

# CSS Reference

---

# Table of Contents

<i>CSS Reference</i> .....	1
A2.1 Style Sheets .....	3
Selectors .....	3
Pseudo Classes .....	4
Rules .....	5
A2.2 Properties .....	6
Text .....	6
Colors and Backgrounds .....	8
Fonts .....	9
Box Model .....	12
Visual Formatting and Positioning .....	16
Generated Content and Lists .....	20
Tables .....	23
Paged Media .....	24
Aural Style Sheets .....	26

## A2.1 Style Sheets

If you have looked into the details of Cascading Style Sheets, much of this appendix will look familiar. However, there are many properties that you will not recognize. Level 2 has taken style sheets to a new level, and this appendix details all the new additions.

Currently, CSS1 is implemented in Netscape Navigator 4+ and Microsoft's Internet Explorer 4+. CSS2 implementation is available to a certain degree in the Internet Explorer 5.0 browser, and expected in Netscape's 5.0 browser as it becomes available. CSS2 is in the recommendation stage and can be found at <http://www.w3.org/TR/REC-CSS2/>.

The properties in this appendix are grouped into areas according to their function. In many cases one property affects another and therefore properties are presented in a logical order. The property groups include the following:

- Text
- Colors and backgrounds
- Fonts
- Box model
- Visual formatting and positioning
- Generated content and lists
- Tables
- Paged media
- Aural style sheets

All properties that are new to the CSS2 specification are marked with an asterisk right after the property name.

### Selectors

Selectors are the tag elements defined at the beginning of a style sheet definition that tell the browser where to apply the style. After the selector, the style definition is included within curly brackets. In this example, `body` is the selector.

```
body {color: blue}
```

Several selectors can be grouped together if they are separated with commas.

```
h1, h2, h3 {font-family: san-serif}
```

In place of selectors, you can use the wildcard. This example applies a font size style to all tags on the page:

```
{font-size: 14pt}
```

Another wild card character is the > sign. This tells the browser to search for child selectors within a certain parent. This example applies the style only to li elements with ol lists:

```
ol > li {list-style-type: decimal}
```

Using class selectors, you can apply different styles to the same tag. A period and a name follow a general selector and the style is applied to the tag whose class attribute matches the class name. The following example applies the style to any h2 tags that have the class attribute equal to "myBlue".

```
h2.myBlue {background-color: blue}
<h2 class="myBlue">This header has a blue
background.</h2>
```

Selectors can also be identified by the id attribute using the # character. The following example matches the style to any tags whose ID attribute is "duckie".

```
#duckie {border-color: yellow}
```

## Pseudo Classes

To access the control of elements that are not referred to by normal tags, CSS2 defines several pseudo classes. An example is the first line of a paragraph. HTML has no way of identifying this element, so a pseudo class called: `first-line` is used. All pseudo classes have colons in front of them. They are located after a selector like the following:

```
p:first-line {color: red}
```

The following are identified pseudo classes in CSS2:

- **:first-child**-This is the first child element of another element.
- **:link**-These are links that have not yet been visited.
- **:visited**-These are visited links.
- **:hover**-This is an element that the cursor is currently over.
- **:active**-This is the currently activated element.
- **:focus**-This is the element that has the focus.
- **:lang**-This defines the current language.
- **:first-line**-This is the first formatted line of a paragraph.
- **:first-letter**-This is the first letter of a paragraph.
- **:before**-This positions content to come before an element.
- **:after**-This positions content to come after an element.

## Rules

Rules are used to access files and documents located outside of the current document. There are five rules defined in CSS2, and all of them begin with the @ character: @charset, @font-face, @import, @media, and @page.

## A2.2 Properties

Properties are the main descriptors of the style sheet language. They appear within brackets and include the property name and, a value separated by a colon. Some properties can include more than one value. These values are typically separated by a single space.

### Text

The text properties include aligning properties such as `text-align` and `word-spacing`, as well as style altering properties such as `text-decoration` and the new `text-shadow` properties.

#### **text-indent**

Description: Defines the length of the indent applied to the first line of text in a block.

Values: Any valid length: Can include negative values. Default is 0.

Any valid percentage.

`inherit`: Takes the same value as its parent.

Example: `p {text-indent: 40px}`

#### **text-align**

Description: Defines how an inline box of text is aligned.

Values: `left`: Aligns text to the right.

`center`: Aligns text to the center.

`right`: Aligns text to the right.

`justify`: Justifies the text.

Any valid string: Defines a string on which table cells will align.

`inherit` -Takes the same value as its parent.

Example: `p {text-align: right}`

#### **text-decoration**

Description: Defines decorations added to the text of an element.

Values: `none`: (default) Applies no text decoration.

`underline`: Underlines the text.

`overline`: Puts a line over the text.

`line-through`: Strikes out the text.

`blink`: Causes the text to blink.

`inherit`: Takes the same value as its parent.

Example: `p {text-decoration: underline}`

#### **text-transform**

Description: Defines capitalization effects to the text of an element.

Values: `none`: (default) Applies no capitalization.  
`capitalize`: Capitalizes the first letter of each word.  
`uppercase`: Capitalizes all letters.  
`lowercase`: Converts all letters to lowercase.  
`inherit`: Takes the same value as its parent.

Example: `h3 {text-transform: uppercase}`

## text-shadow

Description: Describes values to create a text shadow effect. Several lists of shadow values can be included and must be separated by commas. Each separate shadow effect value list must include offset values and can include a blur radius and color.

Values: `none`: (default) Applies no shadow effect.  
`color`: Color of text shadow.

