

CSS Color Module Level 3

W3C Recommendation 18 January 2022



▼ More details about this document

This version:

<https://www.w3.org/TR/2022/REC-css-color-3-20220118/>

Latest version:

<https://www.w3.org/TR/css-color-3/>

Previous version:

<https://www.w3.org/TR/2021/REC-css-color-3-20210805/>

History

<https://www.w3.org/standards/history/css-color-3>

Editor's Draft:

<https://drafts.csswg.org/css-color-3/>

Implementation Report:

https://test.csswg.org/harness/results/css-color-3_dev/grouped/

Feedback:

[GitHub Issues](#) are preferred for discussion of this specification. When filing an issue, please put the text “css-color-3” in the title, preferably like this: “[css-color-3] ...*summary of comment*...”. All issues and comments are [archived](#), and there is also a [historical archive](#).

Editors:

[Tantek Çelik](#) ([Mozilla Corporation](#), and before at [Microsoft Corporation](#))

<tantek@cs.stanford.edu>

Chris Lilley ([W3C](#)) <<https://svgees.us/>>

L. David Baron (W3C Invited Experts) <dbaron@dbaron.org>

Additional Authors:

Steven Pemberton ([CWI](#)) <steven.pemberton@cw.nl>

Brad Pettit ([Microsoft Corporation](#)) <bradp@microsoft.com>

Test Suite:

<http://test.csswg.org/suites/css3-color/nightly-unstable/>

Errata:

<https://www.w3.org/Style/2022/REC-css-color-3-20220118-errata.html>

Copyright © 2022 W3C® ([MIT](#), [ERCIM](#), [Keio](#), [Beihang](#)). W3C [liability](#), [trademark](#) and [permissive document license](#) rules apply.

Abstract

CSS (Cascading Style Sheets) is a language for describing the rendering of HTML and XML documents on screen, on paper, in speech, etc. It uses color-related properties and values to color the text, backgrounds, borders, and other parts of elements in a document. This specification describes color values and properties for foreground color and group opacity. These include properties and values from CSS level 2 and new values.

Status of This Document

This section describes the status of this document at the time of its publication. A list of current W3C publications and the latest revision of this technical report can be found in the [W3C technical reports index](https://www.w3.org/TR/) at <https://www.w3.org/TR/>.

This document was published by the [CSS Working Group](#) as a **Recommendation** using the Recommendation track.

A W3C Recommendation is a specification that, after extensive consensus-building, is endorsed by W3C and its Members, and has commitments from Working Group members to [royalty-free licensing](#) for implementations.

W3C recommends the wide deployment of this specification as a standard for the Web.

This document was produced by a group operating under the [W3C Patent Policy](#). W3C maintains a [public list of any patent disclosures](#) made in connection with the deliverables of the group; that page also includes instructions for disclosing a patent. An individual who has actual knowledge of a patent which the individual believes contains [Essential Claim\(s\)](#) must disclose the information in accordance with [section 6 of the W3C Patent Policy](#).

This document is governed by the [2 November 2021 W3C Process Document](#).

A separate [implementation report](#) shows that each test in the [test suite](#) was passed by at least two independent implementations. However, most tests have now been updated for [CSS Color 4](#) (see [implementation report](#)).

A complete [list of changes](#) to this document is available.

Table of Contents

1. **Introduction**
2. **Dependencies**

- 3. Color properties**
- 3.1. Foreground color: the ‘color’ property
- 3.2. Transparency: the ‘opacity’ property

- 4. Color units**

- 4.1. Basic color keywords
- 4.2. Numerical color values
 - 4.2.1. RGB color values
 - 4.2.2. RGBA color values
 - 4.2.3. ‘transparent’ color keyword
 - 4.2.4. HSL color values
 - 4.2.4.1. HSL examples
 - 4.2.5. HSLA color values
- 4.3. Extended color keywords
- 4.4. ‘currentColor’ color keyword
- 4.5. CSS system colors
 - 4.5.1. CSS2 system colors
- 4.6. Notes on using colors

- 5. Simple alpha compositing**

- 6. Sample style sheet for (X)HTML**

- 7. Profiles**

- 8. Test suite**

- 9. Future features**

- 10. Acknowledgments**

- 11. Changes**

- 12. References**

- 12.1. Normative
- 12.2. Informative

- Index**

- Property index**



1. Introduction

This section is non-normative.

CSS beyond level 2 is a set of modules, divided up to allow the specifications to develop incrementally, along with their implementations. This specification is one of those modules.

This module describes CSS properties which allow authors to specify the foreground color and opacity of an element. This module also describes in detail the CSS `<color>` value type.

It not only defines the color-related properties and values that already exist in [CSS1](#) and [CSS2](#), but also defines new properties and values.

The specification is the result of the merging of relevant parts of the following Recommendations and Working Drafts, and the addition of some new features.

- HTML 4.01 [\[HTML401\]](#)
- CSS 2.0 [\[CSS2\]](#)
- SVG 1.0 [\[SVG10\]](#)
- User Interface for CSS3 (16 February 2000) [\[CSS-UI-3\]](#)

2. Dependencies

Additional terminology is defined in the [Definitions](#) section of [\[CSS21\]](#). Examples of document source code and fragments are given in XML [\[XML10\]](#) or HTML [\[HTML401\]](#) syntax.

3. Color properties

3.1. Foreground color: the ‘[color](#)’ property

<i>Name:</i>	<i>color</i>
<i>Value:</i>	<code><color></code> inherit
<i>Initial:</i>	depends on user agent
<i>Applies to:</i>	all elements
<i>Inherited:</i>	yes
<i>Percentages:</i>	N/A

- Computed value:

- The computed value for basic color keywords, RGB hex values and extended color keywords is the equivalent triplet of numerical RGB values, e.g. `#000000` six digit hex value or `rgb(...)` functional value , with an alpha value of 1.
 - The computed value of the keyword `'transparent'` is the quadruplet of all zero numerical RGBA values, e.g. `rgba(0,0,0,0)` .
 - For all other values, the computed value is the specified value.

This property describes the foreground color of an element's text content. In addition it is used to provide a potential indirect value (`currentColor`) for any *other* properties that accept color values. If the `'currentColor'` keyword is set on the `'color'` property itself, it is treated as `'color: inherit'`.

