Some notes on CSS

John R Hudson*†

10th March 2018

Contents

*The author would welcome notification of any errors or possible misunderstandings.

†This edition presented at the meeting of Bradford GNU/LUG on Tuesday, 13 March 2018.

1 Background

During the 1990s there were several attempts to develop style sheets which separated the presentation of material on a website from its content and structure as defined by HTML. CSS, or Cascading Style Sheets, first published in 1996, differed from other attempts in that it allowed styles to be inherited, thereby avoiding the need to create separate style sheets for every page.

CSS2 was published in 1998 but adoption was slow and there were a number of errors in the specification which were remedied through a series of updates between 2007 and 2011 known as CSS2.1. Since then CSS development, sometimes known as 'Level 3,' has moved to rolling updates.

The updates consulted for these notes are listed in appendix A on page 63. Please note that these notes are intended as a quick reference and not as a replacement for reading the source documentation which is regularly updated (see CSS Current work).

2 Using styles

A style is defined by a rule which, unless otherwise specified, is assumed to be encoded in utf-8.and which may be embedded

- within the `style=" "` attribute of an HTML element

- within the HTML `<style>...  </style>` element

or included in a separate style sheet called from within

- the `<style>...  </style>` element with the `@import "style-sheet.css";` directive (see section 3.1)
- the `<link .  />` element with, for example,
  `<link rel="stylesheet" href="style-sheet.css" type="text/css"/>`

The last option is the preferred option with HTML.

3 Syntax

3.1 @ rules

`@` rules (`@` followed by a keyword) are terminated by a semi-colon and cannot occur in a block.

@counter-style allows the definition of counter styles which can then be used in `list-style-type` properties; a new counter style is defined by

```
@counter-style <counter-style-name> {system: <counter-system>;
symbols: <list of symbols>; negative: " "; prefix: " ";
suffix: " "; range: " "; pad: "<integer> && <symbol>"; spoken: " ";
fallback: " ";}
```

`<counter-style-name>` must not be `"none"` or `"disc"`.

The following styles are predefined for `system`:

```
system: [cyclic | numeric | alphabetic | symbolic | additive |
[fixed <integer>?]
```

where

cyclic cycles through the list of symbols; providing just one symbol enables the definition of a non-standard bullet point

numeric runs through a list of symbols enclosed in single inverted commas interpreting the first symbol as having the value 0 and each symbol as having a place value; there must be at least two symbols in the symbol list

alphabetic runs through a list of symbols as if they were in the `lower-alpha` style, interpreting the first symbol as having the value 1; there must be at least two symbols in the symbol list

symbolic runs through the list of symbols and then doubles and triples them for the second and third runs and so on; it starts like the `upper-alpha` style but. for the second run, it will create AA, BB CC ... rather than AA, AB, AC ...

additive uses the decreasing value comma separated value pairs in a list of symbols defined with the keyword `additive-symbols` rather than `symbols` to create a list using non-decimal number systems such as Roman numerals

fixed runs through a list of symbols and then falls back to whatever is the default counter style.

In addition, the keyword `extends` can be used to extend an existing `list-style-type`; for example:

```
@counter-style decimal-paren { system: extends decimal; suffix: ") "; }
```

will create a decimal style terminated by a parenthesis rather than a full stop.

`symbols: " ";` specifies the symbols to be used

`negative: " ";` specifies how negative numbers are to be displayed

`prefix: " ";` specifies one or more symbols preceding the counter

`suffix: " ";` specifies one or more symbols following the counter; the default value is `"\2E\20"`, that is, full stop followed by space

`range: [[ <integer> | infinite ]{2}]# | auto ];` may take a comma separated list of ranges in which `infinite` may be the first or second value in the range or `auto` which takes the range of the specified system

`pad: <integer> "<symbol>";` takes an integer specifying the amount of padding and a symbol to be used for padding where necessary

`fallback: <system>;` specifies the system to use if the specified system runs out of values

`spoken: [auto | bullets | numbers | words | spell-out | <counter-style-name>];` specifies how speech synthesisers are to interpret the specified system where:

auto if the counter style's system is `alphabetic`, = `spell-out`; if it is `cyclic`, = `bullets`; if it is `extends`, = `auto` for the extended style. Otherwise, it has the same effect as `numbers`.

bullets the UA speaks a cue that represents an unordered list item being read out.

numbers the counter's value is spoken as a number in the content language.

words the counter's value is spoken as normal text in the content language.

spell-out the counter's value is spoken letter-by-letter in the content language.

<counter-style-name> the counter's value is spoken out in the specified style or is treated as `auto`.

The following counter-styles are predefined:

Numeric:

```
decimal, decimal-leading-zero, arabic-indic, armenian, upper-armenian,
lower-armenian, bengali, cambodian, khmer, cjk-decimal, devanagari,
georgian, gujarati, gurmukhi, hebrew, kannada, lao, malayalam,
mongolian, myanmar, oriya, persian, lower-roman, upper-roman, tamil,
telugu, thai, tibetan
```

Alphabetic:

```
lower-alpha, lower-latin, upper-alpha, upper-latin,
cjk-earthly-branch, cjk-heavenly-stem, lower-greek, hiragana,
hiragana-iroha, katakana, katakana-iroha
```

Symbolic:

```
disc, circle, square, disclosure-open, disclosure-closed
```

@import allows the importation of style sheets with, for example,

```
@import "mystyle.css";
l@import url("mystyle.css");
```

Where an `@import` rule is only intended to apply to one media group, this may also be specified

```
@import url("fineprint.css") print;
@import url("bluish.css") projection, tv;
```

`@import` rules may also use the syntax of `@media` rules.

@media introduces conditional rules intended for particular types of media (all other rules apply to all types of media):

```
@media screen and (<property>: <value>), projection and (<property>:
<value>) {...}
```

Note the alternative syntax for the DECLARATION (section 7), for example,

```
@media print {body {font-size: 10pt }}:
@media screen {body {font-size: 13px }};
@media screen, print {body {line-height: 1.2 }};
```

@namespace declares a default namspace, for example,

```
@namespace url("<url>");
```

while a namespace prefix may be declared with

```
@namspace prefix url("<url>");
```

@page introduces rules intended only for use with printed media; it takes the standard form for declarations (section 7):

```
@page: [first | left | right ] {<declaration>; ...;}
```

These cannot be used within `<style>` elements with the `scoped` attribute.

@supports interrogates the user agent to determine what features it supports:

```
@supports (<property>: <value> [!important]) { ... }
```

enabling alternative styling where a feature is or is not supported.

Multiple conditions must be linked with and or `or` and may be preceded by `not` but parentheses must be used to avoid any confusion.

A single declaration may apply to a comma separated list of elements and there can be space separated declarations for different elements but the second and all subsequent elements in the declaration must be preceded by `#` as in this example:

```
@supports ( display: flex ) {
body, #navigation, #content { display: flex; }
#navigation { background: blue; color: white; }
#article { background: white; color: black; } }
```

3.2 Backslash

The backslash may appear:

- in a comment
- outside a string followed by a newline character
- to cancel the meaning of a special CSS character, eg. "\" "
- to introduce a hexadecimal string of up to six digits which represents a character.

Table 1: Media groups

Type				
baille	continuous	tactile	grid	both
embossed	paged	tactile	grid	static
handheld	both	visual, audio, speech	both	both
print	paged	visual	bitmap	static
projection	paged	visual	bitmap	interactive
screen	continuous	visual, audio	bitmap	both
speech	continuous	speech	N/A	both
tty	continuous	visual	grid	both
tv	both	visual. audio	bitmap	both

4 Media types

Certain rules are only relevant to particular media types and `@media` rules (section 3.1) make it possible to have different rules for the same content where different media types are used; the recognised media types are:

all

braille for braille tactile feedback devices.

embossed for paged braille printers.

handheld for handheld devices

print for paged material and for documents viewed on screen in print preview mode.

projection for projected presentations

screen primarily for colour computer screens.

speech for speech synthesizers.

tty for media using a fixed-pitch character grid

tv for low resolution television-type devices .

A media type may belong to four or more media groups (Table 1).

5 Layout

5.1 The containing block

Each element has a containing block; the initial containing block of the root element is the size of the viewport or page area. The positions of the containing blocks of all other elements are determined by the positions of the padding edges (Figure 1) of their ancestor, preceding or adjacent blocks unless their position is `fixed`, in which case it is determined by the viewport or page area.

5.2 Formatting context

Elements may be laid out within a block or an inline formatting context; block boxes, floats, absolutely positioned elements, block containers and most block boxes with overhang create a

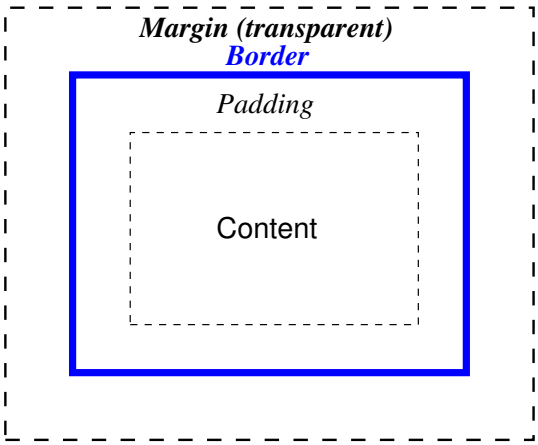

Figure 1: Box model (N.B. the default value of the margin and padding width is 0.)

new block formatting context in which boxes are laid out vertically within their containing box with their left (or right in RTL contexts) edge against the left (or right) edge of the containing box. In an inline formatting context boxes are laid out horizontally between the edges of the containing box and, if their combined length exceeds the width of the containing box, split over more than one line.

Flex boxes have a flexible layout such that the children of a flex container can be laid out in any direction, and can "flex" their sizes, either growing to fill unused space or shrinking to avoid overflowing the parent. Both horizontal and vertical alignment of the children can be easily manipulated. Nesting of these boxes (horizontal inside vertical, or vertical inside horizontal) can be used to build layouts in two dimensions.

Floats are boxes that are shifted left (or right) from the containing box edge; content may flow down their right (or left) side if there is space and inline boxes are either shortened to fit the available space to the right (or left) or placed below the float. Immediately following floats are placed to the right (or left) of the float if there is space or below it otherwise.

Grid layout boxes are like flex boxes except that they are inherently two dimensional.

5.3 Margins, borders and padding

The area occupied by any content is defined by its width and height, plus the width of any margin, border or padding (figure 1). Where the width of all three is zero, the width and height of the area occupied by the content is the same as its content area.

The background property of the margin is always transparent; otherwise, the properties of the content box, padding, borders and margins may be defined in a rule.

In the case of inline boxes, the left and right margins, borders and padding are split between the first and last inline boxes in the element; thus assigning values to the left and right margins, borders and padding of inline boxes should be avoided.

5.4 Page boxes

A page box contains the *page area* and the *margin*. The margins of a page box can be specified within an `@page` rule. Only percentage values can be used within a page box.

5.5 Tables

Tables should only be used for the representation of data relationships, not for layout or presentation purposes.

Tables may be contained in block or inline elements.

Tables are defined by their rows (header and data) so that columns may be defined implicitly from the cells in the rows.

Column and column group elements take the values of the border, background, visibility and width properties of their ancestor elements.

6 Units

6.1 Angles

The units of angles are:

deg: degrees

grad: grads

rad: radians

where a negative value is always equivalent to a positive value.

6.2 Colour

Colour values can be specified

- using the SVG 1.0 colour keywords (see appendix D on page 64):

```
aliceblue antiquewhite aqua aquamarine azure beige bisque
black blanchedalmondblue blueviolet brown burlywood cadetblue
chartreuse chocolate coral cornflowerblue cornsilk crimson cyan
darkblue darkcyan darkgoldenrod darkgray darkgreen darkgrey
darkkhaki darkmagenta darkolivegreen darkorange darkorchid
darkred darksalmon darkseagreen darkslateblue darkslategray
darkslategrey darkturquoise darkviolet deeppink deepskyblue
dimgray dimgrey dodgerblue firebrick floralwhite forestgreen
fuchsia gainsboro ghostwhite gold goldenrod gray green
greenyellow grey honeydew hotpink indianred indigo ivory khaki
lavender lavenderblush lawngreen lemonchiffon lightblue
lightcoral lightcyan lightgoldenrodyellow lightgray lightgreen
lightgrey lightpink lightsalmon lightseagreen lightskyblue
lightslategray lightslategrey lightsteelblue lightyellow lime
limegreen linen magenta maroon mediumaquamarine mediumblue
```

```
mediumorchid mediumpurple mediumseagreen mediumslateblue
mediumspringgreen mediumturquoise mediumvioletred midnightblue
mintcream mistyrose moccasin navajowhite navy oldlace olive
olivedrab orange orangered orchid palegoldenrod palegreen
paleturquoise palevioletred papayawhip peachpuff peru pink plum
powderblue purple red rosybrown royalblue saddlebrown salmon
sandybrown seagreen seashell sienna silver skyblue slateblue
slategray slategrey snow springgreen steelblue tan teal thistle
tomato turquoise violet wheat white whitesmoke yellow
yellowgreen
```

- using sRGB values:

```
color: rgb(r%,g%,b%);
color: rgb([0-255],[0-255],[0-255]);
color: #hex;
```

  Where both digits are the same as in `#ff0000`, the hex string can be shortened to `#f00`.

