

Agenda

1 WHAT IS CSS

2 SYNTAX

3 ADDING STYLES TO THE PAGE

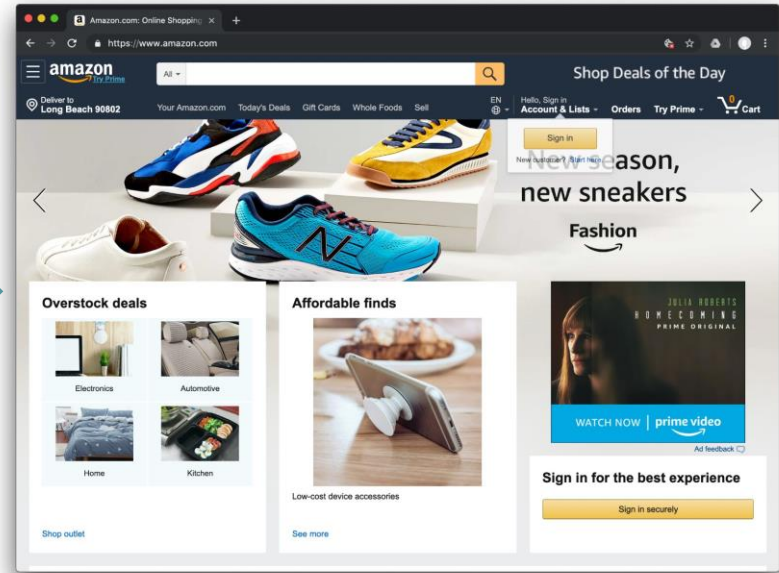
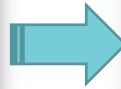
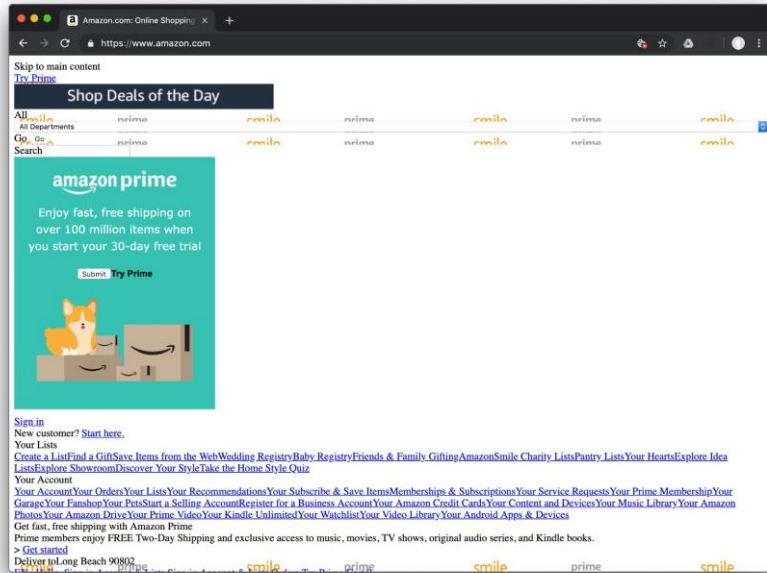
4 SELECTORS

5 COLORS

6 FONTS

```
.screen-reader-text:hover,  
.screen-reader-text:active,  
.screen-reader-text:focus {  
    background-color: #f1f1f1;  
    border-radius: 3px;  
    box-shadow: 0 0 2px 2px rgba(0,  
    clip: auto !important;  
    color: #21759b;  
    display: block;  
    font-size: 14px;  
    font-size: 0.875rem;  
    font-weight: bold;  
    height: auto;  
    left: 5px;  
    line-height: normal;  
    padding: 15px 23px 14px;  
    text-decoration: none;  
    top: 5px;  
    width: auto;  
    z-index: 100000; /* Above WP to  
}
```

Why CSS?



CSS definitions

[Cascading Style Sheets](#) (CSS) are a stylesheet language used to describe the presentation of a document written in HTML or XML (including XML dialects like SVG or XHTML).

CSS describes how elements should be displayed on screen, on paper, in speech, or on other media. CSS is the only document styling language that browsers understand.

- CSS has a [standardized](#) W3C specification.
- CSS1 is now obsolete,
- CSS2.1 is a recommendation,
- CSS3 is splitted into smaller modules, progressing on the standardization track.

