

CSS Data Types

One of Those Things You Have To Know

R. Scott Granneman & Jans Carton

© 2013 R. Scott Granneman

Last updated 2020-07-26

You are free to use this work, with certain restrictions.

For full licensing information, please see the last slide/page.

Notes & URLs for this presentation can be found...

- » underneath the link to this slide show on granneman.com
- » at files.granneman.com/presentations/webdev/CSS-Data-Types.txt

Defined

What's a *data type*?

A way to classify various kinds of data that are allowed as values for CSS properties

Text: `<string>` & `<url>`

Distance: `<length>`

Images: `<color>`

Basic Data Types

<string>

<url>

<length>

<color>

`<string>`

Represents a quoted string of Unicode characters

Covered earlier in *CSS Selectors*

<url>

Represents a *pointer to a web resource* (e.g., image or font)

Was <url>*, then <uri>**, but now back to <url>***
(to stay, hopefully!)

* CSS 1

** CSS 2.1

*** CSS Values and Units Module 3

`<url>` expressed in 2 ways:

- » absolute or relative URL
- » data URL

Absolute	<code>http://baz.com/foo.png</code>	Complete URL
Site root- relative	<code>/foo.htm</code> <code>/bar/foo.png</code>	Web server fills in protocol & domain
Document- relative	<code>foo.png</code>	Same directory
	<code>bar/foo.png</code>	Down into sub-directory
	<code>../bar/foo.png</code>	Up into parent directory

Data URL

Resource inlining: instead of referencing an external image (via an absolute or relative URL), you instead embed the images directly into the style sheet

```
background-image: url(data:image/gif;base64,  
R0lGODlhEAAQAMQAAORHHOVSKu...)
```

