

## HTML

### *What is the internet?*

- Simple, think of it as something as a long piece of wire that connects computers to other computers.
- Computers that are attached to the internet that are on 24/7 are called Servers. These “serve” you all the data you request to be able to access the internet. The computer that the user uses for this is called the client.

The client's browser uses ISP (Internet Service Provider, like comcast or AT&T) by sending a message to it to access the internet which then connects to the DNS (Domain Name System Server). The DSN searched through it's database to find the exact IP address of the website the user is searching for. Once the DNS finds the IP address, it sends it back to the client through the ISP.

### *How do websites work?*

When a DNS sends the information that was requested by the client, it data that is received usually consists of HTML, CSS, and JavaScript.

- HTML code file: is responsible for the content of the website (images, links, text, buttons)
- CSS code file: is responsible for the styling of the website – determines how the website will look. Targets the content of the website that was created using HTML and applies the styling to said elements.
- JavaScript code file: Allows the website to have functionality – it turns a static website into something a user can actually interact with.

Ex - google homepage



- HTML: content of the website (google logo, two buttons, search term box)
- CSS: modifies the content to actually make the website look nice
- JavaScript: makes google functional

## ♦ Introduction to HTML

### *What is HTML?*

A website cannot be created with JUST JavaScript or CSS, but it can be only created with an HTML file. HTML defines the structure and content of a website.

# All You Need to Know

<a>	<canvas>	<dt>	<iframe>	<meta>	<rp>	<sup>
<abbr>	<caption>	<em>	<img>	<meter>	<rt>	<svg>
<address>	<cite>	<embed>	<input>	<nav>	<ruby>	<table>
<area>	<code>	<fieldset>	<link>	<noscript>	<s>	<tbody>
<article>	<col>	<figcaption>	<kbd>	<object>	<samp>	<td>
<aside>	<colgroup>	<figure>	<keygen>	<ol>	<script>	<template>
<audio>	<data>	<footer>	<label>	<optgroup>	<section>	<textarea>
<ch>	<datalist>	<form>	<legend>	<option>	<select>	<tfoot>
<base>	<dd>	<head>	<li>	<output>	<small>	<th>
<bdi>	<del>	<header>	<link>	<p>	<source>	<thead>
<bdo>	<details>	<hgroup>	<main>	<param>	<span>	<title>
<blockquote>	<dfn>	<h1> to <h6>	<map>	<picture>	<strong>	<tr>
<body>	<dialog>	<hr>	<mark>	<pre>	<strong>	<ul>
 	<div>	<html>	<menu>	<progress>	<sub>	<var>
<button>	<dl>	<i>	<menuitem>	<q>	<summary>	<video>
						<wbr>

- **HTML:** HyperText Markup Language
  - **HyperText:** Pieces of text which can link to other documents in the website. The Foundation of how an HTML website works.
  - **Markup Language:** shows what text should be bolded, italicized, etc. Similar to a manuscript edit. This is done through HTML tags.

## Heading Elements

# This is a heading

TABLE OF CONTENTS	
	REVISION HISTORY
	APPROVALS
1.	PURPOSE
2.	POLICY
3.	SCOPE
H2	REFERENCES
4.	DEFINITIONS
5.	ACRONYMS
7.	RESPONSIBILITIES
8.	PROCEDURE
H3	1. Record Type and Retention
	QUALITY SYSTEM RECORDS
10.	DEVICE MASTER RECORDS (DMR)
11.	DEVICE HISTORY FILE (DHF'S)
12.	COMPLAINT AND COMPLAINT INVESTIGATION FILES
12.1	Obsolete Records
12.2	Quality Records
12.3	Quality Manager (QM)
12.4	Non-exhaustive List of Quality Records
13.	RECORDS

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

**Heading 1**

**Heading 2**

**Heading 3**

**Heading 4**

**Heading 5**

**Heading 6**

```
1 <h1>Beetles</h1>
2 <h2>External morphology</h2>
3 <h3>Head</h3>
4 <h4>Mouthparts</h4>
5 <h3>Thorax</h3>
6 <h4>Prothorax</h4>
7 <h4>Pterothorax</h4>
8
```

## Beetles

## External morphology

## Head

### Mouthparts

## Thorax

## Prothorax

## Pterothorax

The purpose of heading elements is to show the document structure.

- Starts with an opening end <h1>
- Ends with an ending tag </h1>
- Content goes in between the opening and closing tags

## Tag vs elements

- **Tag:** Anything inside the angle brackets <> and </> is a
- **Element:** consists of closing and opening tags, and the content between the tags

## Paragraph Element

`<p>This is a paragraph</p>`

This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.

This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph. This is a paragraph.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vestibulum rhoncus at nunc vitae sodales. Cras faucibus odio at nibh aliquam rutrum. Etiam mattis velit non libero sagittis. Pellentesque auctor lacus ut congue vestibulum. Nam urna risus, sollicitudin nec eros aliquam, efficitur dictum ipsum. Donec consequat lacinia viverra. Aenean eget erat velit. Vestibulum fringilla augue vitae est finibus ullamcorper.

