HyperionDev

Full-Stack Web Development

HyperionDev

Basics of HTML and CSS

Full Stack Web Development Session Housekeeping

- The use of disrespectful language is prohibited in the questions, this is a supportive, learning environment for all - please engage accordingly. **(Fundamental British Values: Mutual Respect and Tolerance)**
- No question is daft or silly - **ask them!**
- There are **Q&A sessions** midway and at the end of the session, should you wish to ask any follow-up questions. Moderators are going to be answering questions as the session progresses as well.
- If you have any questions outside of this lecture, or that are not answered during this lecture, please do submit these for upcoming Academic Sessions. You can submit these questions here: **Questions**

Full Stack Web Development Session Housekeeping cont.

- For all **non-academic questions**, please submit a query: **www.hyperiondev.com/support**
- Report a **safeguarding** incident: **www.hyperiondev.com/safeguardreporting**
- We would love your **feedback** on lectures: **Feedback on Lectures**

Objectives

- Understand the basic structure of a HTML document and its key elements
- Apply fundamental CSS styles to HTML elements
- Set up a development environment for HTML and CSS
- Create a simple web page using HTML and CSS

The Terminal

- A **terminal**, also known as a **command-line interface** or **shell**, is a text-based interface used to communicate with the operating system.
- It allows users to execute commands by typing them in, instead of relying solely on a graphical user interface (GUI).
- This is particularly powerful for automation, scripting, and system administration tasks.

Terminal Commands: PWD

- Stands for "print working directory."
- It shows you the current directory you're in.

```
PS E:\practical> pwd

Path
----
E:\practical
```

Terminal Commands: LS

- Lists the files and directories in the current directory.

```
PS E:\practical> ls


    Directory: E:\practical


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         2/22/2024   5:24 PM                Tutorial session
-a----         2/20/2024   1:32 AM             16 example.txt
-a----         2/20/2024   1:13 AM             68 example1.txt
-a----         2/20/2024   1:14 AM             68 example2.txt
-a----         2/19/2024  11:13 PM            683 fileIO.py
-a----         2/22/2024   5:46 PM           3876 oop.py
-a----          3/1/2024   1:45 AM           1971 sandbox.py
```

Terminal Commands: CD

- Stands for "change directory."
- It allows you to navigate between directories.

```
PS E:\practical> cd tutorial_session
PS E:\practical\tutorial_session>
```

Terminal Commands: CD

- **VS Code**: https://code.visualstudio.com/
- **Browsers**:
  - Firefox
  - Chrome
  - Edge
  - Safari

HTML

- **HTML (HyperText Markup Language)** is used to structure and format the content of websites on the World Wide Web.
- In simple words, HTML is the primary building block to create and structure website content.
- Web Developers use it to create a skeleton of modern websites and web apps.

HTML

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

HTML

- **<html>** is the root element of the DOM, and it contains all of the other elements in the code.
- The **<head>** tag contains metadata about the web page, such as the title and any linked CSS or JavaScript files.
- The **<body>** tag contains the main content of the web page, which will be displayed in the web browser's window.

HTML:DOM

- The **Document Object Model (DOM)** is a programming interface for web documents. It represents the page so programs can change the document structure, style, and content dynamically.
- The DOM is a fundamental concept in web development, enabling interaction with web pages.
- When a web page is loaded, the browser creates a Document Object Model of the page.
- The DOM tree represents the structure of an HTML document. Each element in the document is a node in the tree, forming a parent-child relationship.

HTML: Elements

- HTML elements consist of several parts, including the **opening and closing tags**, the **content**, and the **attributes**.
- The **opening tag** consists of the element name, wrapped in angle brackets (< or />). It indicates the start of the element and the point at which the element's effect begins.

HTML: Elements

- The **closing tag** is the same as the opening tag, but with a forward slash (**/**) before the element name. It indicates the end of the element and the point at which the element's effect stops.
- The **content** of the element can be text, other elements, or a combination of both.
- The opening tag, the closing tag, and the content together make up the element.

HTML: Attributes

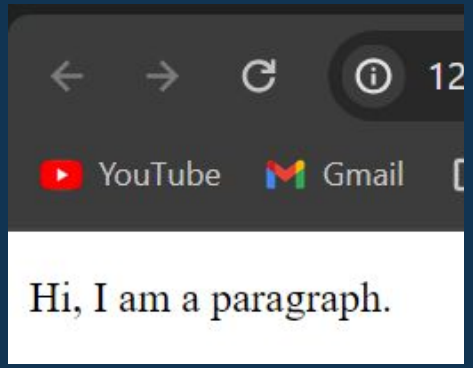
- HTML elements can have **attributes**, which provide additional information about the element.

```
    <p name="new_paragraph">Hi, I am a paragraph.</p>
  </body>
</html>
```

HTML: Paragraphs

- The HTML <p> tag is used to create paragraphs.

