HTML & CSS

What are the Front End Languages?

- **HyperText Markup Language**
- Contains the content and structure of a web page
- Uses tags to create elements on the page

- **Cascading Style Sheets**
- Contains the style and design of a web page
- Uses properties to change the look of elements on the page

- **JavaScript (ECMAScript)**
- Contains the logic and functionality of a web page
- Uses variables and functions to interact with elements on the page

Coding Languages for Web Development

In a web application, the back-end code *generates* the front-end code as the user interacts with the site.

For a simple website, we can write the front-end code directly ourselves.

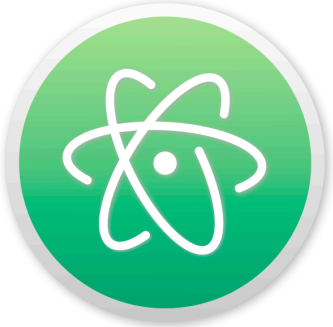
Beginning to Write Code

The first thing we will need is a code editor.

Visual Studio Code

Brackets

Notepad++



Atom

A code editor will allow us to write HTML, CSS and JS.

HTML

- HTML describes the **structure** of a web page.
- It does this using **elements**, and **tags**.

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour=“red” align=“left”>This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

- This tells the browser that the document (file) type is HTML

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour="red" align="left">This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

This is an **element** - a paragraph.

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour="red" align="left">This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

This is another **element** - a H1 heading.

It’s composed of start and end **tags**, **attributes**, and the **content** inside the tags.

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour=“red” align=“left”>This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

Start and end **tags**

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour="red" align="left">This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

attributes

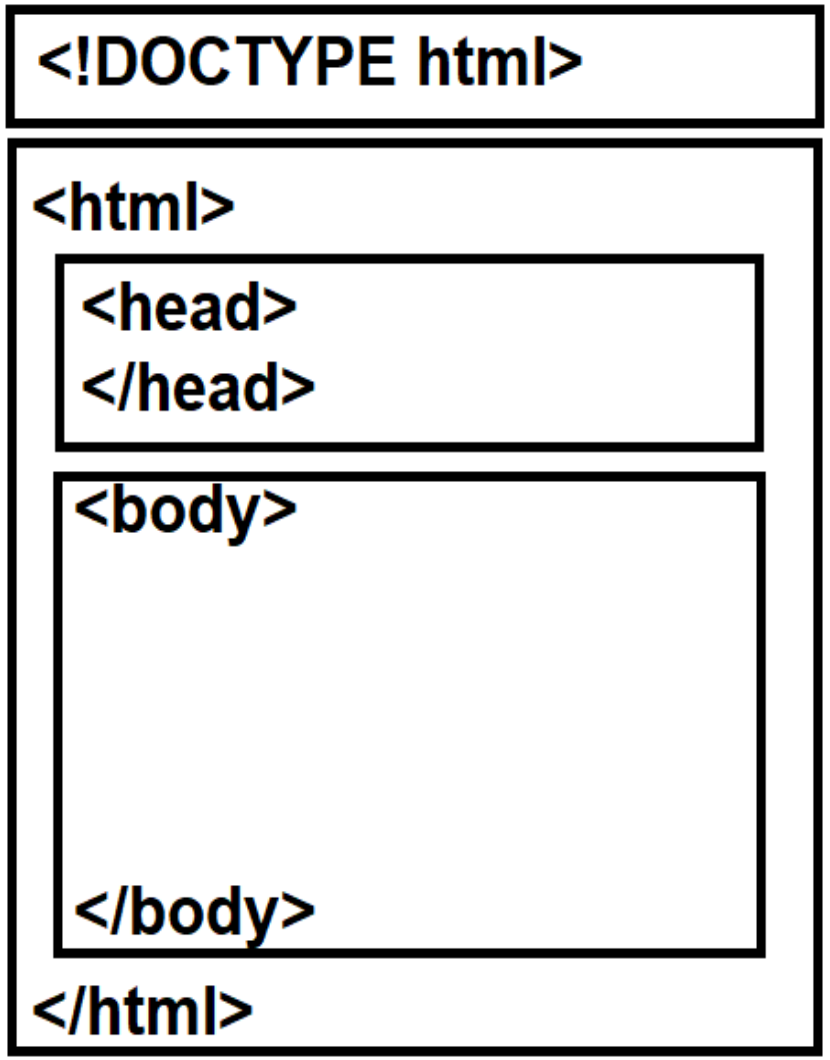
```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour="red" align="left">This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML

content

