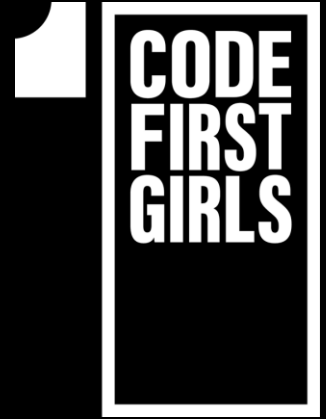


WELCOME TO CFG **YOUR INTRODUCTION** **TO WEB DEVELOPMENT**



TECH SHOULDN'T JUST BE A BOYS CLUB.

COURSE JOURNEY

MODULE 2: CSS

HTML

MODULE 01

CSS



MODULE 02

Recap
Project design

MODULE 03

Javascript
+ Overview, data types
+ Loops, Functions,
scope
+ Objects and the DOM

MODULE 04

Github pages
Frameworks

MODULE 05

Project
presentations
Careers in web
development

MODULE 06

What is CSS and what it is used for?

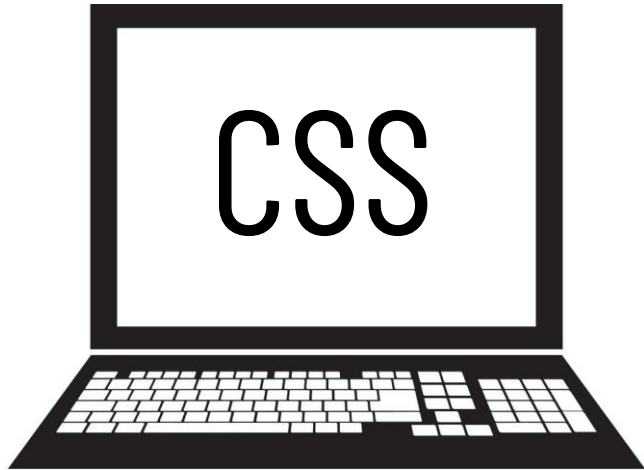
How to link HTML and CSS together

Learn to style a webpage with CSS

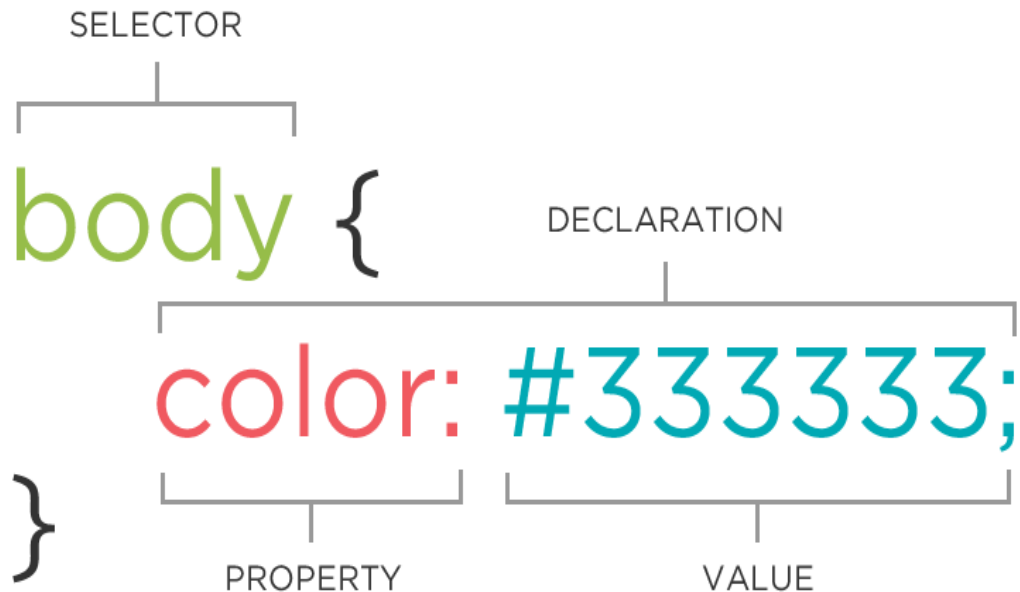
Complete interesting practical exercises

WHAT IS CSS?