First valid length: Horizontal distance to the right of the text. Negative values are to the left of the text.

Second valid length: Vertical distance below the text. Negative values are above the text.

Third valid length: Text shadow blur radius.

`inherit`: Takes the same value as its parent.

Example: `h1 {text-shadow: blue 5px 5px 3px, yellow -2px -2px 3px}`

## letter-spacing

Description: Defines the space between text characters.

Values: `normal`: (default) Applies normal text spacing for the font being used.  
Any valid length: The length of the space between letters.  
`inherit`: Takes the same value as its parent.

Example: `p {letter-spacing: 0.3em}`

## word-spacing

Description: Defines the space between words.

Values: `normal`: (default) Applies normal text spacing for the font being used.  
Any valid length: The length of the space between letters.  
`inherit`: takes the same value as its parent.

Example: `p {word-spacing: 1.3em}`

## white-space

Description: Defines how to handle white-space in an element.

Values: `normal`: (default) Collapses white-space if necessary to fit boxes. This is the same as how HTML handles white-space.  
`pre`: Treats all white-space literally as it appears in code.  
`nowrap`: Collapses all white-space.  
`inherit`: Takes the same value as its parent.

Example: `p {white-space: pre}`

# Colors and Backgrounds

Adding colors and backgrounds to elements creates a visually stimulating Web page. Style sheets include many properties that give your page the zing it needs.

## **color**

Description: Defines the text color.

Values: Any valid color: Colors the text.

`inherit`: Takes the same value as its parent.

Example: `p {color: green}`

`p {color: rgb(0, 255, 0)}`

## ***background-color***

Description: Defines the background color of an element.

Values: Any valid color: Colors the text.

`transparent`: (default) Makes the element's background transparent.

`inherit`: Takes the same value as its parent.

Example: `div {background-color: blue}`

`div {color: rgb(0, 0, 255)}`

## ***background-image***

Description: Defines the background image of an element.

Values: `none`: (default) Sets no background image.

Any valid URL: URL of the background image.

`inherit`: Takes the same value as its parent.

Example: `h1 {background-image: url("texture3.gif")}`

## ***background-repeat***

Description: Defines the direction that the background image is tiled.

Values: `repeat`: (default) Background image repeats both horizontally and vertically.

`repeat-x`: Background image repeats only horizontally.

`repeat-y`: Background image repeats only vertically.

`no-repeat`: Background image doesn't repeat.

`inherit`: Takes the same value as its parent.

Example: `blockquote {background-repeat: repeat-x}`

## background-position

**Description:** Defines the upper-left corner position of the background image. Single values set the horizontal distance and default the vertical offset to 50 percent. Several keywords can be combined.

**Values:** First valid length: Horizontal distance the background image is placed from the left edge. Accepts negative values.

Second valid length: Vertical distance the background image is placed from the top edge. Accepts negative values.

First valid percentage: Percent of the element box the background image is offset from the left edge. Default is 0 percent or upper-left corner.

Second valid percentage: Percent of the element box the background image is offset from the top edge.

`top`: Positions the background image along the top edge.

`center`: Positions the background image in the center of the element box.

`bottom`: Positions the background image along the bottom edge.

`left`: Positions the background image along the left edge.

`right`: Positions the background image along the right edge.

`inherit`: Takes the same value as its parent.

**Example:** `blockquote {background-position: top center}`

## background-attachment

**Description:** Defines whether the background image is fixed to the window or scrolls with the document.

**Values:** `scroll`: (default) Background image scrolls along with the window.

`fixed`: Background image is permanently fixed to its location. Background image repeats only horizontally.

`inherit`: Takes the same value as its parent.

**Example:** `img {background-attachment: fixed}`

## background

**Description:** Shorthand property for defining all background properties at once. If not included, a property is set to its default value.

**Values:** `background-color`: Background color value.

`background-image`: Background image value.

`background-repeat`: Background repeat value.

`background-attachment`: Background attachment value.

`background-position`: Background position value.

`inherit`: Takes the same value as its parent.

**Example:** `p {background: blue url("texture3.gif") repeat fixed top right}`

## Fonts

Font control adds style and flair to your Web pages whether you change the family, size, or weight.

## font-family

Description: Defines a font to use for the element's text. It can include several font families separated by commas. The list order defines the priority.

Values: Font name: Font to use to render the text. Fonts with more than one word need to be in quotes.

Generic font name: Generic font class to use to render the text. Generic fonts include the following: *serif*, *sans-serif*, *cursive*, *fantasy*, and *monospace*.

*inherit*: Takes the same value as its parent.

Example: `body {font-family: "Times Roman", courier, serif}`

## font-style

Description: Defines a font style, such as italic or oblique.

Values: *normal*: (default) Uses the normal font style.

*Italic*: Uses an italic font style.

*oblique*: Uses an oblique or slanted font style.

*inherit*: Takes the same value as its parent.

Example: `span {font-style: italic}`

## font-variant

Description: Defines whether a font is rendered using small caps.

Values: *normal*: (default) Uses the normal font style.

*small-caps*: Renders the font in small caps.

*inherit*: Takes the same value as its parent.

Example: `h4 {font-variant: small-caps}`

## font-weight

Description: Defines how thick text appears.

Values: *normal*: (default) Uses the normal font thickness.

*bold*: Uses a bold font weight.

*bolder*: Uses a bolder font weight.

*lighter*: Uses a lighter font weight.

*100-900*: Number indicates the font thickness. 100 is the lightest (same as *lighter*), 400 is normal, 700 is *bold*, and 900 is *bolder*.