- using RGBA values[1]

```
color: rgba(r%,g%,b%,0.5);
color: rgba([0-255],[0-255],[0-255],0.5);
```

- using HSL values

```
color: hsl(h,s%,l%);
```

- using HSLA values

```
color: hsla(h,s%,l%,1);
```

- using `transparent` which is interpreted as:

```
color: rgba(0,0,0,0);
```

6.3 Counters

Counters are referred to by case-sensitive identifiers;

```
counter(<identifier>)
counter(<identifier>,<list-style-type>)
```

See list-style on page 39.

The default style is `decimal`.

To refer to a sequence of nested counters of the same name, use:

```
counters(<identifier>,<string>)
counters(<identifier>,<string>,<list-style-type>)
```

The `identifier` must not be `none`, `inherit` or `initial`.

[1] A browser which does not recognise RGBA values is expected to ignore them.

Table 2: Font weight

Weight		**bolder**	lighter
100		400	100
200		400	100
300		400	100
400	`normal`	700	100
500	`medium` (where there is a normal weight font)	700	100
600		900	400
700	`bold`	900	400
800		900	700
900		900	700

6.4 Fonts

Fonts[2] are selected on the basis that they support the required glyphs; a user agent should substitute a different font from the same generic font family or use a font from a different font family if a particular glyph is not supported by the specified font or fonts available within that font family.

The generic font families are:

serif for example, Times New Roman, Bodoni, Garamond, Minion Web, ITC Stone Serif, MS Georgia, Bitstream Cyberbit

sans-serif for example, MS Trebuchet, ITC Avant Garde Gothic, MS Arial, MS Verdana, Univers, Futura, ITC Stone Sans, Gill Sans, Akzidenz Grotesk, Helvetica

cursive for example, Caflisch Script, Adobe Poetica, Sanvito, Ex Ponto, Snell Round- hand, Zapf-Chancery

fantasy for example, Alpha Geometrique, Critter, Cottonwood, FB Reactor, Studz

monospace Courier, MS Courier New, Prestige, Everson Mono

Font weights take the values described in Table 2.

If a font with a weight below `400` is missing, each unassigned lower value in descending order is checked followed by each higher value and the closest reasonable value is assigned.

If the weights `400` or `500` are missing, the weights `500` or `400` are checked and then the preceding rule is followed.

If a weight above `500` is missing, each unassigned higher value in ascending order is checked followed by each lower value and the closest reasonable value is assigned.

Font position is calculated relative to the text baselines (figure 2).

6.5 Frequency

The units of frequency are:

[2] A font is a particular set of glyphs in a particular style with a particular weight; a typeface is a set of related fonts for which the term 'font family' is used in the CSS guidelines.

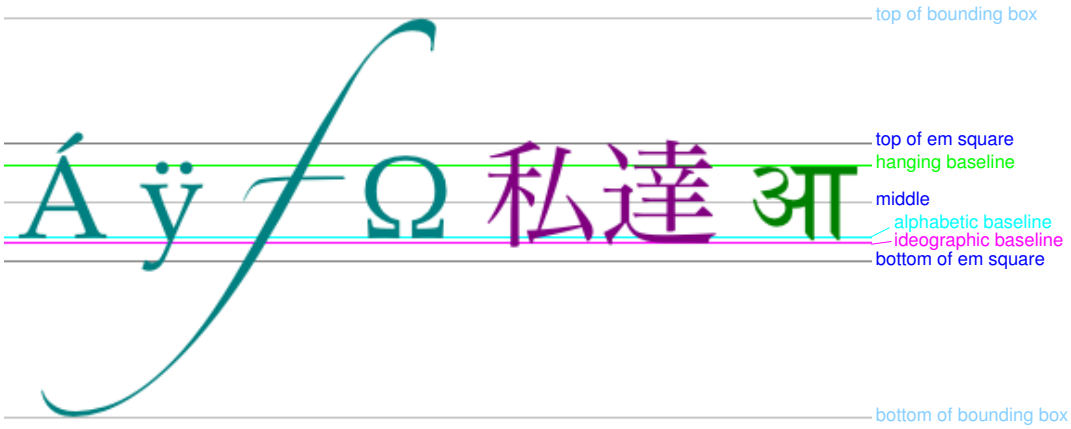

Figure 2: Text baselines

Hz: Hertz

kHz: kilohertz

which may not be negative, and

st: semitone, which may be negative or positive to describe a change of n semitones from a particular frequency.

6.6 Length

Units may be relative or absolute; percentages or relative units are preferred. The relative units of font-size are:

em the em width of the current font

ex the x-height of the current font

Note that an inherited value (see section 10 on page 61) will be absolute, even if computed from a percentage or relative value in the parent document.

The absolute units are:

in inch

cm centimetre

mm millimetre

pt point = 1/72in

pc pica = 12 pt

px pixel = 0.75 pt

In section 9, Properties, where a property takes an absolute value for `<length>`, this value will use one or other of the above units.

6.7 Resolution

The units of resolution are:

dpi dots per CSS inch

dpcm dots per CSS centimetre

6.8 Time

The units of time are:

ms: milliseconds

s: seconds

They may not be negative.

7 The structure of a rule

A CSS rule consists of a SELECTOR containing one or more comma separated HTML elements[3] to which the rule applies followed by any number of declarations; each DECLARATION consists of a property followed by a colon and one or more values terminated by a semi-colon with all the declarations enclosed in braces

SELECTOR	{DECLARATION;	DECLARATION;	...}
`element, element, ...`	`{property: value value ...;`	`property: value value ...;`	`...;}`

8 Selectors

In addition to HTML elements, selectors may contain various CSS elements: attribute and value elements, substring matching elements or pseudo-class elements,[4] optionally ending in a pseudo-element. If any of the elements in a SELECTOR are invalid, the entire rule is ignored.

The following are valid members of a SELECTOR:

***** any element

e element `e`

The specificity of a SELECTOR is calculated by concatenating:

- the number of ID selectors (`e#myid`),
- the number of class (`e.value`), attribute and pseudo-class selectors
- the number of type and pseudo-element selectors.

Attribute and value elements

e[attribute] element `e` when attribute is set

e[attribute="value"] element `e` when the value of attribute exactly equals `"value"`

e[attribute~="value"] element `e` when value `"value"` appears in a list of values for the attribute

e.value element `e` when the `class=" "` attribute is set to `"value"` — equivalent to **e[class~="value"]**. — for example:

e.warning element `e` when the attribute `class="warning"`

e.value1.value2 element `e` when the `class=" "` attribute contains both `"value1"` and `"value2"`

[3] Space separated elements take on a different meaning; see below `e f`.

[4] or HTML elements, not recommended as yet.

Note that `*.value` specifies all elements whose `class=" "` attribute is set to "`value`"

Such an element may also belong to a pseudo-element, e.g. `e.value:visited`.

Substring matching elements

e[attribute^="val"] element `e` the value of whose attribute begins with the string "`val`"

e[attribute$="lue"] element `e` the value of whose attribute ends with the string "`lue`"

e[attribute*="alu"] element `e` the value of whose attribute contains the string "`alu`"

e[attribute|="en"] element `e` the value of whose attribute contains a hyphen separated list of values beginning with the string "`en`"

Pseudo-class elements

e:root element `e` where it is the root of the document

e:nth-child(n) element `e` where it is the `nth` child of its parent

e:nth-last-child(n) element `e` where it is the `nth` child counting from the last child of its parent

e:nth-of-type(n) element `e` where it is the `nth` sibling of its type

e:nth-last-of-type(n) element `e` where it is the `nth` sibling counting from the last sibling of its type

e:first-child element `e` where it is the first child of its parent

e:last-child element `e` where it is the last child of its parent

e:first-of-type element `e` where it is the first sibling of its type

e:last-of-type element `e` where it is the last sibling of its type

e:only-child element `e` where it is the only child of its parent

e:only-of-type element `e` where it is the only sibling of its type

e:empty element `e` where it has no children

e:target element `e` where it is the target of a referring URL

e:lang(la) element `e` if it is in language `la`

e:not(s) element `e` when it does not match simple selector `s`

Pseudo-class selectors in HTML

e:active element `e` where it is a `<link>`, `<a>` or `<area>` element with the `href=" "` attribute, a `<button>` or `<input>` element whose type is in the button state or a `<menuitem>` element that is not `disabled`

e:checked element `e` where it is an `<input>` or `<menuitem>` element whose `type="checkbox"` or "`radio`" or an `<option>` element that has been selected

e:default element `e` where it is the only `<button>` element and/or the only `<input>` element of a button type, an `<input>` element that has been checked or an `<option>` element that has been selected

e:dir(ltr) element `e` whose directionality is `ltr`

e:dir(rtl) element `e` whose directionality is `rtl`

e:disabled element `e` where it is disabled

e:enabled element `e` where it is a `<button>`, `<input>`, `<select>`, `<textarea>` or `<option>` element or an `<optgroup>` or `<menuitem>` element that it is not disabled

e:focus element `e` when it is in focus

(Use the outline property to define any outlining of the four pseudo-class elements above.)

e:hover element `e` when the pointer is hovering over it

e:indeterminate element `e` where it is an `<input>` element whose `type="checkbox"` whose state is indeterminate, an `<input>` element whose `type="radio"` where none of the radio group have been checked or a `<progress>` element without a value

e:in-range element `e` whose value is within the accepted range

e:invalid element `e` where it is any element including a `<form>` or `<fieldset>` element containing elements whose values are invalid

e:link element `e` where it is a `<link>`, `<a>` or `<area>` element with an `href=" "` attribute before the link has been visited

e:optional element `e` where it is an `<input>` , `<select>` or `<textarea>` element whose `required` attribute is not set

e:optional element `e` where it is an `<input>`, `<select>` or `<textarea>` element whose `required` attribute is not set

e:out-of-range element `e` whose value is out of range

e:read-only all elements which do not match the criteria for the `read-write` pseudo-class

e:read-write element `e` which is editable or where it is an `<input>` or `<textarea>` element whose `readonly` attribute is not set

e:required element `e` where it is a required `<input>` element or a `<select>` or `<textarea>` element whose `required` attribute is set

e:valid element `e` where it is any element including a `<form>` or `<fieldset>` element which contains no elements whose values are invalid

e:visited element `e` where it is a `<link>`, `<a>` or `<area>` element with an `href=" "` attribute that has been visited

Pseudo-elements

e::first-line applies a style to the first line of element `e`

e::first-letter applies a style to the first line of element `e`

e::before places content before element `e`; the content inherits any inheritable properties of its element

e::after places content after an element `e`; the content inherits any inheritable properties of its element

Only one of the pseudo elements may be appended to a list of selectors.

The id selector

e#myid element `e` when the attribute `id="myid"`. Note that no two `id=" "` attributes in a document can have the same value.

Combination elements

e f element `f` when it is a descendant of element `e`

e * f element `f` when it is a descendant of any descendent of element `e`

e>f element `f` when it is a child of element `e`

e+f element `f` when it immediately follows a sibling element `e`

e~f element `f` when it follows a sibling element `e`

Combination elements may be combined.

Namespace elements

Elements which have been declared in a namespace (as defined in XML 1.0) may be used as follows:

ns|e elements with name `e` in namespace `ns`

***|e** elements with name `e` in any namespace, including those without a namespace

|e elements with name `e` without a namespace

e if no default namespace has been declared for selectors, this is equivalent to `*|e`. Otherwise it is equivalent to `ns|e` where `ns` is the default namespace.

In the above examples `e` may be `*`.

See section 3.1 for namespace declarations.