Phasellus condimentum tellus sapien, eu gravida augue vehicula non. Aliquam diam orci, pharetra at viverra quis, suscipit a massa. Proin turpis nisi, mollis quis arcu et, sagittis dignissim arcu. Nunc a dignissim eros. Etiam nisi justo, imperdiet nec porta vitae, hendrerit in turpis. Proin risus augue, elementum nec ex et, pharetra aliquam magna. Pellentesque sodales turpis libero, et placerat dolor pretium a. Cras eget nulla vel enim mattis gravida sed sed urna. Fusce lorem nunc, lobortis id interdum at, convallis id risus. Aenean at odio convallis, posuere turpis tincidunt, finibus urna. Maecenas ac veneratis eros. Nunc lobortis, nisi eget suscipit sagittis, nisi ipsum ultrices lectus, id facilisis diam velit a nisi.

Generated 2 paragraphs, 169 words, 1146 bytes of Lorem Ipsum

**Lorem ipsum:** “dummy text” of the printing and typesetting industry. It shows a normal distribution of letters which is more readable to the user. (placeholder text)

## Void Elements

`<hr />` and `<br />`

A void element is an element that does not allow content.

`<hr />` - horizontal rule element

```
<p>This is a paragraph</p>  
<hr />  
<p>This is a paragraph</p>
```

This is a paragraph

This is a paragraph

- Looks similar to a closing tag of an HTML element, however there is a space and forward slash before the end of the tag.
- Divides the content of the HTML file
- Can also look like this → `<hr><br>`

`<br />` - break element

```
<p>  
To see a World in a Grain of Sand<br />  
And a Heaven in a Wild Flower,<br />  
Hold Infinity in the palm of your hand<br />  
And Eternity in an hour.<br />  
</p>
```

To see a World in a Grain of Sand  
And a Heaven in a Wild Flower  
Hold Infinity in the palm of your hand  
And Eternity in an hour.

- Similar to the horizontal rule element
- Used to separate content onto separate lines to convey meaning
- DONT add a break element at the end of a paragraph as it is not good for accessibility! Create new paragraph instead and only use a break element when needed.

## ◆ Intermediate HTML

List Elements: ordered and unordered

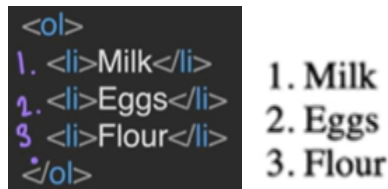
Most websites are full of list (FBI 10 most wanted fugitives, click bait)

`<ul></ul>` - unordered list



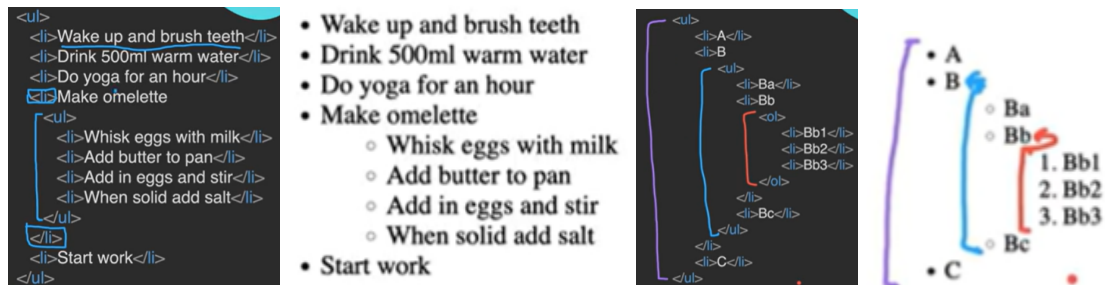
- These two tags are separated by lists items (`<li>content</li>`)

`<ol></ol>` - ordered list



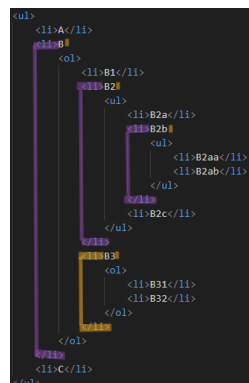
- Similar to unordered list, except that the items are order with numbers based on how the items are in the ordered list

## Nesting and Indentation



- Lists can be nested in another list
- It is important to have indentation as it can be visually difficult to interpret what is being conveyed if there is no indentation

- A
  - B
    1. B1
    2. B2
      - B2a
        - B2aa
        - B2ab
      - B2b
      - B2c
    3. B3
      1. B31
      2. B32
  - C



-When embedding a list into another one, find the list item that you want to create another list in and after the content of the list item, hit enter and create the next list. Do this for every embedded list!

## Anchor Elements HTML attributes

<https://developer.mozilla.org/en-US/docs/Web/HTML/Element/a>

The anchor elements allows to create HyperLinks.

`<a>This is a link</a>` → not an active hyperlink

`<a href="http://www.google.com">This is a link</a>` → correct!

- The additional attribute for the HTML element goes in the opening tag - after the name of the opening tag and just before the closing angle bracket at the end of the opening tag.
- `<tag attribute=value >Content</tag>`
  - Name of element, space
  - Attribute name=value of attribute

```
<tag attribute=value anotherattribute=value>Content</tag>
```

- When having multiple attributes, separate each one by a space. They all will go into the tag

```
<a draggable=true  
This is a link to Google  
</a>
```

- A global attribute can be applied to ANY HTML element.