There are different ways to specify lime green:

```
em { color: lime }           /* color keyword */
em { color: rgb(0,255,0) }   /* RGB range 0-255 */
```

<color>
Color units are defined in a [following](#) section.

3.2. Transparency: the `'opacity'` property

Opacity can be thought of as a postprocessing operation. Conceptually, after the element (including its descendants) is rendered into an RGBA offscreen image, the opacity setting specifies how to blend the offscreen rendering into the current composite rendering. See [simple alpha compositing](#) for details.

Name:	<i>opacity</i>
Value:	<alphavalue> inherit
Initial:	1
Applies to:	all elements

Inherited: no

Percentages: N/A

Computed value: The same as the specified value after clipping the <alphavalue> to the range [0.0,1.0].

<alphavalue>

Syntactically a <number>. The uniform opacity setting to be applied across an entire object. Any values outside the range 0.0 (fully transparent) to 1.0 (fully opaque) will be clamped to this range. If the object has children, then the effect is as if the children were blended against the current background using a mask where the value of each pixel of the mask is <alphavalue>. For SVG, ‘has children’ is equivalent to being a container element [\[SVG11\]](#).

Since an element with opacity less than 1 is composited from a single offscreen image, content outside of it cannot be layered in z-order between pieces of content inside of it. For the same reason, implementations must create a new stacking context for any element with opacity less than 1. If an element with opacity less than 1 is not positioned, then it is painted on the same layer, within its parent stacking context, as positioned elements with stack level 0. If an element with opacity less than 1 is positioned, the ‘z-index’ property applies as described in [\[CSS21\]](#), except that if the used value is ‘auto’ then the element behaves exactly as if it were ‘0’. See [section 9.9](#) and [Appendix E](#) of [\[CSS21\]](#) for more information on stacking contexts. The rules in this paragraph do not apply to SVG elements, since SVG has its own [rendering model](#) ([\[SVG11\]](#), Chapter 3).

4. Color units

A <color> is either a keyword or a numerical specification.

4.1. Basic color keywords

The list of basic color keywords is: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, and yellow. The color names are [ASCII case-insensitive](#).

Color names and sRGB values

Named	Numeric	Color name	Hex rgb	Decimal
		black	#000000	0,0,0
		silver	#C0C0C0	192,192,192
		gray	#808080	128,128,128
		white	#FFFFFF	255,255,255

		<i>maroon</i>	#800000	128,0,0
		<i>red</i>	#FF0000	255,0,0
		<i>purple</i>	#800080	128,0,128
		<i>fuchsia</i>	#FF00FF	255,0,255
		<i>green</i>	#008000	0,128,0
		<i>lime</i>	#00FF00	0,255,0
		<i>olive</i>	#808000	128,128,0
		<i>yellow</i>	#FFFF00	255,255,0
		<i>navy</i>	#000080	0,0,128
		<i>blue</i>	#0000FF	0,0,255
		<i>teal</i>	#008080	0,128,128
		<i>aqua</i>	#00FFFF	0,255,255

```
body {color: black; background: white }
h1 { color: maroon }
h2 { color: olive }
```

4.2. Numerical color values

4.2.1. RGB color values

The RGB color model is used in numerical color specifications. These examples all specify the same color:

```
em { color: #f00 }           /* #rgb */
em { color: #ff0000 }        /* #rrggbb */
em { color: rgb(255,0,0) }
em { color: rgb(100%, 0%, 0%) }
```

The format of an RGB value in hexadecimal notation is a ‘#’ immediately followed by either three or six hexadecimal characters. The three-digit RGB notation (**#rgb**) is converted into six-digit form (**#rrggbb**) by replicating digits, not by adding zeros. For example, **#fb0** expands to **#ffbb00**. This ensures that white (**#ffffff**) can be specified with the short notation (**#fff**) and removes any dependencies on the color depth of the display.

The format of an RGB value in the functional notation is ‘**rgb(**’ followed by a comma-separated list of three numerical values (either three integer values or three percentage values) followed by

‘)’’. The integer value 255 corresponds to 100%, and to F or FF in the hexadecimal notation: `rgb(255,255,255) = rgb(100%,100%,100%) = #FFF`. White space characters are allowed around the numerical values.

All RGB colors are specified in the sRGB color space (see [\[SRGB\]](#)). User agents may vary in the fidelity with which they represent these colors, but using sRGB provides an unambiguous and objectively measurable definition of what the color should be, which can be related to international standards (see [\[COLORIMETRY\]](#)).

Values outside the device gamut should be clipped or mapped into the gamut when the gamut is known: the red, green, and blue values must be changed to fall within the range supported by the device. User agents may perform higher quality mapping of colors from one gamut to another. This specification does not define precise clipping behavior. For a typical CRT monitor, whose device gamut is the same as sRGB, the four rules below are equivalent:

```
em { color: rgb(255,0,0) }      /* integer range 0 - 255 */
em { color: rgb(300,0,0) }      /* clipped to rgb(255,0,0) */
em { color: rgb(255,-10,0) }    /* clipped to rgb(255,0,0) */
em { color: rgb(110%, 0%, 0%) } /* clipped to rgb(100%,0%,0%) */
```

Other devices, such as printers, have different gamuts than sRGB; some colors outside the 0..255 sRGB range will be representable (inside the device gamut), while other colors inside the 0..255 sRGB range will be outside the device gamut and will thus be mapped.

4.2.2. RGBA color values

The RGB color model is extended in this specification to include “alpha” to allow specification of the opacity of a color. See [simple alpha compositing](#) for details. These examples all specify the same color:

```
em { color: rgb(255,0,0) }      /* integer range 0 - 255 */
em { color: rgba(255,0,0,1) }    /* the same, with explicit opacity */
em { color: rgb(100%,0%,0%) }    /* float range 0.0% - 100.0% */
em { color: rgba(100%,0%,0%,1) } /* the same, with explicit opacity */
```

Unlike RGB values, there is no hexadecimal notation for an RGBA value.

The format of an RGBA value in the functional notation is ‘`rgba(`’ followed by a comma-separated list of three numerical values (either three integer values or three percentage values), followed by an [<alphavalue>](#), followed by ‘`)`’. The integer value 255 corresponds to 100%,

`rgba(255,255,255,0.8) = rgba(100%,100%,100%,0.8)`. White space characters are allowed around the numerical values.

