



CSS 1

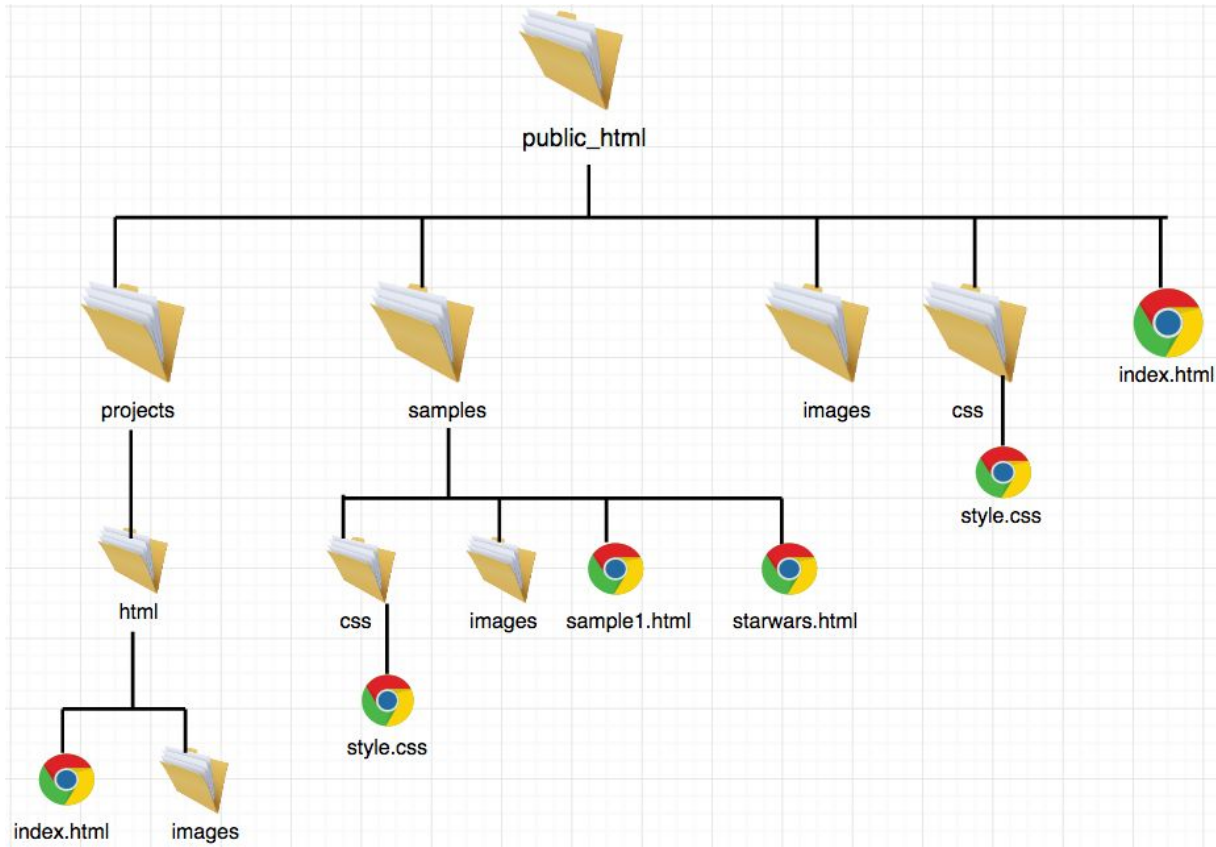
Prep Work

- ❑ Finish the HTML Project if you did not complete it and upload it to the server (send me a link):

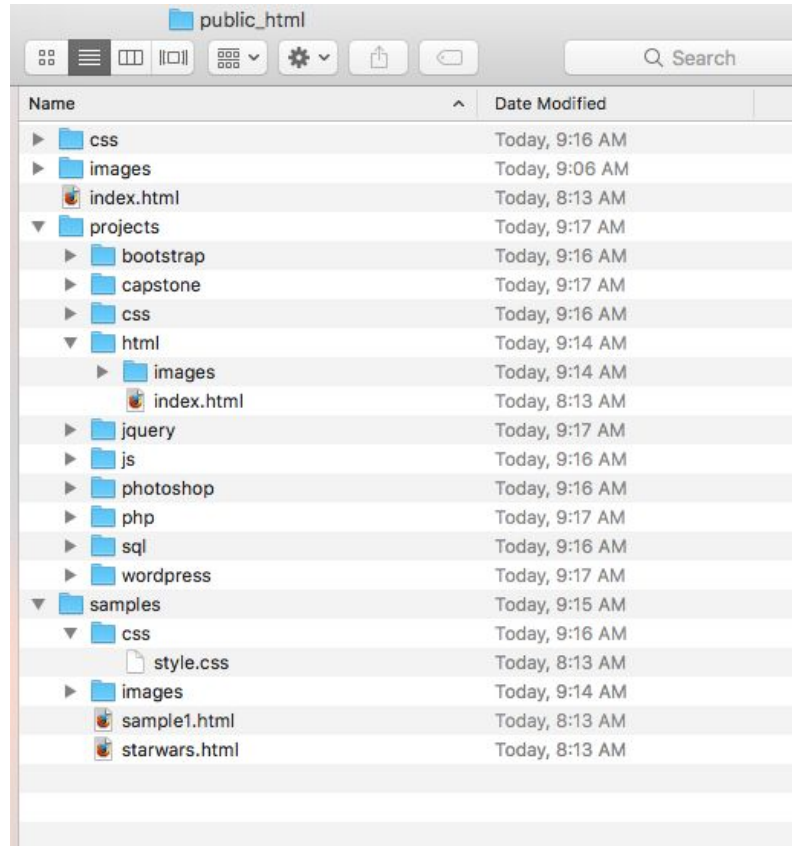
public_html/class-projects/html/safety.html

- ❑ Treehouse
- ❑ Practice using HTML tags

- ❑ Create a `css` folder in your `public_html/class-samples` folder
- ❑ Selectors and Styling Declarations
- ❑ Placement for Different Styling
- ❑ Color
- ❑ Text Formatting
- ❑ Boxes

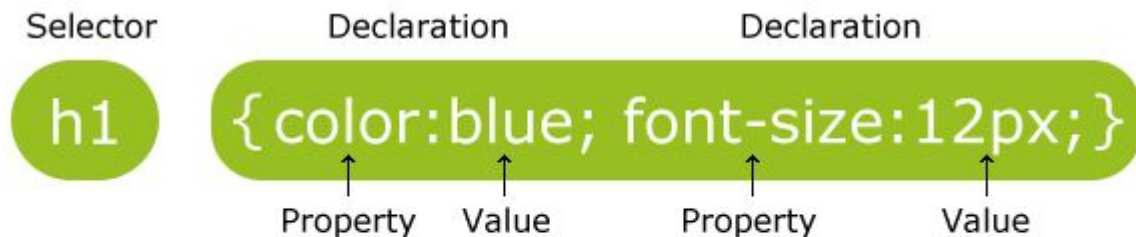


CSS 1





CSS Selectors and Styling



The selector points to the HTML element you want to style.