The types of styles:

- a browser's style
- an author's style
- a user's styles

The basic syntax of CSS

```
p{  
  color: orange;  
}
```

selector

p

declaration

{ color:blue; }

↑
property

↑
value

The basic syntax of CSS

ORDER MATTERS

```
p {  
  color: green;  
}  
p {  
  color: lime;  
}
```

p color will be lime

COMMENTS IN CSS-FILE

```
/*  
  This is a comment  
  Which can span one  
  or multiple lines  
*/  
  
div {  
  width: 200px; /* Another comment here */  
}
```

Adding styles to the page - external styles

example.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Styles</title>
    <link rel="stylesheet" href="css/style.css" />
  </head>
  <body>
    <h1>Heading</h1>
    <p>Content</p>
  </body>
</html>
```

style.css

```
h1 {
  color: #000080;
  font-size: 200%;
  text-align: center;
}
p {
  padding: 20px;
  background: yellow;
}
```

Adding styles to the page - internal styles

```
example.html

<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Internal styles</title>
    <style>
      h1 {
        font-size: 120%;
        font-family: Verdana, Arial, Helvetica, sans-serif;
        color: #333366;
      }
    </style>
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```


Adding styles to the page - inline styles

example.html

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Inline styles</title>
  </head>
  <body>
    <h1 style="font-size: 120%; color: #cd66cc;">Hello, world!</h1>
  </body>
</html>
```

Importing styles

example.html

```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8" /> <title>Styles</title>
<link rel="stylesheet" href="css/style.css">
</head>
<body>
  <h1>Heading</h1>
  <p>Content</p>
</body>
</html>
```

style.css

```
@import "style-2.css"

h1 {
  color: #000080;
  font-size: 200%;
  text-align: center;
}
```

style-2.css

```
body {
  background: #fc0;
}

p {
  font-weight: bold;
}
```

Specify media type: @import

```
style.css

@import "style-screen.css" screen;
@import "style-print.css" print, speech;

h1 {
  color: #000080;
  font-size: 200%;
  text-align: center;
}
```

Media type using media queries

style.css

```
@media screen {  
  body {  
    font-family: Arial, Verdana, sans-serif;  
    color: #333;  
  }  
}  
  
@media print {  
  body {  
    font-family: Times, "Times New Roman", serif;  
    color: #000;  
  }  
}
```

Media type in HTML “media” attribute

example.html

```
<html>
  <head>
    <meta charset="utf-8" />
    <title>Devices</title>
    <link media="print, handheld" rel="stylesheet" href="print.css" />
    <link media="screen" rel="stylesheet" href="main.css" />
  </head>
  <body>
    <p> ... </p>
  </body>
</html>
```

Device types

Value	Description
All	Used for all media type devices
print	Used for printers
screen	Used for computer screens, tablets, smart-phones etc.
speech	Used for screen readers that "read" the page out loud

Deprecated values:

aural, braille, embossed, handheld, projection, tty, tv

How to add styles to the page

All described methods of using CSS can be used either alone or in combination with each other.

In the second case, is necessary to remember their hierarchy.

- Inline style - highest priority
- Internal style, external style - lower priority

Absolute Lengths

cm	centimeters
mm	millimeters
in	inches (1in = 2.54cm)
px	pixels
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

Relative Lengths

em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport
vh	Relative to 1% of the height of the viewport
vmin	Relative to 1% of viewport's smaller dimension
vmax	Relative to 1% of viewport's larger dimension

Property values

```
style.css

/* Strings: */
li:before {
  content: "Hello";
}

/* Numbers: */
p {
  font-weight: 600;
  line-height: 1.2;
}

/* URLs: */
a {
  background: url(warn.png) no-repeat;
}

/* Keywords: */
p {
  text-align: right;
}
```

Color:

- By **hexadecimal** values: #6609CF, #fc0
- By **name**: white, silver, black, lightblue, ...
- RGB: **rgb**(255, 0, 0)
- RGBA: **rgba**(0,255,0,0.3)
- HSL: **hsl**(120,100%, 25%)
- HSLA: **hsla**(120,100%, 50%, 0.3)

Basic selectors

```
style.css

/* ID - Selects the element with id="firstname" */
#firstname {
  width: 520px;
  padding: 100px;
  background: #fc0;
}

/* Class - Selects all elements with class="intro" */
.intro {
  font-size: 11px;
}

/* Type - Selects all <p> elements */
p {
  text-align: right;
  font-size: 1.5rem;
}

/* Universal - Selects all elements */
* {
  font-size: 11px;
}
```

- ID
 - Selects the element with `id="firstname"`
- Class
 - Selects all elements with `class="intro"`
- Type
 - Selects all `<p>` elements
- Universal
 - Selects `all` elements

Combinators

div, p

Groups of selectors

A comma-separated list of selectors represents the union of all elements selected by each of the individual selectors in the list

div p

Descendant combinator

Selects all <p> elements inside <div> elements

div > p

Child combinator

Selects all <p> elements where the parent is a <div> element

div + p

Adjacent sibling combinator

Selects all <p> elements that are placed immediately after <div> elements

p ~ ul

General sibling combinator

Selects every element that are preceded by a <p> element

Descendant Selectors

style.css

```
.main-nav {  
  margin: 0;  
  padding: 0;  
  list-style: none;  
}  
.main-nav li {  
  margin: 0 0 10px 0;  
  padding: 3px;  
  background: #fc0;  
}  
.main-nav a {  
  color: #000;  
}
```

example.html

```
<ul class="main-nav">  
  <li><a href="#">Home page</a></li>  
  <li><a href="#">About me</a></li>  
  <li><a href="#">Contacts</a></li>  
</ul>
```

Home page

About me

Contacts

Child combinator

style.css

