Assignment – 1

- 25/ July/ 2024, Thursday

1) What is the difference between the DOM and HTML?

Ans: -

Here is a table that outlines the key differences between the DOM (Document Object Model) and HTML:

Aspect	HTML	DOM
Definition	HTML (Hypertext Markup Language) is a standard markup language used to create web pages.	DOM (Document Object Model) is a programming interface for web documents.
Nature	Static content that defines the structure and content of a webpage.	Dynamic representation of a web document that can be manipulated via programming languages like JavaScript.
Structure	Consists of tags, attributes, and content that define elements of a webpage.	Represents the document as a tree of objects with nodes, where each node is a part of the document.
Language	Markup language, used to structure content.	Object-based representation, used to interact with and manipulate HTML and XML documents.
Modification	Requires editing the HTML file directly to make changes.	Can be modified dynamically using scripts (e.g., JavaScript) without altering the original HTML file.
Interaction	Browsers read HTML to render the webpage.	Scripts use the DOM to access, update, and change the document content and structure.
Role in Web Development	Provides the basic structure and content of a webpage.	Allows developers to manipulate the structure, style, and content of the webpage dynamically.
Example	`` `<div id="example">Hello, World!</div>` ``	`` `document.getElementById("example").innerText = "Hi!";` ``
Rendering	Parsed by the browser to display the web page as intended.	Manipulated by JavaScript to alter the web page after it is loaded.
Standards	Governed by the World Wide Web Consortium (W3C) and WHATWG.	Defined and maintained by W3C as a standard interface for accessing and updating documents.

Key Points:

- **HTML**: Describes the structure and content of a webpage using elements like headings, paragraphs, links, images, etc.
- **DOM**: Provides a structured representation of the document as a tree, which can be programmatically accessed and manipulated to change the document's appearance and behavior dynamically.

In summary, HTML is the language used to build the webpage, while the DOM is an interface that allows you to interact with and manipulate that webpage.

---

2) What is the difference between the DOMContentLoaded event and the load event?

Ans: -

Here is a table summarizing the differences between the `DOMContentLoaded` event and the `load` event:

Feature	`DOMContentLoaded`	`load`
Triggered When	The initial HTML document has been completely loaded and parsed	The entire page, including all depend resources such as stylesheets and images, has finished loading
Fires Before	External resources (images, stylesheets, etc.) are fully loaded	All resources are fully loaded
Use Case	Execute code as soon as the DOM is ready, without waiting for all resources	Execute code after the entire page ar all resources have fully loaded
Typical Usage	Initializing UI elements, event listeners, etc.	Executing scripts that depend on the content (e.g., measuring the layout, accessing external resources)
Event Object	`DOMContentLoaded` event object	`load` event object
Code Example	`document.addEventListener('DOMContentLoaded', function() { console.log('DOM fully loaded and parsed'); });`	`window.addEventListener('loa function() { console.log('All resources finished loading!') });`
Performance Impact	Generally faster as it does not wait for resources	Slower because it waits for all resourc to load

This table highlights the primary differences and typical use cases for both events in web development.

---

3) What is the difference between innerHTML and innerText?

Ans: -

Sure, here's a table comparing `innerHTML` and `innerText` in JavaScript:

Feature	`innerHTML`	`innerText`
Definition	Gets or sets the HTML markup inside an element.	Gets or sets the plain text content inside an element.
Rendering	Parses HTML content, including tags.	Treats content as plain text, without HTML parsing.
Use Case	Use when you need to insert or extract HTML elements or markup.	Use when you need to get or set only the visible text content.
Performance	Can be slower, especially if the HTML content is complex or needs to be re-rendered.	Typically faster for reading and writing plain text.
HTML Entities	Will render HTML entities as HTML elements (e.g., `<b>Bold</b>`).	Displays HTML entities as plain text (e.g., `<b>Bold</b>`).
Security	Can introduce security risks like XSS (Cross-Site Scripting) if not properly sanitized.	Generally safer as it doesn't parse HTML, but still be cautious with user input.
Formatting	Maintains the formatting of HTML elements (e.g., `<p>`, `<div>`).	Does not preserve HTML formatting; just displays text as is.

Let me know if you need more details on any of these points!

---

4) What is the role of the Window object in the DOM?

Ans: - The `Window` object is a key part of the web browser's environment. Think of it as the global object that represents the entire browser window or tab. Here's a simple breakdown of its roles:

1. Global Scope:
   - It's like a big container where all the JavaScript code runs. When you write JavaScript, you're usually interacting with this container.
2. Manage the Browser Window:

- It helps control and interact with the browser window or tab. For example, you can use it to open new windows, close them, or resize them.

3. Access to Browser Features:

- It provides access to various browser features, like:

1) `alert()`, `confirm()`, and `prompt()` for user interaction.

2) `setTimeout()` and `setInterval()` for timing functions.

3) `localStorage` and `sessionStorage` for saving data.

4. Document and Navigator:

- It gives you access to the `Document` object (which represents the webpage) and the `Navigator` object (which provides information about the browser).

In short, the `Window` object is like the main hub that manages everything happening in a browser tab, from user interactions to handling different browser features.

---

5) What is a DOM node in JavaScript?

Ans: - A DOM node is like a building block of a webpage. The DOM (Document Object Model) represents the structure of a webpage as a tree of these blocks, each called a "node." Here's a simple way to understand it:

1. Types of Nodes:

- There are several types of nodes, including:

a) Element Nodes:

- These represent HTML tags, like `<p>`, `<div>`, or `<a>`. They form the structure of the webpage.

b) Text Nodes:

- These contain the text inside the HTML tags. For example, in `<p>Hello</p>`, "Hello" is a text node.

c) Attribute Nodes:

- These represent attributes of elements, like `class="my-class"` in `<div class="my-class">`.

2. Tree Structure:

- Imagine the webpage as a tree. The `<html>` tag is the root of this tree, and branches are the different tags and content, forming a hierarchy.

3. Manipulating Nodes:

- JavaScript can interact with these nodes to change the content, style, or structure of the webpage. For example, you

can add, remove, or modify elements and their content using JavaScript.

So, a DOM node is just a single part of this tree that represents a piece of the webpage, and you can use JavaScript to work with these parts to update or change the page.
