

INTRODUCTION TO THE DOM

FULL STACK SKILLS BOOTCAMP

INTRODUCTION TO THE DOM

- **Lesson Overview:**

- In this lesson, we will be introduced to:

1. Introducing the DOM
2. Traversing the DOM
3. Updating elements, content and styles
4. Creating Elements dynamically
5. Event Handlers

INTRODUCTION TO THE DOM

- **What is the DOM?**
 - DOM stands for **Document Object Model**.
 - It represents the structure of a web page as a tree of nodes.
 - Each node is an object representing part of the page (like elements, text, attributes).

TRAVERSING THE DOM

- **What does traversing mean?**

Traversing means navigating the DOM structure.

You can move between **parent**, **child**, and **sibling** nodes.

- **Methods to traverse:**

`document.getElementById()` – Access an element by its ID.

`document.querySelector()` – Select an element using a CSS selector.

`parentNode`, `childNodes` – Move between parent and child nodes.

```
const element = document.getElementById('my-element');  
const parent = element.parentNode;
```

UPDATING ELEMENTS, CONTENT, AND STYLES

- **Updating Content:**
- **Modify inner content** using `innerHTML` or `textContent`.

```
document.getElementById('title').textContent = 'New Title';
```

UPDATING ELEMENTS, CONTENT, AND STYLES

- **Updating Styles:**
 - Change CSS styles dynamically using style property.

```
document.getElementById('box').style.backgroundColor = 'blue';
```

UPDATING ELEMENTS, CONTENT, AND STYLES

- **Modifying Attributes:**

- Change attributes like src, href, class.

```
document.getElementById('link').setAttribute('href', 'https://example.com');
```

CREATING ELEMENTS DYNAMICALLY

- **Why create elements dynamically?**
 - Add content dynamically based on user interactions (e.g., adding new tasks to a TODO list).
- **How to create elements:**
 1. Use `document.createElement()`.
 2. Set properties (text, attributes, etc.).
 3. Append the element to the DOM.

Demo...

```
const newElement = document.createElement('div');  
newElement.textContent = 'New Task';  
document.body.appendChild(newElement);
```


EVENT HANDLERS

■ What are event handlers?

- Functions that run in response to user actions like clicks, typing, or submitting a form.

■ How to use them:

- Attach event listeners to elements using `addEventListener()`.

■ Example:

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

COMMON EVENT HANDLERS

- **Common events:**
 - click – Triggered when a user clicks an element.
 - input – Triggered when the user inputs text.
 - submit – Triggered when a form is submitted.

CONCLUSION

- The **DOM** is the bridge between your HTML content and JavaScript.
- **Traversing the DOM** allows you to access and navigate different elements on the page.
- You can **dynamically update content and styles** to create interactive, responsive web experiences.
- By **creating elements dynamically**, you can add new content on the fly, such as user-generated tasks.
- **Event Handlers** help you respond to user actions, bringing your web page to life.

QUESTIONS?