
KINGSLAND
UNIVERSITY

# CSS

# Table of Contents

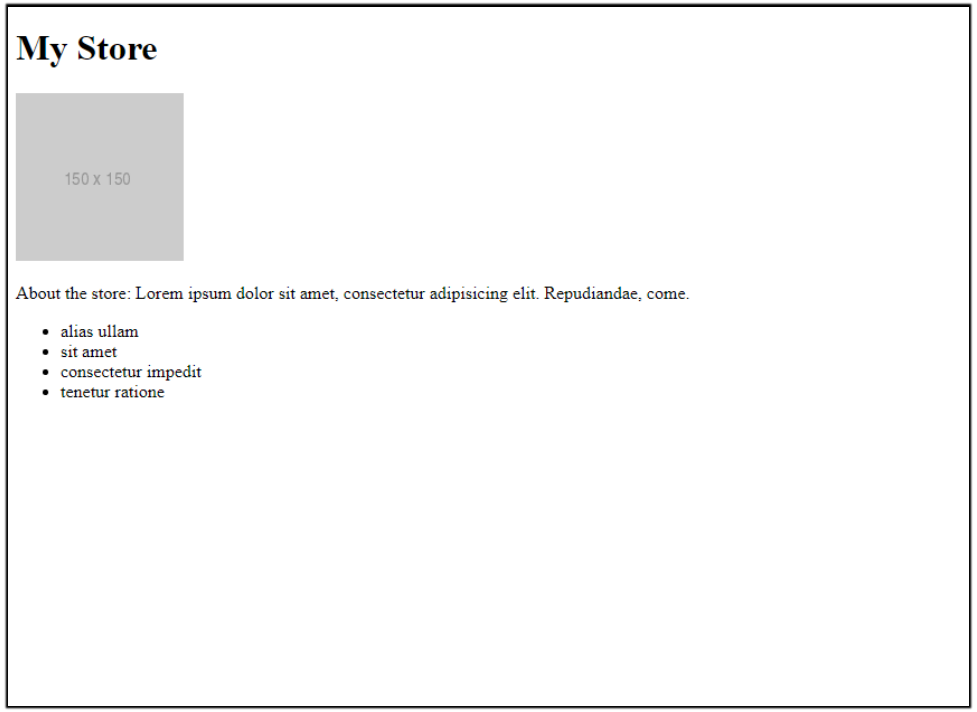

# Introduction to CSS

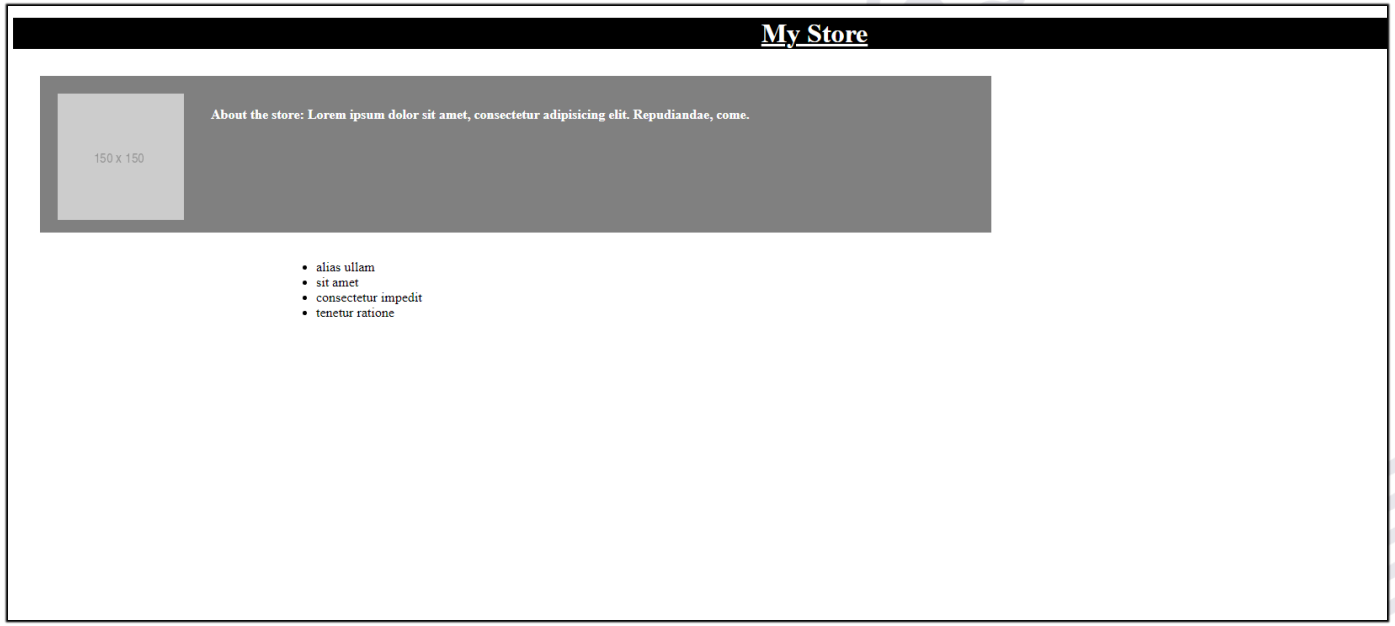
# What is CSS?

- CSS - Cascading Style Sheets – adds styling to an HTML page
- With CSS any and all HTML elements can be manipulated to create a certain effect.
- CSS allows us to separate the styling out of the HTML and into its own area
  - This separates the work and allow us as programmers to focus on one thing at a time

# Why CSS?

- Even a basic page can be made to look a lot nicer with a little bit of styling



# Adding styling without CSS

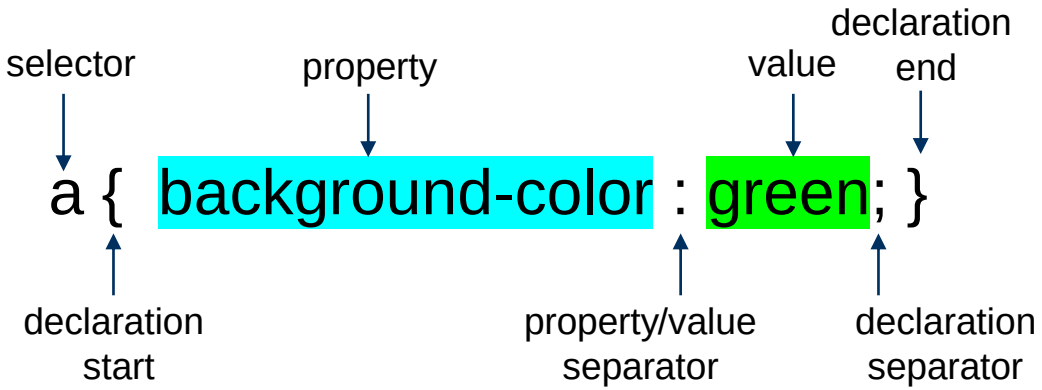
- HTML elements can be individually styled

```
<!DOCTYPE html>
<html lang="en">
     <head>
                <meta charset="UTF-8" />
                <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
                <title>Hello World</title>
     </head>
     <body>
                <h1 background-color= "black" color = "white" > Hello World </h1>
     </body>
</html>
```

- This is nice when your site is small, but as it grows this will become unmanageable

# CSS Syntax

- CSS works by hooking onto **selectors** added to the HTML using **classes** and **identifiers**.
- Once a hook has been added to the HTML we can apply styles to those HTML elements using CSS