Why? To save HTTP requests to the web server

The downside: it makes your style sheets bigger



data:image/gif;base64,R0lGODlhMAAwAPf/APbVbfbQcv769vjek/rqtumVI++tWvnkofC0ZvTKZem4l/LBQt3d3deVc9WOZ/nhmeqaJfzxyvjciPPz8/358fbWgezJtuymLv79/frmpNVsEsNcKfTQW/PMqOmVHfTNbO2kSky9LS0u2tNvXh1Nt4Fu7u7vbYgvHx8frorPG6PfG7Me6wLf301fTNUtir1vj4+NeHVvnmquWJGeugJOB8F+qSMvPFX/G9ReykIu+5Zf766uqaHvrkneqdI+uZNuzs7PG7dPnhnfX19erq6v312NXV1fLBTeiLG+ymPvXVtdhzFNl2JeSYVe2rNMrKyvrrvdJmEvfaJougIOOGGfrqu8JQEuyiP+uhKPvt3fbTdf323e+1Nfvs1fniwe2lJOubIsRSD+mTHPnevvnogfjdj/fp4vT09Pvp1efn5+Tk5PTJTuli4vz38vbUeeWGItnZ2fvo0Pngvs5gEfjhl+bm5u2uL+Dg4Nvb2/vsuueGGOiQHfzy0f316eGDGPzuweylLcxeEfbUgeucJMNWHOqbIPvsv/bTY//9+vfxcumXH+iYHvjdjP317P305fvz7u2rMuynL/rps+yeQfjJjeeVHe6pMu+yMe6tJ/ror/njpffadPCyL+mYJOqZIdZvE+2nI+2pJPbUe+yiIeuch+qYHfjbh+6sJPbYgfvqsfv7++ukK/TNa+mQG+60POGtkuuiIdR5RPzwx8pcD83NzfTLj+yjJu6vJe2vTvbPnt2hgvjej/TKlvLFfv3y5/334u6xOPjdlvLYyfTNbvrpsfvqsvjdwvzyzvPd0fC4N+vp5/fUdOmZKuykLOKhd/HSu/bavfngoPXRePbRftuRYOCNRvH0tvbRc/fZe+Szm/TMWPG0OPfXeeiLKspiJdF0O+qdMOJ/G9LBueSCHvTHXfLCWOnn5fHFnvrvw6868tPbPcfjn2vrn08dcHPbQcN+IPumRIOqRHuqSH/TIYu/AleqgNeOogOu9n8hWD++3OPG/eu+0M/nkmOaKKe6pM////yH5BAEAAAP8ALAAAAAAwADAAAj/AP8JHEiwoEGBjhwdXMiwocN/vXo9nEjRYBcnTrpU3DhRxwIcOjiKXOjFDh1GkLyMXCkQEa4lRVocwYWB5cgKdjJQoMDPTgWbHCkk0bZlx44WLpIIAFqx1qUUR4vsyMCiFtOJjsAdKrIlQoQtLTjU63PVYRAuqX5FkCUrwi8yl/SVZdgFC4AifAzpNcSnxSFmGucaNJCmwJE8eQgQQFwkRTIEggvKsbWJz59UmFMVK/bnGABbciJjiFPhigpiEVKoXq06tYorFdDUXCkgRK4gIOAN+rICW4RUGYILD55K1qYVXYDAAXEkV4ilDfuM4YVgEjwwXzgtcAGA0oHFPcKL/x8PhcABSgBcLOD0BQy8SQh4eWk0cIyBH/Kw79ueqMyBTFBAUQUBKfTwwAN0JEjHgQj2kAkBAVaRwnmJqHdJe/H8gMAYAoCQwxoAMNKDDIpJMqEQA+yySxkstujii2UE00MGKUiiimAwPMALAGjkYsBQFk0TCTQo4DsCIBKYkaYoE SCq5JJNJQgnlk0wyMsADRC4TCQgUDNTID5a4oUkwJ5SJCiplmnkmmmm2meaZbp4QjCZaWDIJfQT5YoMTWkQjhSiAiuKGG4EGSiiggyJaqKCBShHNNZbY4MtBWXTjxDXBCHLNppx2ek0AmwYgqqekChKMO/7YkAVD7LzhhDpSSP/zway01srKrKzkemuttUojxQf+9IOGQyGMY8k8ggyTwLLMNuvss8wOMw05kYwTwkTQhAPLLVzkowIOR5Rzw7jkl1tuOUfgoAIXl9wyRTjGVKREdbCcYu+9p2DCwr4sbMMFvyxggu+9U9SgBEcdzABLKAw37PDDEDcMCxUdjEQPFaBkrPHGHHesMRXnsNREJTMUbPLJKKecQyVN2PTOIqPELPPMNNc8igfv2MREIVP07PPPQAc9BQ9MsNSGBqQkrfTSTDetdBSIrESCBjz4wMPVWGetddaFWK3BoitZ80khNPBQytlon2122mmTzcMnz6ykwBKeYKGI B3jnrYgPWGD/McgieeuNhSef3LOSMYWasYoYjDNe9yqrDOID5IM3LoYHq4BRgj0rxVADFoC0IronzURywSbi+OGHGINcEEkznYjeygVY1EDNSt/4cUEze0AAiROWQLBHCXPEEsscJfT+e/B7XHCBH7Gs1M4MI1QPjBMQILHELDEII5AwMcyBBIQOAFM9SNQ4c1Ij4SBhCuuJNDJDBpY4QAJBpHggBWfzNDJCPBDghWgsxFk4AMeIyiAH6JghQaYoSFmaIAVouCHAoxgD/jAH0cseAg91CAKhNDFIyiCDl0QIgo10MMsLCCSV8whEBt4RRtE0oZXbGAOc8hGQzCgChjAYAhnmAAKjRywgRcowwRIRMEehjAEHzqRiUJEIhCIQEVzvGADDUCBEpsIA1XUpIdCBEIa1MAAPIjDCCJIowiMgIc7qIEISiwjEdLABgbAQQS0eIIen0ALEaQDDgxgQx2AoMQu8hAGYSSCGthwBwY4Eg+OZEMA4KjECVjSBGJcPB2NwEkjwAGQbkWDEEwwAUNGZiQBAQA7

base64

Encodes binary file into text

Convert any image format: JPEG, GIF, PNG, SVG

On macOS & Linux

```
$ base64 star.gif
```

On Windows

```
Base64.exe -e star.gif
```

(see support.microsoft.com/kb/191239)

[address, find IP address of a domain name](#)

[Convert IP address to different formats](#)

[Convert Unicode characters to HTML code numbers and vice versa](#)

[Convert Unicode characters to Unicode escape sequences and vice versa](#)

[Coordinate converter and show map](#)

[Create self-signed SSL certificates online](#)

[CSV to XML converter](#)

[CVS pserver password decoder and encoder](#)

[Decode Certificate Signing Request \(CSR\)](#)

[Decode SSL certificate](#)

[Electronic business card vCard generator](#)

[European clothing standard EN 13402 pictogram generator](#)

[Favicon generator](#)

[Find the BIC numbers for Dutch IBAN numbers](#)

[Free game sound effects](#)

[Free game textures](#)

[Free online practice exams](#)

[Free online SEPA XMI](#)

Input base64 encoder and decoder:

Enter source data *:



-- Or --

Upload source file *:
Max 100 KB.



Choose File no file selected

Conversion method *:



Encode to Base64 string

Select output *:



Output in textbox

Max characters per line *:



75 Allowed values 0 - 999. Use 0 for unlimited characters.

To prevent automated submissions an Access Code has been implemented for this tool.

