JavaScript

4.1 Basics of JavaScript

4.1.1 JavaScript Introduction: -

JavaScript was initially created to "make web pages alive". The programs in this language are called scripts. They can be written right in a web page's HTML and run automatically as the page loads. Scripts are provided and executed as plain text. They don't need special preparation or compilation to run. In this aspect, JavaScript is very different from another language called Java.

When JavaScript was created, it initially had another name: "LiveScript". But Java was very popular at that time, so it was decided that positioning a new language as a "younger brother" of Java would help. But as it evolved, JavaScript became a fully independent language with its own specification called ECMAScript, and now it has no relation to Java at all.

4.1.2 Use of JavaScript: -

JavaScript is a versatile and widely used programming language primarily known for its role in creating interactive and dynamic elements on websites. It was initially developed by Netscape as a client-side scripting language to enhance the user experience of web pages. Over time, JavaScript has evolved into a powerful and versatile language that can be used for both front-end and back-end development. Its versatility and capabilities make it an essential tool for creating dynamic and interactive user experiences.

- **Web Page Interactivity**: JavaScript is used to enhance the interactivity of web pages. It enables features such as dropdown menus, image sliders, interactive forms, and real-time updates without requiring a full page reload.
- **DOM Manipulation**: JavaScript is used to interact with the Document Object Model (DOM) of a web page. This allows developers to dynamically create, modify, and delete HTML and CSS elements, providing a seamless and responsive user experience.
- **Event Handling**: JavaScript enables developers to respond to various user interactions such as clicks, keyboard inputs, and mouse movements. This interaction drives the behavior of web applications and makes them more user-friendly.
- **Client-Side Validation**: JavaScript is commonly used to validate form inputs on the client side before they are submitted to a server. This helps ensure that users provide correct and properly formatted data.
- **AJAX (Asynchronous JavaScript and XML)**: JavaScript enables asynchronous communication with servers, allowing data to be fetched and updated in the background without requiring a full page refresh. This is crucial for creating responsive and dynamic web applications.
- **Animations and Effects**: JavaScript is used to create animations, transitions, and visual effects on web pages. This enhances the visual appeal and engagement of the user interface.
- **Server-Side Development**: With the introduction of technologies like Node.js, JavaScript can now be used for server-side programming. This allows developers to build scalable and efficient server applications using a single language throughout the entire stack.

4.1.3 Way to include JavaScript: -

We can add javascript in html file using <script> tag. Place the javascript code between opening '<script>' and closing '</script>' tags.

```
<html>
    <head>
        <title> JavaScript Demo </title>
    </head>
    <body>
        <script>
            alert("Hello");
        </script>
    </body>
</html>
```

We can add <script> tag in <body> tag as well as in <head> tag also.

We can add external script keep html separate from javascript.

```
<html>
    <head>
        <title> JavaScript Demo </title>
        <script src="script.js"> </script>
    </head>
    <body>
    </body>
</html>
```

4.1.4 Syntax of JavaScript: -

- **Javascript values**: -

  The JavaScript syntax defines two types of values:

  - **Fixed values**: - Fixed values are called Literals.
    1. Numbers are written with or without decimals:
       1000,
       10.00
    2. Strings are text, written within double or single quotes:
       "Hello"
       'Hello'

  - **Variable values**: - Variable values are called Variables. JavaScript uses the keywords var, let and const to declare variables.
    let x=15;

- **Javascript operators**: -
  - **Assignment operator:** -

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
***=**	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y
****=**	x **= y	x = x ** y

  - **Shift assignment operator**: -

Operator	Example	Same As
<<=	x <<= y	x = x << y
>>=	x >>= y	x = x >> y
>>>=	x >>>= y	x = x >>> y

- **Bitwise Assignment Operators: -**

Operator	Example	Same As
&=	x &= y	x = x & y
^=	x ^= y	x = x ^ y
|=	x |= y	x = x | y

- **Logical Assignment Operators: -**

Operator	Example	Same As
&&=	x &&= y	x = x && (x = y)
||=	x ||= y	x = x || (x = y)
??=	x ??= y	x = x ?? (x = y)

- **Javascript is Case sensitive and camel-case: -**
  - **Case sensitive: -**

```
let lastname, lastName;
lastName = "ABC";
lastname = "XYZ";
```

  - **Camel case: -**
    1. **Underscore**: - first_name, last_name, master_card, inter_city.
    2. **Upper Camel Case (Pascal Case)**: - FirstName, LastName, MasterCard,
    3. **Lower Camel Case**: - firstName, lastName, masterCard, interCity.