CASCADING STYLE SHEET



HOW DO I WRITE CSS?



NOW LET'S PRACTICE TOGETHER

TYPOGRAPHY, COLORS & FONTS

MODULE 2: CSS

5 MINS

Exercise 2.0 - set up

*Create HTML file and CSS file, then link them together

Exercise 2.1

* Target an element and change its color.

Exercise 2.2

* Pick a heading tag and change its font size, make it bold and underlined.

7 MINS

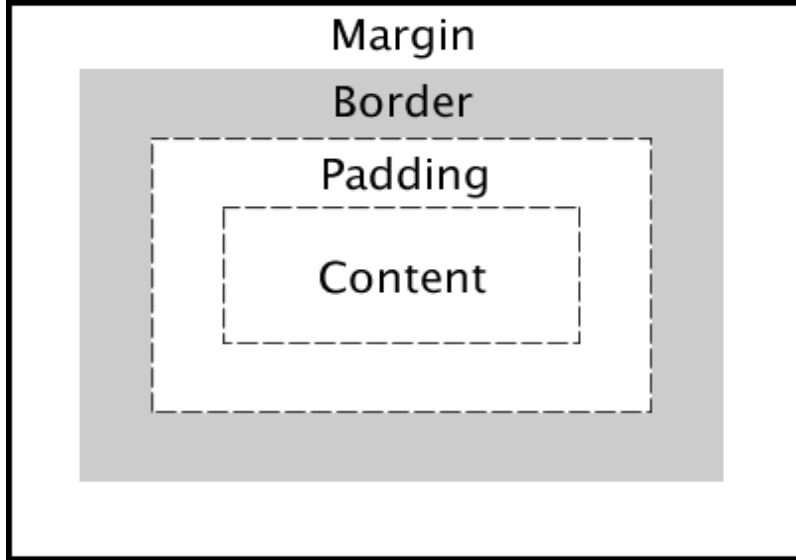
Exercise 2.3

* Import a font from [Google fonts](https://fonts.google.com/), link it to your HTML page and try to use it for one of your lists (ordered/unordered)