## The Image Element

```

```

```

```

Src = source of image → renders the image (tells the image element what the source of the image in)

"Url" = location of image

- Is a self closing tag = void element
- 

Alt Attribute: alternative text description. Describes image

```

```

```
<img alt="Dolphin leaping from the sea" sizes="(max-widt  
h: 399px) 240px, 480px" srcset="https://ichef.bbci.co.uk/
```

- Accessibility for visual impairment users.

## ◆ Multi-Page Websites

### File Paths

A file path is a unique location for a file or a folder on a computer.

- A file path on a computer directs the computer to look inside a specific location on a hard drive.

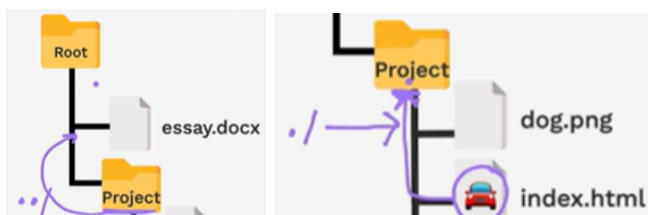
**Absolute:** A file path that is relative to the root of the computer; the file path can direct you to a specific location using the absolute file path

- No matter where you are in the file structure, the computer would know how to go from the very beginning to the desired location

**Relative:** Does not have to be specified from the root. Files can be moved

- More useful for web development
- Usually shorter than the absolute file path.

### Special Characters



**../** : written at the beginning of the relative file path, it means to go up a level from written file

**./** : written at the beginning of the relative file path, it means to stay within the current directory and look within it

```
<h1>All the Animals</h1>
<h2>Rabbit:</h2>

<!-- ./ takes us to all the contents inside Folder0(parent directory),
      Rabbit is inside the directory of Folder0-->

<h2>Cat:</h2>

<!-- ./ takes us to all the contents inside Folder0(parent directory),
      which includes Folder3.cat is inside Folder3-->

<h2>Dog:</h2>

<!-- ../ takes us to 4.0 File Paths(a level UP from parent directory Folder0),
      Dog is on the same level as Folder0 and Folder1. -->

<h2>Fish:</h2>

<!-- ../ takes us to 4.0 File Paths(a level UP from parent directory Folder0),
      then takes us to Folder1 where fish is located-->

<h2>Bird:<

<!-- ../ takes us to 4.0 File Paths(a level UP from parent directory Folder0),
      Folder2 is located inside Folder1, which has bird-->
```

4.0 File Paths

Folder0

Folder3

goal.png

index.html

rabbit.png

solution.html

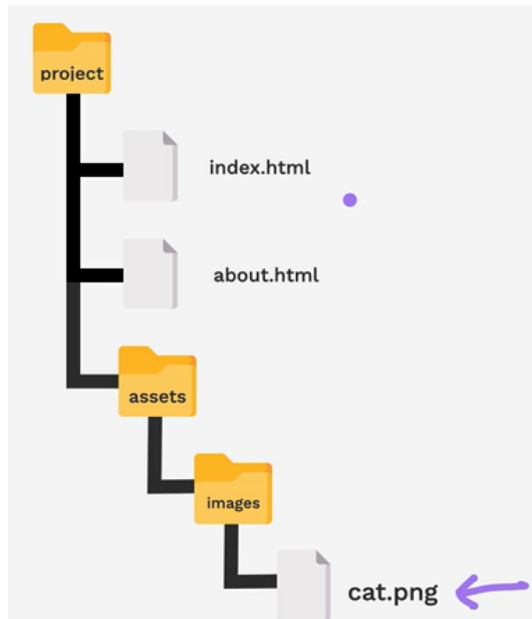
Folder1

Folder2

bird.png

fish.png

dog.png

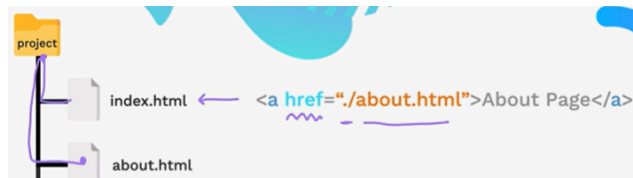


```

```

### *What are WebPages?*

MultiPage website files are kept in the same project folder.



- This anchor tag links the user to the about page FROM the home page(index). The current directory is the Project folder which had the about html file

## The HTML Boilerplate

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>My Website</title>
  </head>

  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```



- `<!DOCTYPE html>`: tells browser that code is in HTML
- `<html lang="en"></html>`: root of document. Inside the opening and closing html tag. Lang is the language the code is written in.
- `<head></head>`: important information of website that is not displayed to the user.
- `<meta charset="UTF-8">`: ensures that the characters on the website are displayed correctly (some charset don't allow emojis or special characters)
- `<title></title>`: Title of the website (displayed in tab bar)
- `<body></body>`: contains the content of the website (text, titles, images, everything we've done so far).

```
index.html > html > head > meta
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta http-equiv="X-UA-Compatible" content="IE=edge">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Document</title>
8 </head>
9 <body>
10
11 </body>
12 </html>
```

- Defines how the website should be displayed relative to the screen it is rendered on
  - vscode shortcut for boilerplate → type “!” click “Enter”
  - Only works if the file is named file.html



## CSS

### What is CSS?

CSS: Cascading Style Sheets. Think of levels that drop down to other levels

Style Sheets: language that allows to specify how things should look in a website.