9 Properties

9.1 align-content

```
align-content: normal | <baseline-position> | <content-distribution> |
<overflow-position>? <content-position>
```

aligns the contents of the box as a whole within the box itself along the block/column/cross axis of the box where

`<baseline-position>` values may be

baseline which is the same as `first`

first which uses the first box's baseline

last which uses the last box's baseline

`<content-distribution>` values may be

space-between which distributes items evenly along the vertical axis; if there is insufficient space, this is the equivalent of `start`

space-around which distributes items evenly along the vertical axis with a half space at either end; if there is insufficient space, this is the equivalent of `center`

space-evenly which distributes items evenly along the vertical axis with a full space at either end

stretch which gives all the items the same width within any existing max-height and max-width constraints.

`<overflow-position>?` may be

safe where, in the case of an overflow, defaults to `start`

unsafe where the existing alignment is honoured regardless of its impact

In the absence of a value, the UA works out the best way of handling the positioning.

`<self-position>` may be

center which centres the item within its container.

start which aligns the item to be flush with the container's start edge in the vertical axis.

end which aligns the item to be flush with the container's end edge in the vertical axis.

flex-start which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s cross-start side.

flex-end which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s cross-end side.

left which aligns the item to be flush with the alignment container's line-left or physical left edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

right which aligns the item to be flush with the alignment container's line-right or physical right edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

The default value is `stretch`.

9.2 align-items

```
align-items: normal | stretch | <baseline-position> |
[ <overflow-position>? <self-position> ]
```

specifies the default `align-self` for all of the child boxes in a box. See `align-self` for an explanation of the permissible values.

The default value is `stretch`.

9.3 align-self

```
align-self: auto | normal | stretch | <baseline-position> |
<overflow-position>? <self-position>
```

specifies the vertical alignment of items in a container along the block/column/cross axis of the container where

`<baseline-position>` values may be

baseline which is the same as `first`

first which uses the first box's baseline

last which uses the last box's baseline

`<content-distribution>` values may be

space-between which distributes items evenly along the vertical axis; if there is insufficient space, this is the equivalent of `start`

space-around which distributes items evenly along the vertical axis with a half space at either end; if there is insufficient space, this is the equivalent of `center`

space-evenly which distributes items evenly along the vertical axis with a full space at either end

stretch which gives all the items the same width within any existing max-height and max-width constraints.

`<overflow-position>?` may be

safe where, in the case of an overflow, defaults to `start`

unsafe where the existing alignment is honoured regardless of its impact

In the absence of a value, the UA works out the best way of handling the positioning.

`<self-position>` may be

center which centres the item within its container.

start which aligns the item to be flush with the container's start edge in the vertical axis.

end which aligns the item to be flush with the container's end edge in the vertical axis.

self-start which aligns the item to be flush with the edge of the container corresponding to the start side in the vertical axis.

self-end which aligns the item to be flush with the edge of the container corresponding to the end side in the vertical axis.

flex-start which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s cross-start side.

flex-end which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s cross-end side.

left which aligns the item to be flush with the alignment container's line-left or physical left edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

right which aligns the item to be flush with the alignment container's line-right or physical right edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

The default value is `auto`.

9.4 aspect-ratio

```
aspect-ratio: <width>/<height> ;
```

specifies the aspect ratio of a media type.

<width>/<height> two positive integers

It is not inherited and there is no default value.

9.5 azimuth

```
azimuth: <angle> | [[ left-side | far-left | left |
center-left | center | center-right | right | far-right |
right-side ] || behind ] | leftwards | rightwards | inherit
```

specifies the spatial audio properties of an aural[5] element.

The azimuth values are:

<angle> refers to the position of the sound source with 0deg referring to the front, 90deg to the right, 180deg behind and 270deg to the left; negative values represent the equivalent positive value.

left-side 270deg. With `behind`, 270deg

far-left 300deg. With `behind`, 240deg

left 320deg. With `behind`, 220deg

center-left 340deg. With `behind`, 200deg

center 0deg. With `behind`, 180deg

center-right 20deg. With `behind`, 160deg

right 40deg. With `behind`, 140deg.

far-right 60deg. With `behind`, 120deg

right-side 90deg. With `behind`, 90deg

behind 180deg

leftwards subtracts 20 degrees

rightwards adds 20 degrees

The default value is `center`.

9.6 background

```
background: [<background-color> || <background-image> ||
<background-repeat> || <background-attachment> ||
<background-position>] | inherit ;
```

offers a shorthand for setting the background properties.

```
background-color: <color> | transparent | inherit ;
```

specifies the background colour (see Colour on page 9).

Setting `background-color: transparent` ensues that links display correctly. The value of `background-color` only applies to table columns if its value is `transparent` for both table rows and table cells.

It is not inherited by default and its default value is `transparent`.

```
background-image: url("<url>") | none | inherit ;
```

specifies the background image for an element. A `background-color` should also be specified; this will take the place of the image if it is not available and will show through any transparent regions of the image. The dimensions of the image will be calculated with reference to the values of the `background-position` property.

It is not inherited and the default value is `none`.

```
background-repeat: repeat | repeat-x | repeat-y | no-repeat |
inherit ;
```

[5]The aural category will be replaced by the audio and speech categories in due course.

specifies whether and how a `background-image` should be repeated (tiled).

The repeat values may be:

repeat repeat horizontally and vertically

repeat-x repeat horizontally

repeat-y repeat vertically

no-repeat do not repeat

It is not inherited and the default value is `repeat`.

```
background-attachment: scroll | fixed |inherit ;
```

specifies whether a `background-image` should scroll or remain fixed to the border area of the content; n.b. the value `scroll` will make an image appear fixed in a scrolling mechanism.

It is not inherited and the default value is `scroll`.

```
background-position: [ [ <percentage> | <length> | left |
center | right ] [ <percentage> | <length> | top | center |
bottom ]? ] | [ [ left | center | right ] || [ top | center |
bottom ] ] | inherit ;
```

specifies the initial position of a `background-image`; the first value specifies its horizontal and the second its vertical position. In the absence of a second value, this is assumed to be `center`.

The position values are:

<percentage> refers to the width and height of the padding block of the image.

<length> refers to the distance horizontally or vertically from the top left corner of the padding box

left = 0% for the horizontal position

center = 50% for the horizontal or vertical position

right = 100% for the horizontal position

top = 0% for the vertical position

bottom = 100% for the vertical position

It is not inherited and the default value is `0% 0%`.

9.7 border

```
border: <border-width> || <border-style> || <border-color> |
inherit ;
```

offers a shorthand for setting the same values for all four borders.

```
border-top: <border-width> || <border-style> ||
<border-color> | inherit ;
border-right: <border-width> || <border-style> ||
<border-color> | inherit ;
border-bottom: <border-width> || <border-style> ||
<border-color> | inherit ;
border-top: <border-width> || <border-style> ||
<border-color> | inherit ;
```

offer shorthands for setting the values of individual borders.

```
border-width: thin | medium | thick | <length> | inherit ;
```

offers a shorthand for setting the same border width for all four borders.

```
border-top-width: thin | medium | thick | <length> | inherit ;
border-right-width: thin | medium | thick | <length> | inherit ;
border-bottom-width: thin | medium | thick | <length> | inherit ;
border-left-width: thin | medium | thick | <length> | inherit ;
```

specify the widths of individual borders. The default value is `medium`.

```
border-style: none | hidden | dotted | dashed | solid |
double | groove | ridge | inset | outset
```

offers a shorthand for setting the same style for all four borders.

The style values may be:

none sets border-width to zero

hidden same as `none` when used in tables where it takes precedence if the value of `border-collapse` is `collapse`

dotted a sequence of dots

dashed a sequence of dashes

solid a single solid line

double two solid lines on either side of the border

groove appears to be carved in the surface

ridge appears to come up from the surface

inset makes the box appear embedded in the surface, the same as `groove` if the value of `border-collapse` is `collapse`

outset makes the box appear proud of the surface, the same as `ridge` if the value of `border-collapse` is `collapse`

```
border-top-style: none | hidden | dotted | dashed | solid |
double | groove | ridge | inset | outset
border-right-style: none | hidden | dotted | dashed | solid |
double | groove | ridge | inset | outset
border-bottom-style: none | hidden | dotted | dashed | solid |
double | groove | ridge | inset | outset
border-left-style: none | hidden | dotted | dashed | solid |
double | groove | ridge | inset | outset
```

specify the style of individual borders.

Values are not inherited and *there is no default value; so a border style must be set.*

```
border-color:<color> | transparent | inherit ;
```

offers a shorthand for setting the same colour for all four borders.

```
border-top-color:<color> | transparent | inherit ;
border-right-color:<color> | transparent | inherit ;
border-bottom-color:<color> | transparent | inherit ;
border-left-color:<color> | transparent | inherit ;
```

specify the colours of individual borders.

Values are not inherited and the default value is the value of the `color` property.

9.8 border-collapse

```
border-collapse: collapse | separate | inherit ;
```

specifies whether and how much space there is borders between table rows, columns and cells

If the value of `border-collapse` is `collapse`, and the value of `border-style` is `hidden`, these borders take precedence over all other borders; if not, wider borders take precedence over narrower ones.and cell border colours take precedence over row, column and table colours.

The default value is `separate`.

```
border-spacing: <length> <length>? | inherit ;
```

specifies the spacing where the value of `border-collapse` is `separate`; if there are two values, the first applies to the horizontal and the second to the vertical borders.

The default value is `0`.

```
empty-cells: show | hide | inherit ;
```

specifies whether borders or backgrounds are to be drawn for empty cells where the value of `border-collapse` is `separate`

The default value is `show`.

9.9 bottom

See position on page 52

9.10 box-decoration-break

```
box-decoration-break: slice | clone
```

specifies, when a break splits a box,

- whether the box's margins, borders, padding, and other decorations wrap the broken edges of the box fragments
- how the background positioning area is derived from or duplicated across the box fragments and how the element's background is drawn within them.

The values are:

clone each box fragment is independently wrapped with the border, padding, and margin.

slice the element is rendered as if with no breaks present, and then sliced by the breaks afterwards; no border and no padding are inserted at a break; no box-shadow is drawn at a broken edge; backgrounds, border-radius, and the border-image are applied to the geometry of the whole box as if it were unbroken.

9.11 break-after

```
break-after: auto | avoid | avoid-page | page | left | right |
recto | verso | avoid-column | column | avoid-region | region
```

specifies page/column/region break behaviour after the generated box. The forced break values `left`, `right`, `recto`, `verso`, `page`, `column` and `region` create a forced break in the flow while the avoid break values `avoid`, `avoid-page`, `avoid-column` and `avoid-region` indicate that content should be kept together.

Since breaks are only allowed between siblings, not between a box and its container, a `break-after` value on a last-child box is propagated to its container.

Values may be:

auto neither force nor forbid a break after the principal box.

avoid avoid a break after the principal box.

9.11.1 Page Break Values

avoid-page avoid a page break after the principal box.

page always force a page break after the principal box.

left force one or two page breaks after the principal box so that the next page is formatted as a left page.

right force one or two page breaks after the principal box so that the next page is formatted as a right page.

recto force one or two page breaks after the principal box so that the next page is formatted as either a left page or a right page, whichever is second in a page spread.

verso force one or two page breaks after the principal box so that the next page is formatted as either a left page or a right page, whichever is first in a page spread.

9.11.2 Column Break Values

avoid-column avoid a column break after the principal box.

column always force a column break after the principal box.

9.11.3 Region Break Values

avoid-region avoid a region break after the principal box.

region always force a region break after the principal box.

9.12 break-before

```
break-before: auto | avoid | avoid-page | page | left | right |
recto | verso | avoid-column | column | avoid-region | region
```

specifies page/column/region break behaviour before the generated box. The forced break values `left`, `right`, `recto`, `verso`, `page`, `column` and `region` create a forced break in the flow

while the avoid break values `avoid`, `avoid-page`, `avoid-column` and `avoid-region` indicate that content should be kept together.

Since breaks are only allowed between siblings, not between a box and its container, a `break-before` value on a first-child box is propagated to its container.

Values may be:

auto neither force nor forbid a break before the principal box.

avoid avoid a break before the principal box.

9.12.1 Page Break Values

avoid-page avoid a page break before the principal box.

page always force a page break before the principal box.

left force one or two page breaks before the principal box so that the next page is formatted as a left page.

right force one or two page breaks before the principal box so that the next page is formatted as a right page.

recto force one or two page breaks before the principal box so that the next page is formatted as either a left page or a right page, whichever is second in a page spread.

verso force one or two page breaks before the principal box so that the next page is formatted as either a left page or a right page, whichever is first in a page spread.