```
ol > li {  
  color: rgb(0, 255, 0);  
}
```

example.html

```
<h2 class="headline">Title of something</h2>  
<p>Lorem ipsum dolor sit amet... </p>  
<ol>  
  <li>Lupus</li>  
  <li>Ursa</li>  
</ol>
```

Title of something

Lorem ipsum dolor sit amet, consectetur
adipiscing elit. In condimentum magna leo, sit
amet ultrices eros eleifend a. Aliquam maximus
feugiat posuere. Cras a ultrices urna.

1. Lupus

2. Ursa

Adjacent sibling combinator

style.css

```
.headline + p {  
  font-weight: bold;  
}
```

example.html

```
<h2 class="headline">Title of something</h2>  
<p>Lorem ipsum dolor sit amet... p</p>  
<p>Donec quis nibh vitae ... </p>
```

Title of something

Lorem ipsum dolor sit amet, consectetur adipiscing elit. In condimentum magna leo, sit amet ultrices

eros eleifend a. Aliquam maximus feugiat posuere. Cras a ultrices urna.

Donec quis nibh vitae tellus tristique euismod id sed purus. Cras quis lacinia sem. Nunc eget purus nec

nibh iaculis suscipit. Morbi quis nunc molestie, ullamcorper nulla at, vehicula ligula.

General combinator

style.css

```
.headline ~ P {  
  color: rgb(0, 255, 0);  
}
```

example.html

```
<p>A paragraph before the headline!</p>  
<p></p>  
<h2 class="headline">Title of something</h2>  
<aside>  
  <p>Lorem ipsum dolor sit amet... </p>  
  <p>In condimentum magna leo... </p>  
</aside>  
<p>Praesent ipsum ex, efficitur... </p>  
<p>Quisque molestie vulputate ex... </p>
```

A paragraph before the headline

Title of something

Praesent ipsum ex,
efficitur mollis eleifend a,
convallis a est.

Quisque molestie
vulputate ex, ac pharetra
metus tristique vel.
Vestibulum scelerisque
dui sed ipsum bibendum,
sed iaculis sem porta.

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. In
condimentum magna
leo.

In condimentum
magna leo, sit amet
ultrices eros eleifend a.
Aliquam maximus
feugiat posuere. Cras a
ultrices urna

Attribute presence and value selectors

[target]

Selects all elements with a "target" attribute (with any value)

[target=_blank]

Selects all elements with target="_blank" (exact value)

[title~="flower"]

Selects elements with a title attribute **containing the word** "flower"

[lang|="en"]

Selects all elements with a lang attribute value **starting with** "en"

Substring matching attribute selectors

`a[href*="w3c"]`

Selects <a> elements
with href attribute value
containing the substring
"w3c"

`a[href^="https"]`

Selects <a> elements
with href attribute value
starting with "https"

`a[href$=".pdf"]`

Selects <a> elements
with href attribute value
ending with ".pdf"

Attribute selectors

style.css

```
[title] {  
  color: maroon;  
}  
a[href] {  
  background: green;  
}  
a[target="_blank"] {  
  background: #ccc;  
  padding-left: 15px;  
}
```

example.html

```
<blockquote title="Some title text">  
  Lorem ipsum dolor sit amet, consectetur adipisicing elit  
</blockquote>  
<p>  
  <a href="#link">Link</a>  
</p>  
<p>  
  <a href="#link" target="_blank">The link will open in a new window</a>  
</p>
```

"Lorem ipsum dolor sit amet, consectetur adipisicing elit"

[Link](#)

[The link will open in a new window](#)

Advanced attributes selectors

```
style.css

/* Attribute value starts with some text */
a[href^="http://"]
{
    color: red;
}

/* If link ends with ".com" */
a[href$=".com"] {
    background: #fc0;
}

/* If link contains "google" */
[href*="google"] {
    background: yellow;
}

/* One of the several attribute values */
[title~="block"] {
    color: green;
}
```

```
example.html

<p>
  <a href="http://gmail.com" target="_blank"> External link to gmail.com</a>
</p>
<p>
  <a href="http://www.yahoo.com">Yahoo.com</a>
</p>
<p>
  <a href="http://google.com.ua">Search the web</a>
</p>
<h3 title="block tag">Heading</h3>
```

External link to gmail.com

Yahoo.com

Search the web

Heading

Pseudo-classes

The pseudo-class concept is introduced to **permit selection** based on information that lies outside of the document tree or **that cannot be expressed using the other simple selectors**.