### How to add CSS/Styles



- Goes into the same line as a particular HTML element
- Goes into the opening tag of the HTML; style attribute which is globally available to all tags which is assigned to a value (the value is the CSS code property that you want to change and the value for desired setting)
- Can be tedious and not recommended, should only be used for specific sections
- TARGETS AN ELEMENT

```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <title>Inline</title>
</head>

<body>
  <h1 style="color: blue;">Style Me in Blue!</h1>
</body>

</html>
```



```
<html>
  <head>
    <style>
      html {
        background: red;
      }
    </style>
  </head>
</html>
```

- **Selector:** comes before a set of curly braces and the CSS goes in between the curly braces. The "selected" element in the example applies to the html element
- This is different from inline because it can be applied anywhere within the same HTML document(html, head, etc)
- Not good for multi page websites
- TARGETS A WEBPAGE

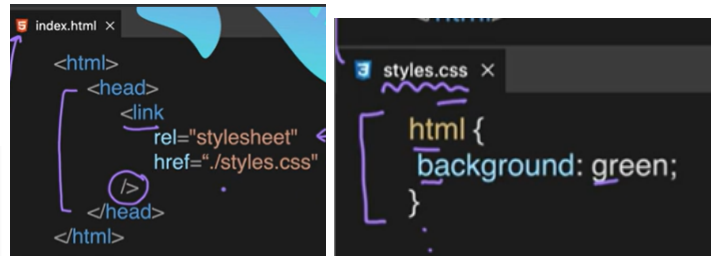
```
<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <title>Internal</title>
  <style>
    h1 {
      color: red;
    }
  </style>
</head>

<body>
  <h1>Style Me in Red!</h1>
</body>

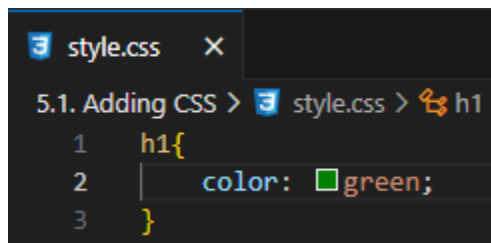
</html>
```

**External**  
<link href="style.css"/>



- Is in a completely different file from the html
- In order to link these two pages together, <link />
  - `rel="stylesheet"` → relationship or role
  - `href="./styles.css"` → location
- Most commonly used in web development
- TARGETS A WEBSITE

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>External</title>
  <link
    rel="stylesheet"
    href="./style.css"
  />
</head>
<body>
  <h1>Style Me in Green</h1>
</body>
</html>
```



```
5.1. Adding CSS > style.css > h1
1  h1{
2    color: green;
3  }
```

## ◆ Introduction to CSS

### CSS Selectors

We see this in the External and Internal styles. Comes before a set of curly braces and the CSS goes in between the curly braces – linked in the head section of the boilerplate and also written in the head section of the boilerplate.

## Element Selector

```
h1 {  
  color: blue  
}
```

color: blue

- 'Property:Value' (value changes property)
- **Element Selector**: selects the part of the HTML to apply rule

## Class Selector

```
.red-heading {  
  color: red  
}
```

```
<h2 class="red-text">Heading 2</h2>  
<h3>Heading 3</h3>  
<p class="red-text">Paragraph</p>
```

- .nameOfClass
- A class can be added as an attribute to any HTML element. Targets specific elements with the class. Multiple types elements can have the same class.
- 

## ID Selector

```
#main {  
  color: red  
}
```

- .nameOfID
- Selects all elements with the particular ID – works similarly to class selector, but only works with a single element in a single HTML file. UNIQUE!
- 

## Attribute Selector

```
p[draggable]{  
  color: red  
}
```

```
<p draggable="true">Drag me</p>  
<p>Don't drag me</p>  
<p>Don't drag me</p>
```

```
p[draggable] {  
  color: red;  
}
```

```
<p draggable="true">Drag me</p>  
<p draggable="false">Don't drag me</p>  
<p draggable="false">Don't drag me</p>
```

```
p[draggable="false"] {  
  color: red;  
}
```

- htmlElement[attribute]
- Specific html elements is applied with the attribute
- The value of the attribute can be selected

## Universal Selector

```
* {  
  color: red  
}
```

```
* {  
  color: red;  
}
```

```
<h1 class="title">Hello</h1>  
<h2 id="heading">World</h2>  
<p draggable="true">  
  This is a website  
</p>
```

- Select all → selects all elements and applies

## ♦ CSS Font Properties

### Colors

[CSS colors](#)

<https://colorhunt.co/>

<https://redketchup.io/color-picker>

### Color Property:

- Color of text

### Font-weight:

**normal**  
**bold**

Keywords

**lighter** ~ 100  
**bolder** ~ 100

Relative to Parent

**number** .

100-900  
light bold

### Font-size

- 1px (pixel) = 1/96th inch (.26mm)
- 1pt (point) = 1/72nd inch (.35mm)
- 1em ("m", full width of m) = 100% of parent
  - If <body> tag is enclosing an <h1> tag, then the h1 would be set to whatever the body tag is set to
  - 2em would be 2 times the size
- 1rem = relative size to the root of the html file
  - If parent is changed at some point, rem does not change (stays relative to parent). ONLY CHANGES if the html element is changed
  - 2rem would be 2 times the root size

## Font-family

<https://fonts.google.com/>