```
<!DOCTYPE html>
<html>
 <head>
  <title>My First HTML</title>
 </head>
 <body>
  <h1 colour="red" align="left">This is the first heading on the page.</h1>
  <p>This is a paragraph.</p>
 </body>
</html>
```

HTML 5 Document Structure



- !Doctype
  - Defines the DTD (Document Type Declaration). Informs the browser which version of HTML to use.
- HTML
  - Overall container for everything HTML.
- Head
  - Contains “hidden” information about the page: metadata, styles, linked style and script files, etc.
- Body
  - Container for everything you wish to display on the webpage. This can include text, images, embedded videos, etc.

HTML

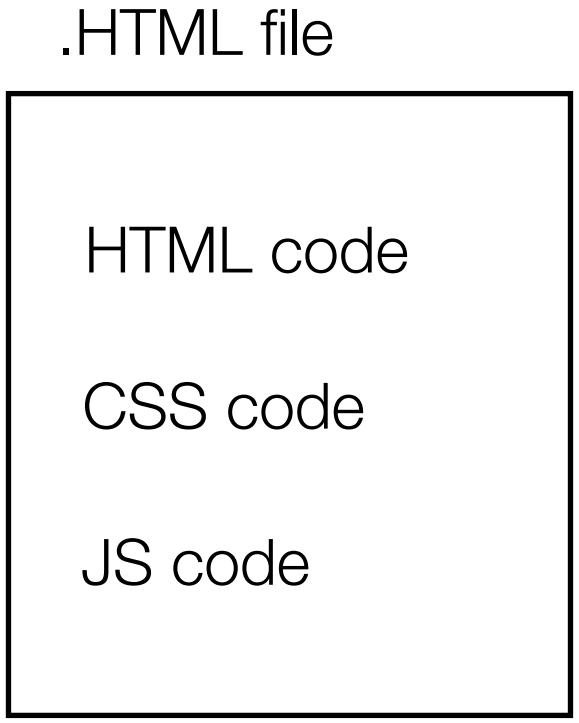
Let’s put the code in a file, open the file in a browser, and see what happens.

```
<!DOCTYPE html>
<html>
  <head>
    <title>My First HTML</title>
  </head>
  <body>
    <h1 colour=“red” align=“left”>This is the first heading on the page.</h1>
    <p>This is a paragraph.</p>
  </body>
</html>
```

HTML 5 Document Structure

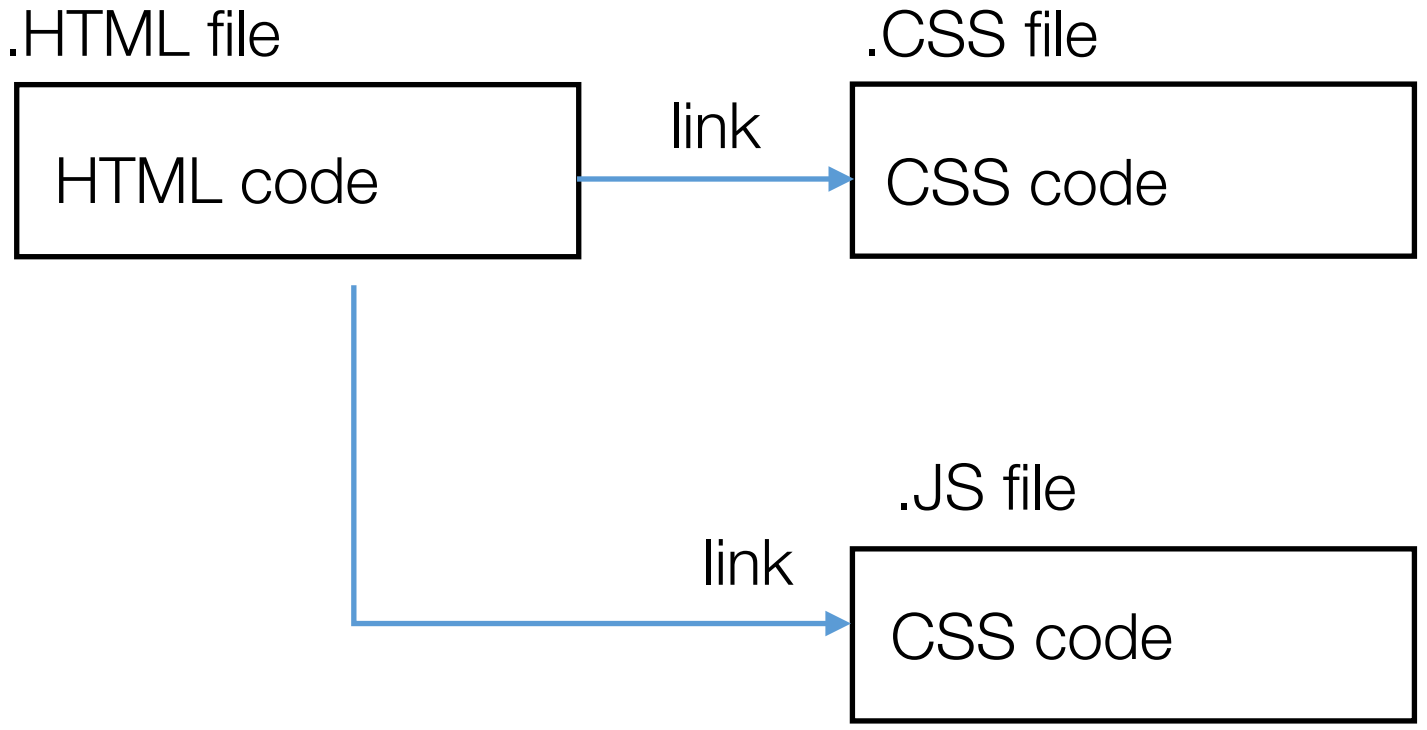
Can be set up like this

Embedded/inline



or this

External



CSS

- CSS describes the appearance of each element in the HTML.
- It does this using **selectors**, **properties**, and **values**.

Inline CSS