*inherit*: Takes the same value as its parent.

Example: `h1 {font-weight: bolder}`

## font-stretch

Description: Defines the font's width.

Values: *normal*: (default) Uses the normal font width.

*wider*: Increases the width by one over current setting.

*narrower*: Decreases the width by one under current setting.

*ultra-condensed*: Defines the tightest width setting.

*extra-condensed*: Looser than the preceding value.

condensed: Looser than the preceding value.  
semi-condensed: Looser than the preceding value.  
semi-expanded: Wider than normal.  
expanded: Wider than the preceding value.  
extra-expanded: Wider than the preceding value.  
ultra-expanded: Defines the widest setting.  
inherit: Takes the same value as its parent.

Example: `body {font-stretch: condensed}`

## font-size

Description: Defines the size of the font.

Values: Absolute size: Uses keywords to express font size. Values include `xx-small`, `small`, `medium` (default), `large`, `x-large`, and `xx-large`.

Relative size: Uses relative keywords to express font size. Values include `larger` and `smaller`.

Any valid length: Defines the absolute font size. Negative values are not accepted.

Valid percentage: Defines the percent increase or decrease from the parent font size.

`inherit`: Takes the same value as its parent.

Example: `body {font-size: 16pt}`

## font-size-adjust

Description: Defines an aspect ratio to maintain when sizing fonts. This enables users to adjust for the text height when resizing.

Values: `none`: (default) Font's aspect ratio ignored.

Any valid number: Number representing the aspect value for the font.

`inherit`: Takes the same value as its parent.

Example: `p {font-size-adjust: 0.45}`

## font

Description: Shorthand property for defining all font properties at once. If not included, a property is set to its default value.

Values: `font-style`: Font style value.

`font-variant`: Font variant value.

`font-weight`: Font weight value.

`font-size`: Font size value.

`line-height`: Line height value.

`font-family`: Font family value.

`inherit`: Takes the same value as its parent.

Example: `body {font: italic bold 16pt 110% impact Garmond sans-serif}`

## Box Model

All elements are enveloped in a box made from the actual content, padding, border, and margins. Learning how to control these properties helps as you lay out your pages.

### margin-top, margin-right, margin-bottom, margin-left

Description: Defines the margin width for the designated side.

Values: Any valid length: Number representing the width of the margin. Default is 0.

Any valid percentage: Percentage of window to use for the width of the padding.

`inherit`: Takes the same value as its parent.

Example: `p {margin-top: 20px}`

### margin

Description: Shorthand property for defining margins for all sides of an element at once. This property can include one to four values. One value sets all margins to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `margin-top`: Width of the top margin.

`margin-right`: Width of the right margin.

`margin-bottom`: Width of the bottom margin.

`margin-left`: Width of the left margin.

`inherit`: Takes the same value as its parent.

Example: `body {margin: 20px 30px 5px}`

### padding-top, padding-right, padding-bottom, padding-left

Description: Defines the padding width for the designated side. Padding separates the text from the border.

Values: Any valid length: Number representing the width of the padding. Default is 0.

Any valid percentage: Percentage of window to use for the width of the padding.

`inherit`: Takes the same value as its parent.

Example: `p {padding-top: 20px}`

### padding

Description: Shorthand property for defining padding widths for all sides of an element at once. This property can include one to four values. One value sets all padding widths to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `padding-top`: Width of the top padding.

`padding-right`: Width of the right padding.

`padding-bottom`: Width of the bottom padding.

`padding-left`: Width of the left padding.

`inherit`: Takes the same value as its parent.

Example: `body {padding: 20px 30px 5px}`

## **border-top-width, border-right-width, border-bottom-width, border-left-width**

Description: Defines the border width for the designated side. The border comes between the padding and margin.

Values: `thin`: Creates a thin weight border.

`medium`: (default) Creates a medium weight border.

`thick`: Creates a thick weight border.

`inherit`: Takes the same value as its parent.

Example: `p {border-top-width: 10px}`

## **border-width**

Description: Shorthand property for defining border widths for all sides of an element at once. This property can include one to four values. One value sets all border widths to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `border-top-width`: Width of the top border.

`border-right-width`: Width of the right border.

`border-bottom-width`: Width of the bottom border.

`border-left-width`: Width of the left border.

`inherit`: Takes the same value as its parent.

Example: `body {border-width: 20px 30px 5px}`

## **border-top-color, border-right-color, border-bottom-color, border-left-color**

Description: Defines the border color for the designated side. The border comes between the padding and margin.

Values: Any valid color: Specifies the border color by setting its red-green-blue (rgb) intensities. The intensity values are rated from 0 from 255.

`inherit`: Takes the same value as its parent.

Example: `p {border-top-color: rgb(255, 0, 255)}`

## **border-color**

Description: Shorthand property for defining border colors for all sides of an element at once. This property can include one to four values. One value sets all border colors to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `border-top-color`: Color of the top border.

`border-right-color`: Color of the right border.

`border-bottom-color`: Color of the bottom border.

`border-left-color`: Color of the left border.

`transparent`: Makes the borders transparent.

`inherit`: Takes the same value as its parent.

Example: `body {border-color: blue red pink}`

## **border-top-style, border-right-style, border-bottom-style, border-left-style**

Description: Defines the border style for the designated side. The border comes between the padding and margin.

Values: `none`: Specifies no border style.

`hidden`: Also specifies no border style, but acts differently for tables.

`dotted`: Creates a series of dots.

`dashed`: Creates a series of dashed lines.