```
h1 {  
  font-family: Helvetica, sans-serif  
}  
  
h2 {  
  font-family: "Times New Roman", serif  
}
```

- Font name, generic font type (Mac users, Windows Users)
- Sans serif = all the edges of letters are at right angles
- Quotations are used when the font family name is longer than 1 word

## Text Align

### ◆ Inspecting CSS

<https://appbrewery.github.io/just-add-css/>

- click three dots on top right in chrome bar
- Got to More tools → developer tool
  - Control + shift + I
- You can also right click on an element then click inspect

“Changes” made in the chrome developer tool will be live, but the website will not be affected. Nothing is changed in the companies servers, only what is loaded on our website.

Computed tab: shows what is actually being applied to CSS property

- three dots in chrome developer tools → more tools → CSS overview → capture overview

<https://appbrewery.github.io/css-inspection/>

Q.1 What is the named color of the body?

A: aliceblue

Once chrome developer tools is open, click on body tag in “elements”, then “styles” in the right bar.

Q.2 What is the font size of the h1?

A: 3rem

- Click head tag, click style tag in “elements”
- Body tag, h1 tag (in styles bar on right)

Q.3 What is the font weight of the h2?

A: 500

- Click head tag, click style tag
- Body tag, h2 tag, computed and styles bar

Q.4 What is the font family of the p?

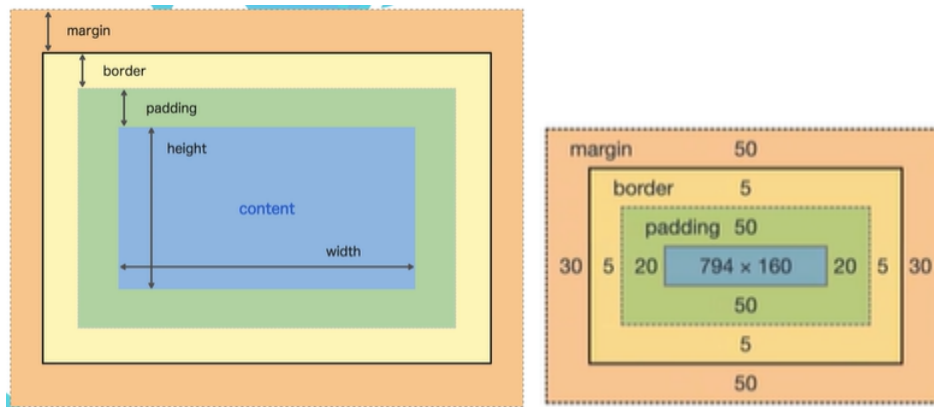
A: Arial

- Click p tag in elements bar, look in computed and styles bar

## ◆ CSS Box Model

(Margin, Padding, Border properties)

<https://appbrewery.github.io/box-model/>



Each element has its own “invisible box” – the dimensions of the boxes are width(horizontal spacing) and height(vertical spacing). Can be defined by pixels or percentages.

### Border

The border of the “box”. takes 3 values (thickness, style, color) separated by space.

- Ex: border: 10px solid black
- When the border is changes, the height and the width of the box is not changed

**Border-top:** top of border, comes after border element

**Border width:** takes 4 values (top, right, bottom, left), can also have 2 values (top and bottom. Right and left).

- Ex: border-width: 0px 10px 20px 30px
- Ex: border-width: 0px 20px (top/bottom, left/right)

### Padding

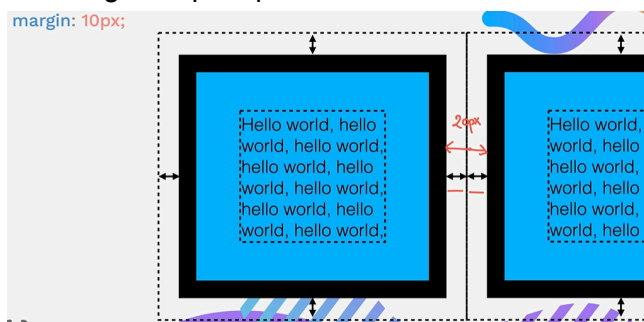
This is basically a “pad” between the box and the border. 4 values

- Ex: padding: 20px → 20px between box and border on all sides
- The height and width for the box is not changed

### Margin

Outside of the border. Like the spacing between elements that you would not get with padding

- Ex: margin: 10px, up to 4 values

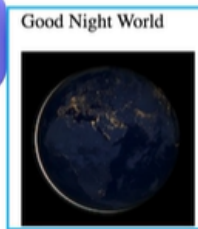


### *Content Division Element*

Useful to group content together (nice and neat!).

`<div>CONTENT</div>` → invisible unless css is applied

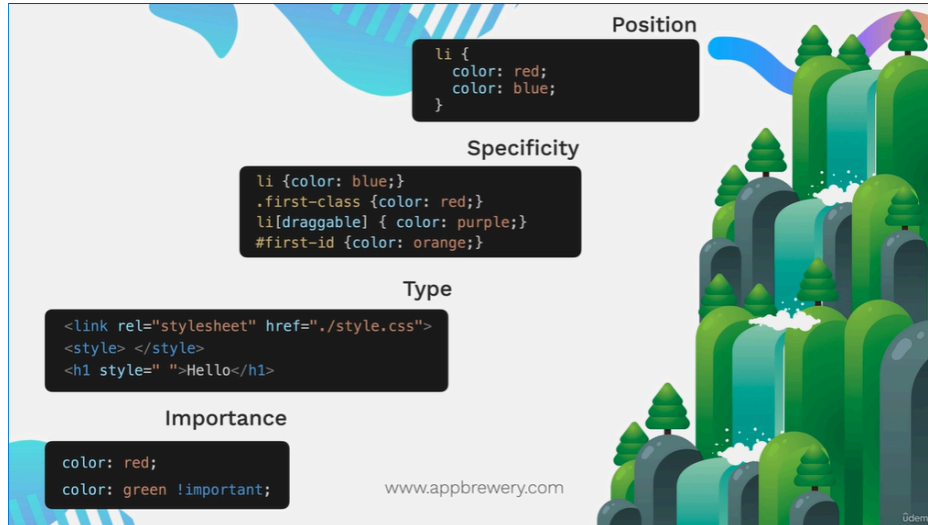
```
<body>
  <div>
    <p>Hello World</p>
    