```
a:hover {  
    border-bottom: 1px solid;  
    background: #cdfcaa;  
}
```


Dynamic pseudo-classes


Link


```
  
a:link {  
  color: #265301;  
}
```

```
  
a:visited {  
  color: #0000ff;  
}
```

User action


```
  
a:hover {  
  border-bottom: 1px solid;  
  background: #CDFEAA;  
}
```

```
  
a:active {  
  background: #265301;  
  color: #CDFEAA;  
}
```


```
  
a:focus {  
  border-bottom: 1px solid;  
  background: #BAE498;  
}
```

Other pseudo-classes


UI element



```
input:enabled {  
  background: #ccc;  
}
```




```
input:disabled {  
  background: #ccc;  
}
```




```
:checked {  
  margin-left: 25px;  
  border: 1px solid blue;  
}
```

Negation (matches-none)



```
a:not([target]){  
  color: #000;  
}
```

Matches-any



```
a:is(ul, ol) > li {  
  color: #000;  
}
```

(Some) structural pseudo-classes

Selector	Example	Example description
:first-child	p:first-child	Selects every <p> element that is the first child of its parent
:first-of-type	p:first-of-type	Selects every <p> element that is the first <p> element of its parent
:last-child	p:last-child	Selects every <p> element that is the last child of its parent
:last-of-type	p:last-of-type	Selects every <p> element that is the last <p> element of its parent
:nth-child(n)	p:nth-child(2)	Selects every <p> element that is the second child of its parent
:nth-last-child(n)	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child
:nth-of-type(n)	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent

Pseudo-classes

```
a:link {  
    color: #036; /* The color of not visited links */  
}  
a:hover {  
    color: #f00; /* The color of links on mouse pointer hovering */  
}  
a:visited {  
    color: #606; /* The color of visited links */  
}  
a:visited:hover {  
    color: #303; /* The color of not visited links on hover */  
}  
a:active {  
    color: #ff0; /* The color of active links */  
}  
b:first-child {  
    color: red; /* The color of the first tag */  
}  
b:last-child {  
    color: green; /* The color of the last tag */  
}
```

Pseudo-elements

Selector	Example	Example description
<code>::after</code>	<code>p::after</code>	Insert something after the content of each <p> element
<code>::before</code>	<code>p::before</code>	Insert something before the content of each <p> element
<code>::first-letter</code>	<code>p::first-letter</code>	Selects the first letter of every <p> element

```
p::first-letter {  
  color: lime;  
  font-size: 300%;  
}
```

T ext Lorem ipsum dolor sit amet,
cum. Esse delectus, quasi aliquam ex

Pseudo-elements

```
● ● ● style.css

p:before {
  content: "";
  display: inline-block;
  width: 20px;
  height: 1em;
  margin-right: 10px;
  background: #f3c;
}

p:after {
  content: " - a Rule";
  color: #666;
}
```



Search method of a lion by a simple sort. - a Rule

Grouping selectors



style.css

```
h1,h2,h3 {  
    font-family: Arial, Helvetica, sans-serif;  
}  
  
/* h1, h2, h3 share the same font-family rule */  
  
h1 {  
    font-size: 160%;  
    color: #003;  
}
```

Inheritance

style.css

```
body {  
  color: green;  
}  
div {  
  color: black;  
}  
div.red {  
  color: red;  
}
```

example.html

```
Hello  
<div>Hello, i'm div!</div>  
<div class="red">Hello, i'm red!</div>
```

Hello
Hello, i'm div!
Hello, i'm red!

!important

!important declaration **overrides** any other CSS declaration.

```
selector {  
  property: property value !important;  
}
```

First paragraph

Second paragraph.

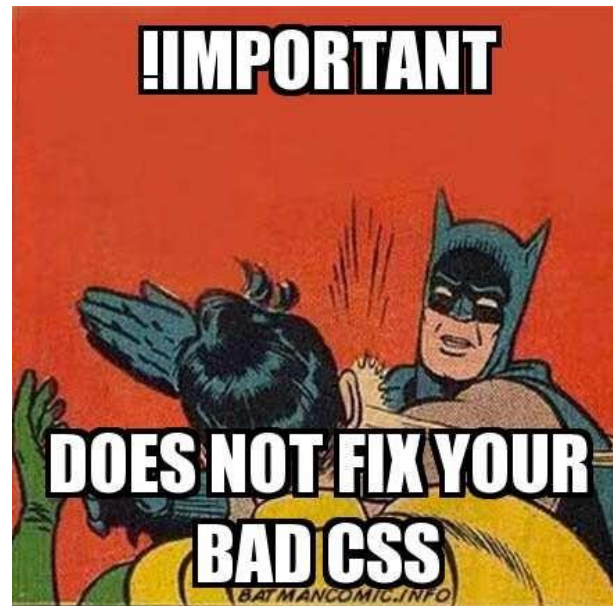
```
example.html  
  