Implementations must clip the red, green, and blue components of RGBA color values to the device gamut according to the rules for the RGB color value composed of those components.

These examples specify effects that are possible with the `rgba()` notation:

```
p { color: rgba(0,0,255,0.5) } /* semi-transparent solid blue  
p { color: rgba(100%, 50%, 0%, 0.1) } /* very transparent solid orange
```

Note. If RGBA values are not supported by a user agent, they should be treated like unrecognized values per the CSS forward compatibility parsing rules ([CSS21], Chapter 4). RGBA values must *not* be treated as simply an RGB value with the opacity ignored.

4.2.3. ‘[transparent](#)’ color keyword

CSS1 introduced the ‘[transparent](#)’ value for the background-color property. CSS2 allowed border-color to also accept the ‘[transparent](#)’ value. The Open eBook(tm) Publication Structure 1.0.1 [OEB101] extended the ‘[color](#)’ property to also accept the ‘[transparent](#)’ keyword. CSS3 extends the color value to include the ‘[transparent](#)’ keyword to allow its use with all properties that accept a <color> value. This simplifies the definition of those properties in CSS3.

transparent

Fully transparent. This keyword can be considered a shorthand for transparent black, `rgba(0,0,0,0)`, which is its computed value.

4.2.4. HSL color values

CSS3 adds numerical hue-saturation-lightness (HSL) colors as a complement to numerical RGB colors. It has been observed that RGB colors have the following limitations:

- RGB is hardware-oriented: it reflects the use of CRTs.
- RGB is non-intuitive. People can learn how to use RGB, but actually by internalizing how to translate hue, saturation and lightness, or something similar, to RGB.

There are several other color schemes possible. Some advantages of HSL are that it is symmetrical to lightness and darkness (which is not the case with HSV for example), and it is trivial to convert HSL to RGB.

HSL colors are encoding as a triple (hue, saturation, lightness). Hue is represented as an angle of the color circle (i.e. the rainbow represented in a circle). This angle is so typically measured in degrees that the unit is implicit in CSS; syntactically, only a <number> is given. By definition red=0=360, and the other colors are spread around the circle, so green=120, blue=240, etc. As an angle, it implicitly wraps around such that -120=240 and 480=120. One way an implementation could normalize such an angle x to the range $[0,360)$ (i.e. zero degrees, inclusive, to 360 degrees, exclusive) is to compute $((x \bmod 360) + 360) \bmod 360$. Saturation and lightness are represented as percentages. 100% is full saturation, and 0% is a shade of gray. 0% lightness is black, 100% lightness is white, and 50% lightness is “normal”.

So for instance:

```
* { color: hsl(0, 100%, 50%) } /* red */
* { color: hsl(120, 100%, 50%) } /* lime */
* { color: hsl(120, 100%, 25%) } /* dark green */
* { color: hsl(120, 100%, 75%) } /* light green */
* { color: hsl(120, 75%, 75%) } /* pastel green, and so on */
```

The advantage of HSL over RGB is that it is far more intuitive: you can guess at the colors you want, and then tweak. It is also easier to create sets of matching colors (by keeping the hue the same and varying the lightness/darkness, and saturation)

If saturation is less than 0%, implementations must clip it to 0%. If the resulting value is outside the device gamut, implementations must clip it to the device gamut. This clipping should preserve the hue when possible, but is otherwise undefined. (In other words, the clipping is different from applying the rules for clipping of RGB colors after applying the algorithm below for converting HSL to RGB.)

Converting an HSL color to sRGB is straightforward mathematically. Here's a sample implementation of the conversion algorithm in JavaScript. It returns an array of three numbers representing the red, green, and blue channels of the colors, normalized to the range $[0, 1]$.

```
function hslToRgb (hue, sat, light) {
  hue = hue % 360;

  if (hue < 0) {
    hue += 360;
  }

  sat /= 100;
  light /= 100;

  function f(n) {
    let k = (n + hue/30) % 12;
    let a = sat * Math.min(light, 1 - light);
    return light - a * Math.max(-1, Math.min(k - 3, 9 - k, 1));
  }
```

```

    }

    return [f(0), f(8), f(4)];
}

```

4.2.4.1. HSL examples

Each table below represents one hue. Twelve equally spaced colors (i.e. at 30° intervals) have been chosen from the color circle: red, yellow, green, cyan, blue, magenta, with all the intermediate colors (the last is the color between magenta and red).

In each table, the X axis represents the saturation while the Y axis represents the lightness.

0° Reds

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

30° Reds-Yellows (=Oranges)

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

60° Yellows

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						

90° Yellow-Greens

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						

60° Yellows

	100%	80%	60%	40%	20%	0%
20%						
10%						
0%						

120° Greens

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

180° Cyans

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

240° blues

	100%	80%	60%	40%	20%	0%
100%						

90° Yellow-Greens

	100%	80%	60%	40%	20%	0%
20%						
10%						
0%						

150° Green-Cyans

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

210° Cyan-Blues

	100%	80%	60%	40%	20%	0%
100%						
90%						
80%						
70%						
60%						
50%						
40%						
30%						
20%						
10%						
0%						

270° Blue-Magentas

	100%	80%	60%	40%	20%	0%
100%						

240° blues

100% 80% 60% 40% 20% 0%

90%

80%

70%

60%

50%

40%

30%

20%

300° Magentas

0% 100% 80% 60% 40% 20% 0%

100%

90%

80%

70%

60%

50%

40%

30%

20%

10%

0%

270° Blue-Magentas

100% 80% 60% 40% 20% 0%

90%

80%

70%

60%

50%

40%

30%

20%

330° Magenta-Reds

0% 100% 80% 60% 40% 20% 0%

100%

90%

80%

70%

60%

50%

40%

30%

20%

10%

0%

4.2.5. HSLA color values

Just as the ‘`rgb()`’ functional notation has the ‘`rgba()`’ alpha counterpart, the ‘`hsl()`’ functional notation has the ‘`hsla()`’ alpha counterpart. See [simple alpha compositing](#) for details. These examples specify the same color:

```
em { color: hsl(120, 100%, 50%) }      /* green */
em { color: hsla(120, 100%, 50%, 1) } /* the same, with explicit op
```

The format of an HSLA color value in the functional notation is ‘`hsla(`’ followed by the hue in degrees, saturation and lightness as a percentage, and an [<alphavalue>](#), followed by ‘`)`’. White space characters are allowed around the numerical values.

