DOM (Document Object Model) - Complete Notes

1. Basic DOM Structure:

- When a webpage loads, the browser creates a "window" object.
- Inside this window object, there's a "document" object representing the entire HTML.
- The document's root node is usually an <html> element.
  - <html> has two main children:
    - a) <head> - contains meta, title, links, etc.
    - b) <body> - contains visible elements like div, p, h1, etc.

2. Head Section:

- Contains metadata like <meta> tags and <title>.
- <title> has a text node inside it.

3. Body Section:

- Contains all visible elements like <div>, <h1>, <p>, etc.
- Each tag can have attributes (e.g., class, id) and children (other elements or text nodes).

4. Accessing DOM Elements:

- document.querySelector("tag/class/id") Returns the first matching element.
- document.querySelectorAll("selector") Returns a NodeList of all matches.
- document.getElementById("id") Returns the element with the given ID.
- document.getElementsByClassName("class") Returns HTMLCollection.
- document.getElementsByTagName("tag") Returns HTMLCollection.

5. Differences:

- textContent vs innerText:

  a) textContent: Returns all text inside an element, including hidden (display: none).

  b) innerText: Returns only visible text (CSS-applied visibility considered).

6. NodeList vs HTMLCollection:

- NodeList supports forEach directly.

- HTMLCollection needs conversion to array for forEach.

7. Modifying DOM Elements:

- .textContent: Changes or gets text.

- .innerHTML: Can get/set HTML content inside an element.

- .setAttribute("attr", "value"): Sets attribute value.

- .getAttribute("attr"): Gets attribute value.

- .classList.add(), .classList.remove(): Manage classes dynamically.

8. Creating & Appending New Elements:

- document.createElement("tagName") Creates a new element.

- element.appendChild(child) Appends a new child node.

- element.append("text" or node) Can append multiple or text.

- element.remove() Removes the element.

9. Traversing the DOM:

- parentElement Gets parent of an element.

- children Gets child elements.

- nextElementSibling / previousElementSibling Navigate between siblings.

10. Event Handling in DOM:

- element.addEventListener("event", callback)

  e.g., button.addEventListener("click", function() { alert("Clicked!"); })

11. Other Useful Properties:

- innerHTML Can dangerously inject HTML (avoid with user input).

- style Inline style manipulation.

- className Entire class string.

- tagName Returns tag name in uppercase.

BEST PRACTICE:

- Avoid using innerHTML for inserting untrusted content (security risk).

- Use querySelector for flexibility with CSS-style selectors.

- Use event delegation for dynamically added elements.

This note covers essential and advanced concepts of DOM for practical frontend development.