**Day7:- Today I learned about DOM Manipulation and Events:**

**DOM:**

The **DOM (Document Object Model)** is a tree-like structure that represents the **HTML** document as an object, allowing access and manipulation of its elements using JavaScript. The **DOM** allows scripts to dynamically change the content, structure, and styles of a web page.

Also, it can be defined as a technology, with the help of which we completely control any **HTML** document with JavaScript. With the help of **DOM**, we can access and change any tag, or class or can change **CSS** style from **DOM**.

Here are some of the most commonly used methods in the DOM API:

* document.getElementById(): Retrieve an element by its unique ID
* document.querySelector(): Retrieve the first element that matches a CSS selector
* document.createElement(): Create a new HTML element
* document.getElementsByClassName(): document.getElementsByClassName() is a method in JavaScript that returns an array-like object of all elements in the document with the specified class name(s).

The syntax to use getElementsByClassName method is:

let elements = document.getElementsByClassName("class-name");

Where class-name is the name of the class you want to search for. The method returns a collection of elements that have the specified class name. You can access individual elements in the collection using an array-like syntax,

for example:

let firstElement = elements[0];

Note that getElementsByClassName returns a live HTMLCollection, which means that the collection is updated automatically as elements are added or removed from the document.

* document.getElementsByTagName()

element.innerHTML: Get or set the HTML content within an element

* element.textContent: The textContent property of an element in JavaScript returns the text content of a node and its descendants as a string. It is used to get or set the text content of an HTML element. It can be used to update or retrieve the text content of an element and is often used to manipulate the text content of a page dynamically with JavaScript.
* element.style: Access the inline styles of an element
* element.classList: Access and manipulate the class names of an element.

element.appendChild():Add a new child element to an element

* element.removeChild():Remove a child element from an element
* element.insertBefore():The insertBefore() method is a JavaScript function used to insert an element into the DOM (Document Object Model) as a child of a specified parent node, before a specified reference element.
* element.replaceChild():The element.replaceChild(newChild, oldChild) method in JavaScript replaces a child node of an element with a new node.

Syntax: element.replaceChild(newChild, oldChild)

* element.setAttribute():Set the value of an attribute on an element
* element.getAttribute():The element.getAttribute(attributeName) method in JavaScript returns the value of the specified attribute of an element.

Syntax: element.getAttribute(attributeName)

* element.hasAttribute():The element.hasAttribute(attributeName) method in JavaScript returns a Boolean indicating whether the specified element has the specified attribute or not.

Syntax: element.hasAttribute(attributeName)

* element.removeAttribute():The element.removeAttribute() method removes an attribute from an HTML element. It takes one argument, the name of the attribute to be removed.
* element.addEventListener():The element.addEventListener() method is used to attach an event listener to an HTML element. It takes two arguments: the type of event to listen for and the function to be called when the event occurs (the event handler or callback).
* element.removeEventListener():The element.removeEventListener() method removes an event listener from an HTML element. It takes two arguments: the type of event to remove and the function that was used as a callback.

**DOM methods in javascript for access html code**

The DOM (Document Object Model) provides a way to access and manipulate HTML elements using JavaScript. Here are some of the most commonly used methods for accessing HTML code in the DOM:

1. document.getElementById(id): Returns the element with the specified id.

For example:

let element = document.getElementById("example");

1. document.querySelector(selector): Returns the first element that matches the specified selector.

For example:

let element = document.querySelector("#example");

1. document.getElementsByClassName(className): Returns a list of elements with the specified class name. For example:

let elements = document.getElementsByClassName("example");

1. document.getElementsByTagName(tagName): Returns a list of elements with the specified tag name. For example:

let elements = document.getElementsByTagName("p");

1. element.innerHTML: Gets or sets the HTML content inside an element. For example:

let element = document.getElementById("example");

let content = element.innerHTML;

These are just a few examples of the methods available in the DOM API for accessing HTML code. Once you have access to an HTML element, you can manipulate its properties, styles, and content using the DOM API.

Resource: <https://www.dynamicwebtraining.com.au/blog/javascript-dom-methods>

**Events:**

Events in JavaScript are actions or occurrences that happen in the browser, such as a user clicking on a button, a page finishing loading, or an element being updated. JavaScript provides a way to respond to these events through event handlers, which are functions that are executed when the event occurs.

The change in the state of an object is known as an **Event**. In html, there are various events which represents that some activity is performed by the user or by the browser. When [javascript](https://www.javatpoint.com/javascript-tutorial) code is included in [HTML](https://www.javatpoint.com/html-tutorial), js react over these events and allow the execution. This process of reacting over the events is called **Event Handling**. Thus, js handles the HTML events via **Event Handlers**.

**For example**, when a user clicks over the browser, add js code, which will execute the task to be performed on the event.

Examples of common events in JavaScript include:

* click
* load
* submit
* change
* mouseover
* hover
* keydown
* focus
* blur
* resize
* scroll.

There are several types of events in JavaScript, including:

* UI events: User interface events such as mouse clicks, scrolling, and keyboard input.
* Form events: Events related to form elements such as submit, change, and focus.
* Document/Window events: Events related to the Document Object Model (DOM) or the window object, such as load, resize, and unload.
* Keyboard events: Keyboard-related events such as keydown, keypress, and keyup.
* Mouse events: Mouse-related events such as click, dblclick, and mouseover.
* Touch events: Touchscreen events such as touchstart, touchmove, and touchend.
* Drag and drop events: Events related to the drag-and-drop functionality, such as dragstart, drag, and drop.
* Media events: Events related to multimedia elements such as audio and video, such as play, pause, and ended.
* Animation events: Animation-related events such as animationstart, animationend, and animationiteration.

These are some of the most common types of events in JavaScript, and they can be handled using event handlers or listeners.

Resource: <https://www.javatpoint.com/javascript-events>

<https://developer.mozilla.org/en-US/docs/Learn/JavaScript/Building_blocks/Events>