Please enter the Access Code as displayed above*:

jH8

*** = required**

Convert

Clear

Output base64 encoder and decoder:

Select all

Clear

Used in many CSS properties, such as `background-image`, `cursor`, & `list-style`

<length>

Represents *distance measurements*: a <number> immediately followed by a unit

Units

- » Absolute
- » Relative (viewport-percentage or font-relative)

`<length>` is used with properties like...

- » `width`
- » `height`
- » `margin`
- » `padding`
- » `border-width`
- » `font-size`
- » `text-shadow`

Note: `<percentage>` is not a `<length>`, even though some of these properties also accept it

Absolute <length>

px

cm

mm

Q

in

pc

pt

Of these, the foundation is **px**, because every **<length>** is eventually calculated as pixels

*Pixel means
picture element*

Pixels are not
just for computers!













Georges Seurat's *A Sunday Afternoon on the Island of La Grande Jatte* (1884)



Resolution is a measurement of the amount of effort it takes to see (or *resolve*) individual pixels

The Roman mosaic is low resolution

The Seurat painting is higher resolution

An iPhone is very high resolution

The basic unit of an electronic display is a *pixel*

All length measurements eventually get counted in
pixels

For years, computer pixels were all $1/72$ of an inch

Apple's iPhone 4 introduced the first high resolution devices that have much smaller pixels (300+ per inch — & my 2019 iPhone 11 Pro Max has 458 ppi)

The size of pixels in modern high resolution devices varies hugely

Thanks to these 2 changes, the size of a pixel in CSS had to be redefined

For good old “standard resolution” displays, a pixel is (still a device pixel, which is) $1/72$ inch

For high resolution displays (& other high resolution media like print) a CSS pixel is now about $1/96$ inch

So now it may take several device pixels to equal 1 CSS pixel

px 96 pixels per inch

cm 1 centimeter = $96\text{px}/2.54$

mm 1 millimeter = 1/10 of 1cm

Q 1 quarter-millimeter = 1/40 of 1cm

in 1 inch = 2.54 cm = 96px

pc 1 pica = 1/6 of 1in = 12 pt

pt 1 point = 1/72 of 1in

Relative <length>

Relative lengths are measurements derived from some other distance

Depending on the unit, this can be based on...

- » size of a specific character
- » line height
- » viewport size

Units are either *viewport-percentage* or *font-relative*

Font-Relative *<length>*

ch

ex

em & rem

These values are relative to the size of the font

Of these, the most common you will see are `em` & `rem`

We will cover those & more in *CSS Typography*

em

Represents the *calculated font-size* of an element

1 em is the height of the current font

rem

Represents the font-size set on the *root element*
(`<html>`)

							
ch	9 [*]	12	2	7	7	27	4.4
ex	6	12	2	3.1	3.2	4	2.1
em	Y	Y	Y	Y	Y	Y	Y
rem	9 [†] 11	12	3.6	5	4 [‡]	4	2.1

* Width of **ch** is that of the 0 glyph, not its surrounding space

† Doesn't work with **font** or pseudo elements

‡ iOS Safari 5.0-5.1 doesn't support **rem** with media queries

Viewport Percentage *<length>*

vh 1% of the viewport's height

vw 1% of the viewport's width

vmin 1% of **vh** or **vw**—whichever is smaller

vmax 1% of **vh** or **vw**—whichever is larger

vi 1% of the initial containing block, in the direction of the root element's *inline* axis

vb 1% of the initial containing block, in the direction of the root element's *block* axis



					ios		
vh	10	12	19	6.1	8	26	4.4
vw	10	12	19	6.1	6.1	26	4.4
vmin	10	12	19	6.1	6.1	26	4.4
vmax	—	—	19	6.1	8	26	4.4
vi	—	—	—	—	—	—	—
vb	—	—	—	—	—	—	—

calc()

`calc()`

CSS function that lets you perform calculations when specifying values for CSS properties

Can be used anywhere you could use a `<length>`, `<frequency>`, `<angle>`, `<time>`, `<percentage>`, `<number>`, or `<integer>`

Most commonly used with `<length>`

Can use $+$, $-$, $\times$, $\&$ $/$ operators

- » $\times$ $\&$ $/$ require at least 1 `<number>`
- » $/$ requires that `<number>` is on the right
- » $+$ $\&$ $-$ must be surrounded by space ($\times$ $\&$ $/$ do not require space, but use it for consistency)
- » $/$ by 0 gives an error
- » Be careful using `calc()` with `<percentage>`s for `width` $\&$ `height` on tables

⚙ HTML

1 <p>
2 Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!
3 </p>

Ph'nglui mglw'nafh
Cthulhu R'lyeh
wgah'nagl fhtagn!