  </div>
  <div>
    <p>Good Night World</p>
    
  </div>
</body>
```



## Intermediate CSS

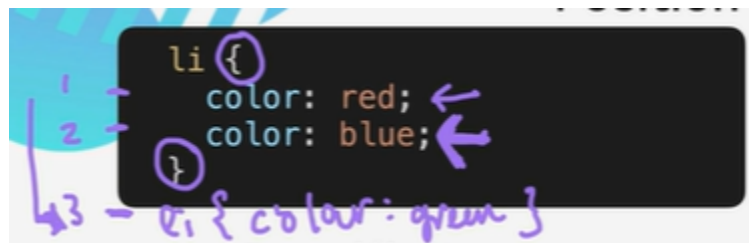
### ♦ The Cascade

If there are different/conflicting CSS rules that target the same element, they will be put into a hierarchy.



### Level of importance

- **Position:** higher or lower position in the CSS – the one that is lower down will be applied (the top one is applied then immediately the lower one is applied). If an element is targeted again with a selector, then that target will be applied to the element.
  - The lower down in a CSS external file or internal style element, the more important it is.



### Specificity

- **Specificity:** refers to how specific a selector is in terms of the element that the rules are being applied to. 4 levels in order (least to most specific)

```
li {color: blue;}
.first-class {color: red;}
li[draggable] { color: purple;}
#first-id {color: orange;}
```

Ex: `<li id="first-id" class="first class" draggable>`

**Class:** will select all the other selected elements that have the class name.

**Attribute:** attribute overrides the class.

**Id:** most specific, it essentially targets a single element with a specific ID.

- Type: 3 styles to apply CSS



```
<link rel="stylesheet" href="./style.css">
<style> </style>
<h1 style="" ">Hello</h1>
```

- **External:** using the link element (linking to a specific file in the in the project folder, can be applied to multiple HTML files if they're linked to the same CSS files)
- **Internal:** through style element, CSS goes in between the brackets (only exists inside 1 webpage)
- **Inline:** CSS is applied through the style attribute, inside the opening tag of an HTML element (applied to particular element)
- **Importance:** keyword applied to any CSS rules.

```
color: red;
color: green !important;
```

- is more important than everything listed above

### ♦ Combining CSS Selectors

How to combine different CSS selectors as to target specific elements to style

```
<p>Yellow Text</p>
<div class="box inner-box">
  <p>White Text</p>
</div>
```

First looks at the element with the inner-box class and the descendants in the div (anything in the div is considered descendent). In this case we are wanting the paragraph element descendent to be white text.

```
p {
  color: yellow;
}
.inner-box p {
  color: white;
}
```

Yellow Text  
White Text

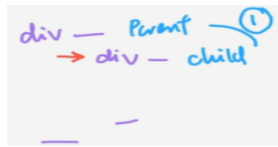
### Group Selector

```
selector, selector {
  color: blueviolet;
}
```

Two selectors separated by a comma. The grouping selects both (or multiple) of the selectors and applies the same style to the entire group.

### Child Selector

```
selector > selector {  
  color: firebrick;  
}
```



Two selectors separated by a greater-than (right-angle bracket) symbol → Parent and child.

- Ex- One div is enclosed in another – this would be a direct descendant and the relationship would be 1 generation. The div that is enclosing the other div is the parent and the enclosed div is the child.
- This is applied ONLY to the direct child of the parent (on left side of right-angle bracket). Child has to be ONE LEVEL inside the parent.

### Descendant Selector

```
selector selector {  
  color: blue;  
}
```

Two selectors not separated by space → ancestor and descendant. The ancestor selector is going to apply the style to the descendant.

### Chaining Selector