The declaration block contains one or more declarations separated by semicolons.

Each declaration includes a property name and a value, separated by a colon.

- CSS Style rules are associated with HTML elements and declared in two parts by Selectors and Declarations.
- The Selector shows what element to apply the rule or style to

Selectors -The selector shows what element/html to apply the rule or style to. Same rule can apply to more than one element if you separate the element names with a comma

- **Selector**-html element you want to change(<p> <h1>, <div>, <header>, <nav>, #id, .class)

Declarations- indicate how element should be styled.

Declarations are placed inside braces { } and contain:

- **Property**-Aspect of the element you want to change(font-family, background-color, width, height)
- **Value**- Settings you want to apply (What you want it to be color name, font name, number of pixels)

When styling with CSS you :

- Separate more than 1 selector (h1, h2, h3) with a comma ','
- Wrap the declaration(s) with curly braces '{', '}'
- Separate more than 1 declaration with a semicolon ';'

```
h1, h2, h3 {  
    font-family: Arial;  
    color: yellow;  
}
```

PROPERTY VALUE

Selectors:

- ▣ Allow you to target rules to specific elements in an HTML document
- ▣ Can be HTML Tags
- ▣ Can be IDs
- ▣ Classes

CSS Selectors

SELECTOR	MEANING	EXAMPLE
UNIVERSAL SELECTOR	Applies to all elements in the document	<code>* {}</code> Targets all elements on the page
TYPE SELECTOR	Matches element names	<code>h1, h2, h3 {}</code> Targets the <code><h1></code> , <code><h2></code> and <code><h3></code> elements
CLASS SELECTOR	Matches an element whose <code>class</code> attribute has a value that matches the one specified after the period (or full stop) symbol	<code>.note {}</code> Targets any element whose <code>class</code> attribute has a value of <code>note</code> <code>p.note {}</code> Targets only <code><p></code> elements whose <code>class</code> attribute has a value of <code>note</code>
ID SELECTOR	Matches an element whose <code>id</code> attribute has a value that matches the one specified after the pound or hash symbol	<code>#introduction {}</code> Targets the element whose <code>id</code> attribute has a value of <code>introduction</code>
CHILD SELECTOR	Matches an element that is a direct child of another	<code>li>a {}</code> Targets any <code><a></code> elements that are children of an <code></code> element (but not other <code><a></code> elements in the page)
DESCENDANT SELECTOR	Matches an element that is a descendent of another specified element (not just a direct child of that element)	<code>p a {}</code> Targets any <code><a></code> elements that sit inside a <code><p></code> element, even if there are other elements nested between them
ADJACENT SIBLING SELECTOR	Matches an element that is the next sibling of another	<code>h1+p {}</code> Targets the first <code><p></code> element after any <code><h1></code> element (but not other <code><p></code> elements)
GENERAL SIBLING SELECTOR	Matches an element that is a sibling of another, although it does not have to be the directly preceding element	<code>h1~p {}</code> If you had two <code><p></code> elements that are siblings of an <code><h1></code> element, this rule would apply to both

The ID selector (combined with HTML Element) Unique

We need ways to select tags in an HTML document. The basic elements like `<h1>`, `<p>`, and `` will often do the job, but our basic set of tags doesn't cover every possible type of page element or layout choice. For this we need ID's and Classes. For example `<ul id="nav">`, this will give us the chance to target this unordered list specifically, so that we may manipulate it uniquely to other unordered lists on our page.

- The ID selector uses the ID attribute of an HTML element to select a **specific** element
- Each element can have only one ID
- The ID of an element should be unique within a page, so the ID selector is used to select one unique element
- Each page can have only one element with that ID
- To select an element with a specific ID, write a hash (#) character, followed by the ID name of the element, e.g.: `#news { color: red; }`

The **.class** selector (combined with HTML Element)

- Selects elements with a specific class attribute
- To select elements with a specific class, write a period (.) character, followed by the name of the class in your CSS stylesheet or within your styling rule
- You can also specify that only specific HTML elements should be affected by a class. To do this, start with the element name, then write the period (.) character, followed by the name of the class
- You can use the same class on multiple elements (those elements usually have same CSS Properties and Values).
- You can use multiple classes on the same HTML element.
- Any styling information that needs to be applied to multiple objects on a page should be done with a class.

The id Selector and The *.class* selector

- Can be combined with HTML Element to apply styling

ID - `#news { color: red; }` CLASS- `.news { color: red; }`

If you want higher specificity, you can combine both the element and ID selectors:

ID - `li#news { color: red; }` CLASS- `li.news { color: red; }`

The `.class` selector

- Let's say you had a class `widget`?
- You can now use the class name `"widget"` as your hook to apply the same set of styling to each one of these. But what if you need one of them to be bigger than the other, but still share all the other attributes? You can apply more than one class to the element:

```
<div class="widget"></div>  
<div class="widget big"></div>  
<div class="widget"></div>
```

- You do not need to make a brand new class name here, just apply a new class right in the class attribute. These classes are space delimited and most browsers support any number of them (actually, it's more like thousands, but way more than you'll ever need).

```
<div class="widget"></div>  
<div class="widget big"></div>  
<div class="widget"></div>
```

NOTE: Classes have no special abilities in the browser, but ID's do have one very important trick up their sleeve. This is the "hash value" in the URL.

If you have a URL like <http://yourdomain.com#comments>, the browser will attempt to locate the element with an ID of "comments" and will automatically scroll the page to show that element. It is important to note here that the browser will scroll whatever element it needs to in order to show that element, so if you did something special like a scrollable DIV area within your regular body, that div will be scrolled too.

This is an important reason why having ID's be absolutely unique is important. So your browser knows where to scroll!



CSS

Placement and Parts

Review on Parts to Styling

Based on Declarations, Properties, and Values

Insert into your HTML source code using 3 possible ways:

- Inline: In the HTML element or tag `<h1 style="color:red;">heading text</h1>`
- Internal: In the HTML page between the head tags `<head><style>h1 {color:red;}</style></head>`
- External: In a separate document `<head><link rel="stylesheet" type="text/css" href="css/mystyle.css"></head>`

Inline is written directly into the html `<>` tag and/or html element

Internal is written/defined in between the `<head><style>HERE</style></head>` tags then by referencing HTML Element/Selector or tag in the body of the page.

External is written/defined in a separate document, linked to CSS document in between the `<head></head>` tags +/- combined HTML Element/selector in the body of the page -written in external document **excludes** the `<style></style>` (Preferred Method)

When using Internal or External CSS your Declarations Properties and Values coincide with your HTML structure. (EXAMPLE:) HTML Syntax -`<h1>Heading</h1>` CSS Syntax - `h1 {color:red;}`

In-line styling -Inside HTML Page affecting a single HTML Element

HTML Page:

```
<!doctype html>

<html>

<head>

    <meta charset="UTF-8">

    <title>In-Line Style</title>

</head>

<body>

    <p style="color:#F9080C;"> This is my red paragraph!</p>

</body>

</html>
```

Result:

This is my red paragraph!