⚙ CSS (SCSS)

1 html {
2 font-size: calc(1em + 1vw)
3 }
4

⚙ JS

⚙ HTML

1

<p>

2

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!

3

</p>

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl
fhtagn!

⚙ CSS (SCSS)

1

html {

2

font-size: calc(1em + 1vw);

3

}

4

⚙ JS

⚙️ HTML

1 <p>
2 Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!
3 </p>

⚙️ CSS (SCSS)

1 html {
2 font-size: calc(1em + 1vw);
3 }
4

⚙️ JS

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!

⚙️ HTML

1

<p>

2

Ph'nglui mglw'nafh

Cthulhu R'lyeh

wgah'nagl fhtagn!

3

</p>

⚙️ CSS (SCSS)

1

html {

2

font-size: calc(1em

+ 1vw);

3

}

⚙️ JS

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!

HTML

1

<div></div>

CSS (SCSS)

1

div {

2

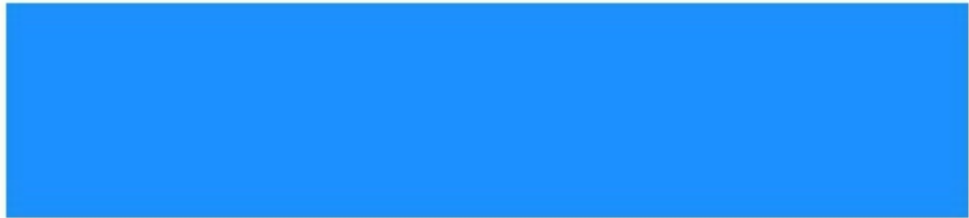
width: calc(100vw - 60px);

3

}

4

JS



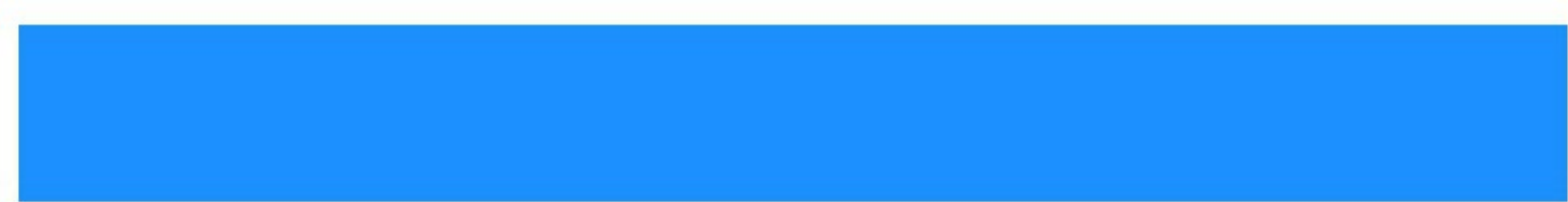
 HTML


```
1 <div></div>
```

 CSS (SCSS)


```
1 div {  
2   width: calc(100vw - 60px);  
3 }  
4
```

 JS



HTML

1

<div></div>

CSS (SCSS)

1

div {

2

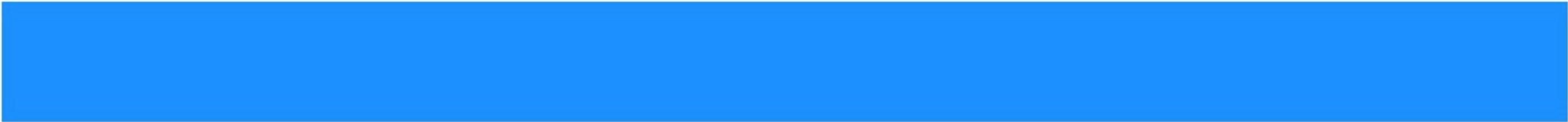
width: calc(100vw - 60px);

3

}

4

JS



					ios		
<code>calc()</code>	9 	12	16	6.1	7	26	67
<code><color></code> value	—	—	59	6	6	—	—
Nested <code>calc()</code>	—	16	48	11	11	51	Y
<code><number></code> value	9	12	48	6	6	31	4.4
<code><gradient></code> color stops	9	12	19	6	6	19	Y
<code>@media</code> expressions	—	79	59	12	12	66	Y

<color>

Represents a *color* in the sRGB color space

Descriptor	Value
Named colors	magenta
6-digit RGB hexadecimal	#RRGGBB
8-digit RGB hexadecimal	#RRGGBBAA
3-digit RGB hexadecimal	#RGB
4-digit RGB hexadecimal	#RGBA