9.12.2 Column Break Values

avoid-column avoid a column break before the principal box.

column always force a column break before the principal box.

9.12.3 Region Break Values

avoid-region avoid a region break before the principal box.

region always force a region break before the principal box.

9.13 break-inside

```
break-inside: auto | avoid | avoid-page | avoid-column |
avoid-region
```

specifies page/column/region break behaviour within the element's principal box.

Values may be:

auto neither force nor forbid a break within the box.

avoid avoid a break within the box.

avoid-page avoid a page break within the box.

avoid-column avoid a column break within the box.

avoid-region avoid a region break within the box.

9.14 caption-side

```
caption-side: top | bottom | inherit ;
```

specifies the position of the caption in a table element. Use `text-align` to adjust the horizontal position of a caption.

The default value is `top`.

9.15 clear

See float on page 32.

9.16 clip

This property is deprecated.

9.17 clip-path

```
clip-path: <clip-source> | [ <basic-shape> || <geometry-box> ] |
none
```

specifies a basic shape or references a `clipPath` element to create a clipping path.

The value for `<clip-source>` is a URL.

The value for `<basic-shape>` is a basic shape function as defined in the CSS Shapes module The default value is `border-box`.

The values for <geometry-box> may be:

```
<shape-box> | fill-box | stroke-box | view-box
```

where

<shape-box> in the absence of a value for `<basic-shape>`, use the edges of the specified box, including any corner shaping as the clipping path.

fill-box use the object bounding box.

stroke-box use the stroke bounding box.

view-box use the nearest SVG viewport.

none do not create a clipping path.

9.18 clip-rule

```
clip-rule: nonzero | evenodd
```

indicates the algorithm which is to be used to determine whether a given point is inside a shape for a clipping region.

The values are:

nonzero determines the "insideness" of a point on the canvas by drawing a ray from that point to infinity in any direction and then examining the places where a segment of the shape crosses the ray. Starting with a count of zero, add one each time a path segment crosses the ray from left to right and subtract one each time a path segment crosses the ray from

right to left. After counting the crossings, if the result is zero then the point is outside the path. Otherwise, it is inside.

evenodd determines the "insideness" of a point on the canvas by drawing a ray from that point to infinity in any direction and counting the number of path segments from the given shape that the ray crosses. If this number is odd, the point is inside; if even, the point is outside.

9.19 color

```
color: <color> | <integer> |transparent | inherit ;
```

specifies the foreground colour for all except media types (see Colour on page 9). or the number of bits per colour of a media type

<integer> bits per colour (only for use with media types (see section 4 on page 7)

transparent = `rgba(0,0,0,0)`

There is no default value other than for media types where the user agent determines it.

```
color: rgba(r%, g%, b%, <alphavalue>);
color: rgba([0-255], [0-255], [0-255], <alphavalue>);
color: hsl(h, s%, l%, <alphavalue>);
```

offers a shorthand for specifying the opacity of a colour of an element other than of an SVG element. The full form:

```
opacity: <alphavalue> | inherit
```

specifies the opacity of a colour of an element other than of an SVG element.

<alphavalue> a number in the range `0.0` (transparent) to `1.0` (opaque).

The default value is `1.0`.

An element whose opacity is less than `1.0` is rendered in a new stacking context at `z-index: 0;`

9.20 color-index

```
color-index: <integer> ;
```

specifies the number of entries in the colour lookup table of a media type (see section 4 on page 7).

<integer> entries in the colour lookup table

It is not inherited and the default value is determined by the user agent.

9.21 column-gap

```
column-gap: normal | <length-percentage>
```

specifies the gutters between columns.

9.22 content

```
content: normal | none | [ <string> | url("<url>") | <counter> |
attr(<identifier>) | open-quote | close-quote | no-open-quote |
no-close-quote ]+ | inherit normal
```

specifies the content of a `:before` or `:after` pseudo-element.

The content values may be:

none no content

normal the same as `none` for the `:before` or `:after` pseudo-elements.

<string> text content

<counter> see Counters on page 10

attr(<identifier>) returns the value of the attribute of the selector which matches `<identifier>`

open-quote introduces an opening quote

close-quote introduces a closing quote

no-open-quote increments the nesting level for quotes

no-close-quote decrements the nesting level for quotes

The style of quotation mark is specified by the quotes property; see page 52.

It is not inherited and the default value is `normal`.

9.23 counter-increment

```
counter-increment: [<identifier> <integer>?]+ | none |
inherit ;
```

specifies the counter (section 6.3 on page 10) to be increment and, optionally, the amount by which it is to be incremented.

It is not inherited and the default value is `none`.

9.24 counter-reset

```
counter-reset: [<identifier> <integer>?]+ | none | inherit ;
```

specifies the counter (section 6.3 on page 10) to be set and optionally, the amount by which it is to be incremented.

It is not inherited and the default value is `none`.

9.25 cue

```
cue: [ <cue-before> || <cue-after> ] ;
```

offers a shorthand for setting cue-before and cue-after. If one value is given, it applies to both properties.

9.26 cue-after

```
cue-after: <uri> <decibel>? | none ;
```

specifies whether and how a cue is to be given after an element.

The values are:

<uri> designates an auditory icon resource. Use an alternative cue, such as a bell sound, if it is not available.

none no auditory icon is used.

<decibel> is a positive or negative number followed by `dB` indicating a change from the computed value of the voice-volume property within the selected element (as a result, the volume level of an audio cue changes when the voice-volume property changes). When omitted, or where the value of voice-volume is `silent`, the value is 0dB. Note that −6.0`dB` to +6.0`dB` normally covers the entire audio range.

The default in `none`.

9.27 cue-before

```
cue-before: <uri> <decibel>? | none ;
```

specifies whether and how a cue is to be given before an element.

The values are the same as for `cue-after`.

The default in `none`.

9.28 cursor

```
cursor: [ [url("<url>"),]* [ auto | crosshair | default | pointer |
move | e-resize | ne-resize | nw-resize | n-resize |
se-resize | sw-resize | s-resize | w-resize | text | wait |
help | progress ] ] | inherit ;
```

specifies the appearance of any pointer.

The cursor values are

url("<url>") if there is a list of URLs, the first readable image is selected

auto determined by the user agent

crosshair a simple crosshair

default platform dependant, normally an arrow

pointer a pointer to a link

move a pointer to something to be moved

e-resize etc. indicates that an edge of the cursor is to be moved relative to the box from the direction specified

text a pointer to text to be selected, often a `I`

wait a pointer indicating that the computer is busy, often a watch or an hourglass

help a pointer indicating that help is available, often a `?`

progress a pointer indicating the progress of an operation

The default value is `auto`.

9.29 device-aspect-ratio

```
device-aspect-ratio: <width>/<height> ;
```

specifies the aspect ratio of the rendering surface of a media type (see section 4 on page 7).

<width>/<height> two positive integers

It is not inherited and there is no default value.

9.30 device-height

```
device-height: <length> | <percentage> | auto | inherit ;
```

specifies the height of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the height of the containing block; for a block whose position is `absolute` this is defined by the box's padding edge (figure 1 on page 8).

auto depends on context

It is not inherited and the default value is `auto`.

9.31 device-width

```
device-width: <length> | <percentage> | auto | inherit ;
```

specifies the width of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the containing block; for a block whose position is `absolute` this is defined by the box's padding edge (figure 1 on page 8).

auto depends on context

It is not inherited and the default value is `auto`.

9.32 direction

```
direction: ltr | rtl | inherit ;
```

specifies the direction of text.

The default value is `ltr`.

9.33 display

```
display: block | inline-block | flex | inline-flex |
grid | inline-grid | inline | list-item |  table |
inline-table | table-row-group | table-header-group |
table-footer-group | table-row | table-column-group |
table-column | table-cell | table-caption | none | inherit ;
```

specifies the type of container within which to display content.

The values are:

block generate a block-level block container.

inline-block generate an inline-level block container.

flex generate a block-level flex container box.

inline-flex generate an inline-level flex container box.

grid generate a block-level grid container box.

inline-grid generate an inline-level grid container box.

The other display values applicable to non-table elements are:

inline generate one or more inline boxes.

list-item generate a principal block box and a marker box.

none causes an element and its descendants not to appear

It is not inherited and the default value is `inline`.

The table related display values.cause an element defined in a non-HTML environment to behave like the relevant HTML table element.

table display like a block level HTML: `table` element

inline-table display like an inline HTML: `table` element

table-row display like an HTML `tr`: element

table-row-group display like an HTML: `tbody` element

table-header-group display like an HTML:`thead` element

table-footer-group display like an HTML: `tfoot` element

table-column display like an HTML:`col` element

table-column-group display like an HTML:`colgroup` element

table-cell display like an HTML: `td` or `th` element

table-caption display like an HTML: `caption` element

9.34 elevation

```
elevation: <angle> | below | level | above | higher | lower |
inherit ;
```

specifies the elevation properties of an aural[6] element.

The elevation values are:

<angle> refers to the position of the sound source with 0deg being level, 90deg directly above and -90deg directly below.

below -90deg

level 0deg

above 90deg

higher increases the angle by 10

lower deceases the angle by 10

The default value is `level`.

9.35 empty-cells

See border-collapse on page 22.

[6]The aural category will be replaced by the audio and speech categories in due course.

9.36 flex

```
flex : none | [ <flex-grow> <flex-shrink>? || <flex-basis> ]
```

offers a shorthand for flex-flow or flex-shrink and flex-basis.

auto specifies the values 1 1 `auto`.

initial specifies the values 0 1 `auto`.

none specifies the values 0 0 `auto`.

9.37 flex-basis

```
flex-basis: content | <width>
```

sets the flex basis.

The values are:

auto

content based on the `flex-item`'s content

<width> any value permitted for the width property.

The default value is `auto`.

9.38 flex-direction

```
flex-direction: row | row-reverse | column | column-reverse
```

specifies how flex items are placed in the flex container, by setting the direction of the flex container's main axis.

The values are:

row the flex container's main axis has the same orientation as the inline axis of the current writing mode. The `main-start` and `main-end` directions are equivalent to the `inline-start` and `inline-end` directions.

row-reverse same as `row`, except the `main-start` and `main-end` directions are swapped.

column the flex container's main axis has the same orientation as the block axis of the current writing mode. The `main-start` and `main-end` directions are equivalent to the `block-start` and `block-end` directions.

column-reverse same as column, except the `main-start` and `main-end` directions are swapped.

The default value is `row`.

9.39 flex-flow

```
flex-flow: flex-direction || flex-wrap
```

offers a shorthand for flex-direction and flex-wrap.

9.40 flex-grow

```
flex-grow: <number>
```

specifies the proportion by which a flex item will grow.

The default value is 0

9.41 flex-shrink

```
flex-shrink: <number>
```

specifies the proportion by which a flex item will shrink.

The default value is 1

9.42 flex-wrap

```
flex-wrap: nowrap | wrap | wrap-reverse
```

controls whether the flex container is single-line or multi-line, and the direction of the cross-axis, which determines the direction new lines are stacked in.

The values are:

nowrap the flex container is single-line.

wrap the flex container is multi-line.

wrap-reverse same as `wrap`.

The default value is `nowrap`.

9.43 float

```
float: left | right | none | inherit ;
```

specifies the position of a float.

It is not inherited and the default value is `none`.

```
clear: left | right | both | none
```

specifies whether the area(s) to the left and right of the float should remain clear of content.

It is not inherited and the default value is `none`.

9.44 font

```
font: [ [ <font-style> || <font-variant> || <font-weight> ]?
<font-size>[/<line-height>]? <font-family> ] | caption | icon |
menu | message-box | small-caption | status-bar | inherit ;
```

offers a shorthand for specifying font properties. For line-height, see page 39,

The font property values are:

caption the font used for captioned controls

icon the font used for icons

menu the font used for menus

message-box the font used in dialogue boxes

small-caption the font used for labelling small controls

status-bar the font used in window status bars

```
font-style: normal | italic | oblique | inherit ;
```

specifies which font style from a particular typeface should be used.

The font style values are:

normal upright

italic italic or cursive

oblique slanted, strictly speaking a slanted upright font but often regarded as the same as *italic*

The default value is `normal`.

```
font-variant: normal | small-caps | inherit ;
```

specifies whether the small caps font is to be used. If the value `small-caps` is specified but no font is available, upper case glyphs will normally be used.