# CSS Syntax

- So how do we get the CSS on the page?
  - We put it inside an HTML tag generally put in the head

```
<!DOCTYPE html>
<html lang="en">
     <head>
               <meta charset="UTF-8" />
               <meta name="viewport" content="width=device-width, initial-
scale=1.0" />
               <title>Hello World</title>
               <style>
                    h1 {
                        background-color : blue;
                    }
               </style>
     </head>
     <body>
               <h1> Hello World </h1>
     </body>
</html>
```

```
1   body {
2     margin: 0;
3     font-family: -apple-system, BlinkMacSystemFont, 'Segoe UI', 'Roboto', 'Oxygen',
4       'Ubuntu', 'Cantarell', 'Fira Sans', 'Droid Sans', 'Helvetica Neue',
5       sans-serif;
6     -webkit-font-smoothing: antialiased;
7     -moz-osx-font-smoothing: grayscale;
8   }
9
10  code {
11    font-family: source-code-pro, Menlo, Monaco, Consolas, 'Courier New',
12      monospace;
13  }
14
```

# CSS Attributes

# Key CSS Attricutes

## Font / Color:

- **color** - sets the color of the text
- **font-size** - sets the size oof the font
- **font-style** - sets italics
- **font-weight** - sets how bold the font is

## Alignment / Spacing:

- **padding (top/right/bottom/left)** - adds space between the element and its own border
- **margin (top/right/bottom/left)** - adds space between the element and surrounding elements
- **float** – forces elements to a side, the center, or the top

## Background:

- **background-color -** sets the background color
- **background-image -** sets a background image

## More at:

- https://www.w3schools.com/tags/

# Problem

- Make a page with the following html

```
<!DOCTYPE html>
<html lang="en">
        <head>
            <meta charset="UTF-8" />
            <meta name="viewport" content="width=device-width, initial-scale=1.0" />
            <title>Awesome Page</title>
        </head>
        <body>
            <h1>Awesome header</h1>
            <div>
                <h2>Super Subsection!</h2>
                <p>
                    Lorem ipsum dolor sit amet consectetur adipisicing elit. Iusto
                    error est numquam debitis alias rem? Suscipit excepturi ea,
                    similique sit hic aliquid atque sapiente dicta molestiae!
                    Delectus a eveniet dicta.
                </p>
            </div>
            <img src="https://via.placeholder.com/150" />
        </body>
</html>
```

With this code your job is to add styling to the elements without adding properties to the html

**The styling should be as follows:**

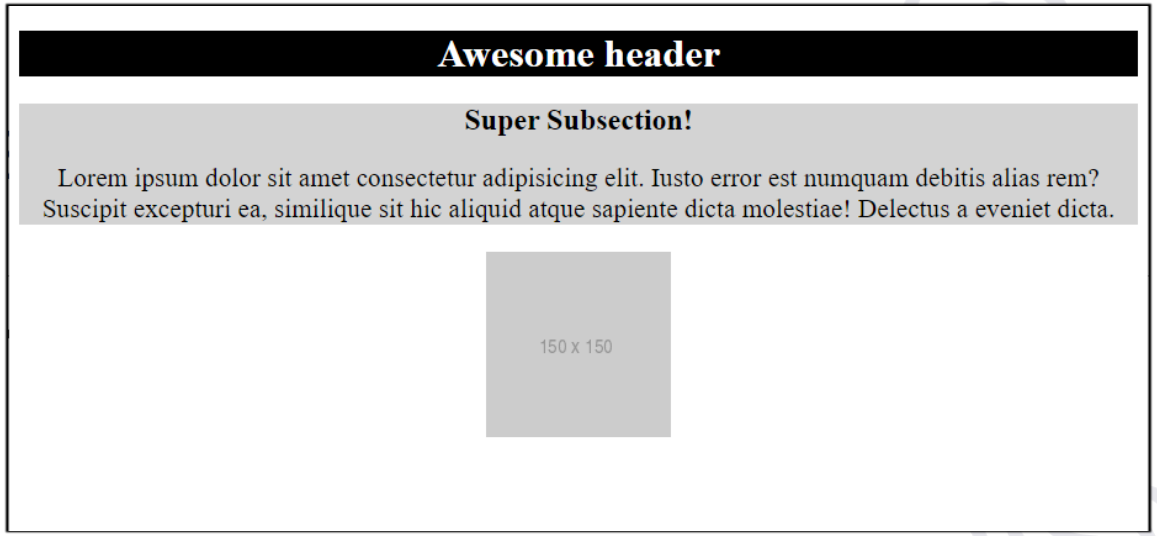
- h1: centered, white text, black background
- div: light grey background
- h2: centered text
- p: centered text, 16pt font size
- img: centered

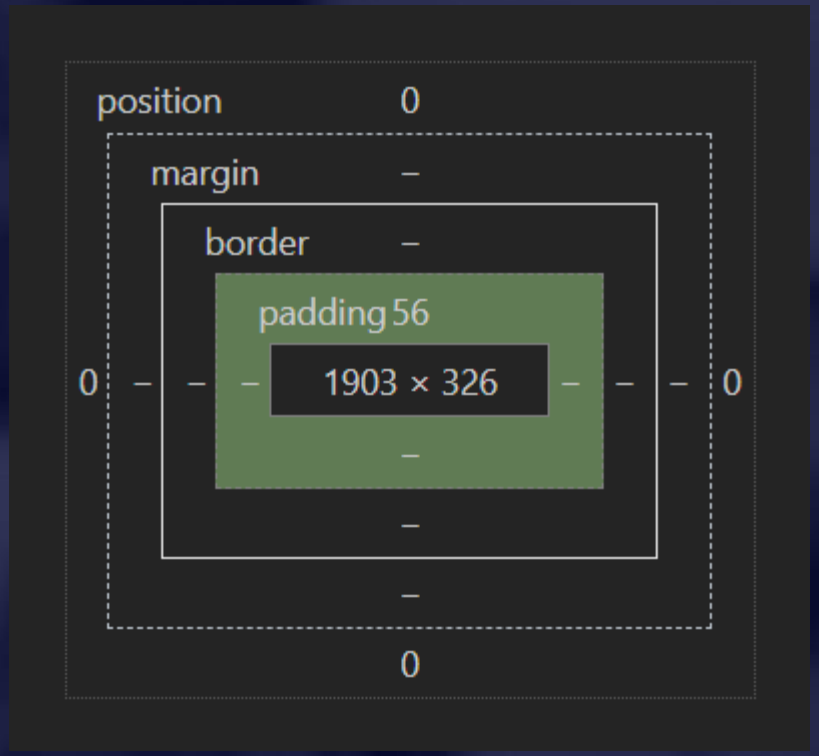
# Solution

## CSS should look like

```
<style>
    h1 {
        text-align: center;
        color: white;
        background-color: black;
    }
    div {
        background-color: lightgray;
        text-align: center;
    }
    p {
        font-size: 12pt;
    }
    img {
        display: block;
        margin-right: auto;
        margin-left: auto;
    }
</style>
```