Internal styling - Inside HTML Page Head Tags using <head><style></style></head>

HTML Page:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Internal Style</title>

<style>

body {
font-family: Arial, Verdana, sans-serif;}
h1, h2 {
color: #ee3e80;}
p {
color: #665544;}

</style>

</head>

<body>
<h1>From Garden to Plate</h1>
<p>A <i>potager</i> is a French term for an
ornamental vegetable or kitchen garden ... </p>
<h2>What to Plant</h2>
<p>Plants are chosen as much for their functionality
as for their color and form ... </p>
</body>

</html>
```

Result:

From Garden to Plate

A *potager* is a French term for an ornamental vegetable or kitchen garden ...

What to Plant

Plants are chosen as much for their functionality as for their color and form ...

- ▣ External style sheets should not contain `<style></style>` inside the CSS document. Just use the selector / element {}, then declarations, attributes and values.
- ▣ CSS Stylesheet Syntax should look like (Below):
- ▣ Can link more than one stylesheet in the `<head>` of the HTML page

```
<head> <link rel="stylesheet" type="text/css" href="css/stylesheetname.css"></head>
```

```
@charset "UTF-8";
```

```
/* CSS Document */
```

```
body {
```

```
    font-family: arial;
```

```
    background-color: blue;}
```

```
h1 {
```

```
    color: black;}
```

- ▣ `href`-tells browser where the CSS document is:
- ▣ `type`- defines type of document you are linking to
- ▣ `rel`- defines relationship of the document to the HTML page

External Styling

HTML Page:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>From Garden to Plate</title>

  <link href="css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>

  <h1> From Garden to Plate</h1>
  <p>A <i>potager</i> is a s French term for an ornamental vegetable or kitchen garden...</p>
  <h2>What to Plant</h2>
  <p>Plants are chosen as much for their functionality as for their color and form...</p>

</body>
</html>
```

CSS Page:

```
/* CSS Document */

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

h1, h2 {
  color:#ee3e80;
}

p{
  color:#665544;
}
```

Result:

From Garden to Plate

A *potager* is a French term for an ornamental vegetable or kitchen garden ...

What to Plant

Plants are chosen as much for their functionality as for their color and form ...

TERMINOLOGY - CSS Stylesheet Pages

- ❑ **UTF-8** is a compromise character encoding that **can** be as compact as ASCII (if the file is just plain English text) but **can** also contain any unicode characters (with some increase in file size). UTF stands for Unicode Transformation Format. The '8' **means** it uses 8-bit blocks to represent a character.
- ❑ **UTF-8** is backwards compatible with ASCII. **UTF-8** is the preferred encoding for web pages.
- ❑ **@charset** at the beginning of the CSS document tells the browser to read the CSS file as **UTF-8**. This is handy if your CSS contains **unicode** character encoding or processing, storage, and transport of text characters. The default for HTML 5 is UTF-8.
- ❑ **Basis for use:** If you have any non-ASCII text in your CSS file, for example non-ASCII characters in **font names, in values of the content property, in selectors (ex , @&#x2011;)** etc., you need to be sure that the CSS parser knows how to transform the bytes into characters correctly, so that it understands your CSS or HTML code.

CSS

Exercise

Create a page called `divs2.html` with **external styling** (`divs2.css`). Add 2 divs: one with an id and one with a class name. Place the word “paragraph” inside both of the divs, and save the file to the `public_html/class-samples` directory.

- Give the first div (with an id) a text color of **blue**
- Give the second div (with a class) a text color of **green**

Add an unordered list with 2 list items (yellow and red)

- Create an id for each list item selector
- Style the first id with a text color of yellow
- Style the second id with a text color red

Paragraph

Paragraph

• Yellow

• Red

CSS Selectors

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>IDs Classes and Lists</title>
<style>
  #div1 {color:blue;}
  .div2 {color:green;}
  ul li#list1 {color:yellow;}
  ul li#list2 {color:red;}

  }
</style>
</head>

<body>

  <div id="div1">Paragraph</div>
  <div class="div2">Paragraph</div>
  <ul>
    <li id="list1">yellow</li>
    <li id="list2">red</li>
  </ul>
</body>
</html>
```

CSS

Exercise

- Create a page called `external.html` with external styling page called `mystyle.css`
- save the html page into `public_html/class-samples` directory
- save the css page into `public_html/class-samples/css` directory

Preview Result Should Look Like this:

From Garden to Plate

A potager is a French term for an ornamental vegetable or kitchen garden ...

What to Plant

Plants are chosen as much for their functionality as for their color and form ...



Color

Options for Applying Colors

- ❑ **Color Name:** (red, blue etc.)
- ❑ **HEX** (#000000)- hexadecimal integers specify the components of the color
- ❑ **RGB** (0,0,0) Red, Green, Blue
- ❑ **RGBA** (0,0,0,1.0) Red, Green, Blue, Alpha
- ❑ **HSL** (120, 100%, 50%) Hue, Saturation, Lightness
- ❑ **HSLA** (120, 100%, 50% , 1.0) Hue, Saturation, Lightness, Alpha

https://www.w3schools.com/cssref/css_colors_legal.asp

Applying Color and Opacity:

□ RGB -Based on RGB Numbers

- RGB color values are supported in all major browsers.

□ RGBA -Red Green Blue Alpha

- RGBA color values are supported in IE9+, Firefox 3+, Chrome, Safari, and in Opera 10+.
- RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity of the object.
- An RGBA color value is specified with: rgba(red, green, blue, alpha). The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

□ HEX Code- Can use last 3 Octets with

- Hexadecimal color values are supported in all major browsers.

□ HSL, HSLA hue, saturation, and lightness

- Hue is a degree on the color wheel (from 0 to 360) - 0 (or 360) is red, 120 is green, 240 is blue. Saturation is a percentage value; 0% means a shade of gray and 100% is the full color. Lightness is also a percentage; 0% is black, 100% is white.

□ Color Names- Not recommended (147 Supported)

color

The `color` property allows you to specify the color of text inside an element. You can specify any color in CSS in one of three ways:

RGB VALUES

These express colors in terms of how much red, green and blue are used to make it up. For example: `rgb(100,100,90)`

HEX CODES

These are six-digit codes that represent the amount of red, green and blue in a color, preceded by a pound or hash # sign. For example: `#ee3e80`

COLOR NAMES

There are 147 predefined color names that are recognized by browsers. For example: `DarkCyan`

We look at these three different ways of specifying colors on the next double-page spread.

CSS3 has also introduced another way to specify colors called HSLA, which you will meet near the end of this chapter on page 255-256.

chapter-11/foreground-color.html

CSS