The default value is `normal`.

```
font-weight: normal | bold | bolder | lighter | 100 | 200 |
300 | 400 | 500 | 600 | 700 | 800 | 900 | inherit ;
```

specifies the font weight to be used (see Fonts on page 11).

The default value is `normal`.

```
font-size: <absolute-size> | <relative-size> | <length> |
<percentage> inherit ;
```

specifies the size of the font to be used.

The font-size values are:

absolute-size `[ xx-small | x-small | small | medium | large | x-large | xx-large ] ;` n.b. `x-small` is not mapped to the HTML sizes 1–6.

relative-size `[larger | smaller]`; that is, increase/decrease the HTML size

length relative to the computed font size of the parent element.

<percentage> refers to the computed font size of the parent element.

The default value is `medium`.

```
font-family: [[ <family-name> | <generic-family> ]
[, <family-name>| <generic-family>]* ] | inherit ;
```

specifies a comma separated list of font families to be used.

The font family values are

family-name for example "Times Roman", Helvetica

generic-family `serif | sans-serif | cursive | fantasy | monospace`

The default value depends on the user agent.

See also Fonts on page 11.

9.45 gap

```
gap: <row-gap> <column-gap>?
```

offers a shorthand for row-gap and column-gap.

9.46 grid

```
grid: <grid-template> | <grid-template-rows> /
[ auto-flow && dense? ] <grid-auto-columns>? |
[ auto-flow && dense? ] <grid-auto-rows>? / <grid-template-columns>
```

offers a shorthand for setting the explicit grid properties (grid-template-rows, grid-template-columns, and grid-template-areas) and the implicit grid properties (grid-auto-rows, grid-auto-columns, and grid-auto-flow) in a single declaration.

9.47 grid-area

```
grid-area: <grid-line> [ / <grid-line> ]{0,3}
```

If four `<grid-line>` values are specified, `grid-row-start` is set to the first value, `grid-column-start` is set to the second value, `grid-row-end` is set to the third value, and `grid-column-end` is set to the fourth value.

When `grid-column-end` is omitted, if `grid-column-start` is a `<custom-ident>`, `grid-column-end` is set to that `<custom-ident>`; otherwise, it is set to `auto`.

When `grid-row-end` is omitted, if `grid-row-start` is a `<custom-ident>`, `grid-row-end` is set to that `<custom-ident>`; otherwise, it is set to `auto`.

When `grid-column-start` is omitted, if `grid-row-start` is a `<custom-ident>`, all four longhands are set to that value. Otherwise, it is set to `auto`.

9.48 grid-auto-columns

```
grid-auto-columns: <track-size>+
```

specifies the size of implicitly-created tracks where grid-template-columns does not create them explicitly.

9.49 grid-auto-flow

```
grid-auto-flow: [ row | column ] || dense
```

specifying exactly how auto-placed items get flowed into the grid.

The values are:

row fill each row in turn, adding new rows as necessary.

column fill each column in turn, adding new columns as necessary.

dense use the dense packing algorithm, which fills in holes earlier in the grid if smaller items come up later. If omitted, a sparse algorithm is used, where placement is forward, never backtracking to fill holes.

The default value is `row`.

9.50 grid-auto-rows

```
grid-auto-rows: <track-size>+
```

specifies the size of implicitly-created tracks where grid-template-rows does not create them explicitly.

9.51 grid-column

```
grid-column: <grid-line> [ / <grid-line> ]?
```

offers a shorthand for grid-column-start/grid-column-end.

9.52 grid-column-end

```
grid-column-end: auto |   <custom-ident> |
[ <integer> && <custom-ident>? ] |   [ span && [ <integer>
|| <custom-ident> ] ]
```

contributes to determining a grid item's size and location within the grid.

The values are:

auto

<custom-ident> attempt to match the grid area's edge to a named grid area.

<integer> && <custom-ident>? contributes the nth grid line to the grid item's placement.

span && [<integer> || <custom-ident>] contributes a grid span to the grid item's placement such that the corresponding edge of the grid item's grid area is n lines from its opposite edge in the corresponding direction.

9.53 grid-column-start

```
grid-column-start: auto |   <custom-ident> |
[ <integer> && <custom-ident>? ] |   [ span && [ <integer>
|| <custom-ident> ] ]
```

contributes to determining a grid item's size and location within the grid.

The values are as for `grid-column-end`.

9.54 grid-row

```
grid-row: <grid-line> [ / <grid-line> ]?
```

offers a shorthand for grid-row-start/grid-row-end.

9.55 grid-row-end

```
grid-row-end: auto |   <custom-ident> |
[ <integer> && <custom-ident>? ] |   [ span && [ <integer>
|| <custom-ident> ] ]
```

contributes to determining a grid item's size and location within the grid.

The values are as for `grid-column-end`.

9.56 grid-row-start

```
grid-row-start: auto |   <custom-ident> |
[ <integer> && <custom-ident>? ] |   [ span && [ <integer>
|| <custom-ident> ] ]
```

contributes to determining a grid item's size and location within the grid.

The values are as for `grid-column-end`.

9.57 grid-template

```
grid-template: none | [ <'grid-template-rows'>
/ <'grid-template-columns'> ] | [ <line-names>? <string> <track-size>?
<line-names>? ]+ [ / <explicit-track-list> ]?
```

offers a shorthand for setting grid-template-columns, grid-template-rows, and grid-template-areas in a single declaration.

none sets all three properties to their initial values (`none`).

9.58 grid-template-areas

```
grid-template-areas: none | <string>+
```

specifies named grid areas, which are not associated with any particular grid item, but can be referenced from the grid-placement properties.

The values are:

none no named grid areas, and likewise no explicit grid tracks, are defined.

<string>+ a row is created for every separate string listed.

The default value is `none`.

9.59 grid-template-columns

```
grid-template-columns: none | <track-list> | <auto-track-list>
```

specifies, as a space-separated track list for the grid's columns, the line names and track sizing functions of the grid.

The values are:

none no explicit grid tracks are created; any rows/columns will be implicitly generated, and their size will be determined by the grid-auto-rows and grid-auto-columns properties.

<track-list> | <auto-track-list> specifies the track list as a series of track sizing functions and line names. Each track sizing function can be specified as a length, a percentage of the grid container's size, a measurement of the contents occupying the column or row, a range using the `minmax()` notation or a fraction of the free space in the grid.

9.60 grid-template-rows

```
grid-template-rows: none | <track-list> | <auto-track-list>
```

specifies, as a space-separated track list for the grid's rows, the line names and track sizing functions of the grid.

The values are the same as `grid-template-columns`.

9.61 height

```
height: <length> | <percentage> | auto | inherit ;
```

specifies the height of a block level containing box or media type.or the minimum height of a table row.

<percentage> refers to the height of the containing block; for a block whose position is `absolute` this is defined by the box's padding edge (Figure 1 on page 8).

auto depends on context

It is not inherited and the default value is `auto`.

9.62 justify-content

```
justify-content: normal | <content-distribution> |
<overflow-position>? [ <content-position> | left | right ]
```

aligns the contents of the box as a whole) within the box itself along the inline/row/main axis of the box where

`<baseline-position>` values may be

baseline which is the same as `first`

first which uses the first box's baseline

last which uses the last box's baseline

`<content-distribution>` values may be

space-between which distributes items evenly along the horizontal axis; if there is insufficient space, this is the equivalent of `start`

space-around which distributes items evenly along the horizontal axis with a half space at either end; if there is insufficient space, this is the equivalent of `center`

space-evenly which distributes items evenly along the horizontal axis with a full space at either end

stretch which gives all the items the same width within any existing max-height and max-width constraints.

`<overflow-position>?` may be

safe where, in the case of an overflow, defaults to `start`

unsafe where the existing alignment is honoured regardless of its impact

In the absence of a value, the UA works out the best way of handling the positioning.

`<self-position>` may be

center which centres the item within its container.

start which aligns the item to be flush with the container's start edge in the horizontal axis.

end which aligns the item to be flush with the container's end edge in the horizontal axis.

flex-start which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s main-start side.

flex-end which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s main-end side.

left which aligns the item to be flush with the alignment container's line-left or physical left edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

right which aligns the item to be flush with the alignment container's line-right or physical right edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

The default value is `flex-start`.

9.63 justify-items

```
justify-items: normal | stretch | <baseline-position> |
[ <overflow-position>? <self-position> ] | [ legacy || [ left | right |
center ] ]
```

specifies the default `justify-self` for all of the child boxes in a box where

`legacy` causes the value to effectively inherit into descendants. It effectively replicates the behaviour of the obsolete HTML `<center>` element.

See `justify-self` for an explanation of the other permissible values.

9.64 justify-self

```
justify-self: auto | normal | stretch | <baseline-position> |
<overflow-position>? [ <self-position> | left | right ]
```

justifies a box within its containing block along the horizontal axis of the container where

`<baseline-position>` values may be

baseline which is the same as `first`

first which uses the first box's baseline

last which uses the last box's baseline

`<content-distribution>` values may be

space-between which distributes items evenly along the horizontal axis; if there is insufficient space, this is the equivalent of `start`

space-around which distributes items evenly along the horizontal axis with a half space at either end; if there is insufficient space, this is the equivalent of `center`

space-evenly which distributes items evenly along the horizontal axis with a full space at either end

stretch which gives all the items the same width within any existing max-height and max-width constraints.

`<overflow-position>?` may be

safe where, in the case of an overflow, defaults to `start`

unsafe where the existing alignment is honoured regardless of its impact

In the absence of a value, the UA works out the best way of handling the positioning.

`<self-position>` may be

center which centres the item within its container.

start which aligns the item to be flush with the container's start edge in the horizontal axis.

end which aligns the item to be flush with the container's end edge in the horizontal axis.

self-start which aligns the item to be flush with the edge of the container corresponding to the start side in the horizontal axis.

self-end which aligns the item to be flush with the edge of the container corresponding to the end side in the horizontal axis.

flex-start which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s main-start side.

flex-end which aligns the `flex-item` to be flush with the edge of the `flex-container` corresponding to the `flex-container`'s main-end side.

left which aligns the item to be flush with the alignment container's line-left or physical left edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

right which aligns the item to be flush with the alignment container's line-right or physical right edge, whichever is in the appropriate axis. If the property's axis is not parallel with either left↔right axis, this value behaves as `start`.

9.65 left

See position on page 52.

9.66 letter-spacing

```
letter-spacing: normal | <length> | inherit ;
```

specifies the inter-character spacing.

<length> refers to additional inter-character spacing and its value may be negative

The default value is `normal`.

9.67 line-height

```
line-height: normal | <number> | <length> | <percentage> |
inherit ;
```

specifies the line height of an inline box.

normal is a reasonable value based on the font size.

<number> is a positive factor applied to the font size

<percentage> refers to the font size.

The default value is `normal`.

9.68 list-style

```
list-style: <list-style-type | symbols(system? <symbol list>) ||
list-style-position || list-style-image ;
```

offers a shorthand for setting list-style-type, list-style-position and list-style-image for elements which have the property value `display: list-item`.

9.69 list-style-image

```
list-style-image: uri | none | inherit ;
```

specifies the image to be used as the marker

The default value in `none`.

9.70 list-style-position

```
list-style-position: inside | outside | inherit ;
```

specifies whether the list marker is placed outside the left edge of the element or in an inline box as the first item of the element's content.

The default value is `outside`.

9.71 list-style-type

```
list-style-type: disc | circle | square | decimal |
decimal-leading-zero | lower-roman | upper-roman |
lower-greek | lower-latin | upper-latin | armenian |
georgian | lower-alpha | upper-alpha | none | inherit ;
```

specifies the style of the marker where

- the value of the property `list-style-image` is `none` or
- it is `<url>` but no image is available to that URL.

decimal displays decimal numbers beginning with 1

decimal-leading-zero displays decimal numbers beginning with 01 to 09 and then 10, 11 ...

The default value is `disc`.

```
list-style: symbols(system ? <symbol list>) ;
```

optionally specifies a system defined by an `@counter-style` rule (section 3.1).

9.72 margin

```
margin: <length> | <percentage> | auto | inherit ;
```

offers a shorthand for specifying the widths of all the margins of non-table elements.