## Your page should look like





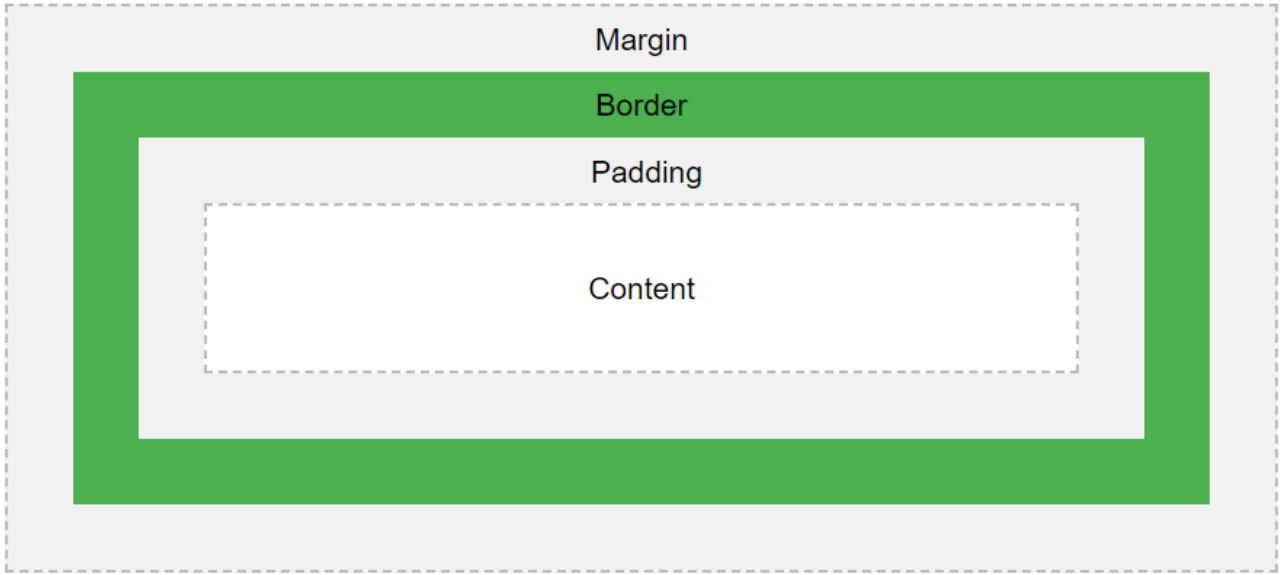

# Box Model

# Boxes, Lots of Boxes

In CSS, every element rests within a series of boxes

Each box has a place it resides within the element:

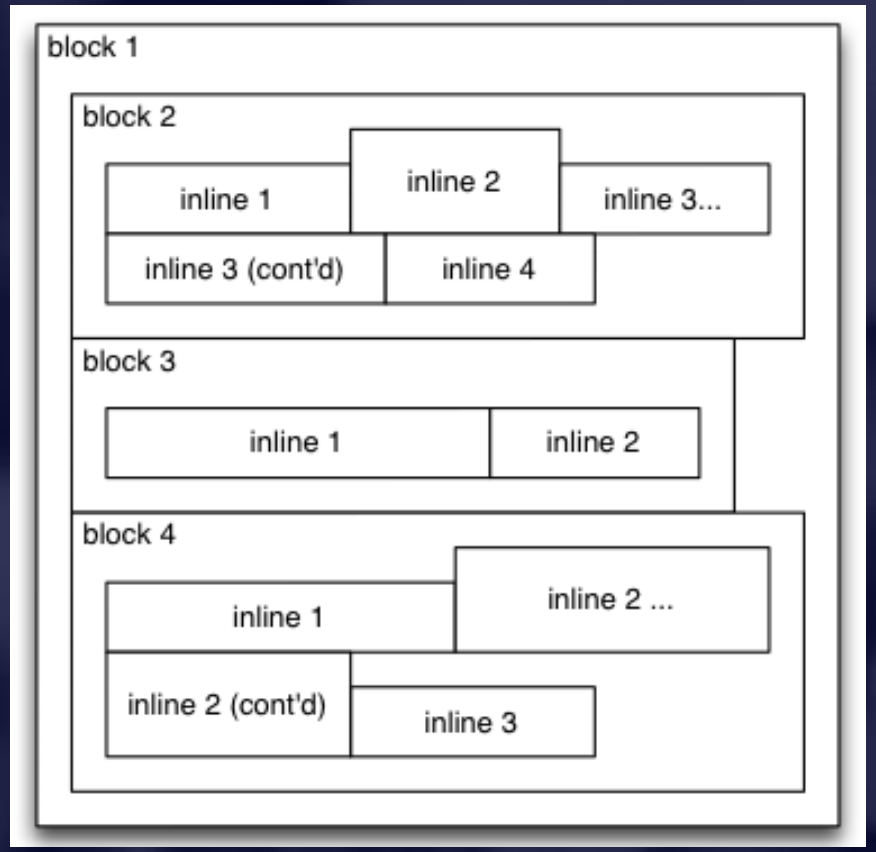
- Content - The content of the box, where text and images appear
- Padding - Clears an area around the content. The padding is transparent
- Border - A border that goes around the padding and content
- Margin - Clears an area outside the border. The margin is transparent



# Properly setting width and height

When setting the width and height properties it is important to remember these few things

- The width and height are set for the content
- To calculate the full size of an element you need to account for the:
  - Element
  - Padding
  - Borders
  - Margins
- The element's width is calculated as follows
  - width + left padding + right padding + left border + right border + left margin + right margin
- The element’s height is calculated as follows
  - height + top padding + bottom padding + top border + bottom border + top margin + bottom margin



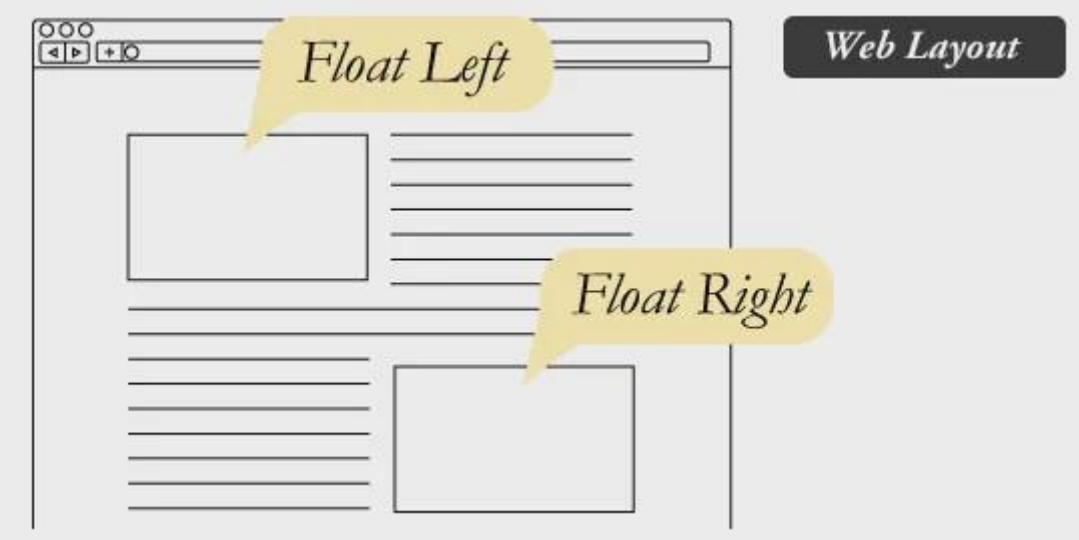

# Flow and Layout Design

# Flow

- Properly applying flow is one of the most difficult parts of CSS
- By default, the flow of HTML is rigid, line by line, in the order of elements
- HTML Elements force this adjacent elements to flow around them

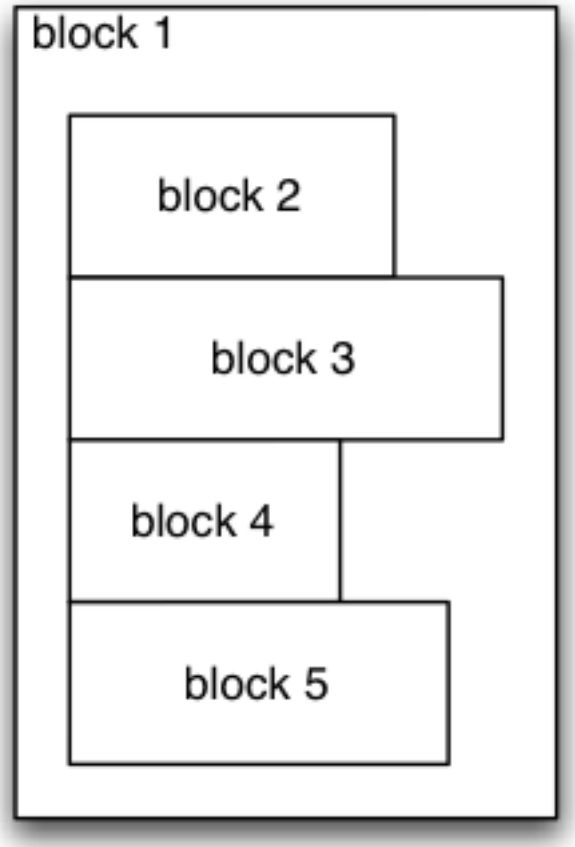
# Flow

- This concept of flow is similar to the wrap-text options in word
- In MS word images can be in line with text and have the text “flow” around the images

