🌳Document Object Model (DOM) - JavaScript Notes

What is the DOM?

- The **DOM** is a **programming interface** for web documents.
- It represents HTML or XML documents as a **tree of nodes**.
- Each HTML element is an **object** that can be accessed and modified using JavaScript.

Example Tree Structure:

```
<html>
  <body>
    <h1>Hello</h1>
    <p>Welcome!</p>
  </body>
</html>
```

DOM Tree:

```
Document
 └── html
      └── body
           ├── h1
           └── p
```

🍂DOM Hierarchy

1. **Document** → root of the DOM tree.
2. **Element** → HTML tags (like `<div>`, `<p>`).
3. **Attribute** → tag properties (`id`, `class`).
4. **Text** → content inside an element.

Accessing DOM Elements

1. By ID

```
let title = document.getElementById("main-title");
```

2. By Class Name

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

3. By Tag Name

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

4. By Query Selector

```
let firstItem = document.querySelector(".item");
```

5. By Query Selector All

```
let allItems = document.querySelectorAll(".item");
```

🐱Modifying Elements

Change Text

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

Change HTML

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

Change Attribute

```
document.getElementById("image").src = "new-image.jpg";
```

Change Style

```
document.getElementById("box").style.backgroundColor = "lightblue";
```

🌓Creating and Adding Elements

```
let newDiv = document.createElement("div");
newDiv.textContent = "New Box";
document.body.appendChild(newDiv);
```

Insert Before

```
let container = document.getElementById("container");
let firstChild = container.firstElementChild;
container.insertBefore(newDiv, firstChild);
```

🪐Removing Elements

```
let element = document.getElementById("oldDiv");
element.remove(); // Modern

// OR
// element.parentNode.removeChild(element);
```

🐈Traversing the DOM

```
let parent = element.parentNode;
let children = element.childNodes;
let first = element.firstElementChild;
let last = element.lastElementChild;
let next = element.nextElementSibling;
let prev = element.previousElementSibling;
```

⚡Attributes

Get / Set / Remove

```
let value = element.getAttribute("class");
element.setAttribute("id", "newId");
element.removeAttribute("style");
```

🌀Event Handling

Inline Event

```
<button onclick="showAlert()">Click Me</button>
```

Using JavaScript

```
let btn = document.getElementById("btn");
btn.onclick = function() {
  alert("Button clicked!");
};
```

Using addEventListener()

```
btn.addEventListener("click", function() {
  alert("Clicked!");
});
```

🪱Event Object

```
btn.addEventListener("click", function(event) {
  console.log(event.type);
  console.log(event.target);
});
```

🦬Common DOM Events

🐭Mouse Events

Event	Description
`click`	When an element is clicked
`dblclick`	Double-click on element
`mouseenter`	Mouse enters element area
`mouseleave`	Mouse leaves element area
`mousemove`	Mouse moves over element

Example:

```
div.addEventListener("mouseenter", () => {
  div.style.backgroundColor = "yellow";
});
```

Keyboard Events

Event	Description
`keydown`	When a key is pressed down
`keyup`	When a key is released
`keypress`	When a key is pressed (deprecated, use keydown)

Example:

```
document.addEventListener("keydown", (e) => {
  console.log("Key pressed:", e.key);
});
```

🦆Form Events

Event	Description
`submit`	When a form is submitted

Event	Description
`change`	When input value changes
`focus`	When an input gets focus
`blur`	When an input loses focus

Example:

```
form.addEventListener("submit", (e) => {
  e.preventDefault();
  console.log("Form submitted!");
});
```

✨DOM Collections

- `HTMLCollection` → Live collection (auto updates)
- `NodeList` → Static collection (does not auto update)

```
document.getElementsByTagName("p"); // HTMLCollection
document.querySelectorAll("p"); // NodeList
```

🍂Common Properties and Methods

Property / Method	Description
`innerHTML`	Gets/sets HTML content
`textContent`	Gets/sets text content
`style`	Access CSS styles
`classList.add()`	Adds a CSS class
`classList.remove()`	Removes a CSS class
`classList.toggle()`	Toggles a CSS class
`appendChild()`	Adds a node at the end
`createElement()`	Creates new element
`querySelector()`	Selects first match

Property / Method	Description
`addEventListener()`	Adds event handler

💐 Best Practices

- Use `querySelector()` / `querySelectorAll()` (modern)
- Avoid inline JS (e.g., `onclick` in HTML)
- Run JS **after DOM loads**:

```
document.addEventListener("DOMContentLoaded", function() {
  // DOM is ready
});
```

🍂 Example

```
<h1 id="title">Hello</h1>
<button id="btn">Change Text</button>

<script>
  const title = document.getElementById("title");
  const btn = document.getElementById("btn");

  btn.addEventListener("click", () => {
    title.textContent = "Welcome to the DOM!";
    title.style.color = "green";
  });
</script>
```