DOM Manipulation

Question 1: What is the DOM (Document Object Model) in JavaScript? How does JavaScript interact with the DOM?

Ans.

The Document Object Model (DOM) is a programming interface for web documents. It represents the structure of an HTML or XML document as a tree of objects, where each node in the tree represents a part of the document, such as an element, attribute, or text. This object-oriented representation allows programming languages, particularly JavaScript, to interact with the content, structure, and style of a web page dynamically.

❖How JavaScript Interacts with the DOM:

JavaScript interacts with the DOM through a set of APIs (Application Programming Interfaces) provided by the browser. These APIs allow JavaScript to:

- Access Elements: JavaScript can select specific elements within the HTML document using various methods provided by the document object. Common methods include:
- document.getElementById('idName'): Selects an element by its unique ID.
- document.getElementsByClassName('classNa me'): Selects all elements with a specific class name.
- document.getElementsByTagName('tagName') : Selects all elements with a specific HTML tag name.
- document.querySelector('cssSelector'): Selects the first element that matches a specified CSS selector.
- document.querySelectorAll('cssSelector') : Selects all elements that match a specified CSS selector.

- Modify Content: Once an element is accessed, JavaScript can modify its content:
- element.innerHTML = 'new HTML': Changes the HTML content inside an element.
- element.textContent = 'new text': Changes the text content inside an element.

- Modify Attributes and Styles: JavaScript can change element attributes and inline styles:
- element.setAttribute('attributeName', 'newValue'): Sets or changes the value of an attribute.
- element.style.propertyName = 'value': Changes the inline CSS style of an element.

- Add and Remove Elements: JavaScript can dynamically add new elements to the DOM or remove existing ones:
- document.createElement('tagName'): Creates a new HTML element.
- parentElement.appendChild(childElement): Adds a new child element to a parent.
- parentElement.removeChild(childElement): Removes a child element from a parent.

- Handle Events: JavaScript can attach event listeners to DOM elements to make web pages interactive:
- element.addEventListener('eventName', function): Executes a function when a specific event (e.g., 'click', 'mouseover') occurs on an element.

Question 2: Explain the methods getElementById(), getElementsByClassName(),and querySelector() used to select elements from the DOM.

Ans.

The Document Object Model (DOM) provides several methods to select and manipulate HTML elements within a web page. Three commonly used methods are getElementById(), getElementsByClassName(), and querySelector().

1. document.getElementById()
   This method retrieves a single element from the document by its unique id attribute.
   - Usage: document.getElementById("elementId")
   - Return Value: It returns the Element object with the specified ID, or null if no element with that ID is found.
   - Key Feature: IDs are intended to be unique within a document. Therefore, this method guarantees the selection of a single, specific element.

   Example:
   HTML
   hii
   JavaScript
   const myDiv = document.getElementById("myDiv");
   console.log(myDiv.textContent); // Output: Hello

2. document.getElementsByClassName() This method retrieves a collection of elements that share a specified class name.

   - Usage: document.getElementsByClassName("className")
   - Return Value: It returns a live HTMLCollection of all elements in the document that have the specified class name. This collection is "live"

meaning it automatically updates if elements with that class are added or removed from the DOM.

- Key Feature: Classes can be applied to multiple elements, allowing for the selection and manipulation of groups of elements.

Example:

HTML

```
<P class=”light”>text11</p>
<P class=”light”>text101</p>
```

JavaScript

```
 const highlightedParagraphs =
document.getElementsByClassName("highlight");  for
(let i = 0; i < highlightedParagraphs.length; i++) {
highlightedParagraphs[i].style.color = "blue";
}
```

3. document.querySelector()
   This method returns the first element within the document that matches a specified CSS selector.

   - Usage: document.querySelector("CSS_selector")
   - Return Value: It returns the first Element object that matches the selector, or null if no matching element is found.
   - Key Feature: It offers a flexible way to select elements using any valid CSS selector, including IDs (#id), classes (.class), tag names (tag), or more complex combinations.

   Examples:

   HTML

```
<div id=”cointainer”>
    <p class=”item”>item 1</p>
    <span class=”item”>item11</span>
</div>
```

JavaScript

```
const firstParagraph = document.querySelector("p"); // Selects the first <p>
const containerDiv = document.querySelector("#container"); // Selects the element with id="container"
const firstItem = document.querySelector(".item"); // Selects the first element with class="item"
```