```
/* color name */
h1 {
  color: DarkCyan;}
/* hex code */
h2 {
  color: #ee3e80;}
/* rgb value */
p {
  color: rgb(100,100,90);}
```

RESULT

Marine Biology

The Composition of Seawater

Almost anything can be found in seawater. This includes dissolved materials from Earth's crust as well as materials released from organisms. The most important components of seawater that influence life forms are salinity, temperature, dissolved gases (mostly oxygen and carbon dioxide), nutrients, and pH. These elements vary in their composition as well as in their influence on marine life.

Above each CSS rule in this example you can see how CSS allows you to add comments to your CSS files. Anything between the `/*` symbols and the `*/` symbols will not be interpreted by the browser. They are shown in grey above.

The use of comments can help you to understand a CSS file (and organise it, by splitting a long document into sections). Here, we have used comments to indicate which method is used to specify each of the different types of colors.

Applying Color and Opacity:

HTML Selector/ID or Class { color: *ColorValue*;

HTML Selector/ID or Class { background-color: *ColorValue*;

- CSS3 introduces the opacity property which allows you to specify the opacity of an element and any of its child elements.
- The value is a number between 0.0 and 1.0 (so a value of 0.5 is 50% opacity and 0.15 is 15% opacity).

HTML Selector/ID or Class { opacity: 0.5;}

Use Hex unless specifying opacity or in need of the Alpha Channel

Color Type HEX - (#000000)

HEX - A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color. All values must be between 00 and FF.

Note: Browser Supported

HTML Page

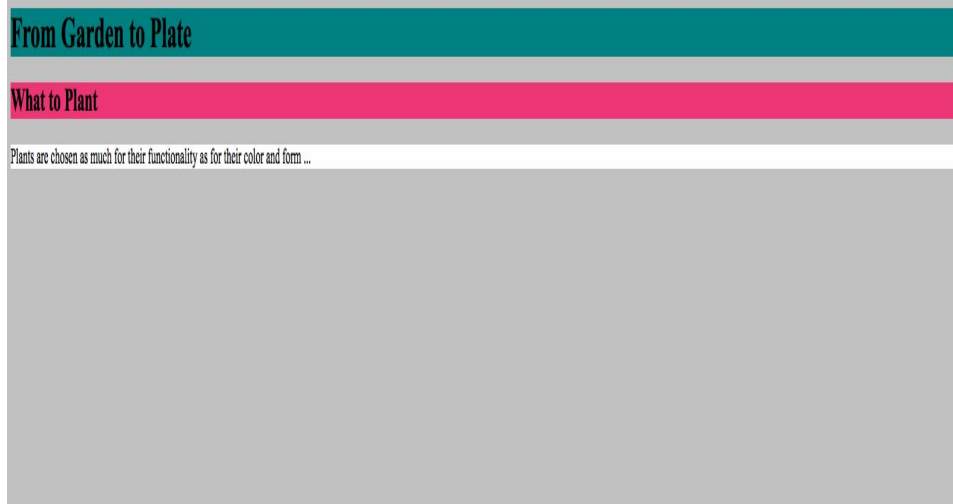
```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
body {
background-color: rgb(200,200,200);}
h1 {
background-color: DarkCyan;}
h2 {
background-color: #ee3e80;}
p {
background-color: white;}
</style>
</head>
<body>

<h1>From Garden to Plate</h1>
<h2>What to Plant</h2>
<p>Plants are chosen as much for their functionality
as for their color and form ... </p>

</body>

</html>
```

Preview Result



HTML Page

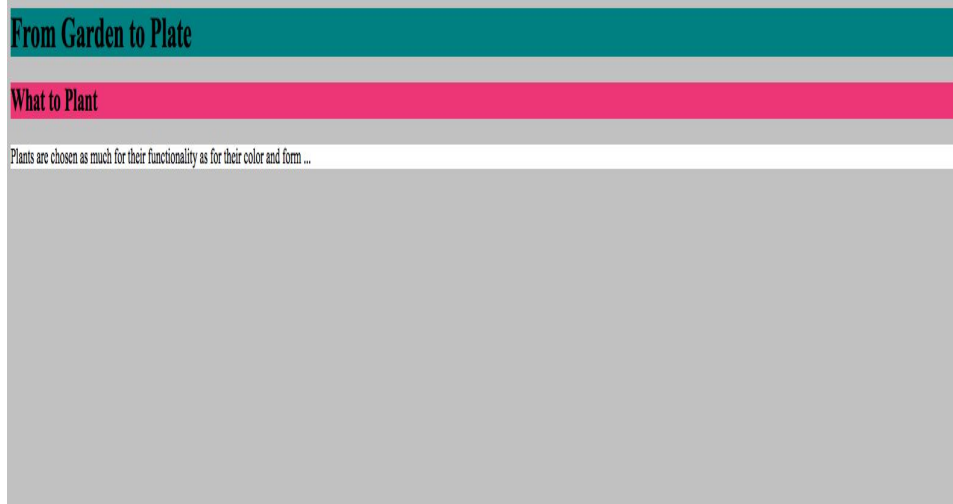
```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
body {
background-color: rgb(200,200,200);}
h1 {
background-color: DarkCyan;}
h2 {
background-color: #ee3e80;}
p {
background-color: white;}
</style>
</head>
<body>

<h1>From Garden to Plate</h1>
<h2>What to Plant</h2>
<p>Plants are chosen as much for their functionality
as for their color and form ... </p>

</body>

</html>
```

Preview Result



Color Type RGB - (255, 0, 0)

RGB -An RGB color value is specified with the `rgb()` function, which has the following syntax:

`rgb(red, green, blue)`

Each parameter (red, green, and blue) defines the intensity of the color and can be an integer between 0 and 255 or a percentage value (from 0% to 100%).

For example, the `rgb(0,0,255)` value is rendered as blue, because the blue parameter is set to its highest value (255) and the others are set to 0.

Also, the following values define equal color: `rgb(0,0,255)` and `rgb(0%,0%,100%)`.

Note: Browser Supported

Color Declarations-Based on Red, Green, Blue, Alpha

rgba stands for
red green blue alpha.

`rgba(255, 0, 0, 0.3)`

While it is sometimes described as a color space, it is actually simply a use of the **RGB** color model, with extra information.

Opacity can be combined/changed using a **4th** value **rgba**

RGB VALUES

Values for red, green, and blue are expressed as numbers between 0 and 255.



`rgb(102,205,170)`

This color is made up of the following values:

102 red
205 green
170 blue

HEX CODES

Hex values represent values for red, green, and blue in hexadecimal code.



`#66cdaa`

The value of the red, 102, is expressed as 66 in hexadecimal code. The 205 of the green is expressed as cd and the 170 of the blue equates to aa.

COLOR NAMES

Colors are represented by predefined names. However, they are very limited in number.



`MediumAquamarine`

There are 147 color names supported by browsers (this color is `MediumAquamarine`). Most consider this to be a limited color palette, and it is hard to remember the name for each of the colors so (apart from white and black) they are not commonly used.

Setting Opacity - (1.0 Full or 0.5 half)

HTML Page