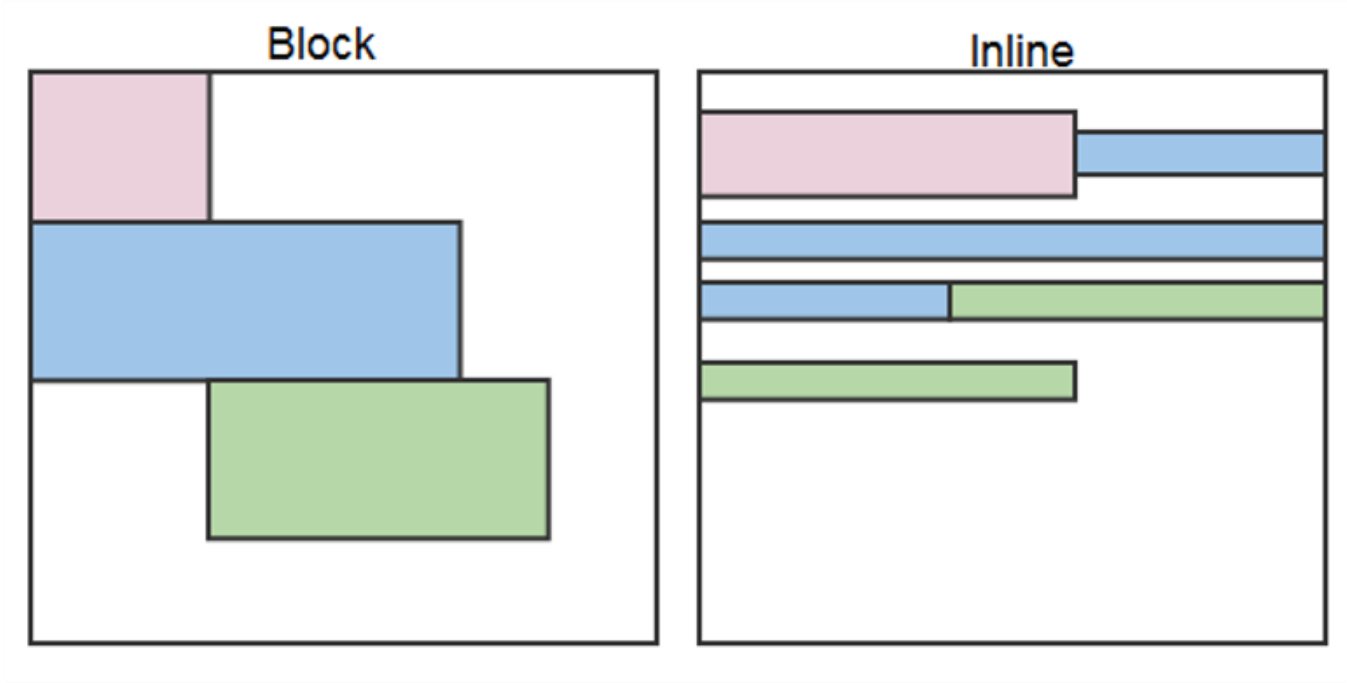


# Block Elements

- By defaut many HTML element are rendered as block elements.
  - paragraphs
  - headers
  - divs
  - etc.
- A block element will take up an entire line of space
  - Unless CSS is used to change the elements properties

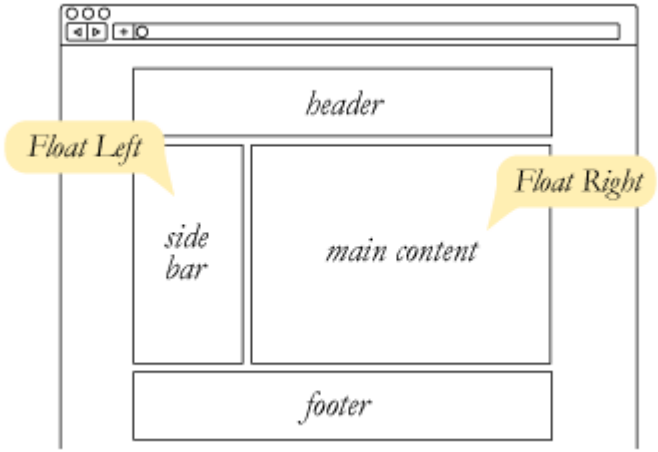


# Block vs. Inline




- Now if we change the properties from block to inline they will compact
- By using float CSS properties we can command many HTML elements to display adjacently  and even right justify elements

# Floating




**CSS:**

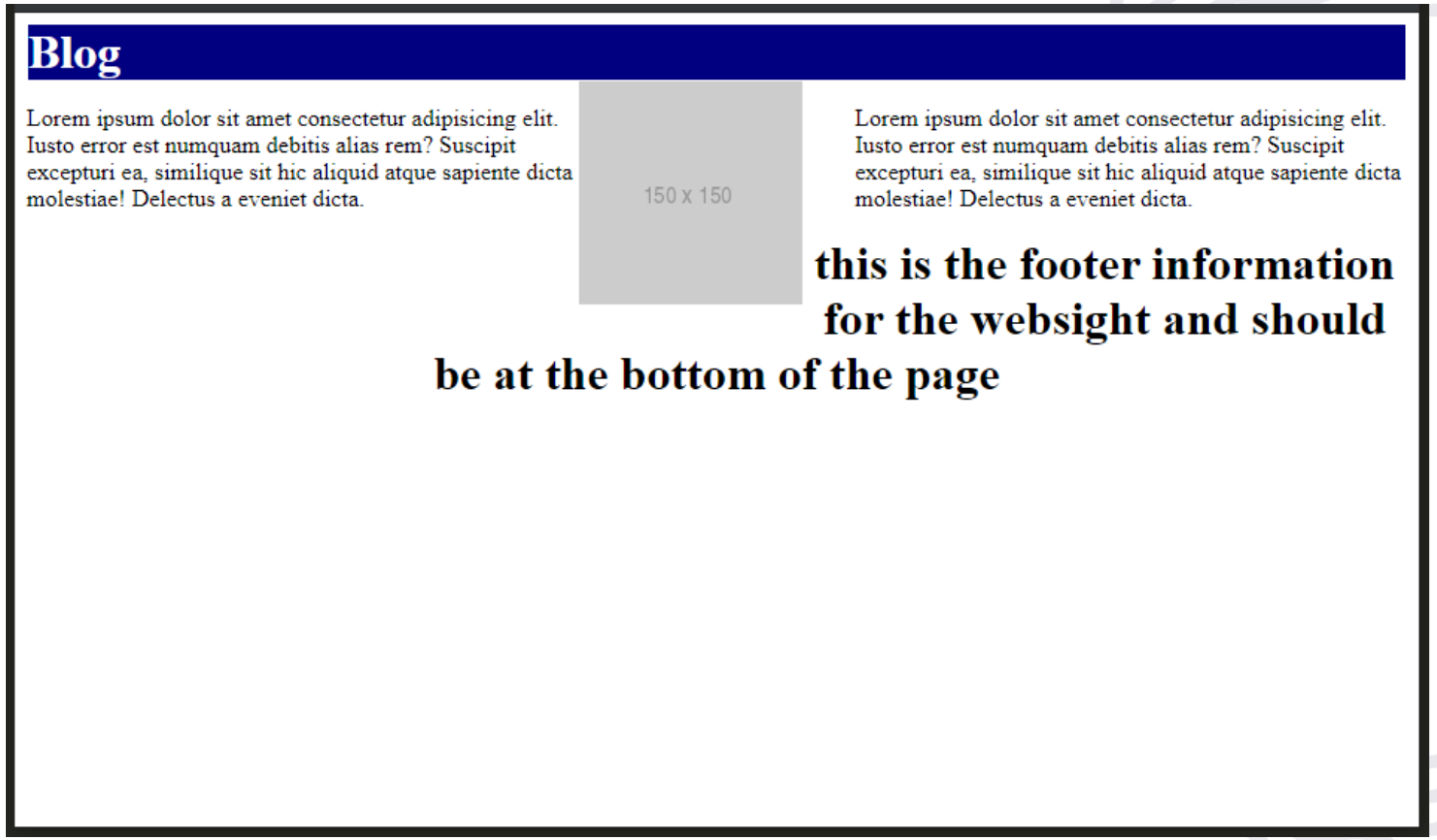
```
<style>
    #sidebar {
        float: left;
    }

    #main-content {
        float: right;
    }
</style>
```