Specifications			Color	Keyword	RGB cubic coordinates	Live Example
CSS3	CSS2	CSS1		black	=rgb(0, 0, 0)	
				silver	=rgb(192, 192, 192)	
				gray ^[*]	=rgb(128, 128, 128)	
				white	=rgb(255, 255, 255)	
				maroon	=rgb(128, 0, 0)	
				red	=rgb(255, 0, 0)	
				purple	=rgb(128, 0, 128)	
				fuchsia	=rgb(255, 0, 255)	
				green	=rgb(0, 128, 0)	
				lime	=rgb(0, 255, 0)	
				olive	=rgb(128, 128, 0)	
				yellow	=rgb(255, 255, 0)	
				navy	=rgb(0, 0, 128)	
				blue	=rgb(0, 0, 255)	
				teal	=rgb(0, 128, 128)	
				aqua	=rgb(0, 255, 255)	
				orange	=rgb(255, 165, 0)	

Color keywords

CSS 1: 16

CSS 2: 1

CSS 3: 128



Rebecca Meyer (2008–2014), namesake for [rebeccapurple](#)

Numeral system

Humans use *base 10*: 0–9

Computers use *base 2 (binary)*: 0 & 1

Colors can be expressed with *hexadecimal*: 0–F (0–9, then A–F)

Hexadecimal notation

#RRGGBB

00: no color at all

FF: full color

#000000: no red, no green, no blue, so black

#FFFFFF: full red, full green, full blue, so white

`#FF0000` (or `#ff0000`): full red, no green, no blue

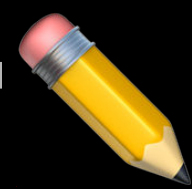
`#00FF00`: no red, full green, no blue

`#0000FF`: no red, no green, full blue

`#FFFFFF`: full red, full green, no blue

`#00FFFF`: no red, full green, full blue

`#E19427`: almost full red, a bit over halfway green, low blue



SIDE NOTE

Hexademical numbers can use either capital or lowercase letters, so these are exactly the same:

#0455BF

#0455bf

#F3F112

#f3f112

#000000

#111111

#222222

#333333

#444444

#555555

#666666

#777777

#888888

#999999

#AAAAAA

#BBBBBB

#CCCCCC

#DDDDDD

#EEEEEE

#FFFFFF

Gradations of gray

Hexadecimal notation may add a 4th pair of digits for transparency: the *alpha channel*

00 represents a fully transparent color & **FF** represent a fully opaque color

#0000FF66: 66 (40%) alpha channel

#F3F112AA: AA (67%) alpha channel

Hexadecimal values & their alpha channel (transparency) equivalents

- » 00 is 0% — completely transparent
- » 33 is 20%
- » 66 is 40%
- » 99 is 60%
- » CC is 80%
- » FF is 100% — completely opaque

3-digit hexadecimal

#RGB

If all three color components (the R, G, & B) are matching pairs, you can abbreviate the hexadecimal notation

#FF0000 → #F00

#33AA77 → #3A7

#BBFF33 → #BF3

#DDEEFF → #DEF

4-digit hexadecimal

#RGBA

Hexadecimal notation may add a 4th hexadecimal number for the alpha channel, if the alpha channel could be expressed by 2 matching numbers

#FF0000 → #F00

#33AA77 → #3A7

#BBFF33 → #BF3

#DDEEFF → #DEF

#FF000033 → #F003

#33AA7766 → #3A76

#BBFF3399 → #BF39

#DDEEFFCC → #DEFC

							
Named colors	3	12	1	1	1	1	1
rebeccapurple	11	12	33	9	8	38	Y
#RRGGBB	3	12	1	1	1	1	1
#RRGGBBAA	—	79	49	10	10	62	Y
#RGB	3	12	1	1	1	1	1
#RGBA	—	79	49	10	10	62	Y

Descriptor	Value
RGB <number>	<code>rgb(255, 0, 51)</code>
RGB <percentage>	<code>rgb(100%, 0%, 20%)</code>
RGB + Alpha	<code>rgba(255, 0, 51, 0.7)</code>
HSL	<code>hsl(348, 100%, 50%)</code>
HSL + Alpha	<code>hsla(348, 100%, 50%, 0.7)</code>

RGB function

`rgb(R,G,B)`

Expressed as either:

- » `<integer>`s between 0 (black) & 255 (white)
- » `<percentage>`s between 0 (black) & 100% (white)

May add a 4th place for alpha channel

`rgba(R, G, B, A)`

The alpha channel is a number between 0 & 1, where:

- » 0 is fully transparent
- » 1 is fully opaque

`rgb(255, 0, 0)`

`rgba(255, 0, 0, 0.5)`

`rgb(0, 255, 201)`

`rgba(0, 255, 201, 0.66)`

Hue

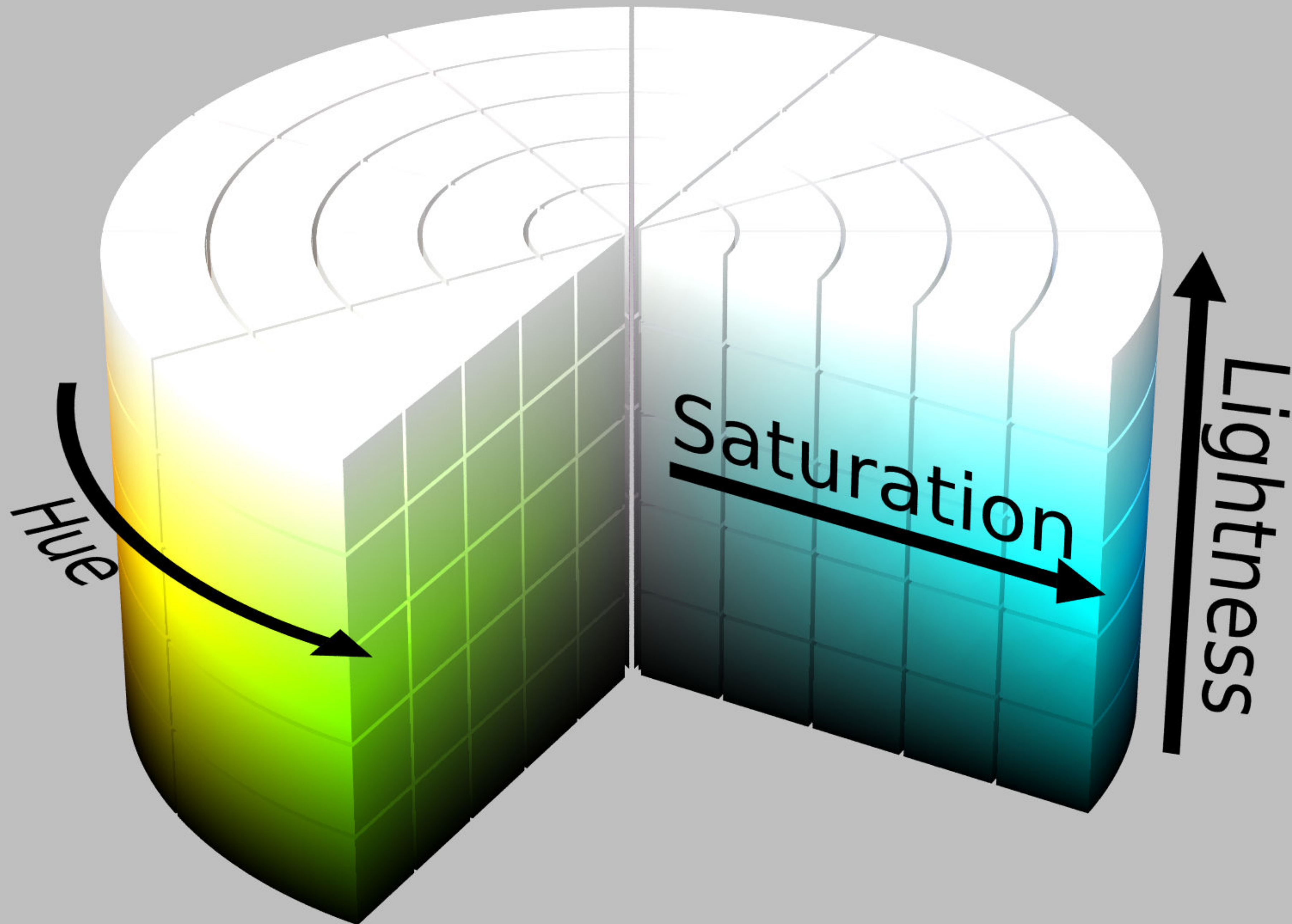
- » color
- » integer between 0 & 360

Saturation (AKA chroma)