`solid`: Creates a solid, non-breaking line.

`double`: Creates a two parallel, solid, non-breaking lines.

`groove`: Creates a 3D carved style border.

`ridge`: Creates a 3D raised style border.

`inset`: Creates a 3D inset style border.

`outset`: Creates a 3D outset style border.

`inherit`: Takes the same value as its parent.

Example: `p {border: top: style: double}`

## **border-style**

Description: Shorthand property for defining border styles for all sides of an element at once. This property can include one to four values. One value sets all border styles to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `border-top-style`: Style of the top border.

`border-right-style`: Style of the right border.

`border-bottom-style`: Style of the bottom border.

`border-left-style`: Style of the left border.

`inherit`: Takes the same value as its parent.

Example: `body {border-style: double solid}`

## **border-top, border-right, border-bottom, border-left**

Description: Shorthand properties for defining several border properties at once for the designated side. Each separate property applies to its named side. The following definitions use the top.

Values: `border-top-width`: Width of the top border.

`border-top-style`: Style of the top border.

`border-top-color`: Color of the top border.

`inherit`: Takes the same value as its parent.

Example: `p {border-top: thin double blue}`

## border

Description: Shorthand property for defining borders for all sides of an element at once. The values are applied equally to all sides of the element.

Values: `border-width`: Width of the border.  
`border-style`: Style of the border.  
`border-color`: Color of the border.  
`inherit`: Takes the same value as its parent.

Example: `body {border: 4px solid red}`

## outline-width

Description: Shorthand property for defining outline widths for all sides of an element at once. This property can include one to four values. One value sets all outline widths to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `thin`: Creates a thin weight border.  
`medium`: (default) Creates a medium weight border.  
`thick`: Creates a thick weight border.  
`inherit`: Takes the same value as its parent.

Example: `body {outline-width: 20px 30px 5px}`

## outline-style

Description: Shorthand property for defining outline styles for all sides of an element at once. This property can include one to four values. One value sets all outline styles to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: `none`: Specifies no border style.  
`dotted`: Creates a series of dots.  
`dashed`: Creates a series of dashed lines.  
`solid`: Creates a solid, non-breaking line.  
`double`: Creates a two parallel, solid, non-breaking lines.  
`groove`: Creates a 3D carved-style border.  
`ridge`: Creates a 3D raised-style border.  
`inset`: Creates a 3D inset-style border.  
`outset`: Creates a 3D outset-style border.  
`inherit`: Takes the same value as its parent.

Example: `body {outline-style: double solid}`

## outline-color

Description: Property for defining outline colors. This property can include one to four values. One value sets all outline colors to that value. Two values set the top and bottom to the first and the left and right to the second. Three values set the top to the first, left and right to the second, and the bottom to the third.

Values: Any valid color: Specifies the border color.  
`invert`: (default) Inverts the colors of the outline.

`inherit`: Takes the same value as its parent.

Example: `body {outline-color: blue red pink}`

## outline

Description: Shorthand property for defining outlines. The values are applied equally to all sides of the element.

Values: `outline-width`: Width of the outline.  
`outline-style`: Style of the outline.  
`outline-color`: Color of the outline.  
`inherit`: Takes the same value as its parent.

Example: `body {outline: 4px solid red}`

## Visual Formatting and Positioning

The display property provides a way to define elements for the style sheet. Once defined, the position properties can place the elements exactly where you want them to go.

### display

Description: Defines the type of display box the element creates. These different types of boxes interact differently with each other as they are laid out on a page.

Values: `inline`: (default) Creates an inline display box.  
`block`: Creates a block display box.  
`list-item`: Creates a list-item inline display box.  
`marker`: Creates generated content to appear before or after a display box. Only used with the `:before` and `:after` pseudo elements.  
`none`: Creates no display box. The element has no effect on the overall layout.  
`run-in`: Creates a box like a block display box depending on its location.  
`compact`: Creates a box like an inline display box depending on its location.  
`table`, `inline-table`, `table-row-group`, `table-column`, `table-column-group`, `table-header-group`, `table-footer-group`, `table-row`, `table-cell`, `table-caption`: Creates a table display box matching the property name.  
`inherit`: Takes the same value as its parent.

Example: `p {display: block}`

### position

Description: Defines the positioning method to use.

Values: `static`: (default) Defines a normal box using default HTML layout.  
`relative`: Positioned box is offset from its normal layout position.  
`absolute`: Positioned box is offset from its containing box's position and they do not affect the layout.  
`fixed`: Positioned box is offset like the absolute model, but is fixed in the browser window and doesn't move when the window is scrolled.  
`inherit`: Takes the same value as its parent.

Example: `img {position: absolute}`

## top, right, bottom, left

Description: Defines the offset width from the designated edge.

Values: `auto`: (default) Enables the browser to select an offset width to position all elements.

Any valid length: Number representing the width from the edge.

Any valid percentage: Percentage of window to offset from the edge.

`inherit`: Takes the same value as its parent.

Example: `ul {top: 20px; right: 40px}`

## width

Description: Defines the width of a display box.

Values: `auto`: (default) Enables the browser to select a width for the display box.

Any valid length: Number representing the width of the display box.

Any valid percentage: Percentage of window to use for the display box width.

`inherit`: Takes the same value as its parent.

Example: `blockquote {width: 260px}`

## min-width, max-width

Description: Defines the minimum or maximum widths of a display box.

Values: Any valid length: Number representing the minimum or maximum widths of the display box.

Any valid percentage: Percentage of window to use for the minimum or maximum widths.

