DOM (Document Object Model)

Introduction

When a webpage is loaded, the browser creates the **DOM (Document Object Model)** of the page.

- The DOM is a **data representation** of the objects that comprise the structure and content of a document on the web.
- It represents an **HTML document in memory**.
- The DOM represents the document as **nodes and objects**, allowing programming languages like JavaScript to interact with the page.
- In both cases (HTML source and DOM), it is the same document, but the **DOM representation allows it to be manipulated dynamically**.

You can inspect the DOM using:

```
console.dir(document);
```

This command displays all properties and methods available on the `document` object.

The DOM follows a **tree-like structure**:

```
window
 └── document
      └── html
           ├── head
           └── body
                └── all sub-nodes
```

DOM Manipulation

DOM Manipulation refers to **selecting, modifying, adding, and deleting HTML elements** using JavaScript.

Selection Methods

1. Selecting by ID

```
document.getElementById("myId");
```

- Selects a single element using its `id`
- Equivalent to `#myId` in CSS

---

2. Selecting by Class Name

```
document.getElementsByClassName("myClass");
```

- Selects elements using class name
- Equivalent to `.myClass` in CSS
- Returns an **HTMLCollection** (array-like object)

---

3. Selecting by Tag Name

```
document.getElementsByTagName("tagName");
```

- Selects elements using tag name
- Returns an **HTMLCollection**

---

4. Query Selector

Used to select **id, class, or tag** automatically using CSS selectors.

```
document.querySelector("selector");
```

- Returns the **first matching element**

```
document.querySelectorAll("selector");
```

- Returns a **NodeList** of all matching elements

DOM Properties

1. `tagName`

```
element.tagName;
```

- Returns the tag name of an element node

---

2. `innerText`

```
element.innerText;
```

- Returns only the **visible text content** of the element and its children
- Does **not** include HTML tags

---

3. `innerHTML`

```
element.innerHTML;
```

- Returns the **HTML content** inside the element
- Includes text as well as nested HTML tags

---

4. `textContent`

```
element.textContent;
```

- Returns **all textual content**, including hidden elements

Attributes

1. Get Attribute

```
element.getAttribute("attr");
```

- Used to get the value of an attribute

---

2. Set Attribute

```
element.setAttribute("attr", "value");
```

- Used to set or update the value of an attribute

Styling Elements

```
element.style.property = "value";
```

- Used to apply CSS styles directly to elements

Example:

```
element.style.color = "red";
```

Insert Elements

1. `append()`

```
node.append(element);
```

- Adds element at the **end** of the node (inside)

---

2. `prepend()`

```
node.prepend(element);
```

- Adds element at the **start** of the node (inside)

---

3. `before()`

```
node.before(element);
```

- Adds element **before** the node (outside)

---

4. `after()`

```
node.after(element);
```

- Adds element **after** the node (outside)

Delete Elements

```
node.remove();
```

- Used to delete the selected node from the DOM