GROUP EXERCISE



BOX-MODEL



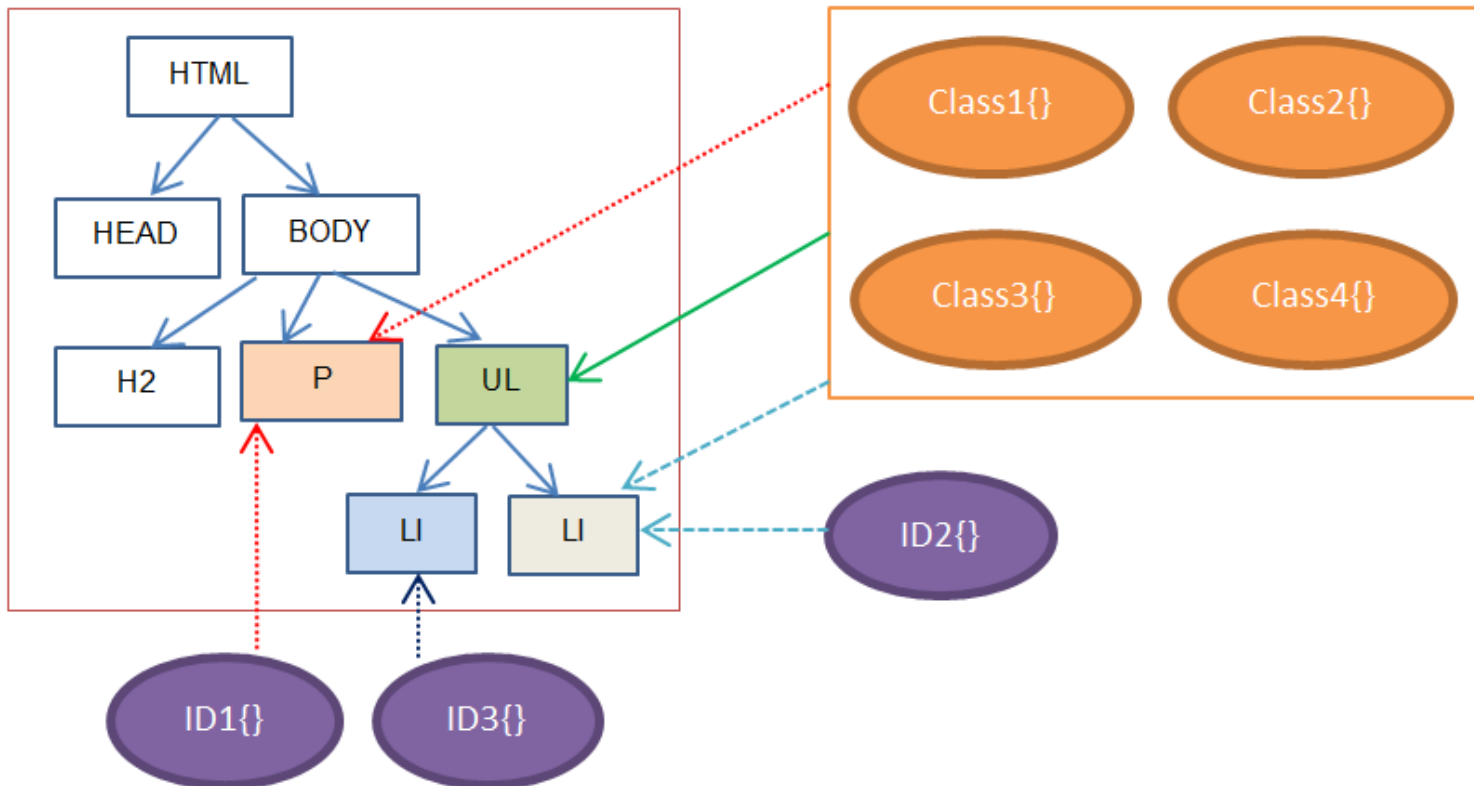
- **Content** - The content of the box, where text and images appear
- **Padding** - Clears an area around the content. The padding is transparent
- **Border** - A border that goes around the padding and content
- **Margin** - Clears an area outside the border. The margin is transparent

GROUP EXERCISE



CSS CLASS & ID

STYLING SPECIFIC ELEMENTS IN A UNIQUE WAY



CSS CLASS & ID

STYLING SPECIFIC ELEMENTS IN A UNIQUE WAY

CLASS SELECTOR	ID SELECTOR
<ul style="list-style-type: none">• Preceded by a dot (.)• Targets any element that contain the given class name in its class attribute• Class can be assigned to any element in HTML and any number of elements can belong to one class	<ul style="list-style-type: none">• Preceded by a hash symbol (#)• Any element can have an ID attribute, but that attribute's value can only be used once within a single document

LET'S PRACTICE TOGETHER

POSITIONING, SPACING & DISPLAY

MODULE 2: CSS

10 MINS

Exercise 2.4

* Enclose two sections of the page into two separate `<div>` tags. Style each of the two divs with different background colors/fonts/different border styles. Try experimenting with the padding and margins to see how you can position content within each div.
(Hint: Use different class names for each div)