- To effectively transform block elements to inline elements we can use float
- Floats are **required** for  building good website layout

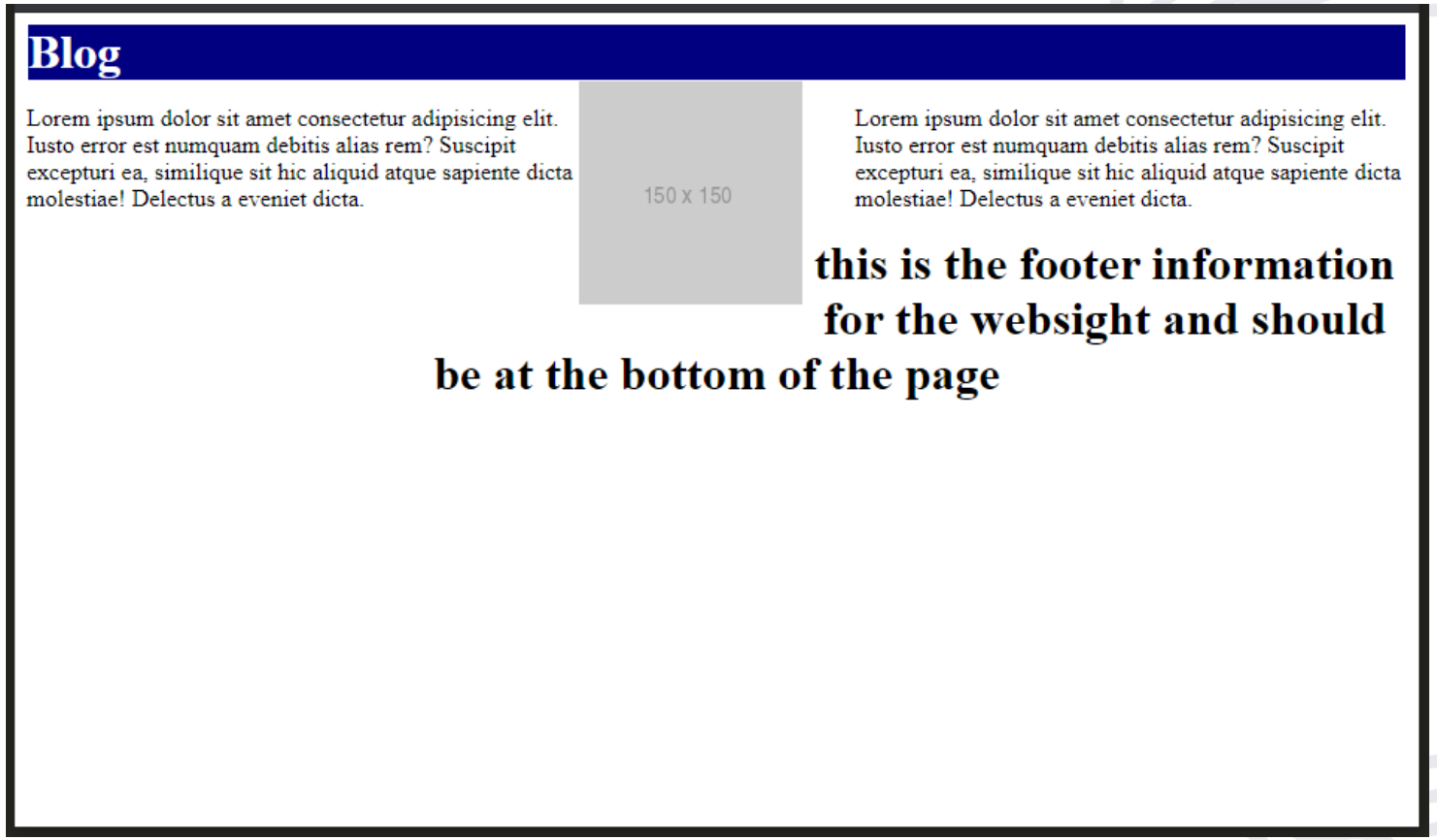
# Too much float

- Floats can get in the way of a websites structure
- We sometimes don’t want an element inline