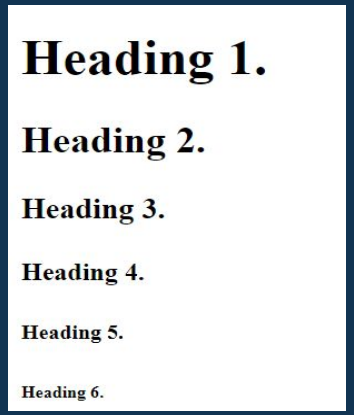
```
  <body>
    <p>Hi, I am a paragraph.</p>
  </body>
</html>
```



HTML: Headings

- The HTML heading tags (<h1> to <h6>) are used to add headings to a webpage.

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



HTML: Comments

- HTML comments are used to insert notes to a web page.

```
<!-- You can't see me -->
<p>
  You cannot see the comment above because it's not supposed to be visible
</p>
</body>
```

Hi, I am a paragraph.

You cannot see the comment above because it's not supposed to be visible

CSS

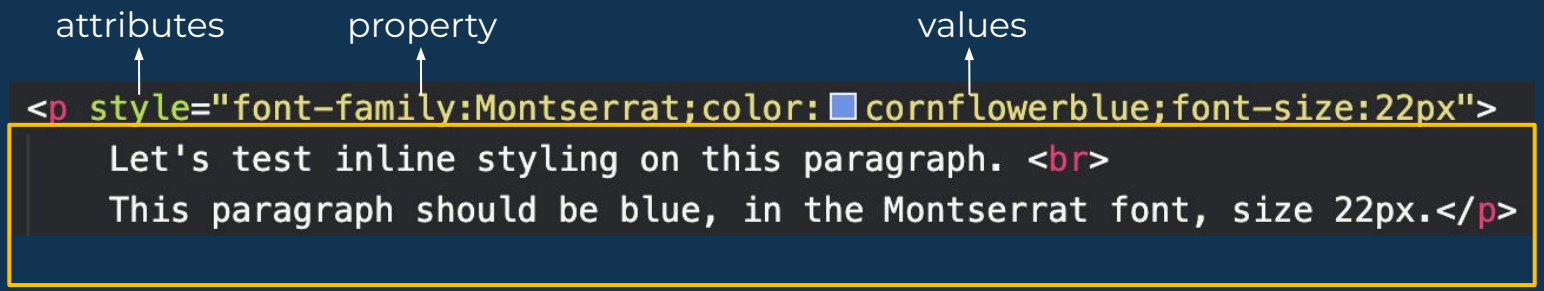
Cascading Style Sheets (CSS) is a language used to change the presentation and styling of a document written in a markup language e.g. HTML

- Helps us create **visually appealing** and **user-friendly** websites.
- HTML structures the content, CSS controls how the content looks.
- CSS uses a **set of rules** written in a **certain syntax** to style HTML.
- We use CSS to create **style sheets**, which define the appearance and layouts of the elements on a webpage.
- The various properties which we can control with CSS can be found here.

Styles: Inline Style

- HTML elements are described using **attributes** and **properties**.
- One of the attributes of an element is **style**, which we can change by **adjusting its properties** using CSS rules.
- Attributes are adjusted **inside the element's beginning tag**.

For example: Text Elements:



Styles: Internal Style

- CSS rules can be defined in the **head** part of the HTML template, inside the **style element**. This is known as **internal CSS**.
- Rules can be defined for every type of element in the HTML document.

```
<head>
    <style>
        p {
            font-style: italic;
            color: chartreuse;
        }
    </style>
</head>
<body>
    <p style="font-family:Montserrat;
    color:cornflowerblue;
    font-size:22px;">
        Let's test inline styling on this paragraph.
        <br>This paragraph should be blue,
        in the Arial font, size 22px.</p>
</body>
```

➔ The style sheet consists of **selectors** and **declarations**

- **Selectors:** indicates which element you want to style
- **Declaration block:** contains one or more declarations, separated by semicolons and enclosed in curly brackets.
- **Declaration:** includes a property and a value separated by a colon

Styling: External CSS

- Another way to define the style for an HTML file is by writing all the style rules in a **separate .css file**. This is called **external CSS**.
- The external file can be **linked** to any HTML file to apply the style rules.
- This method is useful when **applying the same style rule to multiple HTML files**.

