can zoom in and out unless you set <code>user-scalable</code> to <code>no</code> . Zooming by the user is also limited by the <code>minimum-scale</code> and <code>maximum-scale</code> properties.
minimum-scale	Specifies the minimum scale value of the viewport. The default is 0.25. The range is from >0 to 10.0.
maximum-scale	Specifies the maximum scale value of the viewport. The default is 1.6. The range is from >0 to 10.0.

Property	Description
<code>user-scalable</code>	<p>Determines whether or not the user can zoom in and out—whether or not the user can change the scale of the viewport. Set to <code>yes</code> to allow scaling and <code>no</code> to disallow scaling. The default is <code>yes</code>.</p> <p>Setting <code>user-scalable</code> to <code>no</code> also prevents a webpage from scrolling when entering text in an input field.</p>

When referring to the dimensions of a device, you should use the constants described in [Table 2](#) (page 69) instead of hard-coding specific numeric values. For example, use `device-width` instead of `320` for the width, and `device-height` instead of `480` for the height in portrait orientation.

Table 2 Special viewport property values

Value	Description
<code>device-width</code>	The width of the device in pixels. Available on iPhone 1.1.1 and later.
<code>device-height</code>	The height of the device pixels. Available on iPhone 1.1.1 and later.

You do not need to set every viewport property. If only a subset of the properties are set, then Safari on iPhone infers the other values. For example, if you set the scale to `1.0`, Safari assumes the width is `device-width` in portrait and `device-height` in landscape orientation. Therefore, if you want the width to be 980 pixels and the initial scale to be `1.0`, then set both of these properties.

For example, to set the viewport width to the width of the device, add this to your HTML file:

```
<meta name = "viewport" content = "width = device-width">
```

To set the initial scale to `1.0`, add this to your HTML file:

```
<meta name = "viewport" content = "initial-scale = 1.0">
```

To set the initial scale and to turn off user scaling, add this to your HTML file:

```
<meta name = "viewport" content = "initial-scale = 2.3, user-scalable = no">
```

Use the Safari on iPhone console to help debug your webpages as described in [Debugging](#). The console contains tips to help you choose viewport values—for example, it reminds you to use the constants when referring to the device width and height.

Document Revision History

This table describes the changes to *Safari HTML Reference*.

Date	Notes
2008-09-09	Updated for Safari 3.1.
2008-01-15	Moved reference information for the viewport meta key from Safari Web Content Guide for iPhone.
2007-12-11	Added iPhone-specific HTML attributes.
2007-09-04	Reformatted content.
2006-05-23	Corrected typos.
2005-11-09	Corrected typos.
2005-08-11	Corrected typos.
	Corrected typos. Added information on the paragraph tag.
2005-06-28	Corrected a typo.
2005-06-04	New document that describes the HTML tags and properties supported by Safari and the Web Kit.