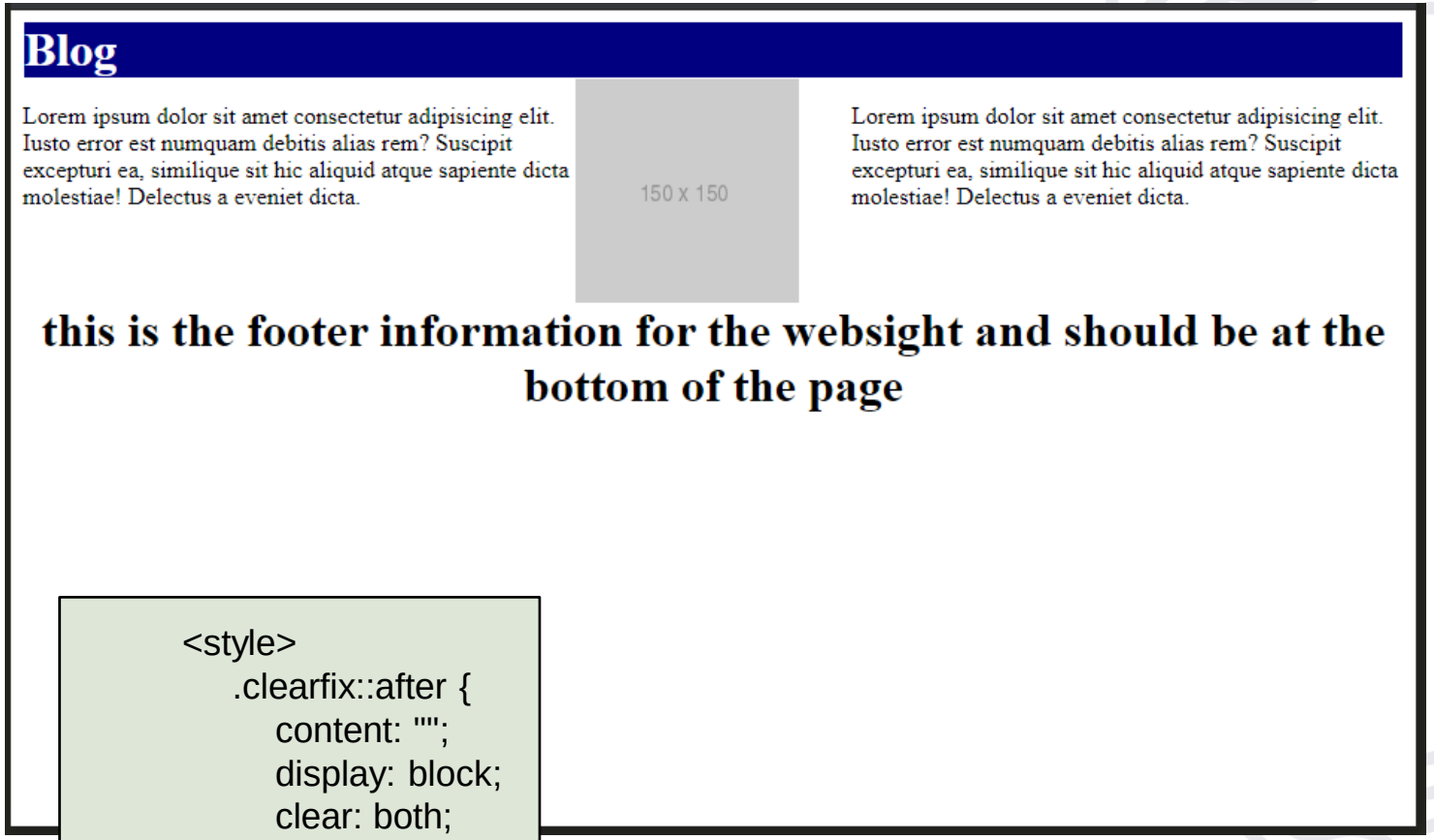
# Clearfix hotfix

- When the elements don't line up we use a fix that has been coined clearfix.
- It makes sure that the content properly displays and it gives floats "structure"
  - so we can more effectively organize a page



# Clearfix hotfix

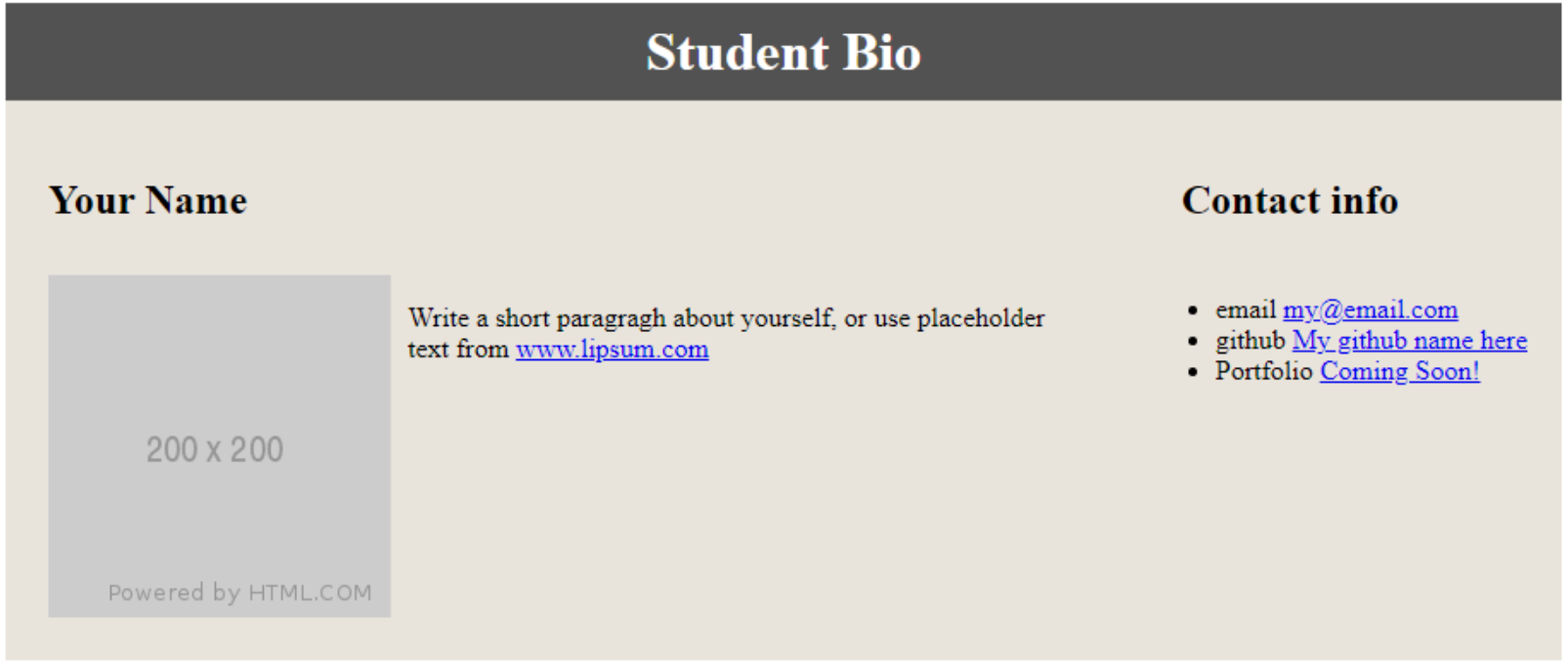
- clearfix eliminates the wrapping of the text
- **::after** : is what a pseudo-element.
  - Its used to style specific parts of an element
- The **clearfix** adds an HTML element, that's hidden from view, after the content of the element with the class "clearfix"
- This is what clears the float




```
<style>
    .clearfix::after {
        content: "";
        display: block;
        clear: both;
    }
</style>
```