```
selectorselector {  
  color: seagreen;  
}
```

Two selectors separated by nothing. Apply the styles to the instances where ALL the selectors are true. SUPER SPECIFIC

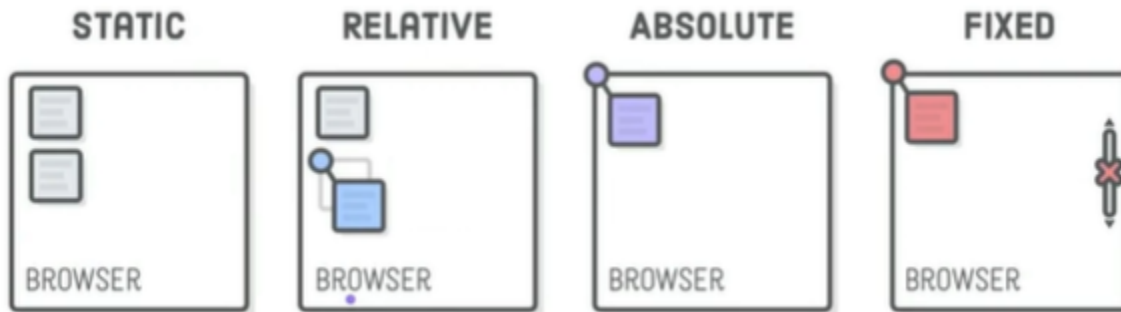
- ex  
 <h1 id="title" class="big heading"> Hello World</h1>  
 h1#title.big.heading → look for h1 with the id of "title", the class of "big" and the class of "heading".
- In the selector, the element must come first and then the class is added or id. Be careful to not use spaces!!

### Combining Combiners:

```
selector selectorselector {  
  font-size: 0.5rem;  
}
```

## ◆ CSS Position Property

### [CSS-Positioning](#)



#### *Static Positioning*

- **HTML default.** Each item flows as they would in our HTML. One on top of the other usually with no spaces in between

#### *Relative Positioning*

- **An item that gets positioned relative to its default position** → takes the original position and applies some changes so that it is relative to where it “should” be. Its only relative to its supposed location

#### *Absolute Positioning*

- **Makes the position relative to the nearest positioned ancestor** → in the example, the div enclosing the div containing the purple box would be the ancestor and the purple box inside the div being enclosed would be the descendant.
- If there is no ancestor, then it will be relative to the browser window. If you want it to be positioned to another item, you have to make sure the ancestor is set to relative position.
- Defaults to the top left of the webpage, will not move with scroll.

#### *Z-Index*

- It **determines which element goes on top of which in the Z axis.** If the element has a higher Z-index than another, then the higher Z-index element will sit on top.
- When an item is set to the absolute position, it is taken out of the original HTML flow and is put in on another layer.

#### *Fixed Positioning*

- **Relative to the top left corner of the BROWSER WINDOW.** When you scroll up and down, it will stay in a fixed position.

## Advanced CSS

### ◆ CSS Display Property

#### [CSS Display Property](#)

##### *Block Display Value*

- Most elements have their display property value set as block, which means that the element displayed will be layout as a full width block.
- This means that if there is another element, it will NOT go onto the same line as the first element (unlike inline)



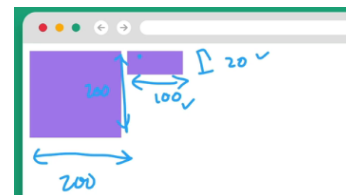
##### *Inline Display Value*

- If there is one element (a span element) and there is another span element, they would be placed on the same line (wrapped).
- The width and height cannot be set because it will default to the size of their content (it will only be as wide as what the element needs to display)



##### *Inline-Block Display Value*

- This is basically a inbetween of inline and block
  - **Inline:** allows word-wrap
  - **Block:** allows to set the height and width



##### *None Display Value*

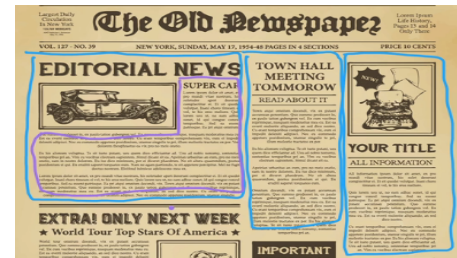
- The element on the screen will basically disappear

##### *<span> element*

- <paragraph> elements usually take up the full width of a webpage, in order for it to basically “wrap up” would be to use the <span> element. It is usually placed between the paragraph opening and ending tag.
- The default display property value is inline → displayed inline with another element

## ◆ CSS Float Property

The float property allows text to wrap around elements (the purple outline is the float text). Setting an image to a float property will move the image left or right and the text will wrap around the images



### Float

- **Left:** The element is floated to the left of its containing block. Other content will flow around it on the right side.
- **Right:** The element is floated to the right of its containing block. Other content will flow around it on the left side.
- **None:** The element does not float and will be displayed in the normal flow of the document. It will not move to the left or right; instead, it will appear where it naturally falls in the layout.
- **Inline-start:** The element is floated to the start edge of its containing block (left side in left-to-right languages and right side in right-to-left languages). This value is part of the Logical Properties and Values specification and is useful for handling text direction in multilingual sites.
- **Inline-end:** The element is floated to the end edge of its containing block (right side in left-to-right languages and left side in right-to-left languages). Similar to inline-start, this value helps in managing content flow based on text direction.



Lorem ipsum dolor sit amet, in velit orci lectus eget diam, dolor.

Et laoreet commodo molestie interdum accumsan, nonummy sed dis, dui ut feugiat interdum aenean ut sed, leo enim nam ipsum morbi, et sed.

Lorem ipsum dolor sit amet, in velit orci lectus eget diam, dolor.

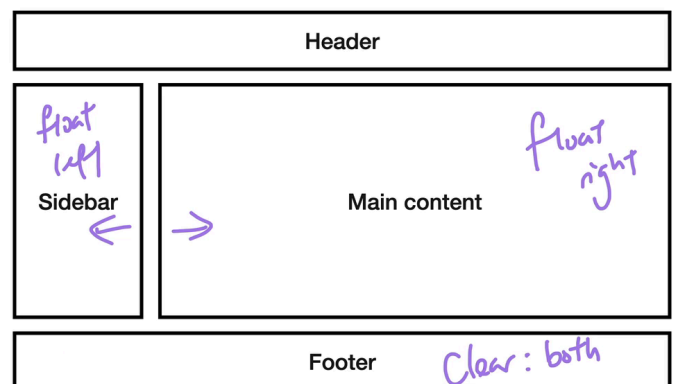


Et laoreet commodo molestie interdum accumsan, nonummy sed dis, dui ut feugiat interdum aenean ut sed, leo enim nam ipsum morbi, et sed.

### Clear Property

This allows a selected element to move below any preceding float elements

- **Left:** The element will move below any preceding floating elements on the left side.
- **Right:** The element will move below any preceding floating elements on the right side.
- **Both:** The element will move below any preceding floating elements on both the left and right sides.
- **None:** The element will not move below floating elements and will be positioned according to normal flow.



## ◆ How To Create Responsive Websites

### [Responsive Example Website](#)

#### Media Query

Is used to apply styles to specific devices or screen sizes. Allows the website to adapt to different device characteristics which ensures better user experience.

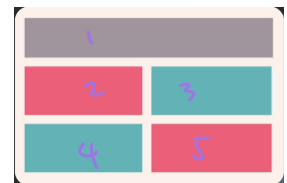
- '@media screen': Targets devices with screens
- 'and (max-width: 600px)': The specified style will be applied to screens with 600px or less.