`none`: No width limit, applies only to the `max-width` property.

`inherit`: Takes the same value as its parent.

Example: `blockquote {min-width: 100px; max-width: 400px}`

## height

Description: Defines the height of a display box.

Values: `auto`: (default) Enables the browser to select a height for the display box.

Any valid length: Number representing the height of the display box.

Any valid percentage: Percentage of window to use for the display box height.

`inherit`: Takes the same value as its parent.

Example: `blockquote {height: 260px}`

## min-height, max-height

Description: Defines the minimum or maximum heights of a display box.

Values: Any valid length: Number representing the minimum or maximum heights of the display box.

Any valid percentage: Percentage of window to use for the minimum or maximum heights.

`none`: No height limit, applies only to the `max-height` property.

`inherit`: Takes the same value as its parent.

Example: `blockquote {min-height: 100px; max-height: 400px}`

## line-height

Description: Defines the line spacing for an element box.

Values: `normal`: (default) Enables the browser to set the value to fit all elements on the page.

Any valid length: Number representing the height of the display box.

Any valid percentage: Percentage of window to use for the box height.

Any valid number: Number of times the font size height.

`inherit`: Takes the same value as its parent.

Example: `blockquote {line-height: 2.2}`

## vertical-align

Description: Defines the vertical positioning inside a line box.

Values: `baseline`: (default) Aligns the box's baseline to its parent baseline.

`middle`: Aligns the box's middle to its parent's baseline.

`top`: Aligns the box's top with the top of the line box.

`bottom`: Aligns the box's bottom to its parent's baseline.

`sub`: Aligns the box's text to be at subscript level to its parent's baseline.

`super`: Aligns the box's text to be at superscript level of its parent's baseline.

`text-top`: Aligns the box's top to the top of the parent's text.

`text-bottom`: Aligns the box's bottom to the bottom of the parent's baseline.

Any valid length: Defines the distance to raise the box's level. Negative values lower its level.

Any valid percentage: Percentage to raise the box's level. Negative values lower its level.

`inherit`: Takes the same value as its parent.

Example: `blockquote {vertical-align: super}`

## float

Description: Defines whether the display box should float to the left or right.

Values: `none`: (default) The display box not float.

`left`: Causes the display box to float to the left and content flows to the right.

`right`: Causes the display box to float to the right and content flows to the left.

`inherit`: Takes the same value as its parent.

Example: `img {float: right}`

## clear

Description: Defines whether content appears adjacent to the side of float box or not.

Values: none: (default) Content not constrained next to float boxes.  
left: Content doesn't appear to the left of a float box.  
right: Content doesn't appear to the right of a float box.  
both: Content doesn't appear to the left or right of a float box.  
inherit: Takes the same value as its parent.

Example: `img {clear: both}`

## overflow

Description: Defines whether a display box is displayed when it overflows the element's box.

Values: visible: (default) The overflowed box is visible and not clipped.  
hidden: The overflowed portion is clipped.  
scroll: The overflowed portion is clipped and any scrollbars are made visible.  
auto: Enables the browser to determine whether overflowed areas are clipped.  
inherit: Takes the same value as its parent.

Example: `pre {overflow: visible}`

## clip

Description: Defines the clipping area for overflowed sections.

Values: auto: (default) Causes the clipping region to have the same size and location as the element's box.  
`rect(top, right, bottom, left)`: The clipping area is defined by the offsets from the top, right, bottom, and left length values.  
inherit: Takes the same value as its parent.

Example: `blockquote {clip: rect(5px, 4px, 2px, 4px)}`

## visibility

Description: Defines whether an element is visible.

Values: visible: Makes the element visible.  
hidden: Makes the element hidden, but it still effects the layout.  
collapse: Same as hidden, except when used on tables.  
inherit: (default) Takes the same value as its parent.

Example: `img {visibility: visible}`

## z-index

Description: Defines the stacking order for elements.

Values: auto: (default) Causes the element box to accept the same stacking order as its parent's box.  
Any valid integer: An integer value representing the stacking order. Lower values have a lower stacking order.  
Inherit: Takes the same value as its parent.

Example: `img {z-index: 3}`

## cursor

Description: Defines how the cursor looks when moved over an element.

Values: `auto`: (default) Cursor determined by the browser.  
`crosshair`: Cursor resembles a crosshair.  
`default`: Cursor is the default cursor for the user's system.  
`pointer`: Cursor resembles a pointer indicating a link.  
`move`: Cursor indicates that something is to be moved.  
`e-resize`, `ne-resize`, `nw-resize`, `n-resize`, `se-resize`, `sw-resize`, `s-resize`, `w-resize`: Cursor indicates a corner position.  
`text`: Cursor text.  
`wait`: Cursor indicates the system is busy.  
`help`: Cursor indicates a help location.  
Any valid URL: URL of a cursor file.  
`inherit`: Takes the same value as its parent.

Example: `img {cursor: pointer}`

## direction

Description: Defines the writing direction for text blocks.

Values: `ltr`: (default) Sets writing direction from left to right.  
`rtl`: Sets writing direction from right to left.  
`inherit`: Takes the same value as its parent.

Example: `body {direction: ltr}`

## unicode-bidi

Description: Enables the text writing direction to be changed.

Values: `normal`: (default) Does not enable other writing directions.  
`embed`: Enables writing direction to be set using the `direction` property.  
`bidirectional-override`: Enables writing direction to be set using the `direction` property. Applies to additional blocks.  
`inherit`: Takes the same value as its parent.