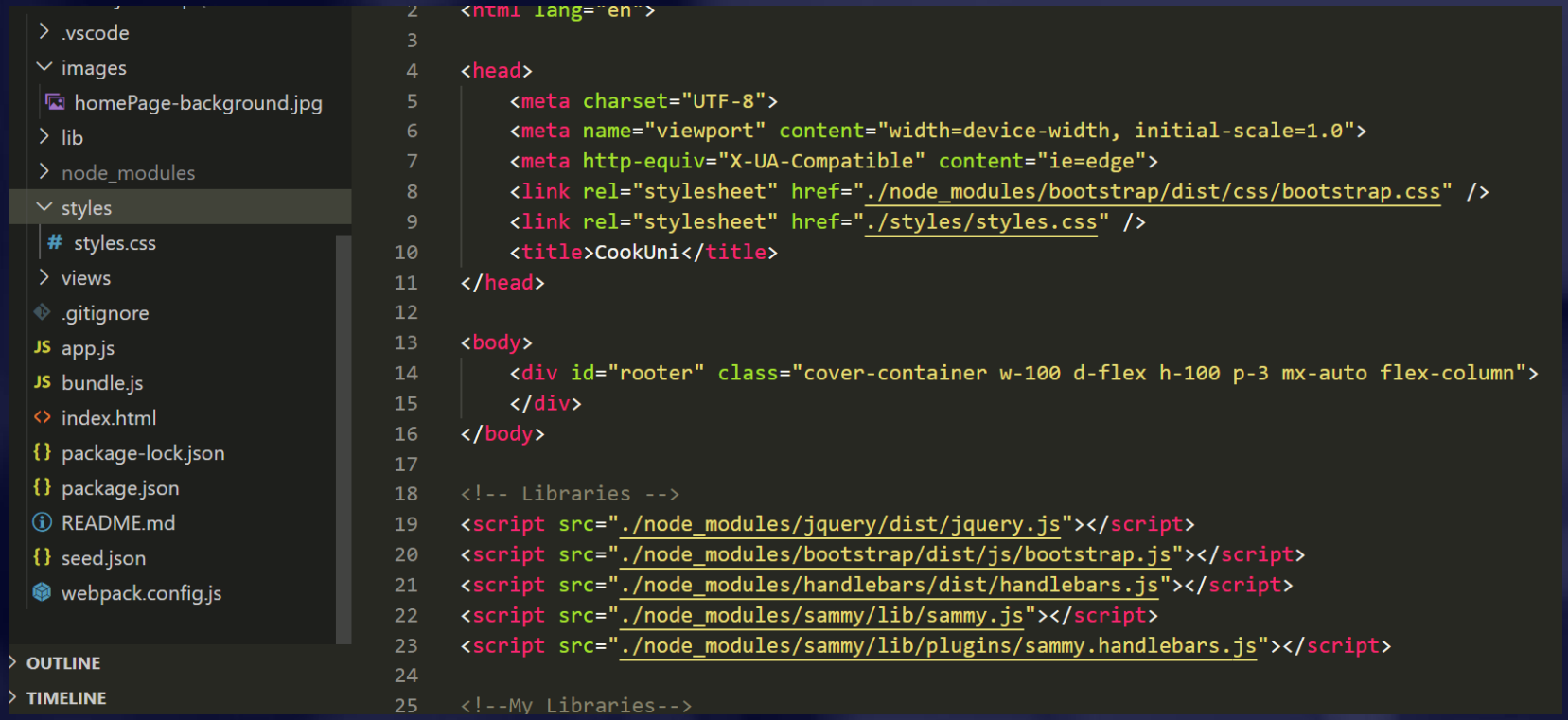
# Problem: Student Bio

- Your job is to create a page that looks like the one below
- Picture can be added with https://via.placeholder.com/200



# Solution

Code together




# Relative file paths

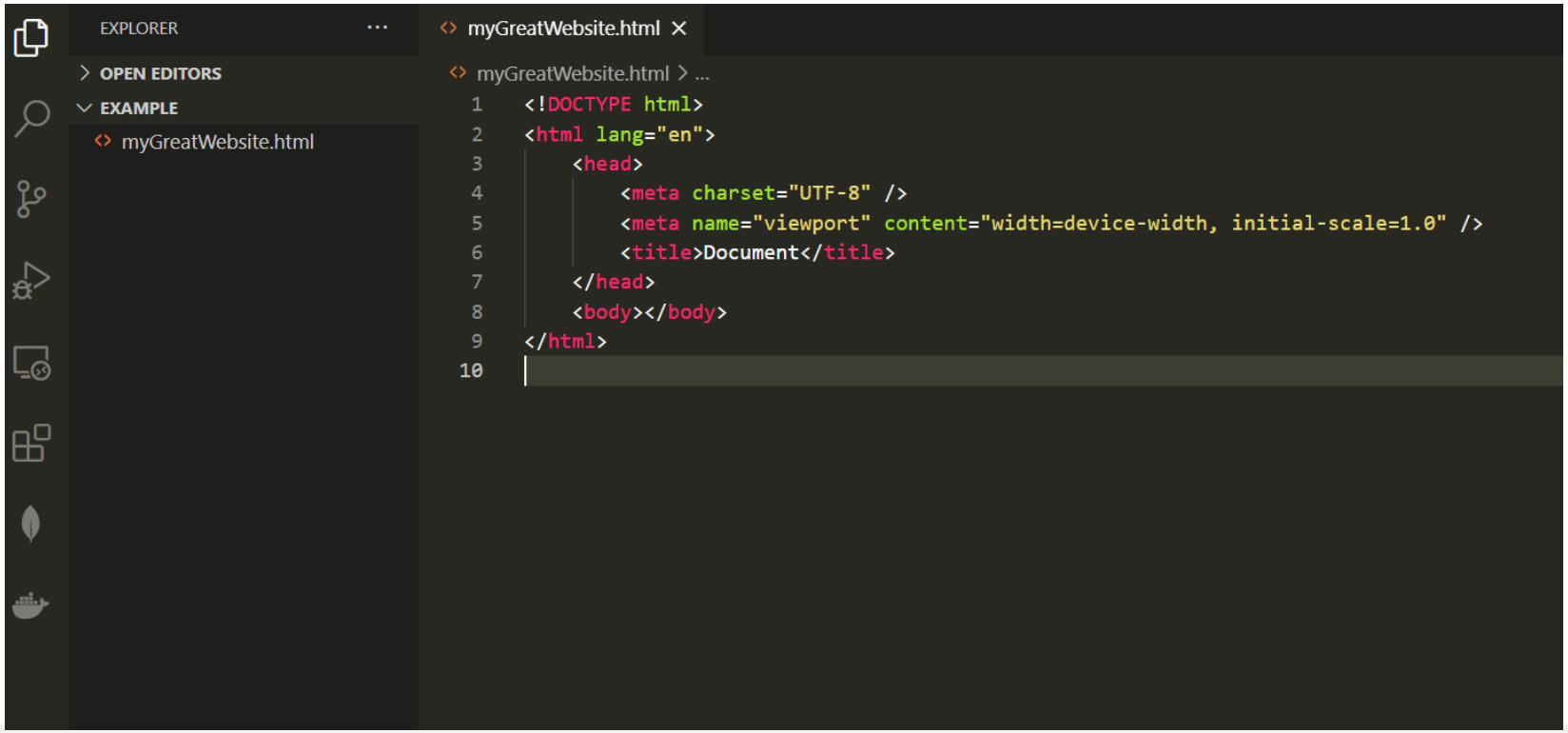
# CSS Files

- At some point the CSS in your html page will get large
- Or there will be a library that you'd like to use
- So here are some tips on adding in files to your HTML page to keep things clean

- As we go through the steps periodically open the html in the browser to see how things change/ if its working

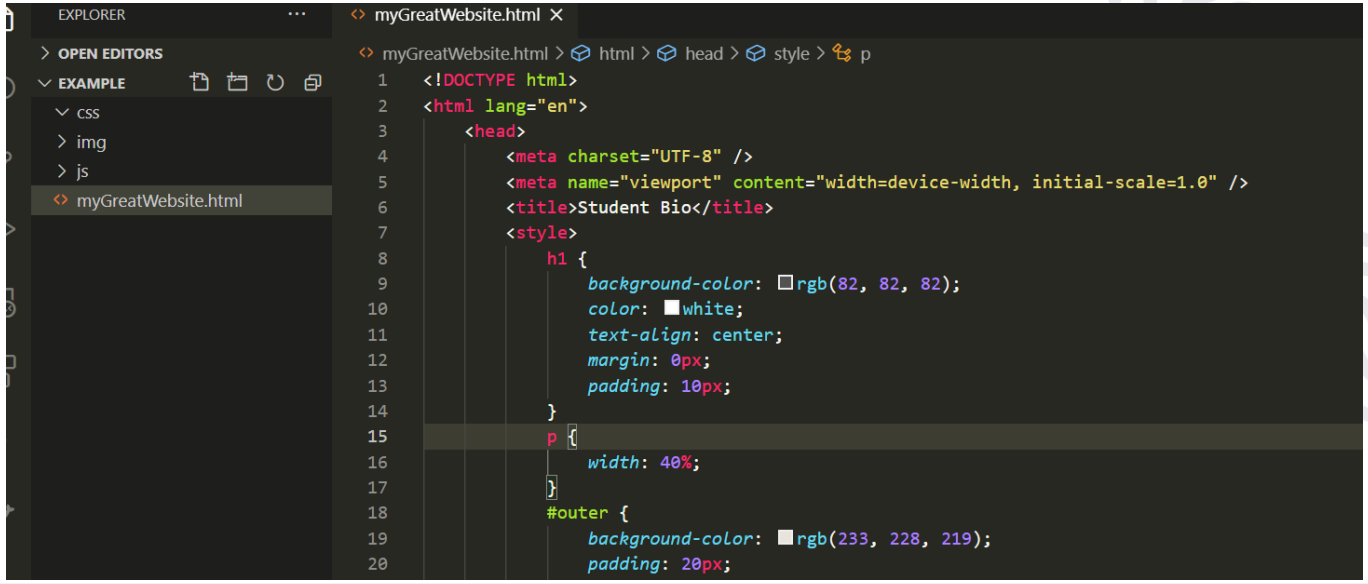
# Steps - 1

- First Make a new folder
- Then open it in your editor
- From here add your html page
- If you've gotten his far everything should look like below