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8" />  
    <title>Document</title>  
    <style>  
      #one {  
        color: red;  
      }  
      #two {  
        color: blue !important; /* No overrides for this */  
      }  
    </style>  
  </head>  
  <body>  
    <p id="one" style="color: orange;">First paragraph</p>  
    <p id="two" style="color: orange;">Second paragraph.</p>  
  </body>  
</html>
```

!important usage

You can use *!important* (it is not an anti-pattern), when your goal is that the property should not be overridden.

Never use *!important* to override other rules!

Do use selectors with proper *specificity* to achieve that.



CSS Specificity

 a 1 x element selector Sith power: 0,0,1	 p a 2 x element selectors Sith power: 0,0,2	 .foo 1 x class selector * Sith power: 0,1,0	 a.foo 1 x element selector 1 x class selector Sith power: 0,1,1
 p a.foo 2 x element selectors 1 x class selector Sith power: 0,1,2	 .foo .bar 2 x class selectors Sith power: 0,2,0	 p.foo a.bar 2 x element selectors 2 x class selectors Sith power: 0,2,2	 #foo 1 x id selector Sith power: 1,0,0
 a#foo 1 x element selector 1 x id selector Sith power: 1,0,1	 .foo a#bar 1 x element selector 1 x class selector 1 x id selector Sith power: 1,1,1	 .foo .foo #foo 2 x class selectors 1 x id selector Sith power: 1,2,0	 style 1 x style attribute Sith power: 1,0,0,0



* Same specificity
class selector =
attribute attribute =
pseudo-classes



!important

CSS Specificity

`nav`

0

IDs

0

Classes, attributes and pseudo-classes

1

Elements and pseudo-elements

`nav > a:hover::before`

0

IDs

1

Classes, attributes and pseudo-classes

3

Elements and pseudo-elements

`ul#primary-nav li.active`

1

IDs

1

Classes, attributes and pseudo-classes

2

Elements and pseudo-elements

[Specificity Calculator](#)

CSS Variables

CSS Variables are entities defined by CSS authors

which contain specific values to be reused

throughout a document

They are set using custom property notation: e.g., `--`

`main-color: black`, and are accessed using the `var()`

function, e.g., `color: var(--main-color)`

[Read more in spec](#)

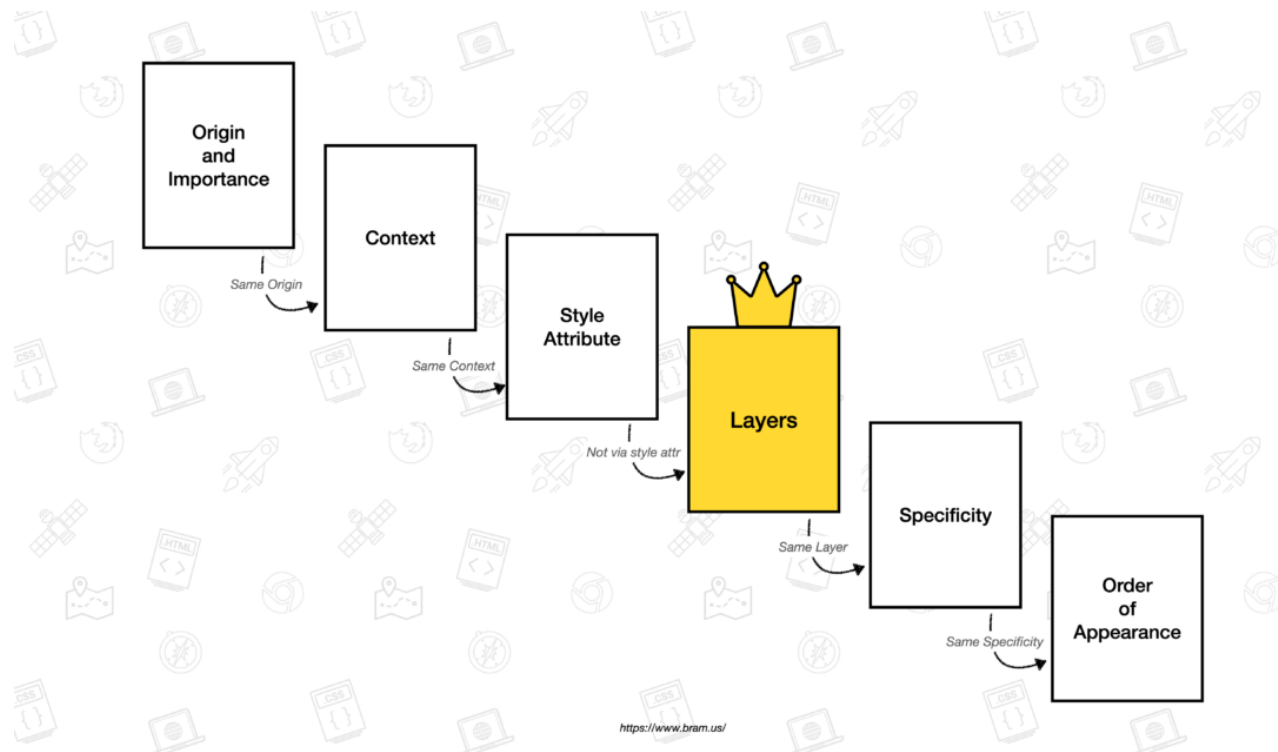
[Read more in MDN](#)

```
style.css

:root {
  --spacing: 1.5em;
  /* declaring a variable */
  --base-colors: {
    /* declaring a mixin */
    color: #fff;
    background-color: color(#fff
shade(+80%));
    /* modifying a color */
  }
}