The margin values are:

<percentage> refers to the width of the containing block (not to the height in the case of `margin-top` and `margin-bottom`)

auto is 0

Up to four values may be specified representing `margin-top`, `margin-right`, `margin-bottom` and `margin-left`. Values may be negative. Omitted values are co-opted as follows:

- 2nd from 1st
- 3rd from 1st
- 4th from 2nd (i.e. from 1st if no 2nd)

Additionally the properties:

```
margin-top: <length> | <percentage> | auto | inherit ;
margin-right: <length> | <percentage> | auto | inherit ;
margin-bottom: <length> | <percentage> | auto | inherit ;
margin-left: <length> | <percentage> | auto | inherit ;
```

may be set separately.

Values are not inherited and the default value is `0`.

Vertically adjoining margins of block content collapse to the greater of the two margin widths (or the positive minus the negative margin width) unless they are the bottom margin.or the top margin of a root element.or the margins of an absolutely positioned box.

Use `margin-right: auto;` and `margin-left: auto;` to centre an element with a defined width in its containing block; Note that, if the element's width is undefined, `margin-right: auto;` and `margin-left: auto` default to `0`.

9.73 mask

```
mask: <mask-reference> || <position> [ / <bg-size> ]? ||
<repeat-style> || <geometry-box> || [ <geometry-box> | no-clip ] ||
<compositing-operator> || <masking-mode>
```

offers a shorthand for setting the mask properties. It also resents `mask-border` to its initial setting.

9.74 mask-border

```
mask-border: <mask-border-source> || <mask-border-slice>
[ / <mask-border-width>? [ / <mask-border-outset> ]? ]? ||
<mask-border-repeat> || <mask-border-mode>
```

offers a shorthand for setting the `mask-border` properties.

9.75 mask-border-mode

```
mask-border-mode: luminance | alpha
```

indicates whether the `<image>` value for `mask-border-source` is treated as luminance mask or alpha mask.

alpha alpha values of the mask border image should be used as the mask values.

luminance luminance values of the mask border image should be used as the mask values.

9.76 mask-border-outset

```
mask-border-outset: [ <length> | <number> ]{1,4}
```

specify the amount by which the mask border image area extends beyond the border box. If it has four values, they set the outsets on the top, right, bottom and left sides in that order. If the left is missing, it is the same as the right; if the bottom is missing, it is the same as the top; if the right is missing, it is the same as the top.

9.77 mask-border-repeat

mask-border-repeat: [stretch | repeat | round | space]{1,2}

specifies how the images for the sides and the middle part of the mask border image are scaled and tiled. The first keyword applies to the horizontal sides, the second to the vertical ones. If the second keyword is absent, it is assumed to be the same as the first.

The values are as for `border-image-repeat`.

9.78 mask-border-slice

```
mask-border-slice: <number-percentage>{1,4} fill?
```

specifies inward offsets from the top, right, bottom, and left edges of the `mask-border-image`, dividing it into nine regions: four corners, four edges and a middle. The middle image part is discarded and treated as fully opaque white (the content covered by the middle part is not masked and shines through) unless the `fill` keyword is present.

9.79 mask-border-source

```
mask-border-source: none | <image>
```

specifies an image to be used as mask border image.

9.80 mask-border-width

```
mask-border-width: [ <length-percentage> | <number> | auto ]{1,4}
```

specifies the border width of the mask border image area.

The values are as for `border-width`.

9.81 mask-clip

```
mask-clip : [ <geometry-box> | no-clip ]#
```

determines the area that is affected by the mask.

<geometry-box> may have the value:

content-box the content is restricted to the content box.

padding-box the content is restricted to the padding box.

border-box the content is restricted to the border box.

margin-box the content is restricted to the margin box.

fill-box the content is restricted to the object bounding box.

stroke-box the content is restricted to the stroke bounding box.

view-box uses the nearest SVG viewport as reference box positioned at the origin of the coordinate system established by the `viewBox` attribute. The dimension of the reference box is set to the width and height values of the `viewBox` attribute.

no-clip the content is not restricted.

9.82 mask-composite

```
mask-composite: add | subtract | intersect | exclude
```

specifies the Porter-Duff compositing operator to be used.

The values are:

add the source is placed over the destination.

subtract the source is placed where it falls outside of the destination.

intersect the parts of source that overlap the destination replace the destination.

exclude the non-overlapping regions of source and destination are combined.

9.83 mask-image

```
mask-image: none | <image> | <mask-source>
```

sets the mask layer image of an element where

mask-source is a URL

none counts as a transparent black image layer.

9.84 mask-mode

```
mask-mode: alpha | luminance | match-source
```

indicates whether to use a luminance mask or alpha mask.

The values are:

alpha use the alpha values of the mask layer image as the mask values.

luminance use the luminance values of the mask layer image as the mask values.

match-source if the value of the `mask-image` property is `mask-source`, use the luminance values of the mask layer image as the mask values.

If the value of the `mask-image` property is `image`, use the alpha values of the mask layer image as the mask values.

9.85 mask-origin

```
mask-origin: <geometry-box>#
```

specifies the mask positioning area for elements rendered as a single box; for elements rendered as multiple boxes specifies which boxes box-decoration-break operates on to determine the mask positioning area.

<geometry-box> may have the value:

content-box use the position relative to the content box.

padding-box use the position relative to the padding box. (For single boxes 00 is the upper left corner of the padding edge, 100% 100% is the lower right corner.)

border-box use the position relative to the border box.

margin-box use the position is relative to the margin box.

fill-box use the position relative to the object bounding box.

stroke-box use the position relative to the stroke bounding box.

view-box use the nearest SVG viewport as reference box positioned at the origin of the coordinate system established by the `viewBox` attribute. The dimension of the reference box is set to the width and height values of the `viewBox` attribute.

For SVG elements without an associated CSS layout box, the values `content-box`, `padding-box`, BORDER-BOX and `margin-box` compute to `fill-box`.

For elements with an associated CSS layout box, the values `fill-box`, `stroke-box` and `view-box` compute to the initial value of `mask-origin`.

9.86 mask-position

```
mask-position: <position>
```

specifies how mask layer images are positioned.

The values are as for `background-position`.

9.87 mask-repeat

```
mark-repeat: repeat-style
```

specifies how mask layer images are tiled after they have been sized and positioned.

The values are as for `background-repeat`

9.88 mask-size

```
mask-size: <background-size>#
```

specifies how mask layer images are sized.

The values are as for `background-size`.

9.89 mask-type

```
mask-type: luminance | alpha
```

defines whether the content of the mask element is treated as as luminance mask or alpha mask.

The values are:

alpha alpha values of the mask should be used.

luminance luminance values of the mask should be used.

9.90 max-aspect-ratio

```
max-aspect-ratio: <width>/<height> ;
```

specifies the maximum aspect ratio of a media type (see section 4 on page 7).

<width>/<height> two positive integers

It is not inherited and there is no default value.aspect-ratio

9.91 max-color

```
max-color: <integer> ;
```

specifies the maximum the number of bits per colour of a media type (see section 4 on page 7).

<integer> bits per colour

There is no default value.

9.92 max-color-index

```
max-color-index: <integer> ;
```

specifies the maximum number of entries in the colour lookup table of a media type (see section 4 on page 7).

<integer> entries in the colour lookup table

It is not inherited and there is no default value.

9.93 max-device-aspect-ratio

```
max-device-aspect-ratio: <width>/<height> ;
```

specifies the maximum aspect ratio of the rendering surface of a media type (see section 4 on page 7).

<width>/<height> two positive integers

It is not inherited and there is no default value.

9.94 max-device-height

```
max-device-height: <length> | <percentage> | none | inherit ;
```

specifies the maximum height of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the height of the containing block

none means there is no limit to the height of the rendering surface

It is not inherited and the default value is `none`.

9.95 max-device-width

```
max-device-width: <length> | <percentage> | none | inherit ;
```

specifies the maximum width of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the width of the containing block

none means there is no limit to the width of the rendering surface

It is not inherited and the default value is `none`.

9.96 max-height

```
max-height: <length> | <percentage> | none | inherit ;
```

specifies the maximum height of an element or media type (see section 4 on page 7).

<percentage> refers to the height of the containing block

none means there is no limit to the height of the element

It is not inherited and the default value is `none`.

9.97 max-monochome

```
max-monochrome: <integer> ;
```

specifies the maximum number of bits per pixel of a monochrome media type (see section 4 on page 7).

<integer> bits per pixel

There is no default value.

9.98 max-resolution

```
max-resolution: <resolution> ;
```

specifies the maximum resolution of the most dense dimension of a media type (see section 4 on page 7)

<resolution> an integer and the units used (see section 6.7 on page 12)

It is not inherited and there is no default value.

9.99 max-width

```
max-width: <length> | <percentage> | none | inherit ;
```

specifies the maximum width of an element or media type (see section 4 on page 7).

<percentage> refers to the width of the containing block

none means there is no limit to the width of the element

It is not inherited and the default value is `none`.

9.100 min-aspect-ratio

```
min-aspect-ratio: <width>/<height> ;
```

specifies the minimum aspect ratio of a media type (see section 4 on page 7).

<width>/<height> two positive integers

It is not inherited and there is no default value.

9.101 min-color

```
min-color: <integer> ;
```

specifies the minimum number of bits per colour of a media type (see section 4 on page 7).

<integer> bits per colour

There is no default value.

9.102 min-color-index

```
min-color-index: <integer> ;
```

specifies the minimum number of entries in the colour lookup table of a media type (see section 4 on page 7).

<integer> entries in the colour lookup table

It is not inherited and there is no default value.

9.103 min-device-aspect-ratio

```
min-device-aspect-ratio: <width>/<height> ;
```

specifies the minimum aspect ratio of the rendering surface of a media type (see section 4 on page 7).

<width>/<height> two positive integers

It is not inherited and there is no default value.

9.104 min-device-height

```
min-device-height: <length> | <percentage> | inherit ;
```

specifies the minimum height of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the height of the containing block

It is not inherited and the default value is `0`

9.105 min-device-width

```
min-device-width: <length> | <percentage> | inherit ;
```

specifies the minimum width of the rendering surface of a media type (see section 4 on page 7).

<percentage> refers to the width of the rendering surface of a media type

It is not inherited and the default value is `0`

9.106 min-resolution

```
min-resolution: <resolution> ;
```

specifies the resolution of the least dense dimension of a media type (see section 4 on page 7)

<resolution> an integer and the units used (see section 6.7 on page 12)

It is not inherited and there is no default value.

9.107 min-monochrome

```
min-monochrome: <integer> ;
```

specifies the minimum number of bits per pixel of a monochrome media type (see section 4 on page 7).

<integer> bits per pixel

There is no default value.

9.108 min-height

```
min-height: <length> | <percentage> | inherit ;
```

specifies the minimum height of an element or media type (see section 4 on page 7).
<percentage> refers to the height of the containing block
It is not inherited and the default value is `0`

9.109 min-width

```
min-width: <length> | <percentage> | inherit ;
```

specifies the minimum width of an element or media type (see section 4 on page 7).
<percentage> refers to the width of the containing block
It is not inherited and the default value is `0`

9.110 monochome

```
monochrome: <integer> ;
```

specifies the number of bits per pixel of a monochrome media type (see section 4 on page 7).
<integer> the number of bits per pixel
The default value is determined by the use agent.

9.111 opacity

See `color` on page 26.

9.112 order

```
order: <integer>
```

specifies the order in which a flex item is displayed where this is not the order in which it appears. −1 puts the item before any others.
The default value is 0.

9.113 orientation

```
orientation: portrait | landscape ;
```

specifies the orientation of a media type (see section 4 on page 7).
portrait the value of the `height` of the media is greater than the value of its `width`
landscape the value of the `width` of the media is greater than the value of its `height`
It is not inherited and there is no default value.

9.114 orphans

```
orphans: <integer> | inherit ;
```

specifies the minimum number of lines to be left at the bottom of a page.
The default value is 2.

9.115 outline

```
outline: [ <outline-color> || <outline-style> ||
<outline-width> ] | inherit ;
```

offers a shorthand for setting the outline properties of an element, Outlines differ from borders in that they can be drawn round any element, they are the same all round the element and they take up no space.

```
outline-color: <color> | invert | inherit ;
```

specifies the `outline-color` if an `outline-style` is defined (see Colour on page 9).

It is not inherited and the default value is `invert`.

```
outline-style: <border-style> | inherit ;
```

specifies the `outline-style` of an element (see `border` on page 20).

It is not inherited and *there is no default value; so an outline style must be set.*

```
outline-width: <border-width> | inherit ;
```

specifies the `outline-width` if an `outline-style` is defined (see `border` on page 20)..