```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
body {
background-color: rgb(200,200,200);}
h1 {
background-color: rgba(234,32,36,1.0);}

p {

background-color: rgba(234,32,36,0.5); }

</style>
</head>
<body>

<h1>From Garden to Plate</h1>

<p>Plants are chosen as much for their functionality
as for their color and form ... </p>

</body>

</html>
```

Preview Result



Color Type HSL, HSLA - (0.75 = 75% or 0.5 = 50%)

HSL stands for (hue, saturation, and lightness) - and represents a cylindrical-coordinate representation of colors.

An HSL color value is specified with: `hsl(hue, saturation, lightness)`. `hsl(120, 100%, 50%)`

- Hue is a degree on the color wheel (from 0 to 360) - 0 (or 360) is red, 120 is green, 240 is blue. Saturation is a percentage value; 0% means a shade of gray and 100% is the full color. Lightness is also a percentage; 0% is black, 100% is white.

An **HSLA** color value is specified with: `hsla(hue, saturation, lightness, alpha)`, where the alpha parameter defines the opacity. `hsla(120, 100%, 50%, 0.3)`

- The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (fully opaque).

Note: Limited Support be careful of browser versions

Color Type HSL, HSLA - (0.75 = 75% or 0.5 = 50%)

HTML Page

```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
body {
background-color: rgb(200,200,200);}
h1 {

background-color: #ffffff;
background-color: hsla(0,100%,100%,0.75);}

p {
background-color: #ffffff;
background-color: hsla(0,100%,100%,0.5);}

</style>
</head>
<body>

<h1>From Garden to Plate</h1>

<p>Plants are chosen as much for their functionality
as for their color and form ... </p>

</body>

</html>
```

Preview Result

From Garden to Plate

Plants are chosen as much for their functionality as for their color and form ...

Color Charts:

http://www.rapidtables.com/web/color/RGB_Color.htm

http://www.w3schools.com/cssref/css_colors.asp

<http://hslpicker.com/>



Text

Applying Fonts and Text Style (attributes):

Options:

- ▣ Family
- ▣ Size
 - ▣ Percentage (%)
 - ▣ Pixel (px)
 - ▣ EM -width of the letter m (em)
- ▣ Reference a URL
- ▣ Weight
- ▣ Style
- ▣ Transform
- ▣ Decoration
- ▣ Spacing
- ▣ Align
- ▣ Indent
- ▣ Shadow
- ▣ First letter / line

NOTE: Set Font Size With Em

To allow users to resize the text (in the browser menu), many developers use em instead of pixels.

The em size unit is recommended by the W3C.

1em is equal to the current font size. The default text size in browsers is 16px. So, the default size of 1em is 16px.

The size can be calculated from pixels to em using this formula: $\text{pixels}/16=\text{em}$

Applying Fonts and Text Style (Syntax):

- ▣ Font-Family- Serif, Verdana, Times
- ▣ Font-Size-
 - ▣ Percentage (%) font-size: 100%;
 - ▣ Pixel (px) font-size: 30px;
 - ▣ EM (em) -width of the letter -based on parent font-size: 1.0em;
- ▣ Reference a URL- @font-face; src: url('fonts/chunkfive.eot');}
- ▣ Font-Weight- bold/normal
- ▣ Font-Style- italic/normal/oblique;
- ▣ Text Transform- uppercase/lowercase/capitalize
- ▣ Text Decoration-none/underline/overline/line-through/blink
- ▣ Letter-Spacing- (0.5em) or Word-Spacing (0.5em)
- ▣ Text-Align-
 - ▣ text-align left/right/justify
 - ▣ vertical-align: top/baseline/bottom ;
- ▣ Text-Indent- 20px
- ▣ Text-Shadow- 1px 1px #666666

Syntax: HTML Selector/ID or Class {font-family: serif, arial, helvetica;}

HTML Selector/ID or Class {font-size: 10px;}

HTML Selector/ID or Class {font-weight: bold;}

```
@font-face {  
  font-family: 'ChunkFiveRegular';  
  src: url('fonts/chunkfive.eot');
```

must first define a name for the font (e.g. myFirstFont), and then point to the font file.

HTML Selector/ID or Class {font-style: italic;}

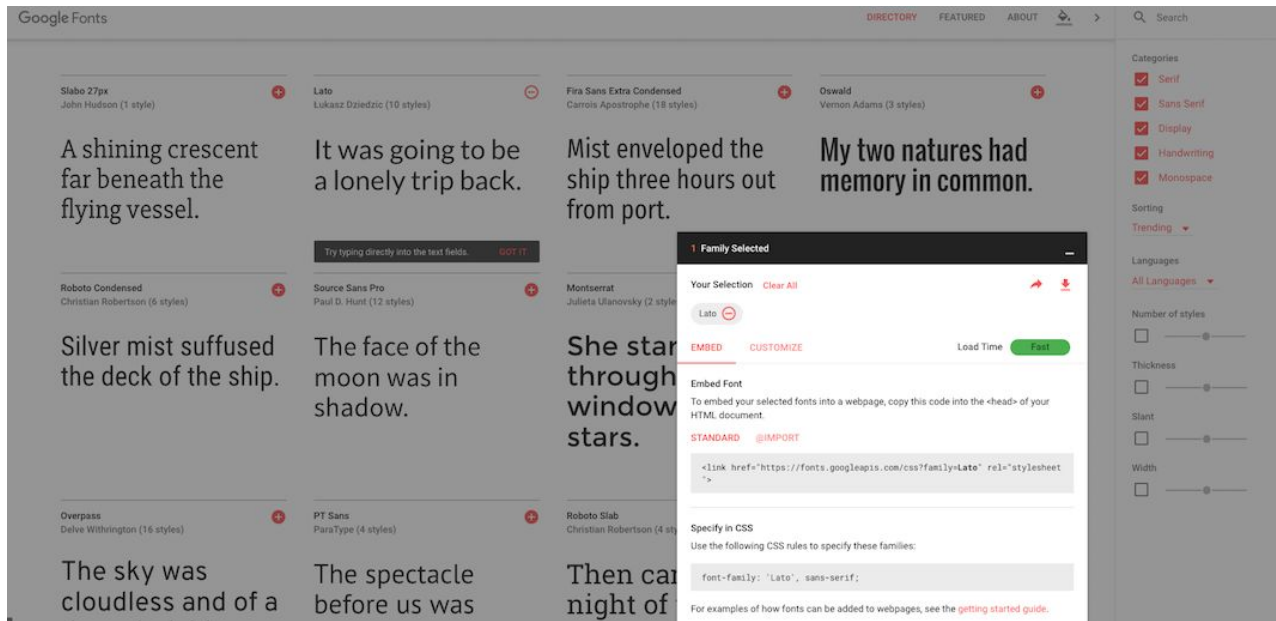
HTML Selector/ID or Class {text-transform: uppercase;}

HTML Selector/ID or Class {text-decoration: none;}

Linking out/Embedding a font library and applying it to your css internal style

Google Fonts: <https://developers.google.com/fonts/>

- Click the +
- Click the - to maximize the link source
- Paste the link source between the `<head></head>` tags



Linking out to a font library and applying it to your css
internal style