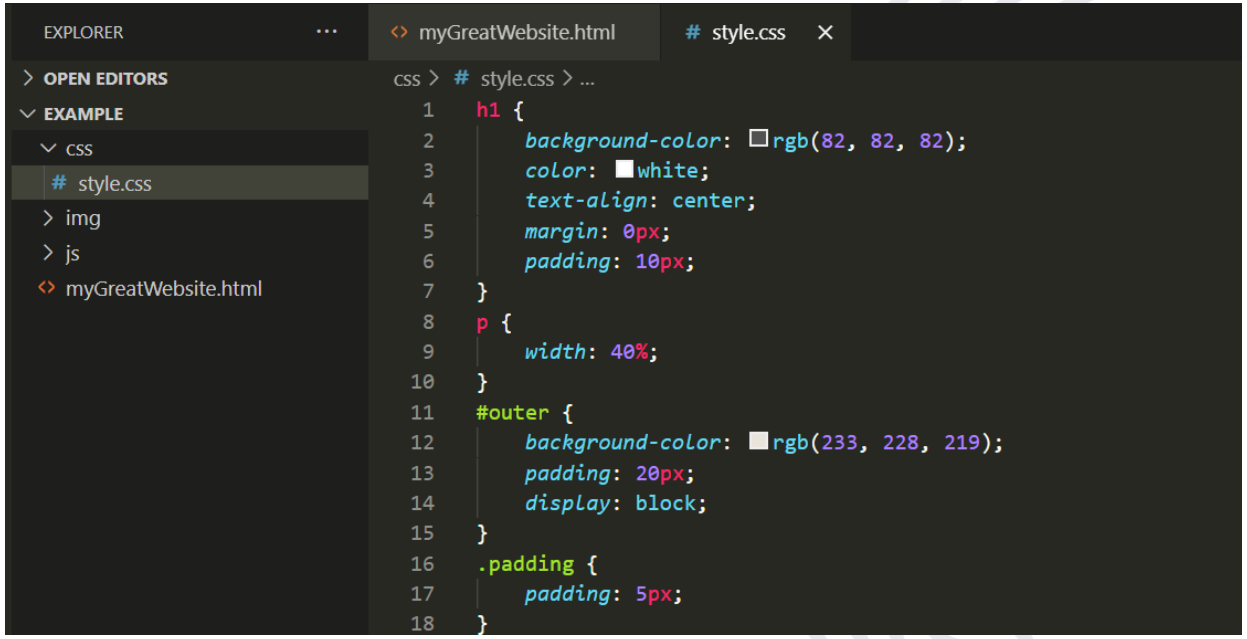
# Steps - 2

- From here we want to make a couple of folders for things we may add to our site later
  - Standard files are:
    - css
    - js (JavaScript)
    - img (images)
- You can also copy in your code form an earlier problem to flesh out the html file
- If you've gotten his far everything should look like below



# Steps - 3

- From here we will create a file in our css folder
  - Call it something like syles.css
  - Naming conventions differ from place to place
    - Best practice is to name it something clear and descriptive as to what the file is doing
  - Make sure the file ends in .css or it is not a css file
- Then copy the css form the HTML we copied before and move it into the css file
  - Remove the style tag as well
- If you've gotten his far everything should look like the right



# Steps - 3

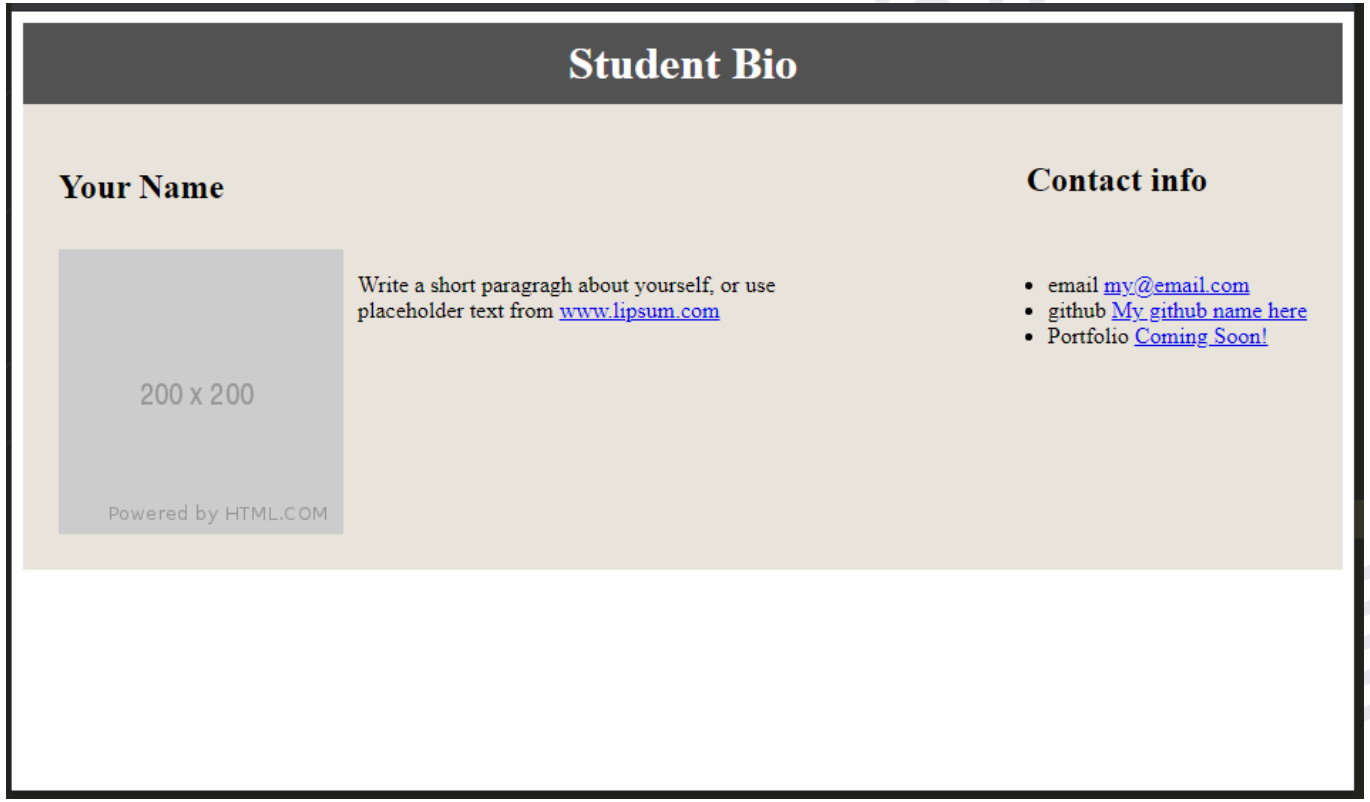
- From here we will link our css to our html file
- First re-open the html file and go to where the style tag was, add a link tag
- Next add the following into the link tag

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

- If you've gotten his far everything should look like below

# Results

- If everything is done correctly it should look the same as before
- But now your css is separated from the html
  - This makes managing websites much easier



# Summary

- The structure of CSS elements
- The different CSS attributes
- The Box model
- Floating elements
- Clearfix
- File structure
- Relative pathing

# Questions?

# License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © Kingsland University – https://kingslanduniversity.com


KINGSLAND
UNIVERSITY
THANK YOU