Lesson 6: Introduction to the Document Object Model (DOM)

Objective

By the end of this lesson, students will understand the basics of the Document Object Model (DOM), how to interact with it using JavaScript, and how to manipulate HTML elements dynamically.

1. What is the DOM?

- **Definition**
  - The Document Object Model (DOM) is a programming interface for web documents.
  - It represents the structure of an HTML document as a tree of objects that can be manipulated programmatically.
- **HTML to DOM**
  - When a web page is loaded, the browser creates a DOM of the page.
  - Each element in the HTML document becomes a node in the DOM tree.
- **Importance of the DOM**
  - The DOM allows JavaScript to dynamically access and update the content, structure, and style of a document.
  - This dynamic interaction enables features like animations, form validations, and content updates without reloading the page.

2. Accessing the DOM

- **Document Object**
  - The `document` object represents the entire HTML document. It is the entry point for interacting with the DOM.

```
console.log(document);  // Outputs the entire DOM structure
```

- **Selecting Elements**
  - **`getElementById`**
    - Selects an element by its `id` attribute.

```
let header = document.getElementById('header');
```

  - **`getElementsByClassName`**
    - Selects all elements with a specific class name.

```
let items = document.getElementsByClassName('item');
```

  - **`getElementsByTagName`**
    - Selects all elements with a specific tag name.

```
let paragraphs = document.getElementsByTagName('p');
```

- **`querySelector` and `querySelectorAll`**
  - `querySelector` : Selects the first element that matches a CSS selector.
  - `querySelectorAll` : Selects all elements that match a CSS selector.

```
let firstItem = document.querySelector('.item');
let allItems = document.querySelectorAll('.item');
```

3. Manipulating DOM Elements

- **Changing Content**
  - **`textContent`**
    - Changes the text content of an element.

```
let header = document.getElementById('header');
header.textContent = 'Welcome to My Website';
```

  - **`innerHTML`**
    - Changes the HTML content of an element.

```
let container = document.getElementById('container');
container.innerHTML = '<p>New content here!</p>';
```

- **Changing Attributes**
  - **`setAttribute` and `getAttribute`**
    - `setAttribute` : Sets a new attribute or changes the value of an existing attribute.
    - `getAttribute` : Gets the value of an attribute.

```
let link = document.querySelector('a');
link.setAttribute('href', 'https://www.example.com');
console.log(link.getAttribute('href'));  // Outputs: https://www.example.com
```

- **Changing Styles**
  - You can change the CSS styles of an element using the `style` property.

```
let header = document.getElementById('header');
header.style.color = 'blue';
header.style.fontSize = '24px';
```

4. Creating and Removing Elements

- **Creating Elements**
  - Use `document.createElement` to create a new HTML element.

```
let newDiv = document.createElement('div');
newDiv.textContent = 'Hello, this is a new div!';
document.body.appendChild(newDiv);
```

- **Removing Elements**
  - Use `removeChild` to remove an existing element.

```
let container = document.getElementById('container');
let child = document.querySelector('.child');
container.removeChild(child);
```

5. Event Handling

- **Introduction to Events**
  - Events are actions or occurrences that happen in the browser, like clicks, keypresses, or page loads.
  - You can write JavaScript to respond to these events using event listeners.
- **Adding Event Listeners**
  - Use `addEventListener` to attach an event handler to an element.

```
let button = document.getElementById('myButton');
button.addEventListener('click', function() {
    alert('Button was clicked!');
});
```

- **Common Events**
  - **`click`** : Triggered when an element is clicked.
  - **`mouseover`** : Triggered when the mouse is over an element.
  - **`keydown`** : Triggered when a key is pressed down.

6. Hands-On Practice

Exercise 1: Changing Text on Button Click

- Create a button that, when clicked, changes the text of a paragraph element.

Exercise 2: Creating and Appending Elements

- Write a script that creates a new list item and appends it to an existing unordered list when a button is clicked.

Exercise 3: Removing Elements

- Write a script that removes a list item from an unordered list when a button is clicked.

7. Homework/Assignment

Assignment 1: Dynamic List

- Create a web page with an input field and a button. When the button is clicked, the text from the input field should be added as a new list item to an unordered list.

Assignment 2: Toggle Visibility

- Create a button that toggles the visibility of a paragraph element on the page.

Assignment 3: Simple Form Validation

- Write a script that checks if an input field is empty when a form is submitted. If it is, display an error message.

8. Recommended Resources

Documentation and Tutorials

- MDN Web Docs: Introduction to the DOM
  - An in-depth introduction to the DOM, how it works, and how to interact with it using JavaScript.
- JavaScript.info: The Document Object Model
  - A detailed guide on working with the DOM, including selecting elements, manipulating content, and handling events.

Video Tutorials

- Traversy Media: JavaScript DOM Crash Course
  - A comprehensive video tutorial that covers the basics of the DOM and how to use JavaScript to manipulate web pages.
- Programming with Mosh: JavaScript DOM Tutorial
  - An easy-to-follow guide on how to interact with the DOM using JavaScript, with practical examples and explanations.