```
<!doctype html>
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="https://fonts.googleapis.com/css?family=Tangerine">
    <style>
      body {
        font-family: 'Tangerine', serif;
        font-size: 48px;
      }
    </style>
  </head>
  <body>
    <div>Making the Web Beautiful!</div>
  </body>
</html>
```

Find Out What Font is on a page or image

<https://www.myfonts.com/WhatTheFont/>

Syntax For Making Text Shadow:

```
HTML Selector/ID or Class {  
color: #666666;  
text-shadow: 1px 1px 0px #000000;}
```

HTML PAGE:

```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
body {
background-color: rgb(200,200,200);}

p{ text-align: left;}

h1 {background-color: #eeeeee;
color: #666666;
text-shadow: 1px 1px 0px #000000;}

h2 {
font-family: Arial, Helvetica;
font-size: 10px;
font-weight: bold;
font-style: italic;
text-transform: uppercase;
text-decoration: underline;}

</style>
</head>
<body>

<h1>From Garden to Plate</h1>

<p>Plants are chosen as much for their functionality
as for their color and form ... </p>

<h2>From Garden to Plate</h2>

</body>

</html>
|
```

Preview Result:

From Garden to Plate

Plants are chosen as much for their functionality as for their color and form ...

FROM GARDEN TO PLATE

□ Styling Links

- Active
- Visited
- Hover

HTML

```
<a href="#">link</a>
```

CSS

```
a:link {  
  color:deeppink;  
  text-decoration:none;  
}
```

```
a:link {  
  
  color: deeppink;  
  
  text-decoration: none;}  
  
a:visited {  
  
  color: black;}  
  
a:hover {  
  
  color: blue;  
  
  text-decoration: underline;}  
  
a:active {  
  
  color: darkcyan;}
```

HTML PAGE:

```
<!doctype html>
<html>
<head>
<title>Background Colors</title>
<style>
a:link {

color: deeppink;

text-decoration: none;}

a:visited {

color: black;}

a:hover {

color: blue;

text-decoration: underline;}

a:active {

color: darkcyan;}

</style>
</head>
<body>

<a href="#">From Garden to Plate</a><br / >
<a href="#">From Garden to Plate</a><br / >
<a href="#">From Garden to Plate</a><br / >

</body>

</html>
```

Result as you hover a link:

From Garden to Plate
From Garden to Plate
From Garden to Plate



Boxes & Layouts

Layout Methods

- Tables -Used in past for page layouts and currently for HTML email or tabular data (Uses HTML)
- The Normal Flow (Uses CSS + Divs, HTML 5)
- Floats (Uses CSS+ DIVS and or HTML5)

Other Methods

- Positioning (Uses CSS)
- Flexbox (Support Limited)
- Grid (Support Limited)

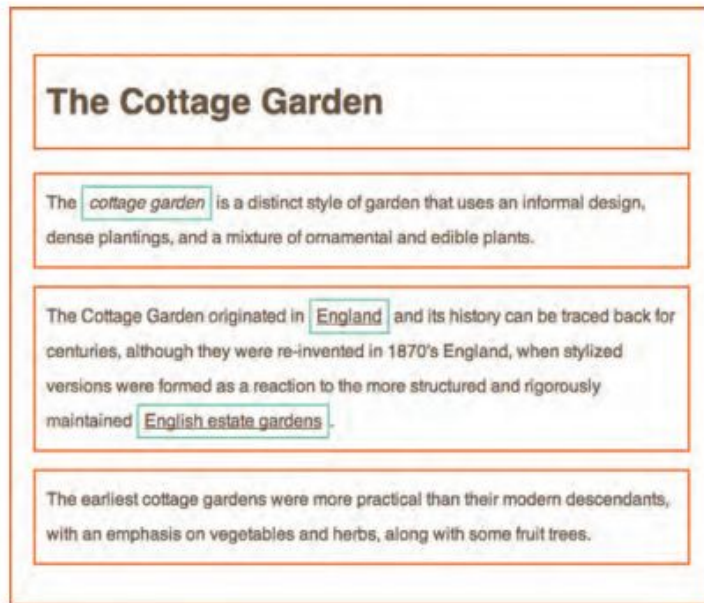
DIVs, and HTML 5 Semantic Grouping :

CSS allows you to create rules that control the way that each individual box (and the contents of that box) is presented.

In this case you can style content inside each box or the box itself including:

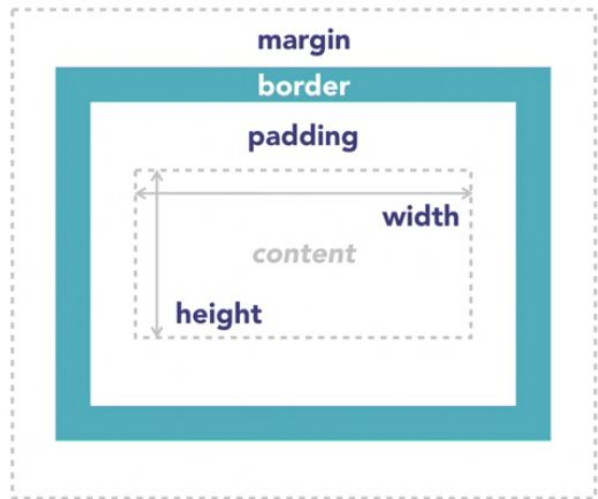
- Background Color
- Text Font
- Border Color
- Its position in the browser

The CSS **Box Model**. All HTML elements can be considered as boxes. In CSS, the term "**box model**" is used when talking about design and layout. The CSS **box model** is essentially a **box** that wraps around HTML elements, and it consists of: margins, borders, padding, and the actual content.



Boxes and Box Model

CSS allows you to create rules that control the way that each individual box (and the contents of that box) is presented.



Box Properties

- Width
- Height
- Padding
- Margin

- **Width**-Sets the width of the content area which is the actual content between the tags
- **Height**-Sets the height of the content area which is the actual content between the tags
- **Padding**- add the space within and element
- **Margin**- Add or remove the space around the element
- **Border**-is added between margin and padding

The Normal Flow

Block Elements

- The normal flow of elements on a page will appear as they are written in the HTML code. Normal elements will appear stacked on one another and are called block elements.
- CSS moves away from the normal flow to create columns and rows, placing elements in specific areas of the page.
- Block elements assume the size as the content between their tags

The Normal Flow

Inline Elements

- Inline elements take up the same space as the content contained in their tags.
- The elements align side by side, starting from the left, as long as there's space inside of the containing element. If there isn't enough space, then the text or elements will move down to a new line.
- If you add a background to the element you can tell what type of element it is. If the color stretches the width of the container, it's a block element. If it only spans the size of the content, then it's an inline element.

Boxes Normal Flow Layout -Method #2

HTML IDs, or Class can be used for marking up boxes grouping content and for layouts using `<div></div>` together with CSS syntax

HTML

