JavaScript

4.1 Basics of JavaScript

4.1.1 JavaScript Introduction

- JavaScript is a widely-used programming language for creating interactive and dynamic web pages.

- It was created by Brendan Eich in 1995

- The language is powered by the V8 engine, a JavaScript engine developed by Google. V8 compiles and executes JavaScript code at high speeds, contributing to the language's efficiency and widespread use.

4.1.2 Use of JavaScript

- JavaScript can be used for the frontend using 2 methods:
    - Writing pure JavaScript.
    - Using libraries and frameworks like ReactJS, AngularJS, jQuery, etc.

- JavaScript can also be used on the server side with technologies like Node.js to build scalable and efficient server applications.

- JavaScript allows developers to add interactivity to web pages, validate forms, handle events, and create animations.

4.1.3 Way to include javascript

1. **Writing the JavaScript code inside the HTML page:**
    - JavaScript can be written inside an HTML page using the ‘<script>’ tag
    - We can place the script tag inside the head tag or inside the body tag or both.
      Note: Writing javascript at the end of the body section is generally preferred because it helps the browser to load and display the main content of the page before it processes the JavaScript resulting in a faster page load time.
    - Eg:

```
<body>
    <script>
        search_query = document.getElementById("query");
    </script>
</body>
```

2. Using script tag to use external JS file
    - To include an external JavaScript file, we can use the script tag with the attribute src.

```
<script src="script.js"></script>
```

    - **Defer** attribute is used to make sure that script is executed after the HTML content is parsed.

```
<script src="script.js" defer></script>
```

4.1.4 Syntax of JavaScript

1. Variable Declaration

- Variables can be declared using var, let, or const.
- Data types supported in JavaScript are string, number, boolean, null, undefined, object, array, and function.
- As Javascript is loosely typed there is no need of defining the type of variable explicitly.
- Eg:

```
var name = "Raj"
let age = 20
const hasDegree = true
```

2. DOM Manipulation

- Inorder to change HTML content we need to interact with DOM.
- A specific element can be selected using ID
- Multiple elements can also be selected using Tags and classes.
- Eg:

```
document.getElementById("title").innerHTML = "This is GUI training";
```

3. Adding Event Listeners

- Using event listeners we can execute a function on an event caused by the user.

```
document.getElementById("submit").addEventListener("click",
function() {
        alert("Form Submitted");
    });
```

4. **Commenting**
- Use // can be used to comment a single line
- /* */ is used to comment multiple lines.

Eg:

```
- // I am Raj
-     /* This is GUI Training
-         For full stack developer */
```

4.1.5 Basic event of javascript

- Events in JavaScript allow you to respond to interactions or occurrences on a web page, such as user actions or system events.
- By attaching event handlers to elements, you can execute JavaScript code when specific events take place

Event Types:

- **Click**: Triggered when a mouse click occurs.
- **Keydown/Keyup**: Fired when a key on the keyboard is pressed or released.
- **Mouseover/Mouseou**t: Occurs when the mouse pointer enters or leaves an element.
- **Submit**: Triggered when a form is submitted.
- **Load**: Triggered when a webpage or an element finishes loading.

Event Handling:

- To handle events using JavaScript, we have to attach event handlers. They can be attached using an inline event handler or add them dynamically through scripts.

```
// Inline event handler
<button id=”myButton” onclick="showMessage()">Click Me</button>

// Add event listener
var element = document.getElementById("myButton");
element.addEventListener("click", showMessage);
```

Event Object:

- Event handlers receive an event object containing information about the event.
- We can access properties like “event.target” to identify the element that triggered the event.

```
function showMessage(event) {
    var targetElement = event.target;
    // Process on the element
}
```

Preventing Default Actions:

- Using event.preventDefault() we can prevent the default behavior associated with an event.
- Eg:
    - Preventing a form from submitting
    - Link from navigating.
    -

```
function preventLink(event) {
    event.preventDefault();
    // Action to perform after onlick of link
}
```

4.1.6 Basic Validation with javascript

- Form validation using JavaScript ensures that user-submitted data is accurate, complete, and properly formatted before it is processed or submitted to a server.

- Validation using JavaScript can also be called **Frontend sanitization**.

- **Note**: If User disables JavaScript from the browser, Validation cannot be performed. In this case, **Backend Sanitization** should always be done.

  Eg:

HTML

```html
<form id="RegForm">
        <input type="text" id="name" placeholder="Name">
        <input type="email" id="email" placeholder="Email">
        <input type="password" id="password" placeholder="Password">
        <button type="submit">Submit</button>
    </form>
```

JS:

```javascript
document.getElementById("RegForm").addEventListener("submit",function(even
t){
    var name = document.getElementById("name").value;
    var email = document.getElementById("email").value;
    var password = document.getElementById("password").value;

    if (name === "" || email === "" || password === "") {
        alert("All fields are required");
        event.preventDefault(); // Prevent form submission
    }
});
```