.some-class {
  padding: var(--spacing);
  width: calc(100% - var(--
spacing));
  /* dynamically calculating a
value */
  @apply --base-colors;
}
```

Cascading



[Read more about cascade layers](#)

Cascading

Cascading refers simultaneous use of different style rules to elements by connecting multiple style files, inheritance of properties and other methods.

The higher style rule is placed in this list, the lower its priority and vice versa:

1. Browser's style
2. Author's style
3. User's style
4. The author's style adding !Important
5. The user's style adding !Important

A few examples of common CSS properties

```
style.css

selector {
  background: #ccc;
  border: #ccc solid 1px;
  color: #ccc;
  cursor: pointer;
  display: block;
  font-family: Arial, Helvetica,
sans-serif;
  font-size: 14px;
  font-style: italic;
  font-weight: bold;
  line-height: 1.3;
  text-align: left;
  text-indent: -10px;
  vertical-align: top;
}
```

```
style.css

selector {
  visibility: hidden;
  white-space: nowrap;
  list-style: none;
  outline: #fc0 dotted 1px;
  margin: 10px 5px 20px 3px;
  padding: 20px 10px;
  width: 100%;
  height: 400px;
  max-height: 50px;
  max-width: 100em;
  min-height: 1em;
  min-width: 50%;
  opacity: 0.5;
  overflow: hidden;
}
```

Reset | Normalize

The goal of a reset stylesheet is to **reduce browser inconsistencies** in default line heights, margins and font sizes of headings, and so on

<https://meyerweb.com/eric/tools/css/reset/>

<https://github.com/murtaugh/HTML5-Reset/blob/master/assets/css/reset.css>

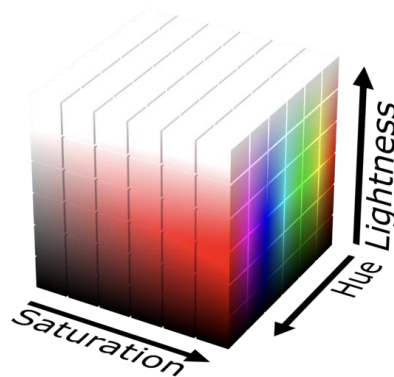
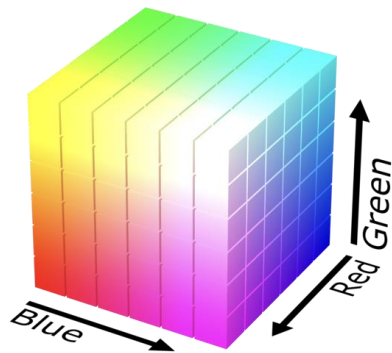
Normalize.css makes browsers render all elements more consistently and in line with modern standards. It precisely targets only the styles that need normalizing.

<http://nicolas.github.io/normalize.css/>

```
reset.css

html, body, div, span, applet, object, iframe, h1,
h2, h3, h4, h5, h6, p, blockquote, pre, a, abbr,
acronym, address, big, cite, code, del, dfn, em,
img, ins, kbd, q, s, samp, small, strike, strong,
sub, sup, tt, var, b, u, i, center, dl, dt, dd, ol,
ul, li, fieldset, form, label, legend, table,
caption, tbody, tfoot, thead, tr, th, td, article,
aside, canvas, details, embed, figure, figcaption,
footer, header, hgroup, main, menu, nav, output,
ruby, section, summary, time, mark, audio, video {
  margin: 0;
  padding: 0;
  border: 0;
  font-size: 100%;
  font: inherit; vertical-align: baseline;
}
/* ... and more */
```

Colors



Name	Hex	Rgb	Hsl
purple	#800080	rgb(128, 0, 128)	hsl(300deg 100% 25%)
lime	#00ff00	rgb(0, 255, 0)	hsl(120deg 100% 50%)
aqua	#00ffff	rgb(0 255 255)	hsl(180deg 100% 50%)

<https://www.w3.org/wiki/CSS/Properties/color/keywords>


RGB color values

The `rgb()` function accepts the RGB value in three parameters — providing the red, green and blue hues respectively



style.css

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



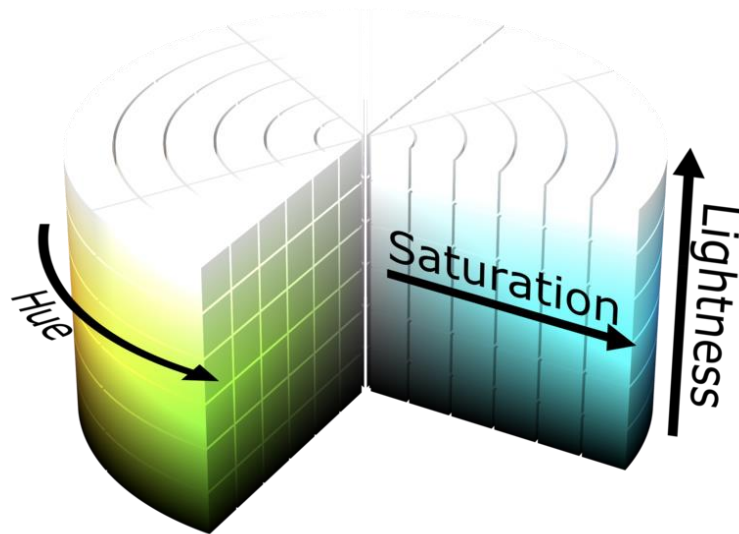
style.css

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

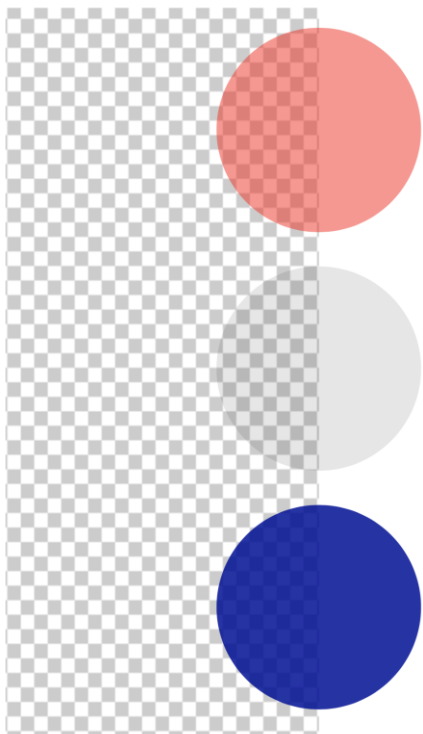

HSL

- **Hue** a value ranging from 0 to 360, defines which color you want.
- **Saturation** percentage, ranging from 0% to 100%, defines how much of that color you want.
- **Lightness** percentage, ranging from 0% to 100%, defines how bright you want that color to be