```
<div class= "nav"></div>
```

CSS

```
.nav {  
  width:1024px;  
  height: 300px;  
}
```

We use **html(5) selectors, classes or ids** to define containers or boxes and use divs or html5 semantic grouping elements to group them

Attributes for boxes:

- ❑ width in pixels - **(px)** or percentages **(%)** HTML Selector/ID or Class {width: 1000px;}
- ❑ height in pixels- **(px)** or percentages **(%)** HTML Selector/ID or Class {height: 1000px;}
- ❑ background color-(box background) HTML Selector/ID or Class {background-color: red;}
- ❑ border:
 - placement** -(top, right, bottom, left) HTML Selector/ID or Class { border-top-style:dotted;}
 - width** -(thick, medium thin) HTML Selector/ID or Class { border-width:medium;}
 - style**- (solid, dotted, dashed, double, groove, ridge, inset, outset)
HTML Selector/ID or Class { border-style:dotted;}
 - color** (text color inside the box) - HTML Selector/ID or Class {color: red;}
 - border-color** HTML Selector/ID or Class {border-color: red;}
 - padding** - (space between inside border and content in the box)
HTML Selector/ID or Class {padding:0px 0px 1px 2px;}
- ❑ margins - (space outside the border) HTML Selector/ID or Class {margin: 0px 0px 1px 2px;}
- ❑ float - (where the box is positioned) HTML Selector/ID or Class {float:left;}
- ❑ clear - (returns to the next line or clears one side) HTML Selector/ID or Class {clear:both;}

HTML PAGE:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Untitled Document</title>
<style>
  h1 {
    width: 250px;
    height: 200px;
    background-color:lightblue;
  }

  h2 {
    width: 200px;
    height: 150px;
    background-color:red;
    padding:20px;
    border-bottom:dotted;
  }

</style>
</head>

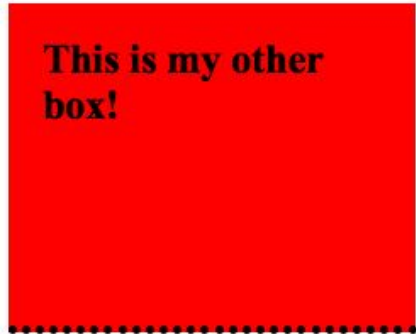
<body>
  <h1>This is my box!</h1>
  <h2>This is my other box!</h2>

</body>
</html>
```

Preview Result:



This is my box!



**This is my other
box!**

Box Formatting Options

Padding -Places space within an element

Same for border which is between margin and padding - (top ,right, bottom, left)

HTML Selector/ID or Class { padding: 1px 5px 3px 1px; }

HTML Selector/ID or Class { padding-left: 1px; }

Margin -Places space around an element

Same as padding/border - (top ,right bottom left)

HTML Selector/ID or Class {margin-left: 10px;}

HTML Selector/ID or Class {margin: 10px 10px 10px 10px; }

NOTE:

*If you want to center a box on the page horizontally (or center it inside the container that it sits in), you can set the **left-margin** and **right-margin** to **auto**. This must also be combined with the **width** property*

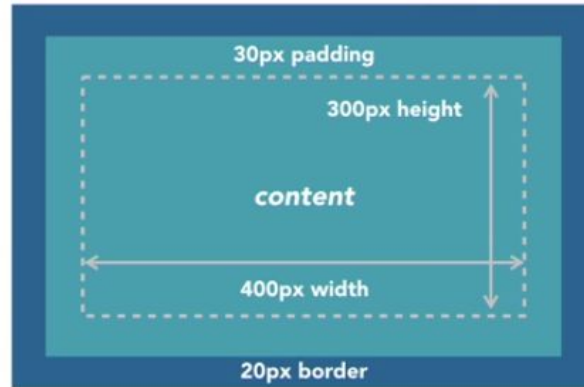
HTML Selector/ID or Class {
margin: 10px auto 10px auto; }

Or

HTML Selector/ID or Class {
margin: 0 auto;}

Box Formatting Options

***Remember All elements increase in size when adjusting any of the properties



$400 \text{ width} + 30 \text{ left padding} + 30 \text{ right padding} + 20 \text{ left border} + 20 \text{ right border} = 500\text{px}$

$300 \text{ height} + 30 \text{ left padding} + 30 \text{ right padding} + 20 \text{ left border} + 20 \text{ right border} = 400\text{px}$

Box Normal Flow -Locations

Float -specifies whether or not an element should float and where. Elements after a floating element will flow or fill around it (**none, left, right, inherit**)

```
HTML Selector/ID or Class { float: left; }
```

Clear -is used to control the behavior of the floating elements property and will clear the float so it does not wrap. Also specifies on which side of an element floating elements are not allowed to float(**left, right, both, none**)

```
HTML Selector/ID or Class { clear: both;}
```


Rendering

-Moz and Webkit are a web browser rendering, rendering engine used by Safari and Chrome (among others, but these are the popular ones). The *-webkit* prefix on CSS selectors are properties that only this engine is intended to process, very similar to *-moz* properties.

These are the vendor-prefixed properties offered by the relevant rendering engines (*-webkit* for Chrome and Safari; *-moz* for Firefox, *-o* for Opera, *-ms* for Internet Explorer). Typically they're used to implement new, or proprietary CSS features, prior to final clarification/definition by the W3.

This allows properties to be set specific to each individual browser/rendering engine in order for inconsistencies between implementations to be safely accounted for. The prefixes will, over time, be removed (at least in theory) as the unprefixed, the final version, of the property is implemented in that browser.

Additional Box Formatting

border-radius Sets the corners of the box

CSS Syntax:






```
HTML Selector/ID or Class{
border: 5px solid #cccccc;
border-radius: 10px;
-moz-border-radius: 10px;
-webkit-border-radius: 10px;
}
```

box-shadow creates a shadow on the box (needs size and color attributes)

CSS Syntax:

```
HTML Selector/ID or Class {
width: 300px;
height: 100px;
background-color: yellow;
box-shadow: 10px 10px 5px #888888;
}
```

Browser Support:

Property					
border-radius	5.0 4.0 -webkit-	9.0	4.0 3.0 -moz-	5.0 3.1 -webkit-	10.5

CSS Block and Inline Elements

display element is used for **HTML Selector/ID or Class** to change the behavior of elements in the normal flow

- ❑ `display:inline;`

- ❑ `display:none;`

- ❑ `display:block`

- ❑ `display:inline-block`

- ❑ used with block elements like `` to make them display inline

- ❑ If used with an inline element like `` the inline element would ignore width and height unless you use `display:block;`

- ❑ If using `display:inline-block` it will apply characteristics of both inline and block

HTML PAGE:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Boxes and Containers</title>
<style>
    li {
        display:inline;
        text-decoration:none;
        margin:10px;
        width:100px;
        background-color:lightblue;
    }

</style>
</head>
<body>
<ul>

    <li><a href="#">Link 1</a></li>
    <li><a href="#">Link 2</a></li>
    <li><a href="#">Link 3</a></li>