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

➔ In the **head** part of the HTML file, in a **link element** define

- **href:** define the name and path of your file (relative to the current working directory)
- **rel:** describes the type of relation the external file is to the HTML (i.e. stylesheet)
- **type:** tells the browser what sort of file it is (only necessary for old browsers)

Best Approached To Styling

- Styling is applied depending on which rules are **closest to the element**.
- Inline styling will be applied to individual elements **overwriting the internal or external CSS** defined for the whole web page.
- Internal styling will overwrite any external styling defined.
- **External CSS** should be chosen over internal CSS where possible
  - **Readability:** separating CSS code and HTML makes code easier to read and follow.
  - **Maintainability:** updating and debugging styling rules is easier since only external CSS files need to change or be replaced.

CSS Selectors

CSS selectors attach to the HTML elements on web pages which allows for customized styling

- There are three common CSS selectors that we will look at:
  - **Element selector**
    - The same style is applied to elements with the same tag.
  - **ID selector**
    - Styles are applied to specific elements using a unique ID.
  - **Class selector**
    - The same style is applied to elements in the same class.

Element Selectors

- The most basic type of CSS selector.
- Style rules are defined for all elements of the same type of tag.
- The selector pinpoints an **element tag** and applies **the same style** to **all elements with that specific tag name**.

For example: Styling the body element

```
body {
    background-color: aliceblue;
    outline-width: 5px;
    outline-color: darkcyan;
    outline-style: groove;
}
```

ID Selectors

- ID selectors apply styles to HTML elements which are identified by its **unique ID name**.
- The ID of an element is an **attribute** defined at the beginning of the HTML tag. The value assigned to this attribute must be **unique**.
- The ID selector is called using a **hash (#)**, followed by the **ID name**.

```
<!-- Here we will be testing ID selectors -->
<h2 id="heading2"> Welcome everyone! </h2>
```

```
#heading2 {
    text-align: center;
    font-family: Montserrat, Helvetica;
    font-size: 26px;
    font-style: italic;
    color: darkgoldenrod;
}
```

Class Selectors

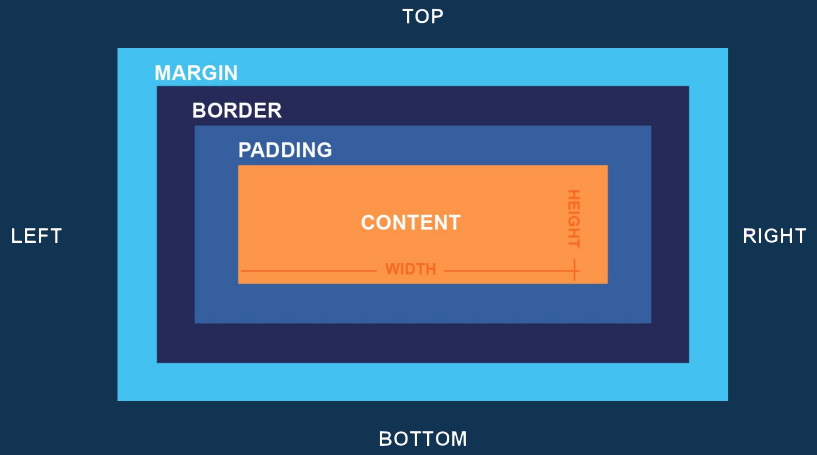
- Class selector aims to change **all HTML elements associated with a specific class**.
- **Class** is also an **attribute**, defined like an ID, but it is **not unique**.
- It is called using a **dot (.) followed by the class name**.
- The **element tag** belonging to that class can be referenced as well.

```css
.endingMessage {
    text-align: center;
    font-family: Bubbly;
    font-size: 20px;
    color: darkslateblue;
    margin-bottom: -18px;
}

p.endingMessage {
    padding-bottom: 20px;
}
```

```html
<h3 class = endingMessage>
    Thank you for joining us :)
</h3>

<p class = endingMessage>
    Please let us know if you have any
    questions regarding the code presented.
</p>
```

The Box Model

- A **rectangle** is created for each element in the HTML document.
- The **box model** describes how the **padding, border, and margin** are added to the content to create the rectangle.
- Each area is surrounded by a perimeter called an **edge**.



CSS Validator

- An important step in your development journey is **testing** and **debugging** your code.
- Using tools like VSCode allows us to identify errors in our **syntax** and **formatting**, but some errors may go unnoticed.
- We can use other tools like this CSS Validation Service, to check our CSS code as well.
- When our code doesn't behave as expected, or our web pages don't look the way we intended, understanding how to **identify errors** is an important first step before we can **debug**.

CSS Validator

- An important step in your development journey is **testing** and **debugging** your code.
- Using tools like VSCode allows us to identify errors in our **syntax** and **formatting**, but some errors may go unnoticed.
- We can use other tools like this CSS Validation Service, to check our CSS code as well.
- When our code doesn't behave as expected, or our web pages don't look the way we intended, understanding how to **identify errors** is an important first step before we can **debug**.

Hyperiondev

Questions and Answers

HyperionDev

Thank You for attending!