```
@media screen and (max-width: 600px) {  
  body {  
    background-color: lightblue;  
  }  
}
```

#### CSS Grid

Allows to create complex, responsive web designs. It can handle columns and rows (two-dimensional grid-based system).

- **Grid Container:** The parent elements that holds the grid items. Define the grid container by setting 'display: grid;' or 'display: inline-grid;', on the parent element.
- **Grid Items:** the child elements of the grid container, which automatically become grid items



#### Basic Grid Properties

```
<div class="grid-container">  
  <div class="first card"></div>  
  <div class="card"></div>  
  <div class="card"></div>  
  <div class="card"></div>  
  <div class="card"></div>  
</div>  
  
.grid-container {  
  display: grid; ←  
  grid-template-columns: 1fr 1fr;  
  grid-template-rows: 100px 200px 200px;  
  gap: 30px;  
}  
  
.first {  
  grid-column: span 2;  
}  
  
.card {  
  background-color: blue;  
}
```

- **Grid Columns and Rows:** Defines the number of columns and rows in the grid (There are 2 columns and 3 rows)
  - The columns are of equal width
  - Starting from the top; the first row is 100px, and the second and third rows are 200px
- **Fractional Units:** 'fr' unit, divides the screen into columns (there are 2 columns of equal width)
- **Grid Gap:** Space in between the columns (30px)
- **Placing Items :** Placement of grid items can be controlled (.first is the class of the first child div, which will span 2 columns)

## CSS Flexbox



Allows to create 1D layout, excels at distributing space along a single axis (Horizontal or vertical).

- **Flex Container:** The parent elements that holds the flex items. Define the grid container by setting 'display: flex;' or 'display: inline-flex;', on the parent element.
- **Flex Items:** Child elements inside the flex container

```
<div class="flex-container">
  <div class="first card"></div>
  <div class="second card"></div>
  <div class="card"></div>
  <div class="card"></div>
</div>
```

### Basic Flexbox Properties

- **Flex Direction:** Controls the directions in which the flex items are placed (row(default, horizontal), row-reverse, column-reverse)
- **Justify Contents:** Aligns items along the main axis (values: flex-start, flex-end, center, space-between, space evenly)
- **Align Items:** Aligns items along the cross axis (values: stretch, flex-start, flex-end, center, baseline)
- **Flex Wrap:** Controls if flex items wrap. By default, items try to fit onto one line.
- **Align Content:** Aligns flex lines when there is extra space in the cross axis (only with flex wrapped items). (values: flex-start, flex-end, center, space-between, space-around, stretch)
- **Flex:** a shorthand property got flex-grow/shrink/basis. Defines how a flex item will grow, shrink, and what its base size is. (width)
- **Order:** Controls order of flex items, regardless of HTML order.

```
.flex-container {
  display: flex;
}

.card {
  background: blue;
  border: 30px solid white;
  height: 100px;
  flex: 1;
}

.first {
  flex: 2;
}

.second {
  flex: 0.5;
}
```

### Bootstrap Framework



Not built-in to CSS, built on top of flexbox. 12 Division system → the website can be divided in 12 equal portions. The portions stay in their sizes based on the percentages they take up on their window size.