</ul>

</body>
</html>
```

Preview Result:

[Link 1](#) [Link 2](#) [Link 3](#) |

HTML PAGE:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>Div</title>

    <link href="css/style.css" rel="stylesheet" type="text/css" />
</head>
<body>

    <div class="header"> <h1>City Gallery</h1></div>

    <div class="sidebar">
        <p>London</p>
        <p>Paris</p>
        <p>Tokyo</p>
    </div>

</body>
</html>
```

CSS:

```
@charset "UTF-8";

/* CSS Document */

.header {

    background-color:#000000;
    width:1024px;
    height:200px;
}

h1{
    color:#ffffff;
    text-align:center;
}

.sidebar {

    background-color:#5F5F5F;
    color:#ffffff;
    width:150px;
    float:left;
```

Result:

City Gallery

London

Paris

Tokyo

Method #3

HTML 5 Selectors/ IDs, or Class can be used for marking up boxes grouping content and for layouts similar to `<div></div>`

- ❑ `<header></header>`
- ❑ `<nav></nav>`
- ❑ `<main></main>`
- ❑ `<section></section>`
- ❑ `<article></article>`
- ❑ `<aside></aside>`
- ❑ `<footer></footer>`

HTML PAGE:

```
<!doctype html>
<html>
<head>
<meta charset="UTF-8">
<title>HTML5 -CSS</title>
<link href="css/style.css" rel="stylesheet" type="text/css" />
</head>

<body>

    <header>This is my Header Info</header>

    <nav>
    <a href="#">Link 1 </a>
    <a href="#">Link 1 </a>
    <a href="#">Link 1 </a>
    </nav>

    <section>
        <article>Article 1</article>
        <aside>Aside 1</aside>
    </section>

    <footer>&copy; 2017</footer>

</body>
</html>
```

HTML 5 and CSS

CSS:

```
header {
    width:1024px;
    height:90px;
    background-color:lightblue;
}
nav {
    width:1024px;
    height:40px;
    background-color:black;
}

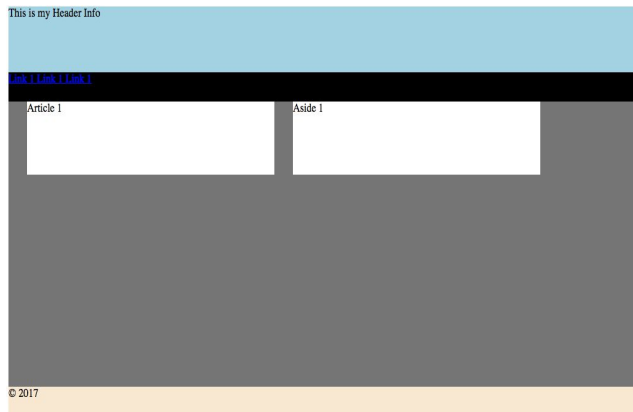
section {
    width:1024px;
    height:390px;
    background-color:gray;
}

article{
    width:400px;
    height:100px;
    background-color:white;
    float:left;
    margin-left:30px;
}

aside{
    width:400px;
    height:100px;
    background-color:white;
    float:left;
    margin-left:30px;
}

footer {
    width:1024px;
    height:40px;
    background-color:antiquewhite
}
```

Result:



This is my Header Info

[Link 1](#) [Link 1](#) [Link 1](#)

Article 1

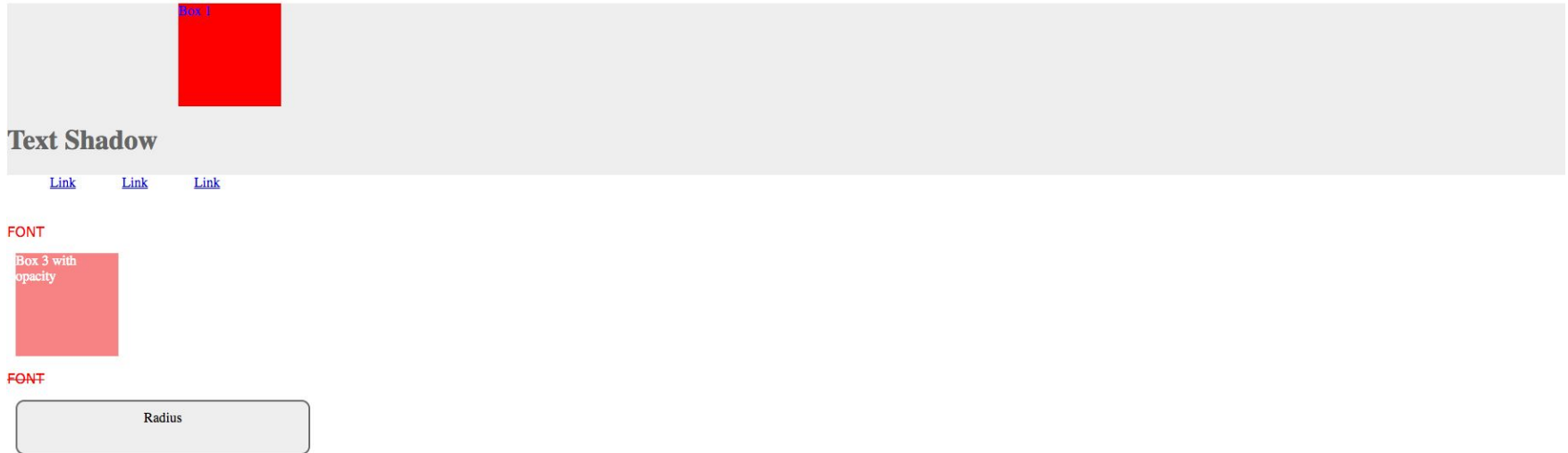
Aside 1

CSS

Exercise

CSS

1. Create a new HTML page called `css1.html` that looks similar to this
2. Link the HTML page to an external stylesheet called `style3.css` (links should hover)
3. Place HTML Page in `public_html/class-samples/` folder
4. Save the `style3.css` page in `public_html/class-samples/css` folder



Create part of the Cartoon Network website in the next slide using HTML 5 and CSS- (Instructor will send images)

1. Name the page `cartoon.html` and save it to `public_html/class-samples/`
2. Link the CSS page to an external stylesheet called `cartoon.css` and save it to `public_html/class-samples/css`
3. Links on page should hover light blue when mouse passes over them.
4. Upload the `images`, `cartoon.css` and `cartoon.html` files up to the web server and send the instructor the link

By Using this site you agree to Cartoon Network's

[Terms of Use](#)



[Games](#)

[Video](#)

[Community](#)

[Apps](#)

[Shop](#)



© Cartoon Network 2017



Homework

HOMEWORK:

- Finish Safety html project and email link to me.
- Finish Cartoon CSS sample and post to web server (email the link to me)
public_html/class-samples/cartoon.html
- Finish <https://teamtreehouse.com/library/how-to-make-a-website>
- Start <https://teamtreehouse.com/library/introduction-to-html-and-css>
- Start CSS Basics <https://teamtreehouse.com/library/css-basics>
- Practice CSS and HTML

End