GROUP EXERCISE



PRACTICE

10 mins

Exercise 2.7

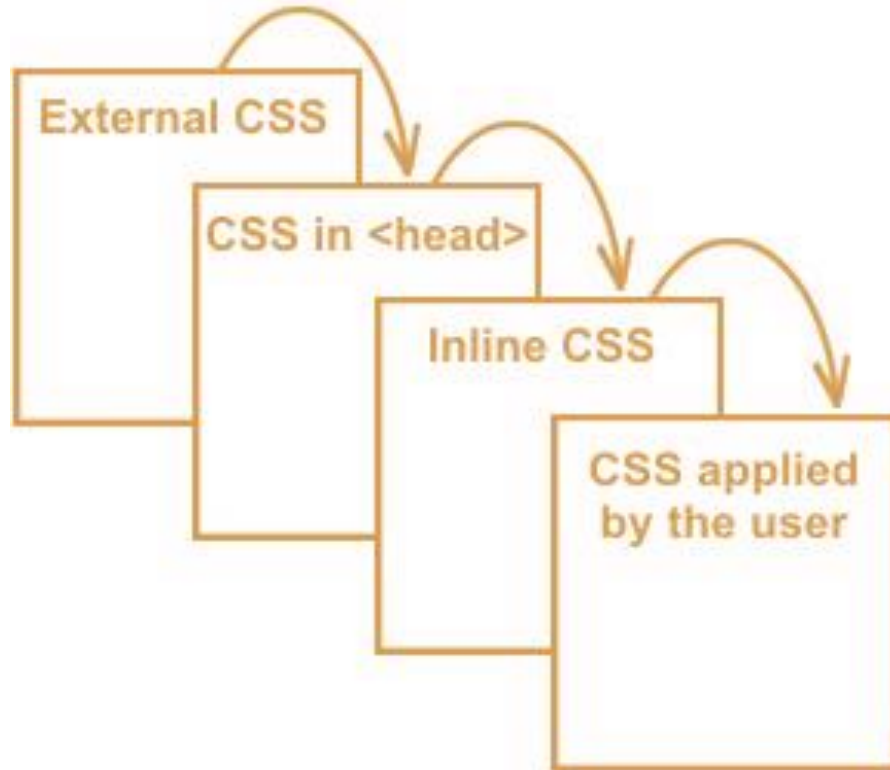
* Find the exercise-starter code in your student folder/on slack

1. Download the code and unzip it
2. Open the folder 'exercise-starter-code' in VScode (Take care to open the folder and not individual files)
3. Go through the code and style the elements according to the instructions in the HTML page

GROUP EXERCISE



WHY ARE THEY CALLED CASCADING STYLE SHEETS?



QUIZ TIME!

What colour would the word 'Style' be?

Welcome to the wonderful world of Cascading Style Sheets

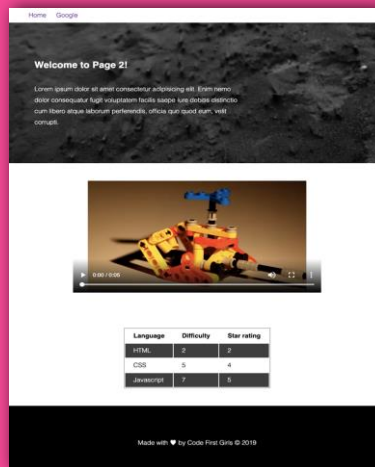
```
<h4>Welcome to the wonderful world of  
  <span class="color color1">Cascading </span>  
  <span class="color color2" id="specificColor">Style </span>  
  <span class="color color3">Sheets</span>  
</h4>
```

```
#specificColor {  
  color: ■brown;  
}  
  
.color {  
  font-size: 30px;  
}  
  
.color1 {  
  color: ■aqua;  
}  
  
.color2 {  
  color: ■#ac2399;  
}  
  
.color3 {  
  color: ■rgb(27, 125, 48);  
}
```

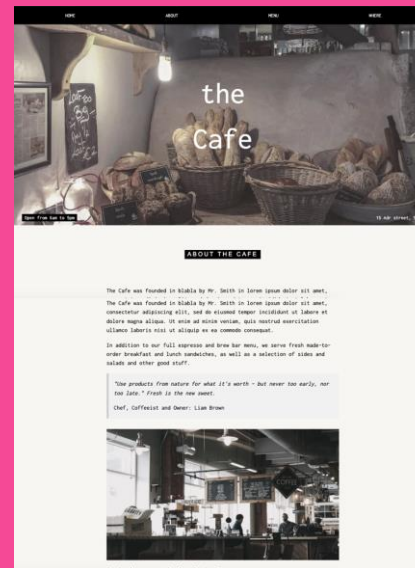
+ Homework Task

Style an index.html page according to the layout specified or choose a simple website you like and try to recreate it.