- **Comments in javascript: -**
  - **Single line comments: -**

```
// This is a single-line comment
```

  - **Multi-line comments: -**

```
/* This is a
   multi-line comment */
```

- **Variables and Datatypes: -**
  - **Declare variables: -**

```
let name = "John";

const age = 30;

var score = 95; // Avoid using 'var' for variable declaration
```

  - **Data types: -**

```
let text = "Hello, World!";

let number = 42;

let isTrue = true;

let someValue = null;

let undefinedValue = undefined;
```

- **Function in JavaScript: -**

```
function name(parameter1, parameter2, parameter3) {
  // code to be executed
}
```

- **Object in JavaScript: -**
  - **Definition: -**

```
const person = {
  firstName: "John",
  lastName: "Doe",
  age: 50,
  eyeColor: "blue"
};
```

  - **Accessing object Property: -**

```
objectName.propertyName
            OR
objectName["propertyName"]
```

- **String methods: -**
  - String length: -
  - String slice(): -
  - String substring()
  - String substr()
  - String replace()
  - String replaceAll()
  - String toUpperCase()
  - String toLowerCase()
  - String concat()
  - String trim()
  - String trimStart()
  - String trimEnd()
  - String padStart()
  - String padEnd()
  - String charAt()
  - String charCodeAt()
  - String split()
  - String indexOf()
  - String lastIndexOf()
  - String search()
  - String match()
  - String matchAll()
  - String includes()
  - String startsWith()
  - String endsWith()

- **Array's methods: -**
  - Array length
  - Array toString()
  - Array pop()
  - Array push()
  - Array shift()
  - Array unshift()
  - Array join()
  - Array delete()
  - Array concat()
  - Array flat()
  - Array splice()
  - Array slice()

4.1.5 Basic event of JavaScript: -

- **Mouse events: -**
  - **click**: When click on an element
  - **dblclick** : When double clicks on an element
  - **mousedown**: Press a mouse button over an element
  - **mousemove**: User moves the mouse over an element
  - **mouseover**: mouse over an element
  - **mouseout**: User moves the mouse out of an element
  - **mouseup**: Releases a mouse button over an element

- **Keyboard events: -**
  - **keydown:** when press a key or holding down a key
  - **keypress:** when press a key or holding down a key
  - **keyup:** When keyboard key is released

- **Form events: -**
  - **blur**: when form element loses focus
  - **change**: When the content of form element changed
  - **focus**: When elements get focused
  - **reset**: event occurs when a form is reset
  - **select**: event occurs when user selects some text
  - **submit**: event occurs when form is submit

```
<!DOCTYPE html>
<html>

<head>
  <title>Mouse Event Example</title>
</head>

<body>
  <button id="colorChanger">Click me to change the color</button>
  <div id="changeThis">This is the element to change</div>

  <script>
    const button = document.getElementById('colorChanger');
    const elementToChange = document.getElementById('changeThis');
```

```
            button.addEventListener('click', function() {
elementToChange.style.backgroundColor = 'lightblue';

    });
  </script>
</body>

</html>
```

4.1.6 Basic validation with JavaScript: -

Basic validation with JavaScript typically involves checking user input to ensure that it meets certain criteria or constraints before processing it further. This can be useful for forms on websites to ensure that users enter valid data.

Basic validation such as emptiness, confirm password, length validation etc.. Data format validation such as Email validation, mobile number validation, etc.. We can validate data using RegExp also in JavaScript.

Example:

```
var pattern = "^[\\w]+$';

var regex = new RegExp(pattern);

if(regex.test(teststring){

        alert(valid);

}

else{

        Alert(invalid);

}
```

n+	String contains at least one n
n*	String contains zero or more occurrence of n
n?	String contains zero or one occurrence of n
n$	String end with n
^n	String start with n
n{x}	String contains sequence of x n's
n{x,y}	String contains a sequence of x to y n's
N{x,}	String contains sequence of at least x n's