Welcome to the CoGrammar Lecture: DOM Manipulation

The session will start shortly...

Questions? Drop them in the chat. We'll have dedicated moderators answering questions.

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**

Skills Bootcamp
8-Week Progression Overview

Fulfil 4 Criteria to Graduation

✅ Criterion 1: Initial Requirements

Timeframe: First 2 Weeks
Guided Learning Hours (GLH): Minimum of 15 hours
Task Completion: First four tasks

Due Date: 24 March 2024

✅ Criterion 2: Mid-Course Progress

60 Guided Learning Hours

Data Science - **13 tasks**
Software Engineering - **13 tasks**
Web Development - **13 tasks**

Due Date: 28 April 2024

CoGrammar

Skills Bootcamp Progression Overview

✅ Criterion 3: Course Progress

Completion: All mandatory tasks, including Build Your Brand and resubmissions by study period end
Interview Invitation: Within 4 weeks post-course
Guided Learning Hours: Minimum of 112 hours by support end date (10.5 hours average, each week)

✅ Criterion 4: Demonstrating Employability

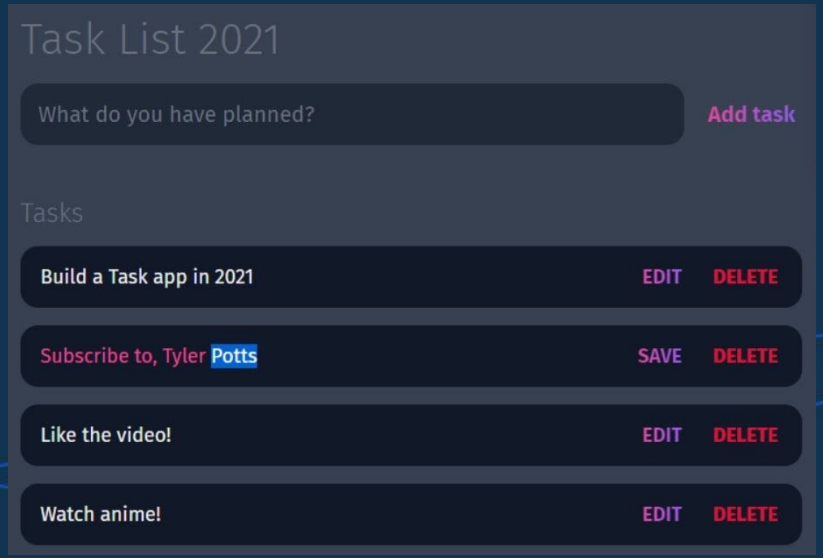
Final Job or Apprenticeship Outcome: Document within 12 weeks post-graduation
Relevance: Progression to employment or related opportunity

Agenda

- Understand the Document Object Model (DOM) and its significance in web development.
- Explore the hierarchical structure of the DOM as a tree data structure.
- Learn various methods for DOM traversal to navigate through the DOM tree.
- Gain proficiency in manipulating the DOM dynamically using JavaScript.

Imagine

- Imagine you're building a web application similar to a to-do list app. You want users to be able to **add** new tasks, **mark** tasks as completed, and **remove** tasks when they're done. How would you approach implementing these functionalities using JavaScript?



CoGrammar

DOM...DOM...DOMMMMM

- What is the **Document Object Model (DOM)**?
  - The **Document Object Model (DOM)** is a programming interface for web documents.
  - It represents the **structure** of HTML documents as a hierarchical **tree** of **objects**.
  - Each **node** in the tree corresponds to a **part** of the document, such as elements, attributes, or text content.
  - The DOM **provides** a structured representation of the document, allowing **scripts** to **dynamically** access, modify, and manipulate its content and structure.