Implementations must clip the hue, saturation, and lightness components of HSLA color values to the device gamut according to the rules for the HSL color value composed of those components.

These examples specify effects that are possible with the `hsla()` notation:

```
p { color: hsla(240, 100%, 50%, 0.5) } /* semi-transparent solid bl
p { color: hsla(30, 100%, 50%, 0.1) } /* very transparent solid or
```

4.3. Extended color keywords

The table below provides a list of the X11 colors [\[X11COLORS\]](#) supported by popular browsers with the addition of gray/grey variants from SVG 1.0. The resulting list is precisely the same as the [SVG 1.0 color keyword names](#). This specification extends their definition beyond SVG. The two color swatches on the left illustrate setting the background color of a table cell in two ways: The first column uses the named color value, and the second column uses the respective numeric color value.

Named	Numeric	Color name	Hex rgb	Decimal
		<i>aliceblue</i>	#F0F8FF	240,248,255
		<i>antiquewhite</i>	#FAEBD7	250,235,215
		<i>aqua</i>	#00FFFF	0,255,255
		<i>aquamarine</i>	#7FFFD4	127,255,212
		<i>azure</i>	#F0FFFF	240,255,255
		<i>beige</i>	#F5F5DC	245,245,220
		<i>bisque</i>	#FFE4C4	255,228,196
		<i>black</i>	#000000	0,0,0
		<i>blanchedalmond</i>	#FFEBCD	255,235,205
		<i>blue</i>	#0000FF	0,0,255
		<i>blueviolet</i>	#8A2BE2	138,43,226
		<i>brown</i>	#A52A2A	165,42,42
		<i>burlywood</i>	#DEB887	222,184,135
		<i>cadetblue</i>	#5F9EA0	95,158,160
		<i>chartreuse</i>	#7FFF00	127,255,0
		<i>chocolate</i>	#D2691E	210,105,30
		<i>coral</i>	#FF7F50	255,127,80
		<i>cornflowerblue</i>	#6495ED	100,149,237
		<i>cornsilk</i>	#FFF8DC	255,248,220

		<i>crimson</i>	#DC143C	220,20,60
		<i>cyan</i>	#00FFFF	0,255,255
		<i>darkblue</i>	#00008B	0,0,139
		<i>darkcyan</i>	#008B8B	0,139,139
		<i>darkgoldenrod</i>	#B8860B	184,134,11
		<i>darkgray</i>	#A9A9A9	169,169,169
		<i>darkgreen</i>	#006400	0,100,0
		<i>darkgrey</i>	#A9A9A9	169,169,169
		<i>darkkhaki</i>	#BDB76B	189,183,107
		<i>darkmagenta</i>	#8B008B	139,0,139
		<i>darkolivegreen</i>	#556B2F	85,107,47
		<i>darkorange</i>	#FF8C00	255,140,0
		<i>darkorchid</i>	#9932CC	153,50,204
		<i>darkred</i>	#8B0000	139,0,0
		<i>darksalmon</i>	#E9967A	233,150,122
		<i>darkseagreen</i>	#8FBC8F	143,188,143
		<i>darkslateblue</i>	#483D8B	72,61,139
		<i>darkslategray</i>	#2F4F4F	47,79,79
		<i>darkslategrey</i>	#2F4F4F	47,79,79
		<i>darkturquoise</i>	#00CED1	0,206,209
		<i>darkviolet</i>	#9400D3	148,0,211
		<i>deeppink</i>	#FF1493	255,20,147
		<i>deepskyblue</i>	#00BFFF	0,191,255
		<i>dimgray</i>	#696969	105,105,105
		<i>dimgrey</i>	#696969	105,105,105
		<i>dodgerblue</i>	#1E90FF	30,144,255
		<i>firebrick</i>	#B22222	178,34,34
		<i>floralwhite</i>	#FFFAF0	255,250,240
		<i>forestgreen</i>	#228B22	34,139,34
		<i>fuchsia</i>	#FF00FF	255,0,255
		<i>gainsboro</i>	#DCDCDC	220,220,220
		<i>ghostwhite</i>	#F8F8FF	248,248,255
		<i>gold</i>	#FFD700	255,215,0
		<i>goldenrod</i>	#DAA520	218,165,32
		<i>gray</i>	#808080	128,128,128

		<i>green</i>	#008000	0,128,0
		<i>greenyellow</i>	#ADFF2F	173,255,47
		<i>grey</i>	#808080	128,128,128
		<i>honeydew</i>	#F0FFF0	240,255,240
		<i>hotpink</i>	#FF69B4	255,105,180
		<i>indianred</i>	#CD5C5C	205,92,92
		<i>indigo</i>	#4B0082	75,0,130
		<i>ivory</i>	#FFFFFF	255,255,240
		<i>khaki</i>	#F0E68C	240,230,140
		<i>lavender</i>	#E6E6FA	230,230,250
		<i>lavenderblush</i>	#FFF0F5	255,240,245
		<i>lawngreen</i>	#7CFC00	124,252,0
		<i>lemonchiffon</i>	#FFFACD	255,250,205
		<i>lightblue</i>	#ADD8E6	173,216,230
		<i>lightcoral</i>	#F08080	240,128,128
		<i>lightcyan</i>	#E0FFFF	224,255,255
		<i>lightgoldenrodyellow</i>	#FAFAD2	250,250,210
		<i>lightgray</i>	#D3D3D3	211,211,211
		<i>lightgreen</i>	#90EE90	144,238,144
		<i>lightgrey</i>	#D3D3D3	211,211,211
		<i>lightpink</i>	#FFB6C1	255,182,193
		<i>lightsalmon</i>	#FFA07A	255,160,122
		<i>lightseagreen</i>	#20B2AA	32,178,170
		<i>lightskyblue</i>	#87CEFA	135,206,250
		<i>lightslategray</i>	#778899	119,136,153
		<i>lightslategrey</i>	#778899	119,136,153
		<i>lightsteelblue</i>	#B0C4DE	176,196,222
		<i>lightyellow</i>	#FFFFE0	255,255,224
		<i>lime</i>	#00FF00	0,255,0
		<i>limegreen</i>	#32CD32	50,205,50
		<i>linen</i>	#FAF0E6	250,240,230
		<i>magenta</i>	#FF00FF	255,0,255
		<i>maroon</i>	#800000	128,0,0
		<i>mediumaquamarine</i>	#66CDAA	102,205,170
		<i>mediumblue</i>	#0000CD	0,0,205