```
<p style="color:green;
font-style:italic;">
This is a paragraph.</p>
```

Embedded/External CSS

```
p {
    color: green;
    font-style: italic;
}
```

HTML

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

CSS Selectors

Selectors allow us to change the style of specific elements.

HTML

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

<p class="daveBold">This paragraph will be bold.</p>

<p id="daveRight">This paragraph will be right-aligned</p>
```

Embedded/External CSS

```
p {
  color: green;
}

.daveBold {
  font-weight: bold;
}

#daveRight {
  text-align: right;
}
```

CSS Selectors

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

This selects **all paragraph elements on the page** and colours their text green.

CSS Selectors

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

This selects **all paragraph elements on the page** and colours their text green.

```
.daveBold {
  font-weight: bold;
}
```

This selects **any elements with the class ‘daveBold’** and makes the text bold.

CSS Selectors

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

This selects **all paragraph elements on the page** and colours their text green.

```
.daveBold {
  font-weight: bold;
}
```

This selects **any elements with the class ‘daveBold’** and makes the text bold.

```
#daveRight {
  text-align: right;
}
```

This selects **only the element with the id ‘daveRight’** and right-aligns the text.

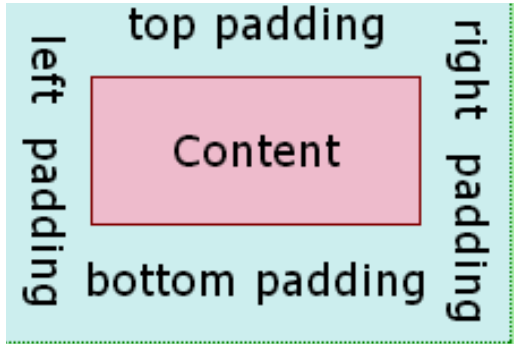
CSS Box Model

For every HTML element we can imagine it has a series of boxes wrapped around it.

Content: Whatever the element in the HTML is.

CSS Box Model

For every HTML element we can imagine it has a series of boxes wrapped around it.

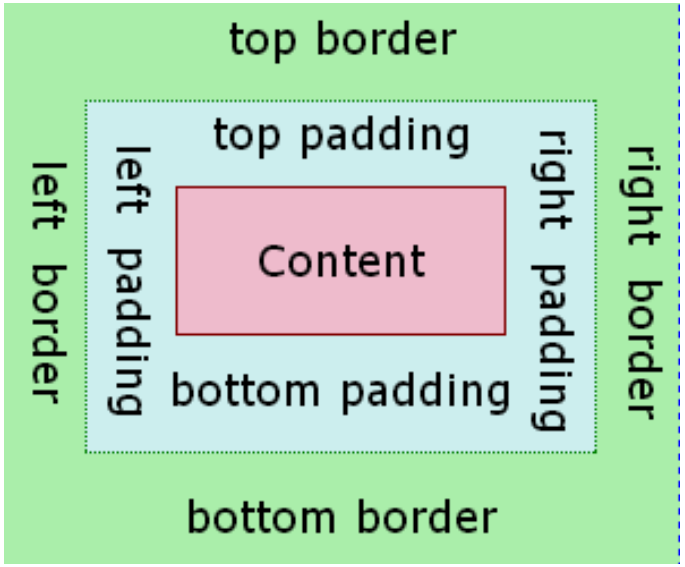


Content: Whatever the element in the HTML is.