It is not inherited and the default value is `medium`.

9.116 overflow

```
overflow: visible | hidden | scroll | auto | inherit ;
```

specifies how any overflow from a block should be handled.

The overflow values are:

visible no clipping

hidden the content is clipped and no scrolling interface is provided

scroll the content is clipped and a scrolling mechanism is present.

auto a scrolling mechanism is provided if necessary.

It is not inherited and the default value is `visible`.

9.117 padding

```
padding: <length> | <percentage> | inherit ;
```

offers a shorthand for setting the values for the padding width.

<percentage> refers to the the width of the containing block (not to the height in the case of `padding-top` and `padding-bottom`)

Up to four values may be specified representing `padding-top`, `padding-right`, `padding-bottom` and `padding-left`. Values must be 0 or positive. Omitted values are co-opted as follows:

- 2nd from 1st
- 3rd from 1st
- 4th from 2nd (i.e. from 1st if no 2nd)

Alternatively, the width of each padding may be specified separately with:

```
padding-top: <length> | <percentage> | inherit ;
padding-right: <length> | <percentage> | inherit ;
padding-bottom: <length> | <percentage> | inherit ;
padding-left: <length> | <percentage> | inherit ;
```

Values are not inherited and The default value is `0`.

The colour of padding is specified by the `background` property.

9.118 page-break-after

An alias for break-after. the value `always` defaults to `page`.

It is not inherited and the default value is `auto`.

9.119 page-break-before

An alias for break-before. the value `always` defaults to `page`.

It is not inherited and the default value is `auto`.

9.120 page-break-inside

An alias for break-inside.

It is not inherited and the default value is `auto`.

9.121 pause

```
pause: <pause-before> <pause-after>? ;
```

offers a shorthand for pause-before and pause-after.

9.122 pause-after

```
pause-after: <time> | none | x-weak | weak | medium |
strong | x-strong ;
```

specifies the length of a pause after an element.

It is not inherited and the default value is `none`.

The values are:

<time> non-negative absolute time units (seconds and milliseconds, e.g. "+3s", "250ms").

none 0ms.

x-weak | **weak** | **medium** | **strong** | **x-strong** monotonically non-decreasing implementation-dependent values indicating the break strength between elements.

The default in `none`.

9.123 pause-before

```
pause-before: <time> | none | x-weak | weak | medium |
strong | x-strong ;
```

specifies the length of a pause before an element.

The values are the same as for `pause-after`.

The default in `none`.

9.124 place-content

```
place-content: <align-content> <justify-content>?
```

offers a shorthand for align-content and justify-content.

9.125 place-items

```
place-items: [ normal | stretch | <baseline-position> |
<self-position> ] [ normal | stretch | <baseline-position> |
<self-position> ]?
```

offers a shorthand for align-items and justify-items.

9.126 place-self

```
place-self: <align-self> <justify-self>?
```

offers a shorthand for align-self and justify-self.

9.127 play-during

```
play-during: url("<url>") [ mix || repeat ]? | auto | none |
inherit ;
```

specifies whether and how a background sound is to be played during an aural[7] element.

The play-during values are:

url("<url>") the link to an audio file to be played during the spoken content

mix play the sound from the parent element's `play-during` property along with the current background sound

repeat repeat the audio file if it is too short to fill the time to the end of the spoken content

auto play the parent element's `play-during` background sound (n.b. `inherit` restarts rather than continues the parent element's sound)

none play no sound during the spoken content (unless suppressed the parent element's sound sill continue behind the next spoken element)

It is not inherited and the default value is `auto`.

[7] The aural category will be replaced by the audio and speech categories in due course.

9.128 position

```
position: static | relative | absolute | fixed | inherit ;
```

The position values are:

static the normal position within normal flow

relative the position within normal flow offset by some factor; n.b. this does not affect the position of subsequent elements in normal flow

absolute the position specified is outside normal flow and does not affect the positioning of subsequent elements in normal flow

fixed the position specified is outside the normal flow and fixed by reference to some point; it does not move when scrolled.

it is not inherited and the default value is `static`.

```
top: <length> | <percentage> | auto | inherit ;
right: <length> | <percentage> | auto | inherit ;
bottom: <length> | <percentage> | auto | inherit ;
left: <length> | <percentage> | auto | inherit ;
```

specify the absolute position of a box.

<percentage> refers to the height or width of the containing block as appropriate

auto depends on context.

Values are not inherited and the default value is `auto`.

9.129 quotes

```
quotes: [<string> <string>]+ | none | inherit ;
```

specifies the opening and closing quotation marks to be used.

The first pair of `<string>` values specify the outermost quotation mark style, subsequent pairs the style for each subsequent level of nesting.

The default value depends on the user agent.

9.130 resolution

```
resolution: <resolution> ;
```

specifies the resolution of a media type (see section 4 on page 7).

<resolution> an integer and the units used (see section 6.7 on page 12)

The media type must use square pixels; if it does not, use max-resolution or min-resolution instead.

It is not inherited and there is no default value.

9.131 rest

```
rest: <rest-before> <rest-after>?
```

offers a shorthand for rest-before and rest-after.

9.132 rest-after

```
rest-after: <time> | none | x-weak | weak | medium |
strong | x-strong
```

specifies a silence with a specific duration that occurs after the speech synthesis rendition of an element.

The values are:

<time> non-negative absolute time units (seconds and milliseconds, e.g. "+3s", "250ms")

none 0ms.

x-weak | weak | medium | strong | x-strong monotonically non-decreasing prosodic breaks in speech output between elements. The exact time is implementation-dependent.

The default in `none`.

As opposed to the pause properties, the rest is inserted between the element's content and any `cue-before` or `cue-after` content. Adjoining rests are treated additively.

9.133 rest-before

```
rest-before: <time> | none | x-weak | weak | medium |
strong | x-strong
```

specifies a silence with a specific duration that occurs before the speech synthesis rendition of an element.

The values are as for `rest-after`.

The default in `none`.

9.134 right

See position.

9.135 row-gap

```
row-gap: normal | <length-percentage>
```

specifies the gutters between rows.

9.136 scan

```
scan: progressive | interlace ;
```

specifies the scanning process where `media="tv"`

It is not inherited and there is no default value.

9.137 speak

```
speak: auto | never | always ;
```

specifies determines whether or not to render text audibly.

The values are:

auto resolves to `never` when `display` is `none`; otherwise it is equivalent to `always` if `visibility` is `visible` and to `never` otherwise.

never causes an element not be rendered.

always the element is rendered audibly.

The default value is `auto`.

9.138 speak-as

```
speak-as: normal | spell-out || digits || [ literal-punctuation |
no-punctuation ]
```

determines in what manner text gets rendered audibly.

The values are:

normal use the language-dependent pronunciation rules for rendering the element's content.

spell-out spell the text one letter at a time (useful for acronyms and abbreviations).

digits speak numbers one digit at a time, for instance, "twelve" would be spoken as "one two", and "31" as "three one".

literal-punctuation name punctuation such as semicolons, braces, and so on aloud rather than rendering it naturally as appropriate pauses.

no-punctuation do not render punctuation whether spoken or rendered as pauses.

The default in `normal`.

9.139 table-layout

```
table-layout: auto | fixed | inherit ;
```

specifies whether a table layout is defined by the fixed widths, if any, of the table, its border, columns and cells or by an algorithm.

The default value is `auto`.

9.140 text-align

```
text-align: left | right | center | justify | inherit ;
```

specifies the alignment of a block of text.

`justify` should be avoided because there are no standard hyphenation rules and user agents use different algorithms to represent it.

The default value is `left` in `ltr` and `right` in `rtl` blocks

9.141 text-decoration

```
text-decoration: none | [ underline || overline ||
line-through || blink ] | inherit ;
```

specifies whether the text is to be decorated.

underline draw a line with the value of the color property under the line

overline draw a line with the value of the color property over the line

line-through draw a line with the value of the color property at the x-height of the line

blink alternative between `visibility: visible` and `visibility: hidden`

Only the value of the `color` property of a decoration is inherited by an element's descendants.

The default value is `none`.

9.142 text-indent

```
text-indent: <length> | <percentage> | inherit ;
```

specifies the indentation of the first line of a block.

<percentage> refers to the width of the containing block

A negative value creates a hanging indent which should be supported by specifying sufficient `padding`.

It is inherited if it is an inline block, The default value is `0`.

9.143 text-transform

```
text-transform: capitalize | uppercase | lowercase |
none | inherit ;
```

specifies whether text is to be transformed.

The text transform values are:

capitalize makes the first letter of each word `uppercase`

uppercase makes all letters `uppercase`

lowercase makes all letters `lowercase`

none makes no effects

The default value is `none`.

9.144 top

See `position` on page 52.

9.145 unicode-bidi

```
unicode-bidi: normal | embed | bidi-override | inherit ;
```

The unicode-bidi values are:

normal follow the Unicode `bidi` algorithm

embed embed an additional inline box that follows the `direction` property in an inline box

bidi-override override the algorithm where it conflicts with the `direction` property

It is not inherited and the default value is `normal`.

9.146 vertical-align

```
vertical-align: baseline | sub | super | top | text-top |
middle | bottom | text-bottom | <percentage> | <length> | inherit ;
```

specifies the position of the content of inline boxes and table cells..

The vertical align values are:

baseline of the parent box or the first row that a cell spans

sub the subscript position of the parent box (not applicable to table elements); note that this value does not affect font size

super the superscript position of the parent box (not applicable to table elements); note that this value does not affect font size

top the top of the highest content area of the subtree of the parent element and all child elements

text-top the top of the parent element's content area (not applicable to table elements)

middle the x-height of the parent box or the middle of the table row

bottom the bottom of the lowest content area of the subtree of the parent element and all child elements

text-bottom the bottom of the parent element's content area (not applicable to table elements)

<**percentage**> refers to the line height; a positive value raises the content above the baseline, a negative value lowers it below the baseline (not applicable to table elements)

<**length**> a positive value raises the content this distance above the baseline, a negative value lowers it this distance below the baseline (not applicable to table elements)

It is not inherited and the default value in `baseline`.

9.147 view-mode

```
windowed | floating | fullscreen | maximized | minimized
```

specifies how visual media are to be presented; the values represent:

windowed in a window with window decorations visible

floating as if windowed but without window decorations visible

fullscreen maximised without window decorations visible

maximized maximised with window decorations visible

minimized minimised

9.148 visibility

```
visibility: visible | hidden | collapse | inherit ;
```

specifies how boxes generated within an element will appear.

The visibility values are

visible boxes are visible

hidden boxes are generated but not visible, that is, they affect the normal flow; to prevent any effects on normal flow use display: `none`

collapse when applied to table rows, row groups columns or columns groups, has the same effect as `hide` in a spreadsheet (applies only to table elements)

The default value is `visible`.

9.149 voice-balance

```
voice-balance: <number> | left | center | right | leftwards |
rightwards
```

controls the spatial distribution of audio output across a lateral sound stage.

The values are:

<number> −100 represents the left side, and 100 represents the right side. 0 represents the centre point

left −100

center 0

right 100

leftwards Moves the sound to the left, by subtracting 20 from the inherited `voice-balance` value up to −100

rightwards Moves the sound to the right, by adding 20 to the inherited `voice-balance` value up to 100

The default value is `center`.

9.150 voice-duration

```
voice-duration: auto | <time>
```

specifies how long it should take to render the selected element's content (not including audio cues, pauses and rests).

The values are:

auto uses the voice-rate to calculate duration.

<time> specifies a value in non-negative time units (seconds and milliseconds, e.g. "+3s", "250ms").

The default value is `auto`.

9.151 voice-family

```
voice-family:[[<name> | <generic-voice>],]* [<name> |
<generic-voice>] | preserve ;
```

specifies specifies a prioritized list of component values that are separated by commas to indicate that they are alternatives. Each component value potentially designates a speech synthesis voice instance.

The comma separated voice family values are:

<name> specific voice instances (e.g., Mike, comedian, mary, carlos2, "valley girl"). Voice names must either be given quoted as strings, or unquoted as a sequence of one or more identifiers.

<**generic-voice**> a space separated list of values covering:

- `<age>` `child`, `young` and `old`
- `<gender>` `male`, `female`, or `neutral`
- `<integer>` indicating the preferred variant (e.g. "the second male child voice"). Only positive integers (i.e. excluding zero) are allowed. The value "1" refers to the first of all matching voices.

preserve inherit the `voice-family` value.