		<i>mediumorchid</i>	#BA55D3	186,85,211
		<i>mediumpurple</i>	#9370DB	147,112,219
		<i>mediumseagreen</i>	#3CB371	60,179,113
		<i>mediumslateblue</i>	#7B68EE	123,104,238
		<i>mediumspringgreen</i>	#00FA9A	0,250,154
		<i>mediumturquoise</i>	#48D1CC	72,209,204
		<i>mediumvioletred</i>	#C71585	199,21,133
		<i>midnightblue</i>	#191970	25,25,112
		<i>mintcream</i>	#F5FFFA	245,255,250
		<i>mistyrose</i>	#FFE4E1	255,228,225
		<i>moccasin</i>	#FFE4B5	255,228,181
		<i>navajowhite</i>	#FFDEAD	255,222,173
		<i>navy</i>	#000080	0,0,128
		<i>oldlace</i>	#FDF5E6	253,245,230
		<i>olive</i>	#808000	128,128,0
		<i>olivedrab</i>	#6B8E23	107,142,35
		<i>orange</i>	#FFA500	255,165,0
		<i>orangered</i>	#FF4500	255,69,0
		<i>orchid</i>	#DA70D6	218,112,214
		<i>palegoldenrod</i>	#EEE8AA	238,232,170
		<i>palegreen</i>	#98FB98	152,251,152
		<i>paleturquoise</i>	#AFEEEE	175,238,238
		<i>palevioletred</i>	#DB7093	219,112,147
		<i>papayawhip</i>	#FFEFD5	255,239,213
		<i>peachpuff</i>	#FFDAB9	255,218,185
		<i>peru</i>	#CD853F	205,133,63
		<i>pink</i>	#FFC0CB	255,192,203
		<i>plum</i>	#DDA0DD	221,160,221
		<i>powderblue</i>	#B0E0E6	176,224,230
		<i>purple</i>	#800080	128,0,128
		<i>red</i>	#FF0000	255,0,0
		<i>rosybrown</i>	#BC8F8F	188,143,143
		<i>royalblue</i>	#4169E1	65,105,225
		<i>saddlebrown</i>	#8B4513	139,69,19
		<i>salmon</i>	#FA8072	250,128,114

		<i>sandybrown</i>	#F4A460	244,164,96
		<i>seagreen</i>	#2E8B57	46,139,87
		<i>seashell</i>	#FFF5EE	255,245,238
		<i>sienna</i>	#A0522D	160,82,45
		<i>silver</i>	#C0C0C0	192,192,192
		<i>skyblue</i>	#87CEEB	135,206,235
		<i>slateblue</i>	#6A5ACD	106,90,205
		<i>slategray</i>	#708090	112,128,144
		<i>slategrey</i>	#708090	112,128,144
		<i>snow</i>	#FFFAFA	255,250,250
		<i>springgreen</i>	#00FF7F	0,255,127
		<i>steelblue</i>	#4682B4	70,130,180
		<i>tan</i>	#D2B48C	210,180,140
		<i>teal</i>	#008080	0,128,128
		<i>thistle</i>	#D8BFD8	216,191,216
		<i>tomato</i>	#FF6347	255,99,71
		<i>turquoise</i>	#40E0D0	64,224,208
		<i>violet</i>	#EE82EE	238,130,238
		<i>wheat</i>	#F5DEB3	245,222,179
		<i>white</i>	#FFFFFF	255,255,255
		<i>whitesmoke</i>	#F5F5F5	245,245,245
		<i>yellow</i>	#FFFF00	255,255,0
		<i>yellowgreen</i>	#9ACD32	154,205,50

4.4. ‘[currentColor](#)’ color keyword

CSS1 and CSS2 defined the initial value of [the ‘border-color’ property](#) to be “the value of the ‘[color](#)’ property” but did not define a corresponding keyword. This omission was recognized by SVG, and thus [SVG 1.0 introduced the ‘currentColor’ value](#) for the ‘fill’, ‘stroke’, ‘stop-color’, ‘flood-color’, and ‘lighting-color’ properties. CSS3 extends the color value to include the ‘[currentColor](#)’ keyword to allow its use with all properties that accept a <color> value. This simplifies the definition of those properties in CSS3.

currentColor

The value of the ‘[color](#)’ property. The used value of the ‘[currentColor](#)’ keyword is the computed value of the ‘[color](#)’ property. If the ‘[currentColor](#)’ keyword is set on the ‘[color](#)’ property itself, it is treated as ‘color: inherit’.

4.5. CSS system colors

4.5.1. CSS2 system colors

Deprecated. In addition to being able to assign color keyword values to text, backgrounds, etc., [CSS2](#) allowed authors to specify colors in a manner that integrated them into the user's graphic environment.

For systems that do not have a corresponding value, the specified value should be mapped to the nearest system color value, or to a default color. Note that some profiles of CSS may not support System Colors at all.

The following lists additional values for color-related CSS values and their general meaning. Any color property can take one of the following names. Although these are [ASCII case-insensitive](#), it is recommended that the mixed capitalization shown below be used, to make the names more legible.

ActiveBorder

Active window border.

ActiveCaption

Active window caption.

AppWorkspace

Background color of multiple document interface.

Background

Desktop background.

ButtonFace

The face background color for 3-D elements that appear 3-D due to one layer of surrounding border.

ButtonHighlight

The color of the border facing the light source for 3-D elements that appear 3-D due to one layer of surrounding border.

ButtonShadow

The color of the border away from the light source for 3-D elements that appear 3-D due to one layer of surrounding border.

ButtonText

Text on push buttons.

CaptionText

Text in caption, size box, and scrollbar arrow box.

GrayText

Grayed (disabled) text. This color is set to #000 if the current display driver does not support a solid gray color.