Example: `img {unicode-bidi: embed}`

## Generated Content and Lists

With these properties, you have control over the style of your list boxes and how the numbers or bullets are presented. They make it easy to have your list count by twos starting from seven.

## content

Description: Used with the `:before` and `:after` pseudo elements to generate content.

Values: Any valid string: String to appear before or after the element.  
Any valid URL: URL to an external file to appear before or after an element.  
`counter()`: Defines a counter with a name to insert the value controlled by the `counter-increment` and `counter-reset` properties.

`open-quote, close-quote`: Enables quote marks to be included. Used with the `quotes` property.

`no-open-quote, no-close-quote`: Inserts no quote marks.

`attr()`: Inserts the value of an attribute for the element.

`inherit`: Takes the same value as its parent.

Example: `pre:after {content: "thank you and good-night."}`

## quotes

Description: Defines the pairs of quotation marks to use for each level of embedded quote marks.

Values: First valid string: Pair of characters to use for the outmost quotation marks.

Second valid string: Pair of characters to use for inner quotation marks.

`none`: No quote marks are created.

`inherit`: Takes the same value as its parent.

Example: `q {quotes: '"" '"" '<' '>'}`

## counter-increment

Description: Increases the value of the specified counter.

Values: `none`: (default) Counter is not incremented.

Counter name and valid number: Identifies the counter and accepts an integer value, that counter is incremented. Negative values are valid.

`inherit`: Takes the same value as its parent.

Example: `h1 {counter-increment: MyCounter 2}`

## counter-reset

Description: Resets the value of a specified counter.

Values: `none`: (default) Counter is not reset.

Counter name and valid number: Identifies the counter and accepts an integer value that the counter is reset. Negative values are valid.

`inherit`: Takes the same value as its parent.

Example: `h1 {counter-reset: MyCounter 2}`

## marker-offset

Description: Defines the distance between a list marker (such as a bullet) and the text.

Values: `auto`: (default) Enables the browser to determine the spacing.

Any valid length: The space between a marker and the text.

`inherit`: Takes the same value as its parent.

Example: `h1 {marker-offset: 12px}`

## list-style-type

Description: Defines the list style to be applied to the list markers.

Values: `disc`: (default) Creates a disc-shaped bullet.

`circle`: Creates a circular-shaped bullet.

`square`: Creates a square-shaped bullet.

`decimal`: Numbers lists using decimal numbers, beginning with 1.  
`decimal-leading-zero`: Numbers lists using decimal numbers padded with a zero, such as 01, 02, 03, and so on.  
`lower-roman`: Numbers lists using lowercase Roman numerals.  
`upper-roman`: Numbers lists using uppercase Roman numerals.  
`hebrew`: Numbers lists using Hebrew numerals.  
`georgian`: Numbers lists using Georgian numerals.  
`armenian`: Numbers lists using Armenian numerals.  
`CJK-ideographic`: Numbers lists using ideographic numerals.  
`lower-latin`, `lower-alpha`: Uses lowercase ASCII characters.  
`upper-latin`, `upper-alpha`: Uses uppercase ASCII characters.  
`lower-greek`: Uses lowercase Greek characters.  
`hiragana`: Uses Japanese hiragana characters.  
`hiragana-iroha`: Uses Japanese hiragana iroha characters.  
`katakana-iroha`: Uses Japanese katakana iroha characters.

Values: `none`: No marker is used.  
`inherit`: Takes the same value as its parent.

Example: `ol {list-style: upper-alpha}`

## list-style-image

Description: Defines the image of a list marker.

Values: `none`: (default) Sets no marker image.  
Any valid URL: URL of the marker image.  
`inherit`: Takes the same value as its parent.

Example: `ul {list-style-image: url("bullet3.gif")}`

## list-style-position

Description: Defines the location of the list box markers.

Values: `inside`: Markers appear within the element box.  
`outside`: (default) Markers appear outside the element box.  
`inherit`: Takes the same value as its parent.

Example: `h1 {list-style-position: inside}`

## list-style

Description: Shorthand property for defining all list style properties at once. If not included, a property is set to its default value.

Values: `list-style-type`: Marker type.  
`list-style-position`: Marker position.  
`list-style-image`: Marker image.  
`inherit`: Takes the same value as its parent.

Example: `ul {list-style: circle inside url("bullet4.gif")}`

# Tables

Table control is new to CSS2. These properties enable you to define the style, spacing, and layout of your tables.

## caption-side

Description: Defines the position of a table caption relative to the table.

Values: `top`: (default) Positions the caption at the top of the table.  
`right`: Positions the caption to the right of the table.  
`bottom`: Positions the caption at the bottom of the table.  
`left`: Positions the caption to the left of the table.  
`inherit`: Takes the same value as its parent.

Example: `table {caption-side: top}`

## table-layout

Description: Defines how the table is laid out.

Values: `auto`: (default) Enables the browser to decide how to layout the table.  
`fixed`: Tables are laid out using a fixed method.  
`inherit`: Takes the same value as its parent.

Example: `table {table: layout: fixed}`

## border-collapse

Description: Defines how the table borders are displayed.

Values: `collapse`: (default) Collapses the table cell borders into a common border.  
`separate`: Keeps each table cell's border separated.  
`inherit`: Takes the same value as its parent.

Example: `table {border-collapse: separate}`

## border-spacing

Description: Defines the spacing between table borders. Only one length value applies equally to both horizontal and vertical directions.

Values: First valid length: Defines the horizontal width separating table cell borders.  
Second valid length: Defines the vertical width separating table cell borders.  
`inherit`: Takes the same value as its parent.

