JavaScript

Day 5

<u>JavaScript HTML DOM Elements (Nodes)</u>

📌 What is a Node?

In the DOM, each part of an HTML document is a **node**:

- The document itself → **Document Node**
- HTML elements → **Element Nodes**
- Text inside elements → **Text Nodes**
- Comments → **Comment Nodes**

🏗️ Creating New Elements

✅ Steps to Add a New Element

1. **Create a new element:**

```
const para = document.createElement("p");
```

2. **Create a text node:**

```
const node = document.createTextNode("This is new.");
```

3. **Append text to the element:**

```
para.appendChild(node);
```

4. **Append the new element to an existing element:**

```
const element = document.getElementById("div1");
element.appendChild(para);
```

📝 insertBefore() — Insert at Specific Position

Instead of adding the new element at the end, you can insert it **before another child**:

```
element.insertBefore(para, child);
```

❌ Removing Elements

✅ Using remove() (Newer Browsers)

```
document.getElementById("p1").remove();
```

🔁 Using removeChild() (For Older Browsers)

1. Get the **parent** element.
2. Get the **child** to remove.
3. Use:

```
parent.removeChild(child);
```

Example:

```
const parent = document.getElementById("div1");
const child = document.getElementById("p1");
parent.removeChild(child);
```

✅ Shortcut using parentNode:

```
const child = document.getElementById("p1");
child.parentNode.removeChild(child);
```

🔁 Replacing Elements

✅ Use replaceChild():

```
const para = document.createElement("p");
const node = document.createTextNode("This is new.");
para.appendChild(node);
```

```
const parent = document.getElementById("div1");
const child = document.getElementById("p1");
parent.replaceChild(para, child);
```

📚 JavaScript HTML DOM Collections

🔹 What is an HTMLCollection?

- An **HTMLCollection** is an **array-like object** that contains a list of HTML elements.
- It is **not a true array**, but you can access items using index numbers like an array.

📌 How to Get an HTMLCollection

Use the getElementsByTagName() method to get a collection of elements with the same tag name:

```
const myCollection = document.getElementsByTagName("p");
```

This collects **all <p> elements** from the document.

🔢 Accessing Elements by Index

HTMLCollections use **zero-based indexing**:

```
myCollection[0]; // First <p> element
myCollection[1]; // Second <p> element
```

📏 HTMLCollection Length

You can check how many elements are in the collection using .length:

```
myCollection.length;
```

Useful when looping through all elements.

🔁 Loop Through HTMLCollection

You can use a for loop to change or access each element:

✅ Example: Change all <p> text color to red

```
const myCollection = document.getElementsByTagName("p");

for (let i = 0; i < myCollection.length; i++) {
  myCollection[i].style.color = "red";
}
```

🔍 Important Notes

- HTMLCollection updates automatically if the document changes (live collection).
- HTMLCollection can be returned by:
    - getElementsByTagName()
    - getElementsByClassName()
    - children property of an element

📚 JavaScript HTML DOM Node Lists

🔷 What is a NodeList?

- A **NodeList** is a **collection of nodes** (elements, text, or comments) from the document.
- It is **similar to an HTMLCollection**, but not the same.

📌 When Do You Get a NodeList?

You get a **NodeList** from:

- document.querySelectorAll() ✅
- childNodes property ✅
- (Sometimes) getElementsByClassName() in old browsers ⚠️

Example: Select all <p> elements

```
const myNodeList = document.querySelectorAll("p");
```

🔢 Accessing Items in a NodeList

You can use **index numbers** just like arrays:

```
myNodeList[0]; // First <p> node
myNodeList[1]; // Second <p> node
```

📏 NodeList Length

Use .length to find out how many nodes are in the NodeList:

```
myNodeList.length;
```

🔁 Loop Through NodeList

You can loop through all nodes using a for loop:

✅ Example: Change text color of all <p> tags to red

```
const myNodeList = document.querySelectorAll("p");

for (let i = 0; i < myNodeList.length; i++) {
  myNodeList[i].style.color = "red";
}
```

🧠 Key Differences: HTMLCollection vs NodeList

Feature	HTMLCollection	NodeList
Returns elements only	✅ Yes	✅ Yes (if from querySelectorAll)
Can include text/comments	❌ No	✅ Yes (e.g. from childNodes)
Live updates (auto-update)	✅ Yes (e.g. getElementsByTagName)	❌ No (static snapshot)
Loop with forEach()	❌ Not always	✅ Often (modern browsers)