DOM...DOM...DOMMMMM

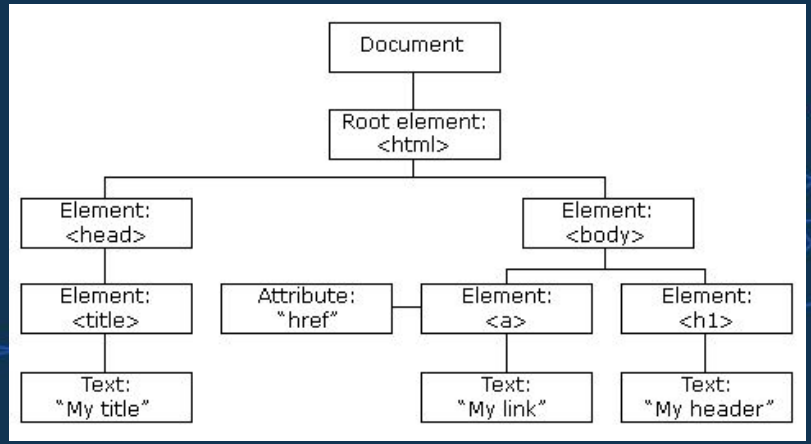
- What is **DOM** Manipulation?
  - DOM manipulation refers to the process of **dynamically** altering the structure, content, or style of web documents using **scripting languages** like JavaScript.
  - It allows developers to create **interactive** and **dynamic** web pages by **accessing** and **modifying** elements in the Document Object Model (DOM).

DOM...DOM...DOMMMMM

- Significance:
  - Enables developers to respond to user actions, update content dynamically, and create engaging user experiences without page reloads.
  - Forms the foundation for modern web applications, facilitating tasks such as form validation, content updates, and animation effects.
  - Empowers developers to create responsive and interactive interfaces, enhancing usability and user engagement.

Tree...2...1

- The DOM **tree** is hierarchical, with a single **root** node.
- Nodes can have parents, children, and siblings.
- Organizes document elements in a **structured** manner.
- Facilitates **traversal** and **manipulation**.
- Provides a logical **representation** of document elements.



CoGrammar

Let’s Breathe!

Let’s take a small break before moving on to the next topic.

CoGrammar

Don’t You...Forget About Me

- The **Document** Object represents the entire HTML document as a tree structure.
- It serves as the entry point to the web page's content, allowing **manipulation** and **interaction** with its elements.
- The **document** object serves as the root node of the Document Object Model (DOM) tree.

Don’t You...Forget About Me

- Offers various **properties** and **methods** for interacting with the document's structure and content.
- Serves as a fundamental component for **dynamic** web development, enabling developers to create **responsive** and **interactive** user interfaces.

Moving through the madness

- DOM **traversal** involves navigating through the DOM tree to access or manipulate elements.

```
// Retrieve the element with the ID "myDiv"
var element = document.getElementById("myDiv");

// Retrieve all elements with the class "container"
var containers = document.getElementsByClassName("container");

// Retrieve all list item elements
var listItems = document.getElementsByTagName("li");

// Retrieve the first paragraph element within a container
var paragraph = document.querySelector(".container p");

// Retrieve all paragraph elements within a container
var paragraphs = document.querySelectorAll(".container p");
```

It’s Morphin’ Time

- Adding elements:

```
// Create a new paragraph element
let paragraph = document.createElement("p");
let heading = document.createElement("h1");

// Add text content to the paragraph
paragraph.textContent = "This is a new paragraph.";
heading.textContent = "I love pie";

// Append the paragraph to the body element
document.body.appendChild(paragraph);
document.body.insertBefore(heading, paragraph);
```

It’s Morphin’ Time

- Modifying Elements:

```
// Change inner HTML content of an element
document.getElementById("myElement").innerHTML = "<strong>New content</st

// Set text content of an element
document.getElementById("myElement").textContent = "Updated text content"

// Set attribute value of an element
document.getElementById("myElement").setAttribute("class", "new-class");
```

It’s Morphin’ Time

- Removing Elements:

```
// Get the element to remove
let elementToRemove = document.getElementById("toRemove");

// Remove the element from the DOM
elementToRemove.remove();
```

Questions and Answers

CoGrammar

Thank you for attending

SKILLS FOR LIFE
SKILLS BOOTCAMPS

Department for Education

CoGrammar