```
body {  
  background: hsl(30, 100%, 50%);  
  color: hsl(30, 100%, 75%);  
  font-size: 1.3em;  
}
```



Color opacity



```
style.css

.red {
  background-color: #ff000080;
}

.smoke {
  background-color: rgba(0, 0, 0, 0.1);
}

.ink {
  background-color: hsla(240, 100%, 30%, 0.15);
}
```

Typography



Web Safe Browser Fonts examples

Sans Serif

- Verdana
- Arial
- Helvetica
- Tahoma
- Trebuchet Ms

Serif

- Times New Roman
- Georgia
- Palatino
- Cambria

Monospace

- Courier New
- Lucida Console

Cursive

- Comic Sans Ms

```
style.css

body {
  font-family: Arial, Helvetica, sans-serif;
}
```

Should i use comic sans?

will your document be viewed by the public?



**don't use
comic sans!**

CSS Font Properties Example

```
style.css

p {
  font-style: italic;
  font-variant: small-caps;
  font-weight: bold;
  font-size: 1.5rem;
  line-height: 120%;
  font-family: "Oswald", sans-serif;
}
```

https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Fonts

font-size units



PX

If you need fine-grained control, renders the letters exactly that number of pixels in height

Use this to avoid dependency on parent components!



EM

1em is equal to the current font-size of the element in question.

By default **1em = 16px**. If you were to go and set a font-size of 20px on your body, then 1em = 20px.



%

Just like em's the very nature of percentage sizing is that it is relative. It also cascades in the same way.

If a parent has the font-size of 20px and the child has a font-size of 50%, it will be 10px.



REM

Inherited from the root element (html) and do not cascade.

EM vs REM

```
style.css

html {
  font-size: 100%; /* =16px */
}
main {
  font-size: 125%;
}
main p {
  font-size: 1em; /* =20px */
  /* The 'em' size is relative to its parent. */
}
div p {
  font-size: 1rem; /* =16px */
  /* The 'rem' size is relative to the root element. */
}
```

Font-size

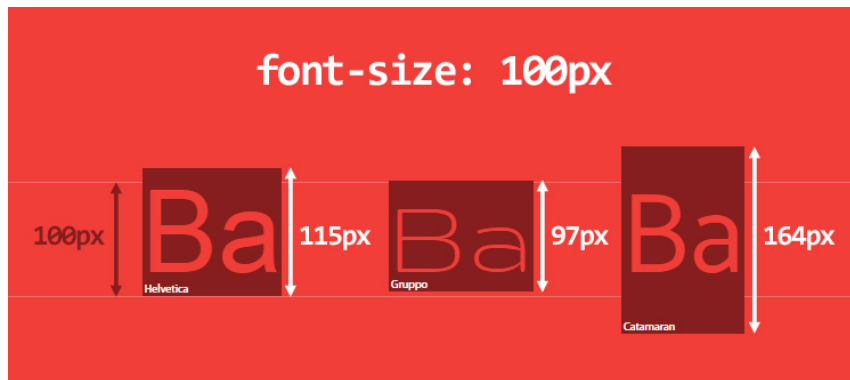
```
style.css

p {
  font-size: 100px;
}
.a {
  font-family: Helvetica;
}
.b {
  font-family: Gruppo;
}
.c {
  font-family: Catamaran;
}
```