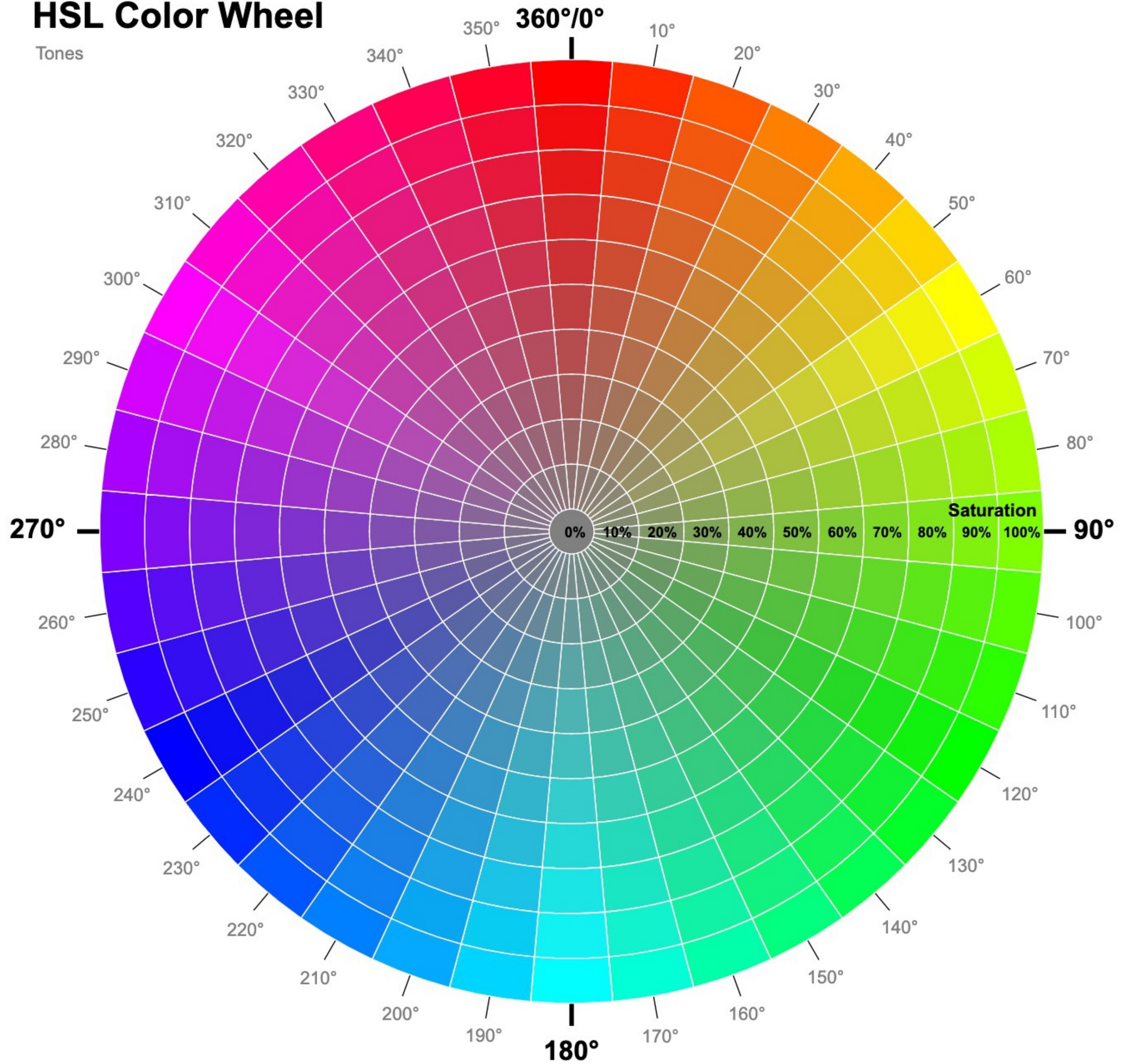
- » intensity or purity of a hue
- » percentage between 0 & 100%

Lightness

- » relative degree of black or white mixed with a hue
- » percentage between 0 & 100%



HSL Color Wheel



May add a 4th place for alpha channel

`hsla(H,S,L,A)`

The alpha channel is a number between 0 & 1, where:

- » 0 is fully transparent
- » 1 is fully opaque

`hsl(360,100%,100%)`

`hsla(360,100%,100%,0.5)`

`hsl(165,87%,100%)`

`hsla(165,87%,100%,0.66)`

					iOS		
rgb ()	4	12	1	1	1	1	1
rgba ()	9	12	3	3.1	5.1	1	2.3
hsl ()	9	12	1	3.1	5.1	1	2.3
hsla ()	9	12	3	3.1	5.1	1	2.3

Descriptor	Value
Special keywords	transparent
	currentColor

transparent

Fully transparent

currentcolor

Value of the *selected element's font* *color*, allowing you to use the color value on properties that do not receive it by default

 HTML

1

```
<p>Ph'nglui mglw'nafh Cthulhu R'lyeh  
wgah'nagl fhtagn!</p>
```

 CSS

1

```
p {  
2   color: green;  
3   border-color: currentcolor;  
4 }
```

 JS

Ph'nglui mglw'nafh Cthulhu R'lyeh wgah'nagl fhtagn!

					ios		
transparent	9	12	3	3.1	3.2	1	Y
currentcolor	9	12	1.5	4	3.2	1	Y

More Info

CSS Values and Units Module Level 3

Editor's Draft, 24 September 2018



► Specification Metadata

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

Abstract

This CSS module describes the common values and units that CSS properties accept and the syntax used for describing them in CSS property definitions.