Highlight

Item(s) selected in a control.

HighlightText

Text of item(s) selected in a control.

InactiveBorder

Inactive window border.

InactiveCaption

Inactive window caption.

InactiveCaptionText

Color of text in an inactive caption.

InfoBackground

Background color for tooltip controls.

InfoText

Text color for tooltip controls.

Menu

Menu background.

MenuText

Text in menus.

Scrollbar

Scroll bar gray area.

ThreeDDarkShadow

The color of the darker (generally outer) of the two borders away from the light source for 3-D elements that appear 3-D due to two concentric layers of surrounding border.

ThreeDFace

The face background color for 3-D elements that appear 3-D due to two concentric layers of surrounding border.

ThreeDHighlight

The color of the lighter (generally outer) of the two borders facing the light source for 3-D elements that appear 3-D due to two concentric layers of surrounding border.

ThreeDLightShadow

The color of the darker (generally inner) of the two borders facing the light source for 3-D elements that appear 3-D due to two concentric layers of surrounding border.

ThreeDShadow

The color of the lighter (generally inner) of the two borders away from the light source for 3-D elements that appear 3-D due to two concentric layers of surrounding border.

Window

Window background.

WindowFrame

Window frame.

WindowText

Text in windows.

For example, to set the foreground and background colors of a paragraph to the same foreground and background colors of the user's window, write the following:

```
p { color: WindowText; background-color: Window }
```

4.6. Notes on using colors

Although colors can add significant amounts of information to document and make them more readable, please consider the W3C Web Content Accessibility Guidelines [\[WCAG20\]](#) when including color in your documents.

- [1.4.1 Use of Color: Color is not used as the only visual means of conveying information, indicating an action, prompting a response, or distinguishing a visual element](#)

5. Simple alpha compositing

When drawing, implementations must handle alpha according to the rules in [Section 14.2 Simple alpha compositing](#) of [\[SVG11\]](#). (If the ‘color-interpolation’ or ‘color-rendering’ properties mentioned in that section are not implemented or do not apply, implementations must act as though they have their initial values.)

6. Sample style sheet for (X)HTML

This appendix is informative, not normative. This style sheet could be used by an implementation as part of its default styling of HTML4, XHTML1, XHTML1.1, XHTML Basic, and other XHTML Family documents.

```
html {  
    color: black;  
    background: white;  
}
```

```
/* traditional desktop user agent colors for hyperlinks */  
:link    { color: blue; }  
:visited { color: purple; }  
  
/* default focus outline */
```

```
:focus {
    outline: 1px dotted; /* or 1px dotted invert */
}
```

7. Profiles

Each specification using CSS3 Color must define the subset of CSS3 Color features it allows and excludes, and describe the local meaning of all the components of that subset.

Non normative examples:

CSS3 Color profile

Specification	HTML4
Accepts	Basic color keywords RGB six digit hex color values ‘color’ property ‘opacity’ property RGB three digit hex color values and RGB functional notation color value
Excludes	RGBA color values HSL and HSLA color values Extended color keywords ‘currentColor’ color value CSS2 UI Colors ‘transparent’ color value
Extra constraints	none.

CSS3 Color profile

Specification	CSS level 1 ‘color’ property
Accepts	Basic color keywords RGB color values ‘opacity’ property RGBA color values HSL and HSLA color values
Excludes	Extended color keywords ‘currentColor’ color value CSS2 UI Colors

	‘transparent’ color value
Extra constraints	none.
CSS3 Color profile	
Specification	CSS level 2
	‘color’ property
	Basic color keywords
Accepts	RGB color values
	CSS2 UI Colors
	‘transparent’ color value
	‘opacity’ property
	RGBA color values
Excludes	HSL and HSLA color values
	Extended color keywords
	‘currentColor’ color value
	‘transparent’ color value not valid for
Extra constraints	‘color’ property.
	‘orange’ color value (part of Extended color keywords) is accepted in CSS level 2 revision 1
CSS3 Color profile	
Specification	SVG 1.0 and 1.1
	‘color’ property
	‘opacity’ property
	Basic color keywords
Accepts	RGB color values
	CSS2 UI Colors
	Extended color keywords
	‘currentColor’ color value
	RGBA color values
Excludes	HSL and HSLA color values
	‘transparent’ color value
Extra constraints	‘currentColor’ color value not valid for
	‘color’ property.

8. Test suite

A CSS Color Module [Test Suite](#) has been developed, although further tests may be added. This test suite is intended to allow user agents to verify their basic conformance to the specification. This test suite does not pretend to be exhaustive and does not cover all possible numerical color values. These tests are available at https://test.csswg.org/harness/suite/css-color-3_dev/.

9. Future features

This section is non-normative.

A number of features related to color management, such as the ‘@color-profile’ at-rule for specifying ICC profiles, a means to control the rendering intent of those profiles, and a way to use those profiles (the ‘color-profile’ property) were removed from CSS Color 3 due to lack of implementation interest at the time.

Development of related features continued in CSS Color 4 and 5; see the ‘[color\(\)](#)’ function, the ‘[@color-profile](#)’ at-rule, and the ‘[rendering-intent](#)’ descriptor.

10. Acknowledgments

Thanks to Brad Pettit both for writing up color-profiles, and for implementing it. Thanks to Steven Pemberton for a write up on HSL colors. Thanks especially to the feedback from Marc Attinasi, Bert Bos, Joe Clark, fantasai, Patrick Garies, Tony Graham, Ian Hickson, Susan Lesch, Alex LeDonne, Cameron McCormack, Krzysztof Maczyński, Chris Moschini, Chris Murphy, Christoph Päper, David Perrell, Jacob Refstrup, Dave Singer, Jonathan Stanley, Andrew Thompson, Russ Weakley, Etan Wexler, David Woolley, Boris Zbarsky, Steve Zilles, the XSL FO subgroup of the XSL working group, and all the rest of the [www-style](#) community. And thanks to Chris Lilley for being the resident CSS Color expert.