Example: `table {border-spacing: 4px}`

## empty-cells

Description: Defines how to render the border of empty cells.

Values: `show`: (default) Enables the borders of empty cells to be seen.  
`hide`: Hides the borders of empty cells.  
`inherit`: Takes the same value as its parent.

Example: `table {empty-cells: show}`

## speaking-header

Description: Enables a screen reader to speak table headers.

Values: `once`: (default) Causes the header to be spoken only once for each column of cells.

`always`: Causes the header to be spoken each time for a column of cells.

`inherit`: Takes the same value as its parent.

Example: `table {speaking-header: once}`

## column-span

Description: Defines the number of columns to span.

Values: Any valid number: The number of columns to span. Default is 1.

`inherit`: Takes the same value as its parent.

Example: `table {column-span: 3}`

## row-span

Description: Defines the number of rows to span.

Values: Any valid number: The number of rows to span. Default is 1.

`inherit`: Takes the same value as its parent.

Example: `table {row-span: 3}`

## Paged Media

These properties enable you to split your page content into pre-defined pages that output correctly to a printer or external device.

### size

Description: Defines the size and orientation of a page.

Values: `auto`: (default) Enables the browser to determine the page size.

First valid length: Sets the page width.

Second valid length: Sets the page height.

`landscape`: Sets the page orientation to landscape.

`portrait`: Sets the page orientation to portrait.

`inherit`: Takes the same value as its parent.

Example: `p {size: 8.5in 11in portrait}`

### marks

Description: Enables printed pages to have crop and cross marks.

Values: `none`: (default) No printing marks are included.

`crop`: Displays crop marks.

`cross`: Displays registration marks.

`inherit`: Takes the same value as its parent.

Example: `p {marks: crop cross}`

## page-break-before

Description: Defines the page breaks for a page.

Values: `auto`: (default) Enables the browser to determine the page breaks.  
`always`: Always forces a page break before a box.  
`avoid`: Avoids placing a page break before a box.  
`left`: Always forces a page break before a box so that the next page is on the left.  
`right`: Always forces a page break before a box so that the next page is on the right.  
`inherit`: Takes the same value as its parent.

Example: `p {page-break-before: avoid}`

## page-break-after

Description: Defines the page breaks for a page.

Values: `auto`: (default) Enables the browser to determine the page breaks.  
`always`: Always forces a page break after a box.  
`avoid`: Avoids placing a page break after a box.  
`left`: Always forces a page break after a box so that the next page is on the left.  
`right`: Always forces a page break after a box so that the next page is on the right.  
`inherit`: Takes the same value as its parent.

Example: `p {page-break-after: avoid}`

## page-break-inside

Description: Defines the page breaks for a page.

Values: `auto`: (default) Enables the browser to determine the page breaks.  
`avoid`: Avoids placing a page break within a box.  
`inherit`: Takes the same value as its parent.

Example: `p {page-break-inside: avoid}`

## page

Description: Identifies a page with a name.

Values: `auto`: (default) Enables the browser to identify pages.  
Any valid name: Gives a page a name. The name can be any string.

Example: `p {page: Mypage}`

## orphans

Description: Defines the minimum number of lines of text that must be available for the next page or column if a page break occurs within this paragraph.

Values: Any valid number: An integer defining the minimum number of text lines which must be used for starting the next page or column.  
`inherit`: Takes the same value as its parent.

Example: `p {orphans: 4}`

## widows

Description: Defines the minimum number of lines of text that must be available for the current page or column if a page break occurs within this paragraph.

Values: Any valid number: An integer defining the number of sentences that must be left on the top of a page. Default is 2.

`inherit`: Takes the same value as its parent.

Example: `p {widows: 4}`

## Aural Style Sheets

As a way to define Web pages for individuals with visual handicaps, aural style sheets enable designers to specify how screen readers interpret Web pages.

## volume

Description: Defines the loudness of text read by a screen reader.

Values: Any valid number, 0100: An integer ranged between 0 and 100 with 0 being minimum and 100 being maximum.

Any valid percentage, 0100: A percentage increase or decrease from the current value.

`silent`: No sound emitted.

`x-soft`: Quietest level of sound, same as 0.

`soft`: Quiet level of sound, same as 25.

`medium`: (default) Normal level of sound, same as 50.

`loud`: Loud level of sound, same as 75.

`x-loud`: Loudest level of sound, same as 100.

`inherit`: Takes the same value as its parent.

Example: `body {volume: soft}`

## speak

Description: Defines how the words are spoken.

Values: `normal`: (default) Words are spoken normally.

`none`: Words are not spoken.

`Spell-out`: Words are spelled letter by letter.

`inherit`: Takes the same value as its parent.

Example: `span {speak: spell-out}`

## pause-before

Description: Causes a pause before the element is read.

Values: Any valid time: The amount of time to pause before reading the element.

Any valid percentage: The percent to pause before reading the element.

`inherit`: Takes the same value as its parent.

Example: `span {pause-before: 500ms}`

## pause-after

Description: Causes a pause after the element is read.

Values: Any valid time: The amount of time to pause after reading the element.

Any valid percentage: The percent to pause after reading the element.

*inherit*: Takes the same value as its parent.

Example: `span {pause-after: 500ms}`

## pause

Description: Shorthand property for setting the `pause-before` and `pause-after` the element is read. If only one time or percent value is given, it applies to both before and after.

Values: First valid time: The amount of time to pause before reading the element.

Second valid time: The amount of time to pause after reading the element.