9.152 voice-pitch

```
voice-pitch: <frequency> && absolute | [[x-low | low | medium |
high | x-high] || [<frequency> | <semitones> | <percentage>]]
```

specifies the baseline pitch of the generated speech output which depends on the voice-family.

The values are:

<**frequency**> specifies a value in frequency units (Hertz or kiloHertz, e.g. "100Hz", "+2kHz"). If `absolute` is not specified, a negative value represents a decrement, and a positive value represents an increment, relative to the inherited value.

absolute If specified, `frequency` is an absolute value which must be positive.

<**semitones**> specifies a relative change (decrement or increment) of *n*st.

<**percentage**> non-negative percentage value representing a change relative to the given keyword value or to the default value.

x-low | **low** | **medium** | **high** | **x-high** implementation and voice specific monotonically non-decreasing pitch levels.

The default value is `medium`.

9.153 voice-range

```
voice-range: <frequency> && absolute | [[x-low | low | medium |
high | x-high] || [<frequency> | <semitones> | <percentage>]]
```

specifies the variability in the "baseline" pitch from the average pitch of the speech output.

The values are:

<**frequency**> specifies a value in frequency units (Hertz or kiloHertz, e.g. "100Hz", "+2kHz"). If `absolute` is not specified, a negative value represents a decrement, and a positive value represents an increment, relative to the inherited value.

absolute If specified, `frequency` is an absolute value which must be positive.

<**semitones**> specifies a relative change (decrement or increment) of *n*st.

<**percentage**> non-negative percentage value representing a change relative to the given keyword value or to the default value.

x-low | **low** | **medium** | **high** | **x-high** implementation and voice specific monotonically non-decreasing pitch levels.

The default value is `medium`.

9.154 voice-rate

```
voice-rate: [normal | x-slow | slow | medium | fast |
x-fast] || <percentage>
```

manipulates the rate of generated synthetic speech in terms of words per minute.

The values are

normal the default rate for the currently active voice.

x-slow | **slow** | **medium** | **fast** | **x-fast** implementation and voice-specific monotonically non-decreasing speaking rates

<**percentage**> non-negative percentage value representing a change relative to the given keyword value or to the default value.

The default value is `medium`.

9.155 voice-stress

```
voice-stress: normal | strong | moderate | none | reduced
```

manipulates the strength of emphasis.

The values are:

normal the default emphasis produced by the speech synthesizer.

none no emphasis.

moderate | **strong** monotonically increasing emphasis which is more than `normal`.

reduced Effectively the opposite of emphasizing a word.

The default value is `normal`.

9.156 voice-volume

```
voice-volume: silent | [[x-soft | soft | medium | loud | x-loud] ||
<decibel>]
```

allows authors to control the amplitude of the audio waveform generated by the speech synthesiser, and to adjust the relative volume level of audio cues within the selected element.

The values are:

silent no sound is generated

x-soft minimum audible level

soft intermediate level between `x-soft` and `medium`

medium user‘s preferred volume level

loud intermediate level between `medium` and `x-loud`

x-loud maximum tolerable volume level

<**decibel**> is a positive or negative number followed by `dB` indicating a change from the value specified before it. Note that -6.0`dB` to $+6.0$`dB` normally covers the entire audio range.

The default value is `medium`.

9.157 white-space

```
white-space: normal | pre | nowrap | pre-wrap | pre-line |
inherit
```

specifies whether and how white space is to be added to text.

The property values are

normal collapse sequences of white space; remove existing newline characters, converting linefeed characters as appropriate; break lines to fill boxes

pre do not collapse sequences of white space; break lines only where there is a newline.

nowrap collapse sequences of white space; convert linefeed characters as appropriate; suppress line breaks

pre-wrap do not collapse sequences of white space; break lines where there is a newline.and to fill boxes

pre-line collapse sequences of white space; remove existing newline characters; break lines where there is a newline.and to fill boxes

The default value is `normal`.

9.158 widows

```
widows: <integer> | inherit ;
```

specifies that minimum number of lines to be left at the top of a page.

The default value is 2.

9.159 width

```
width: <length> | <percentage> | auto | inherit ;
```

specifies the width of a block level containing box or media type or the minimum width of a table column..

<percentage> refers to the containing block; for a block whose position is `absolute` this is defined by the box's padding edge (Figure 1 on page 8).

auto depends on context

It is not inherited and the default value is `auto`.

9.160 will-change

```
will-change: auto | <animateable-feature>#
```

provides a rendering hint to the user agent, stating what kinds of changes the author expects to perform on the element.

The values are:

auto no particular intent

contents animate or change something about the element's contents in the near future. Best used on elements near the bottom of the document tree, containing as little of the document as possible.

<custom-ident> animate or change the property with the given name on the element in the near future.

The default value is `auto`.

9.161 word-spacing

```
word-spacing: normal | <length> | inherit ;
```

specifies the inter-word spacing. Note that `text-align: justify` normally adds inter-word space.

<length> refers to the additional inter-word spacing and may be negative.

The default value is `normal`.

9.162 z-index

```
z-index: auto | <integer> | inherit ;
```

specifies the stack level of an element. Stack levels with a higher integer value are deemed to be closer to the user. Where two or more elements have the same stack level, they are stacked back to front according to their order in the document tree.

The background and borders always have a lower stack level than the lowest element in a stacking context.

auto is 0

It is not inherited and the default value is `auto`.

10 Inheritance

The default values for all attributes are held in the *user-agent* (browser) *style sheet*; these may be overridden by the

user style sheet which is in turn overridden by the

author style sheet unless a rule in the user style sheet has the value `!important` as the final value in a declaration

This is to enable users with specific accessibility needs to control the presentation of content.

Unless the value is inherited, an element with an `id=" "` attribute takes precedence over an element with a `class=" "` attribute which takes precedence over any other element.

Where two or more rules might apply to the same element, the one with the highest specificity gains precedence; where two or more rules have the same specificity, the last one gains precedence.

The values of all attributes, whether applying to the root element or inherited from a parent element, are then resolved to

computed values which are absolute values; except where a computed value includes, for example, a percentage, this value is then turned into a

used value or an absolute value derived after ascertaining the dimensions of other elements and/or from applying the percentage; it is then possible that the user agent may not be able to apply this value but only an approximation of it, known as the

actual value

Applying the attribute `inherit` to a property of the root element causes it to be inherited when it would not otherwise have been inherited.

Where property values would not normally be inherited, this has been stated in section 9 Properties.

References

Meyer, E. A. (2000). *Cascading style sheets: the definitive guide.* Sebastopol CA/Cambridge: O'Reilly.

A Sources for this document

These notes are based on Meyer (2000)
Cascading Style Sheets Level 2 Revision 1 (CSS 2,1) Specification 07 June 2011
CSS Color Module Level 3 7 June 2011
CSS Color Module Level 3 Editor's Draft 19 December 2012
CSS Namespaces Module 29 September 2011
CSS Namespaces Module Editor's Draft 8 June 2013
Media queries 19 June 2012
The 'view-mode' Media Feature W3C Recommendation 19 June 2012
Selectors Level 3 W3C Recommendation 29 September 2011
Selectors Level 3 Editor's Draft 2 January 2013
CSS Snapshot 2017

B The grammar of selectors

`*`	0 or more times
`+`	1 or more times
`?`	optional
`|`	separates two or more alternatives: exactly one of them must occur
`[ ]`	grouping.

C Other symbols used in CSS specifications

`{n1,n2}`	repeat `n1` one or more times up to `n2`
`/* ... */`	comment: cannot be nested
`||`	separates two or more options: one or more of them must occur, in any order.
`&&`	separates two or more components, all of which must occur, in any order.

Several juxtaposed words mean that all of them must occur, in the given order.

Juxtaposition is stronger than the double ampersand, the double ampersand is stronger than the double bar, and the double bar is stronger than the bar.

D SVG 1.0 colours

	HEX		HEX
aliceblue	#F0F8FF	antiquewhite	#FAEBD7
aqua (=cyan)	#00FFFF	aquamarine	#7FFFD4
azure	#F0FFFF	beige	#F5F5DC
bisque	#FFE4C4	black	#000000
blanchedalmond	#FFEBCD	blue	#0000FF
blueviolet	#8A2BE2	brown	#A52A2A
burlywood	#DEB887	cadetblue	#5F9EA0
chartreuse	#7FFF00	chocolate	#D2691E
coral	#FF7F50	cornflowerblue	#6495ED
cornsilk	#FFF8DC	crimson	#DC143C
cyan (=aqua)	#00FFFF	darkblue	#00008B
darkcyan	#008B8B	darkgoldenrod	#B8860B
darkgray (=darkgrey)	#A9A9A9	darkgreen	#006400
darkgrey (=darkgray)	#A9A9A9	darkkhaki	#BDB76B
darkmagenta	#8B008B	darkolivegreen	#556B2F
darkorange	#FF8C00	darkorchid	#9932CC
darkred	#8B0000	darksalmon	#E9967A
darkseagreen	#8FBC8F	darkslateblue	#483D8B
darkslategray (=darkslategrey)	#2F4F4F	darkslategrey (=darkslategray)	#2F4F4F
darkturquoise	#00CED1	darkviolet	#9400D3
deeppink	#FF1493	deepskyblue	#00BFFF
dimgray (=dimgrey)	#696969	dimgrey (=dimgray)	#696969
dodgerblue	#1E90FF	firebrick	#B22222
floralwhite	#FFFAF0	forestgreen	#228B22
fuchsia (=magenta)	#FF00FF	gainsboro	#DCDCDC
ghostwhite	#F8F8FF	gold	#FFD700
goldenrod	#DAA520	gray	#808080
green	#008000	greenyellow	#ADFF2F
grey	#808080	honeydew	#F0FFF0
hotpink	#FF69B4	indianred	#CD5C5C
indigo	#4B0082	ivory	#FFFFF0
khaki	#F0E68C	lavender	#E6E6FA
lavenderblush	#FFF0F5	lawngreen	#7CFC00
lemonchiffon	#FFFACD	lightblue	#ADD8E6
lightcoral	#F08080	lightcyan	#E0FFFF
lightgoldenrodyellow	#FAFAD2	lightgray (=lightgrey)	#D3D3D3
lightgreen	#90EE90	lightgrey (=lightgray)	#D3D3D3
lightpink	#FFB6C1	lightsalmon	#FFA07A
lightseagreen	#20B2AA	lightskyblue	#87CEFA
lightslategray (=lightslategrey)	#778899	lightslategrey (=lightslategray)	#778899
lightsteelblue	#B0C4DE	lightyellow	#FFFFE0
lime (= Unix green)	#00FF00	limegreen	#32CD32
linen	#FAF0E6	magenta (=fuchsia)	#FF00FF

	HEX		HEX
maroon	#800000	mediumaquamarine	#66CDAA
mediumblue	#0000CD	mediumorchid	#BA55D3
mediumpurple	#9370DB	mediumseagreen	#3CB371
mediumslateblue	#7B68EE	mediumspringgreen	#00FA9A
mediumturquoise	#48D1CC	mediumvioletred	#C71585
midnightblue	#191970	mintcream	#F5FFFA
mistyrose	#FFE4E1	moccasin	#FFE4B5
navajowhite	#FFDEAD	navy	#000080
oldlace	#FDF5E6	olive	#808000
olivedrab	#6B8E23	orange	#FFA500
orangered	#FF4500	orchid	#DA70D6
palegoldenrod	#EEE8AA	palegreen	#98FB98
paleturquoise	#AFEEEE	palevioletred	#DB7093
papayawhip	#FFEFD5	peachpuff	#FFDAB9
peru	#CD853F	pink	#FFC0CB
plum	#DDA0DD	powderblue	#B0E0E6
purple	#800080	red	#FF0000
rosybrown	#BC8F8F	royalblue	#4169E1
saddlebrown	#8B4513	salmon	#FA8072
sandybrown	#F4A460	seagreen	#2E8B57
seashell	#FFF5EE	sienna	#A0522D
silver	#C0C0C0	skyblue	#87CEEB
slateblue	#6A5ACD	slategray (=slategrey)	#708090
slategrey (=slategray)	#708090	snow	#FFFAFA
springgreen	#00FF7F	steelblue	#4682B4
tan	#D2B48C	teal	#008080
thistle	#D8BFD8	tomato	#FF6347
turquoise	#40E0D0	violet	#EE82EE
wheat	#F5DEB3	white	#FFFFFF
whitesmoke	#F5F5F5	yellow	#FFFF00
yellowgreen	#9ACD32