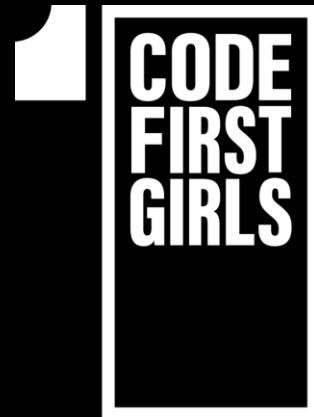
Option 1
(Solution code available)



Option 2
(No solution code available)

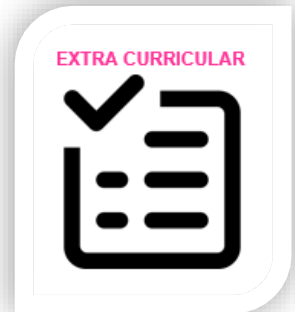


THANK YOU
HAVE A GREAT
WEEK!



SELF STUDY TOPICS

THESE CONCEPTS ARE USEFUL TO KNOW –
WE ENCOURAGE YOU TO SELF STUDY



FLEXBOX

Flex boxes can adjust in size according to content inside them. Individual items within a flex container may also be automatically reordered and rearranged to suit the available layout space

01 Download and unzip the 'flex-exercise-starter-code' from the slack channel.

02 Save and remove all the files you have open in VScode currently

03 Open the extracted **folder** (not individual files) called 'flex-exercise-starter-code' folder.

04 Check that your flex-exercise folder contains:

index.html

CSS

→ main.css

Exercise 2.5

* Follow directions in the flex-exercise-starter-code > main.css file (sent to you/available on slack)

(Refer to <https://css-tricks.com/snippets/css/a-guide-to-flexbox/> for extra tips and tricks)

10 mins



FLEXBOX FROGGY

REFERENCE MATERIALS



HOW TO IMPLEMENT CSS

Inline CSS

BAD PRACTICE

```
<p style="color: blue;">This is a paragraph.</p>
```

Styles are specific to that element and next to impossible to overwrite. Also very impractical if you want the same styles on multiple elements

Internal CSS

BAD PRACTICE

```
<head>
  <style type = text/css>
    body {background-color: blue;}
    p { color: yellow;}
  </style>
</head>
```

While better than inline, internal CSS is specific to the page it's written on so can't be reused across a whole site with multiple pages

External CSS

BEST PRACTICE

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

Code is kept separate so can be used across multiple pages

WILDCARD

```
/* WILDCARD */  
* {  
  /* overlays every item on the page with an outline.*/  
  /* VERY useful for development */  
  outline: 1px solid ■orange;  
}
```

DEFAULT STYLES

```
/* ELEMENTS */  
body {  
  /* Its common to remove default styles */  
  margin: 0;  
  padding: 0;  
}
```

TYPOGRAPHY, COLORS & FONTS

+ Colors and Decoration

- Names - red, blue, yellow etc
- RGB - rgb(255,255,255)
- Hexadecimal - #abc123
- Hex = most popular, hsl = least popular

+ Size

- Pixels - eg 2px
- Percentage - eg 20%
- Em and rem - eg 2em
- Pixels most used, em and rem close behind
- Em and rem used a lot when scaling to larger screens

+ Font family and weight

- Font-family - defaults vs custom
- Custom fonts out of scope for the lesson
- Using ctrl + space at 'font-family: ' shows you all the available fonts

```
h1 {  
  color: blue;  
  text-decoration: underline;  
}
```