```
example.html

<p>
  <span class="a">Ba</span>
  <span class="b">Ba</span>
  <span class="c">Ba</span>
</p>
```

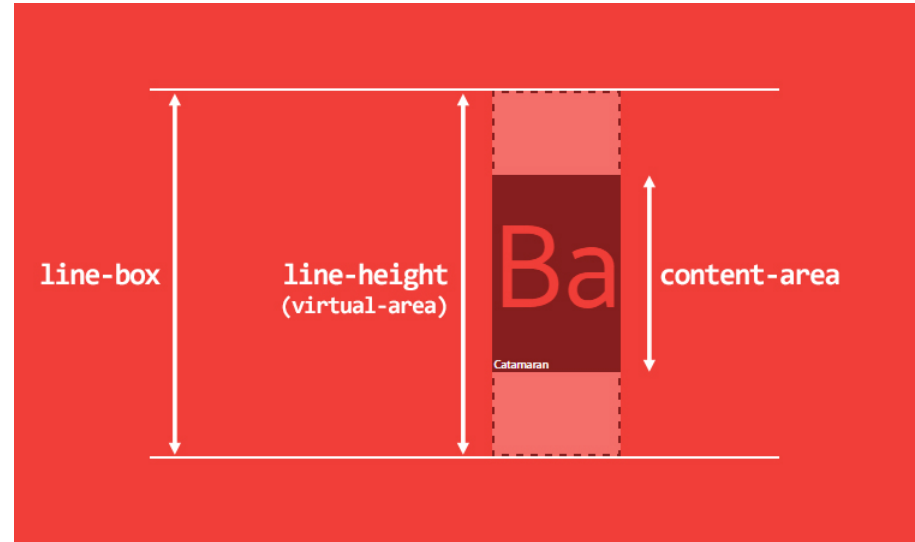
Actual size can be different for each font due to Em-square.



<https://iamvdo.me/en/blog/css-font-metrics-line-height-and-vertical-align>

Line-height

- The content-area height is defined by the font metrics
- The virtual-area height is the line-height, and it is the height used to compute the line-box's height



Line-height examples

style.css

```
div {  
  line-height: normal;  
}  
  
div {  
  line-height: 1.2;  
}  
  
div {  
  line-height: 21px;  
}  
  
div {  
  line-height: 150%;  
}
```

line-height: normal

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Proin
consectetur tristique libero
ultrices luctus.

line-height: 1.2

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Proin
consectetur tristique libero
ultrices luctus.

line-height: 21px

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Proin
consectetur tristique libero
ultrices luctus.

line-height: 150%

Lorem ipsum dolor sit
amet, consectetur
adipiscing elit. Proin
consectetur tristique libero
ultrices luctus.

Web fonts

1. HAPPY FRIDAY!

Marujo Dotface

2. *I Hope you have a*

Salamander Script

3. Great Weekend

Skitch & Skitch Fill Layered

4. *Time with Family*

Truth Unvarnished

5. AND A SUNDAY NAP

Gentil Bold

https://developer.mozilla.org/en-US/docs/Learn/CSS/Styling_text/Web_fonts

Font formats



Web Open Font Format

.woff files are supported by all modern browsers



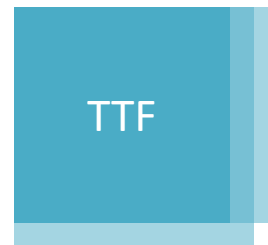
Embedded OpenType

.eot files for older Internet Explorer versions (< 8)



Scalable Vector Graphics

.svg files are supported by all modern browsers



TrueType Font

.ttf .otf files partial support in IE

@font-face declaration

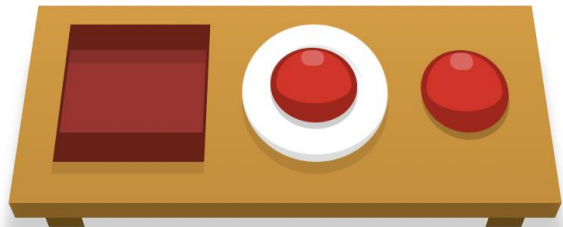
```
style.css

@font-face {
  font-family: 'MyWebFont';
  src: url('webfont.eot'); /* IE9 Compat Modes */
  src: url('webfont.eot?#iefix') format('embedded-opentype'), /* IE6-8 */
       url('webfont.woff2') format('woff2'), /* Super Modern Browsers */
       url('webfont.woff') format('woff'), /* Pretty Modern Browsers */
       url('webfont.ttf') format('truetype'), /* Safari, Android, iOS */
       url('webfont.svg#svgFontName') format('svg'); /* Legacy iOS */
}

body {
  font-family: 'MyWebFont', Fallback, sans-serif;
}
```

<https://css-tricks.com/snippets/css/using-font-face/>

Games – practicing...



<https://flukeout.github.io/> - ...selectors

<https://flexboxfroggy.com/> - ...flexbox

<https://cssgridgarden.com/> - ...css grid

<https://cssbattle.dev/> - replicate targets with the smallest possible CSS code

THANK YOU!