CSS is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, etc.

Status of this document

This is a public copy of the editors' draft. It is provided for discussion only and may change at any moment. Its publication here does not imply endorsement of its contents by W3C. Don't cite this document other than as work in progress.

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

CSS Values and Units Module Level 4

Editor's Draft, 5 July 2020



► Specification Metadata

Copyright © 2020 [W3C](#)® ([MIT](#), [ERCIM](#), [Keio](#), [Beihang](#)). W3C [liability](#), [trademark](#) and [permissive document license](#) rules apply.

Abstract

This CSS module describes the common values and units that CSS properties accept and the syntax used for describing them in CSS property definitions.

[CSS](#) is a language for describing the rendering of structured documents (such as HTML and XML) on screen, on paper, etc.

Status of this document

This is a public copy of the editors' draft. It is provided for discussion only and may change at any moment. Its publication here does not imply endorsement of its contents by W3C. Don't cite this document other than as work in progress.

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

Thank you!

scott@granneman.com

www.granneman.com

ChainsawOnATireSwing.com

@scottgranneman

jans@websanity.com

websanity.com

CSS Data Types

One of Those Things You Just Have To Know

R. Scott Granneman & Jans Carton

© 2014 R. Scott Granneman

Last updated 2020-07-26

You are free to use this work, with certain restrictions.

For full licensing information, please see the last slide/page.

Changelog

2020-07-26 2.3: Added HSL Color Wheel screenshot; added gradients behind alpha channel colors in RGB & HSL; added slide showing Hex gradations of gray

2020-07-17 2.2: Removed URLs in slides, as they are in the Notes; updated `calc()` screenshots; added iPhone 11 Pro Max to retina display

Changelog

2020-07-13 2.1: Added latest screenshot of CSS Values and Units Module Level 4; improved screenshots for siblings & parents; divided `<color>` into sub-sections for easier understanding; added details re: `#RRGGBBAA` & `#RGBA`; added `rebeccapurple`; added compatibility charts for *Font-Relative* `<length>`; updated table for *Absolute* `<length>`; updated compatibility chart for `rgb()` & `hsl()`; added example for `currentcolor`; added `calc()` explanations & examples

Changelog

2020-07-02 2.0: Hid `<ident>` & `<custom-ident>` as they're really about how to read the spec; hid `<integer>`, `<number>`, & `<percentage>` because they're too obvious; moved `<gradient>`, `<image>`, `<shape>/<basic-shape>`, `<filter-function>`, & `<blend-mode>` to *CSS Effects*; moved `<ratio>` & `<resolution>` to Media Types & Media Features; moved `<time>`, `<timing-function>`, `<transform-function>`, & `<angle>` to CSS Effects; moved `<frequency>` to *Accessibility in For Developers Only*; bumped up to 2.0 because of so many big changes

Changelog

2018-10-24 1.5: Removed **e** notation as being wrong; added More Info chapter

2018-09-02 1.4: Updated compatibility tables for **<blend-mode>**; updated theme to Granneman 1.4; fixed minor formatting issues; updated list of data types; reorganized everything; added more to **<url>**; added section on **<custom-ident>**

Changelog

2015-01-24 1.3: Added screenshots for `em` & `rem`

2014-10-20 1.2: Added `<integer>` to list

2014-08-06 1.1: Included browser support for `<color>`

Licensing of this work

This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License.

To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/>.

You are free to:

- » *Share* — copy and redistribute the material in any medium or format
- » *Adapt* — remix, transform, and build upon the material for any purpose, even commercially

Under the following terms:

Attribution. You must give appropriate credit, provide a link to the license, and indicate if changes were made. You may do so in any reasonable manner, but not in any way that suggests the licensor endorses you or your use. Give credit to:

Scott Granneman • www.granneman.com • scott@granneman.com

Share Alike. If you remix, transform, or build upon the material, you must distribute your contributions under the same license as the original.

No additional restrictions. You may not apply legal terms or technological measures that legally restrict others from doing anything the license permits.

Questions? Email scott@granneman.com