11. Changes

Changes since the [05 August 2021 Recommendation](#)

The following editorial corrections were made:

- Sample HSL values & sample conversion code.
- Clarified case-insensitive is ASCII-case-insensitive.
- Removed "Media" from property definition tables
- Replaced "Call for implementation of dropped features" with "Future features"

Changes since the [19 June 2018 Recommendation](#)

- Candidate Correction 1: Sample HSL values & conversion code.
- Candidate Correction 2: Clarified case-insensitive.
- Candidate Correction 3: Removed "Media" from property definition tables
- Candidate Correction 4: Replaced "Call for implementation of dropped features"

Changes since the [5 December 2017 Candidate Recommendation](#)

- A non-substantive [typo](#) was corrected (stray semicolon)
- Removed trailing link to the no longer present `appearance` property
- Updated this changes section
- Date and boilerplate updates for Proposed Recommendation

Changes since the [2011 June 07 Recommendation](#)

This document differs from the [2011 June 07 Recommendation](#) in that it contains the following changes:

- Mark Introduction section as non-normative.
- Don't use SVG-specific terminology in the definition of opacity.
- Describe the effect of opacity on z-ordering more clearly and make it clear that "treated as" does not imply a change in the computed value.
- Make '[currentColor](#)' be a computed value and resolve at used value time, instead of being resolved in the computed value. ([test](#), [results](#)).
- Updated references to current versions

For changes in earlier drafts, see [changes in previous Recommendation](#) and previous versions linked therefrom.

Prior dispositions of comments:

- [Disposition of comments on the 2008 Last Call](#).
- [Disposition of comments on the 2003 Candidate Recommendation](#).
- [Disposition of comments on the 2003 Last Call](#).

12. References

12.1. Normative

[COLORIMETRY]

Colorimetry, Third Edition. CIE 15:2004. ISBN 978-3-901906-33-6

[CSS21]

Bert Bos; et al. *Cascading Style Sheets Level 2 Revision 1 (CSS 2.1) Specification*. 7 June 2011. W3C Recommendation. URL: <http://www.w3.org/TR/2011/REC-CSS2-20110607>

[SRGB]

Multimedia systems and equipment - Colour measurement and management - Part 2-1: Colour management - Default RGB colour space - sRGB. IEC 61966-2-1 (1999-10) ISBN: 2-8318-4989-6 - ICS codes: 33.160.60, 37.080 - TC 100 - 51 pp. as amended by Amendment A1:2003. URL: <https://webstore.iec.ch/publication/6168>

[SVG11]

Erik Dahlström; et al. *Scalable Vector Graphics (SVG) 1.1 (Second Edition)*. 16 August 2011. W3C Recommendation. URL: <http://www.w3.org/TR/2011/REC-SVG11-20110816/>

12.2. Informative

[CSS-UI-3]

Tantek Çelik; Florian Rivoal. *CSS Basic User Interface Module Level 3 (CSS3 UI)*. 21 June 2018. W3C Recommendation. URL: <https://www.w3.org/TR/2018/REC-css-ui-3-20180621/>

[CSS2]

Bert Bos; et al. *Cascading Style Sheets, level 2 (CSS2) Specification*. 07 June 2011. W3C Recommendation. URL: <http://www.w3.org/TR/2011/REC-CSS2-20110607>

[HTML401]

Dave Raggett; Arnaud Le Hors; Ian Jacobs. *HTML 4.01 Specification*. 24 December 1999. W3C Recommendation. URL: <http://www.w3.org/TR/1999/REC-html401-19991224>

[OEB101]

Open eBook(tm) Publication Structure 1.0.1. Open eBook Forum(tm). 02 July 2001. URL: <https://www.loc.gov/preservation/digital/formats/fdd/fdd000054.shtml>

[SVG10]

Jon Ferraiolo. *Scalable Vector Graphics (SVG) 1.0 Specification*. 4 September 2001. W3C Recommendation. URL: <http://www.w3.org/TR/2001/REC-SVG-20010904>

[WCAG20]

Ben Caldwell; et al. *Web Content Accessibility Guidelines (WCAG) 2.0*. 11 December 2008. W3C Recommendation. URL: <http://www.w3.org/TR/2008/REC-WCAG20-20081211/>

[X11COLORS]

Wikipedia *X11 color names*. URL: https://en.wikipedia.org/wiki/X11_color_names

[XML10]

C. M. Sperberg-McQueen; et al. *Extensible Markup Language (XML) 1.0 (Fifth Edition)*. 26 November 2008. W3C Recommendation. URL: <http://www.w3.org/TR/2008/REC-xml-20081126/>

Index

- `ActiveBorder`, [4.5.1](#).
- `ActiveCaption`, [4.5.1](#).
- `aliceblue`, [4.3](#).
- `<alphavalue>`, [3.2.](#), [4.2.2.](#), [4.2.5](#).
- `antiquewhite`, [4.3](#).
- `AppWorkspace`, [4.5.1](#).
- `aqua`, [4.1.](#), [4.3](#).
- `aquamarine`, [4.3](#).
- `azure`, [4.3](#).
- `Background`, [4.5.1](#).
- `beige`, [4.3](#).
- `bisque`, [4.3](#).
- `black`, [4.1.](#), [4.3](#).
- `blanchedalmond`, [4.3](#).
- `blue`, [4.1.](#), [4.3](#).
- `blueviolet`, [4.3](#).
- `brown`, [4.3](#).
- `burlywood`, [4.3](#).
- `ButtonFace`, [4.5.1](#).
- `ButtonHighlight`, [4.5.1](#).
- `ButtonShadow`, [4.5.1](#).
- `ButtonText`, [4.5.1](#).
- `cadetblue`, [4.3](#).
- `CaptionText`, [4.5.1](#).
- `chartreuse`, [4.3](#).

- chocolate, [4.3](#).
- color, [3.1](#).
- <color>, [4](#).
- color:<color>, [3.1](#).
- color-interpolation, [5](#).
- color-rendering, [5](#).
- compositing, [5](#).
- coral, [4.3](#).
- cornflowerblue, [4.3](#).
- cornsilk, [4.3](#).
- crimson, [4.3](#).
- currentColor, [4.4](#), [3.1](#), [3.1](#).
- cyan, [4.3](#).
- darkblue, [4.3](#).
- darkcyan, [4.3](#).
- darkgoldenrod, [4.3](#).
- darkgray, [4.3](#).
- darkgreen, [4.3](#).
- darkgrey, [4.3](#).
- darkkhaki, [4.3](#).
- darkmagenta, [4.3](#).
- darkolivegreen, [4.3](#).
- darkorange, [4.3](#).
- darkorchid, [4.3](#).
- darkred, [4.3](#).
- darksalmon, [4.3](#).
- darkseagreen, [4.3](#).
- darkslateblue, [4.3](#).
- darkslategray, [4.3](#).
- darkslategrey, [4.3](#).
- darkturquoise, [4.3](#).