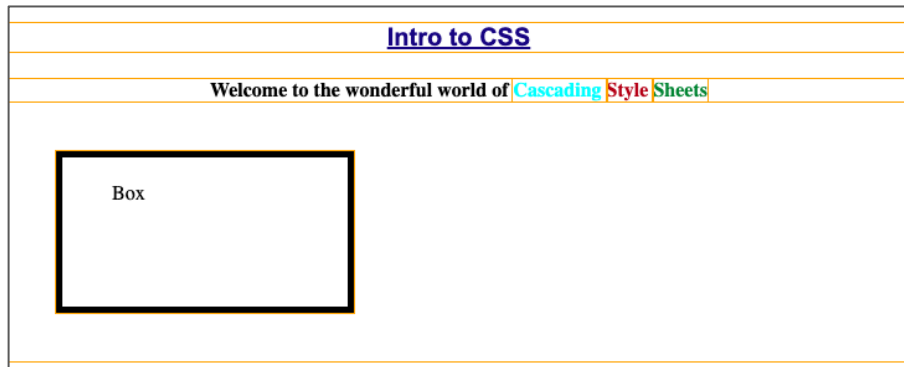
```
h1 {  
  color: blue;  
  text-decoration: underline;  
  font-size: 20px;  
}
```

```
h1 {  
  color: blue;  
  text-decoration: underline;  
  font-size: 20px;  
  font-family: Arial, Helvetica, sans-serif;  
  font-weight: bold;  
}
```

SPACING

Spacing

- Content
- Padding
- Border
- Margin



```
<div class="box">Box</div>
```

```
.box {  
  /* content */  
  width: 150px;  
  height: 80px;  
  /* padding - top-right-bottom-left */  
  padding: 20px 40px 20px 40px; /* or 20px 40px; */  
  /* border - width-style-color */  
  border: 5px solid black;  
  /* margin */  
  margin: 40px;  
}
```

DISPLAY

Display

- Block
- Inline
- Flexbox

/* Flexbox is a powerful way to position your elements */

```
.hero {  
  width: 100%;  
  height: 200px;  
  /* Use the below 3 for perfect centering */  
  display: flex;  
  align-items: center;  
  justify-content: center;  
}  
  
.text {  
  width: 300px;  
  text-align: justify;  
}
```

```
<div class="hero">  
  <p class="text">  
    Lorem ipsum dolor sit amet consectetur adipisicing elit.  
    Quam, soluta quas consequatur facere numquam  
    voluptatum  
    labore, eum ratione dolor placeat assumenda fugit iusto  
    accusamus atque, porro eveniet tenetur  
    perferendis quis!  
  </p>  
</div>
```

Lorem ipsum dolor sit amet consectetur
adipisicing elit. Quam, soluta quas
consequatur facere numquam voluptatum
labore, eum ratione dolor placeat assumenda
fugit iusto accusamus atque, porro eveniet
tenetur perferendis quis!

CLASSES AND IDS

- + Classes are a way to target as many different elements as you want. They don't have to be the same type.

- They are declared on the element with a class attribute
- They are referenced in CSS with a dot before the class name

```
<div class="color">I have a class!</div> .color {  
                                         | font-size: 30px;  
                                         }
```

- + IDs are unique to a page. If you use an ID, you can only use it once in the document as it is unique
 - They are declared on the element with an ID attribute
 - They are generally used for targeting specific elements with JavaScript logic and not for styling.

WHEN DO I USE CLASSES VS IDS

Use classes wherever possible


It is generally regarded as bad practice to use IDs to apply styling for several reasons:

- Class specificity is lower than ID specificity, meaning its harder to overwrite when using IDs
- Classes can be reused, IDs cannot
- A consistent convention - Using only the class attribute to define styles is easier for others to understand instead of a combination of the class and id attributes
- An element can have several classes, but only one ID

THINGS TO REMEMBER

!important

- Eg `color: red !important;`
- Almost impossible to overwrite
- There's nearly always a better way
- Some CSS frameworks use it

```
/* !important */
thead th {
  background:  rgb(0, 255, 55) !important;
}
```

Using IDs as selectors

- IDs are unique
- Better used for targeting with logic (eg Javascript which we'll cover)
- Using IDs to style is not efficient, they are better used to target specific elements with logic (JavaScript)
- If you want to be taken seriously as a developer, it's good to follow best practices. This also makes your code easy to read for others you work with.

```
<hr id="h-rule">
#h-rule {
  margin: 20px 0;
}
```