Padding: A transparent area around the content.

CSS Box Model

For every HTML element we can imagine it has a series of boxes wrapped around it.



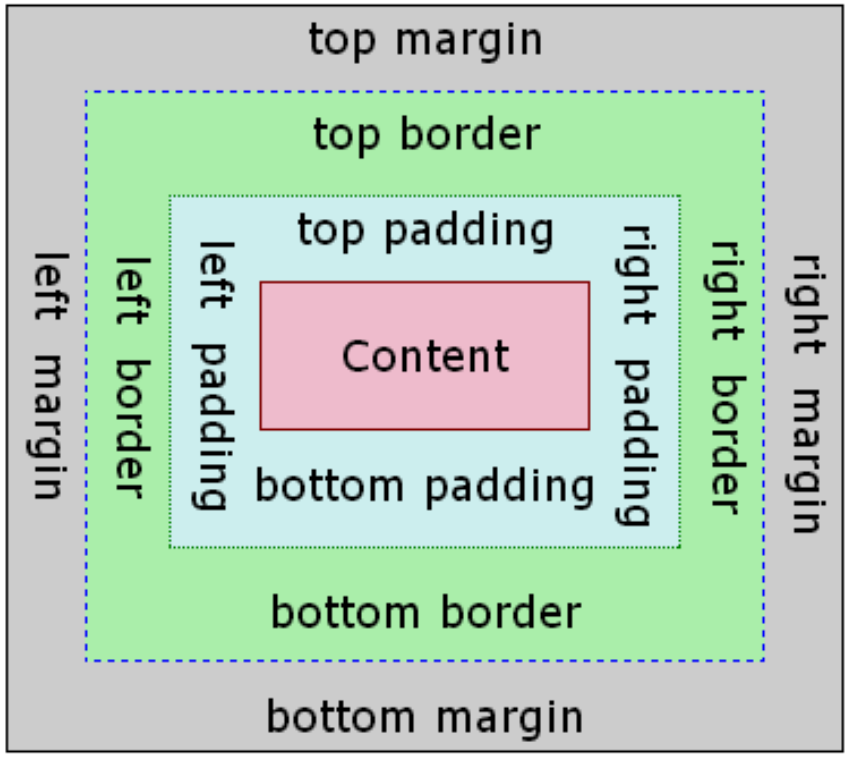
Content: Whatever the element in the HTML is.

Padding: A transparent area around the content.

Border: A styled border that goes around the padding.

CSS Box Model

For every HTML element we can imagine it has a series of boxes wrapped around it.



Content: Whatever the element in the HTML is.

Padding: A transparent area around the content.

Border: A styled border that goes around the padding.

Margin: A transparent area around the border.

Separation of Content and Presentation

- The general idea is that the “presentation and style” (CSS) should be separated from the “content” of a web page (HTML).

- This principle allows us as designers or coders to have our content and style be interchangeable between different code-bases.

- CSS Zen Garden
- W3Schools Intro to CSS
- Hacker News - CSSZG Thread

Semantic Tagging

- CSS is there to make HTML look good to people.
- There is a way to make HTML look good to machines as well.
- *Semantic* tagging allows us to tell a computer what purpose our elements in the HTML have.
- This is useful for SEO, screen reading tools and other accessibility features.

Semantic Tagging

Without

```
<div class="header">
  <img src="logo.png" alt="Company Logo">
  <a href="homepage.html">Home</a>
  <a href="services.html">Services</a>
  <a href="contact.html">Contact</a>
</div>
```

With

```
<header>
  <img src="logo.png" alt="Company Logo">
  <nav>
    <a src="homepage.html">Home</a>
    <a src="services.html">Services</a>
    <a src="contact.html">Contact</a>
  </nav>
</header>
```

Semantic Tagging

- HTML.com - Semantic Markup (Cheat Sheet)
- Semantic HTML for Meaningful Webpages
- https://developer.mozilla.org/en-US/docs/Learn/Accessibility/HTML

Additional Resources

- w3Schools
- Mozilla Developer Network
- Codepen
- CSSDeck
- Stack Overflow

Challenges

1 - w3Schools

Set up an account, and work through the HTML & CSS tutorials, Exercises and Quizzes

2 - Figma to HTML/CSS

Make a (very) simple design in Figma and try to recreate it in HTML & CSS