Any valid percentage: The percent to pause before reading the element.

Any valid percentage: The percent to pause after reading the element.

*inherit*: Takes the same value as its parent.

Example: `span {pause: 500ms 300ms}`

## cue-before

Description: Causes a cue before the element is read.

Values: Any valid URL: URL of an audio file to play before reading the element.

*none*: No audio is played before the element is read.

*inherit*: Takes the same value as its parent.

Example: `span {cue-before: url("bell.wav")}`

## cue-after

Description: Causes a cue after the element is read.

Values: Any valid URL: URL of an audio file to play after reading the element.

*none*: No audio is played after the element is read.

*inherit*: Takes the same value as its parent.

Example: `span {cue-after: url("bell2.wav")}`

## cue

Description: Shorthand property that causes a cue before and after the element is read. If only one URL is given, it applies to both before and after.

Values: First valid URL: URL of an audio file to play before reading the element.

Second valid URL: URL of an audio file to play after reading the element.

*none*: No audio is played before the element is read.

*inherit*: Takes the same value as its parent.

Example: `span {cue: url("ding.wav") url("dong.wav")}`

## play-during

Description: Defines an audio file to be played in the background while text is being read.

Values: Any valid URL: URL of an audio file to play in the background while reading the element.

mix: Mix the current audio with the parent audio file and play both together.

repeat: Repeat the audio until all the text has been read.

auto: (default) Enable the parent elements audio to continue to play.

none: No background audio is played.

inherit: Takes the same value as its parent.

Example: `body {play-during: url("chatter.wav") mix}`

## azimuth

Description: Defines the spatial location of an audio file horizontally around the listener's head.

Values: Any valid angle: An angle value between zero to 360 degrees. Negative values are not allowed.

left-side: Sound from the left side of the head, or 270 degrees.

far-left: Sound from the distant left of the head, or 300 degrees.

left: Sound from the left of the head, or 320 degrees.

center-left: Sound from the center left of the head, or 340 degrees.

center: Sound from the center of the head, or zero degrees.

center-right: Sound from the center right of the head, or 20 degrees.

right: Sound from the right of the head, or 40 degrees.

far-right: Sound from the distant right of the head, or 60 degrees.

left-side: Sound from the right side of the head, or 270 degrees.

leftwards: Sound moved to the left of the current location.

rightwards: Sound moved to the right of the current location.

behind: Sound moved to behind the head at that location.

inherit: Takes the same value as its parent.

Example: `h1 {azimuth: left-side}`

## elevation

Description: Defines the spatial location of an audio file vertically around the listener's head.

Values: Any valid angle: An angle value between 90 to -90 degrees. Negative values are allowed.

below: Sound from below the head, or -90 degrees.

level: Sound from the front of the head, or 0 degrees.

above: Sound from the above of the head, or 90 degrees.

higher: Sound moved up form the current location.

inherit: Takes the same value as its parent.

Example: `h1 {elevation: above}`

## speech-rate

Description: Defines how quickly the element text as it's read.

Values: Any valid number: The speaking rate in words-per-minute.

- `x-slow`: 80 words per minute.
- `slow`: 120 words per minute.
- `medium`: (default) 180 to 200 words per minute.
- `fast`: 300 words per minute.
- `x-fast`: 500 words per minute.
- `faster`: Causes the words to be read faster than the current speed, adds 40 words per minute.
- `slower`: Causes the words to be read slower than the current speed, subtracts 40 words per minute.
- `inherit`: Takes the same value as its parent.

Example: `body {speech-rate: fast}`

## voice-family

Description: Defines the voice type to use to read the element's text. It can include several voice families separated by commas. The list order defines the priority.

Values: Voice name: Voice to use to read the text.

Generic voice name: Generic voice class to use to read the text. Generic voices include: `male`, `female`, and `child`.

`inherit`: Takes the same value as its parent.

Example: `body {voice-family: Bob, male}`

## pitch

Description: Defines the pitch of the element text.

Values: Any valid frequency: The pitch in Hertz (Hz).

- `x-low`: Lowest pitch.
- `low`: Low pitch.
- `medium`: (default) Average pitch.
- `high`: Higher than normal pitch.
- `x-high`: Highest pitch.
- `inherit`: Takes the same value as its parent.

Example: `body {pitch: high}`

## pitch-range

Description: Defines the pitch range of the element text as it's read.

Values: Any valid number: A value between zero and 100 that defines the pitch range. The default value, 50, is normal inflection.

`inherit`: Takes the same value as its parent.

Example: `body {pitch-range: 50}`

## stress

Description: Defines the stress of the element text as it's read.

Values: Any valid number: A value between zero and 100 that defines the pitch range. The default value, 50, is normal inflection.

`inherit`: Takes the same value as its parent.

Example: `body {stress: 50}`

## **richness**

Description: Defines the richness of the element text as it's read.

Values: Any valid number: A value between zero and 100 that defines the pitch range. The default value, 50, is normal inflection.

`inherit`: Takes the same value as its parent.

Example: `body {richness: 50}`

## **speak-punctuation**

Description: Defines how punctuation is spoken.

Values: `code`: Punctuation is spoken literally.

`none`: Punctuation is not spoken.

`inherit`: Takes the same value as its parent.

Example: `body {speak-punctuation: code}`

## **speak-numeral**

Description: Defines how numbers are spoken.

Values: `digits`: Numbers are spoken as individual digits.

`continuous`: Numbers are spoken as a full number.

`inherit`: Takes the same value as its parent.

Example: `body {speak-numeral: digits}`