- darkviolet, [4.3](#).
- deeppink, [4.3](#).
- deepskyblue, [4.3](#).
- dimgray, [4.3](#).
- dimgrey, [4.3](#).
- dodgerblue, [4.3](#).
- firebrick, [4.3](#).
- floralwhite, [4.3](#).
- forestgreen, [4.3](#).
- fuchsia, [4.1](#), [4.3](#).
- gainsboro, [4.3](#).
- ghostwhite, [4.3](#).
- gold, [4.3](#).
- goldenrod, [4.3](#).
- gray, [4.1](#), [4.3](#).
- GrayText, [4.5.1](#).
- green, [4.1](#), [4.3](#).
- greenyellow, [4.3](#).
- grey, [4.3](#).
- Highlight, [4.5.1](#).
- HighlightText, [4.5.1](#).
- honeydew, [4.3](#).
- hotpink, [4.3](#).
- hsl(), [4.2.4](#).
- hsla(), [4.2.5](#).
- InactiveBorder, [4.5.1](#).
- InactiveCaption, [4.5.1](#).
- InactiveCaptionText, [4.5.1](#).
- indianred, [4.3](#).
- indigo, [4.3](#).
- InfoBackground, [4.5.1](#).

- InfoText, [4.5.1](#).
- ivory, [4.3](#).
- khaki, [4.3](#).
- lavender, [4.3](#).
- lavenderblush, [4.3](#).
- lawngreen, [4.3](#).
- lemonchiffon, [4.3](#).
- lightblue, [4.3](#).
- lightcoral, [4.3](#).
- lightcyan, [4.3](#).
- lightgoldenrodyellow, [4.3](#).
- lightgray, [4.3](#).
- lightgreen, [4.3](#).
- lightgrey, [4.3](#).
- lightpink, [4.3](#).
- lightsalmon, [4.3](#).
- lightseagreen, [4.3](#).
- lightskyblue, [4.3](#).
- lightslategray, [4.3](#).
- lightslategrey, [4.3](#).
- lightsteelblue, [4.3](#).
- lightyellow, [4.3](#).
- lime, [4.1](#), [4.3](#).
- limegreen, [4.3](#).
- linen, [4.3](#).
- magenta, [4.3](#).
- maroon, [4.1](#), [4.3](#).
- mediumaquamarine, [4.3](#).
- mediumblue, [4.3](#).
- mediumorchid, [4.3](#).
- mediumpurple, [4.3](#).

- `mediumseagreen`, [4.3](#).
- `mediumslateblue`, [4.3](#).
- `mediumspringgreen`, [4.3](#).
- `mediumturquoise`, [4.3](#).
- `mediumvioletred`, [4.3](#).
- `Menu`, [4.5.1](#).
- `MenuText`, [4.5.1](#).
- `midnightblue`, [4.3](#).
- `mintcream`, [4.3](#).
- `mistyrose`, [4.3](#).
- `moccasin`, [4.3](#).
- `navajowhite`, [4.3](#).
- `navy`, [4.1](#), [4.3](#).
- `oldlace`, [4.3](#).
- `olive`, [4.1](#), [4.3](#).
- `olivedrab`, [4.3](#).
- `opacity`, [3.2](#).
- `orange`, [4.3](#).
- `orangered`, [4.3](#).
- `orchid`, [4.3](#).
- `palegoldenrod`, [4.3](#).
- `palegreen`, [4.3](#).
- `paleturquoise`, [4.3](#).
- `palevioletred`, [4.3](#).
- `papayawhip`, [4.3](#).
- `peachpuff`, [4.3](#).
- `peru`, [4.3](#).
- `pink`, [4.3](#).
- `plum`, [4.3](#).
- `powderblue`, [4.3](#).
- `purple`, [4.1](#), [4.3](#).

- red, [4.1.](#), [4.3.](#)
- #rgb, [4.2.1.](#)
- rgb(), [4.2.1.](#)
- rgba(), [4.2.2.](#)
- rosybrown, [4.3.](#)
- royalblue, [4.3.](#)
- #rrggbb, [4.2.1.](#)
- saddlebrown, [4.3.](#)
- salmon, [4.3.](#)
- sandybrown, [4.3.](#)
- Scrollbar, [4.5.1.](#)
- seagreen, [4.3.](#)
- seashell, [4.3.](#)
- sienna, [4.3.](#)
- silver, [4.1.](#), [4.3.](#)
- skyblue, [4.3.](#)
- slateblue, [4.3.](#)
- slategray, [4.3.](#)
- slategrey, [4.3.](#)
- snow, [4.3.](#)
- springgreen, [4.3.](#)
- sRGB, [4.2.1.](#), [4.1.](#)
- steelblue, [4.3.](#)
- system colors, [4.5.1.](#)
- tan, [4.3.](#)
- teal, [4.1.](#), [4.3.](#)
- thistle, [4.3.](#)
- ThreeDDarkShadow, [4.5.1.](#)
- ThreeDFace, [4.5.1.](#)
- ThreeDHighlight, [4.5.1.](#)
- ThreeDLightShadow, [4.5.1.](#)

- ThreeDShadow, [4.5.1](#).
- tomato, [4.3](#).
- transparent, [4.2.3](#), [3.1](#).
- turquoise, [4.3](#).
- violet, [4.3](#).
- wheat, [4.3](#).
- white, [4.1](#), [4.3](#).
- whitesmoke, [4.3](#).
- Window, [4.5.1](#).
- WindowFrame, [4.5.1](#).
- WindowText, [4.5.1](#).
- yellow, [4.1](#), [4.3](#).
- yellowgreen, [4.3](#).

Property index

Property Values	Initial	Applies to	Inh.	Percentages
<u>color</u> <color> inherit	depends on user agent	yes		N/A
<u>opacity</u> <alphavalue> inherit 1		no		N/A

