📘 JavaScript

Day 2 Class Notes

HTML DOM

The **HTML DOM (Document Object Model)** is a programming interface that allows JavaScript to access and manipulate HTML documents dynamically. It treats the document as a **tree of nodes**, where each part of the document (elements, attributes, text) is a node. This enables real-time interactivity and content changes without reloading the page.

🌳 DOM Tree Structure

The DOM represents the structure of an HTML document as a tree:

- document is the root node.
- Tags like <html>, <head>, <body> are child nodes.
- Each HTML element (like <h1>, <p>) is a node.
- Text inside elements is stored in text nodes.

Example DOM Structure:

```
<html>
  <body>
    <h1>Title</h1>
    <p>Paragraph</p>
  </body>
</html>
```

📂 Accessing Elements in the DOM

JavaScript provides different methods to select and access elements:

1. **getElementById()**

   ```
   document.getElementById("demo").innerHTML = "Hello!";
   ```

2. **getElementsByClassName()**

   ```
   let items = document.getElementsByClassName("example");
   items[0].style.color = "red";
   ```

3. **getElementsByTagName()**

   ```
   let paras = document.getElementsByTagName("p");
   paras[0].style.fontWeight = "bold";
   ```

4. **querySelector()** – Selects the **first** element that matches the CSS selector.

   ```
   let element = document.querySelector(".box");
   element.style.backgroundColor = "yellow";
   ```

5. **querySelectorAll()** – Selects **all** matching elements.

   ```
   let elements = document.querySelectorAll("p.highlight");
   elements.forEach(el => el.style.fontSize = "18px");
   ```

✍️ Changing HTML Content and Attributes

1. **innerHTML** – Gets/sets HTML content.

   ```
   document.getElementById("demo").innerHTML = "<b>Hello</b>";
   ```

2. **innerText** – Gets/sets text content only.

   ```
   document.getElementById("demo").innerText = "Hello";
   ```

3. **value** – Used for input fields.

   ```
   let name = document.getElementById("username").value;
   ```

🎨 Changing CSS Styles using JavaScript

You can dynamically apply CSS styles:

```
document.getElementById("box").style.color = "blue";
document.getElementById("box").style.backgroundColor = "lightgray";
```

🛠️ DOM Methods – Creating & Modifying Elements

1. **createElement()** – Create a new element:

   ```
   let newPara = document.createElement("p");
   newPara.innerText = "This is a new paragraph";
   ```

2. **appendChild()** – Add element to the page:

   ```
   document.body.appendChild(newPara);
   ```

3. **removeChild()** – Remove an element:

   ```
   let parent = document.getElementById("container");
   let child = document.getElementById("oldItem");
   parent.removeChild(child);
   ```

4. **replaceChild()** – Replace one node with another:

   ```
   let parent = document.getElementById("container");
   let oldNode = document.getElementById("oldItem");
   let newNode = document.createElement("p");
   newNode.innerText = "New content";
   parent.replaceChild(newNode, oldNode);
   ```

📝 JavaScript Form Validation

Ensure users fill required fields before submitting a form.

Example:

```
<form onsubmit="return validateForm()">
  <input type="text" id="username">
  <input type="submit" value="Submit">
</form>

<script>
function validateForm() {
  let name = document.getElementById("username").value;
  if (name == "") {
    alert("Name must be filled out");
    return false;
  }
  return true;
}
</script>
```
