Tutorial No. 07

Title: JavaScript Methods and Function, DOM Manipulation.

KJSCE/IT/SYBTech/SEM IV/WP1/2022-23

Batch:A2 Roll No.:16010421063

Tutorial:7 Aim: To implement javascript methods, functions and Event Handling to manipulate DOM

Resources needed: Notepad++, Web Browser

Theory:

JavaScript is a scripting language produced by Netscape for use within HTML Web pages. JavaScript is loosely based on Java and it is built into all the major modern browsers. JavaScript is a lightweight, interpreted programming language, Complementary to and integrated with Java, Complementary to and integrated with HTML, Open and cross-platform and is case sensitive.

Placing JavaScript in HTML document:

There is a flexibility given to include JavaScript code anywhere in an HTML document.
But there are the following most preferred ways to include JavaScript in your HTML file:

1. Script in <head>...</head> section.
2. Script in <body>...</body> section.
3. Script in <body>...</body> and <head>...</head> sections.
4. Script in and external file and then include in <head>...</head> section. An

example of it is shown below:

```
<html>
<body>
<script language="javascript" type="text/javascript">
<!--
document.write("Hello World!")
//-->
</script>
</body>
</html>
```

Looping and Control statements in JavaScript:

- if statement syntax:

```
if (expression){
   Statement(s) to be executed if expression is true
}
```


- if else statement syntax:

```
if (expression){
   Statement(s) to be executed if expression is true
}else{
   Statement(s) to be executed if expression is false
```

```
}
```

- else if ladder syntax:

```
if (expression 1){
  Statement(s) to be executed if expression 1 is true
}else if (expression 2){
  Statement(s) to be executed if expression 2 is true
}else if (expression 3){
  Statement(s) to be executed if expression 3 is true
}else{
  Statement(s) to be executed if no expression is true
}
```

- switch statement syntax:

```
switch (expression)
{
 case condition 1: statement(s)
                   break;
 case condition 2: statement(s)
                   break;
  ...
 case condition n: statement(s)
                   break;
 default: statement(s)
}
```

- While Loop

```
while (expression){
  Statement(s) to be executed if expression is true
 }
```

- do- while Loop

```
do{
  Statement(s) to be executed;
} while (expression);
```


- for Loop

```
for(initialization; test condition; iteration statement){
Statement(s) to be executed if test condition is true
}

 for (variablename in object){
```

```
    Statement or block to execute
}
```

Syntax for JavaScript functions:

```
function concatenate(first, last)
{
  var full;
  full = first + last;
  return full;
}
```

To invoke a function somewhere later in the script, you would simple need to write the name of that function.

Javascript Dialog boxes:

JavaScript supports three important types of dialog boxes. These dialog boxes can be used to raise and alert, or to get confirmation on any input or to have a kind of input from the users:

- Alert Dialog Box:
  An alert dialog box is mostly used to give a warning message to the users.
  ```
  alert("Warning Message");
  ```

- Confirmation Dialog Box:
  A confirmation dialog box is mostly used to take user's consent on any option. It displays a dialog box with two buttons: OK and Cancel.
  ```
  var retVal = confirm("Do you want to continue ?");
  ```
- Prompt Dialog Box:
  The prompt dialog box is very useful when you want to pop-up a text box to get user input. Thus it enables you to interact with the user. The user needs to fill in the field and then click OK.

  ```
  var retVal = prompt("Enter your name : ", "your name here");
  ```

In built objects in JavaScript:

A String object encapsulates a sequence of characters, enclosed in quotes

properties include :

- length : stores the number of characters in the string

- charAt(index) : returns the character stored at the given index (as in indices start at 0)
- substring(start, end) : returns the part of the string between the start (inclusive) and end (exclusive) indices
- toUpperCase() : returns copy of string with letters uppercase ● toLowerCase() : returns copy of string with letters lowercase

Arrays store a sequence of items, accessible via an index since JavaScript is loosely typed, elements do not have to be the same type. To create an array, allocate space using new (or can assign directly):

```
items = new Array(10); // allocates space for 10 items
items = new Array(); // if no size given, will adjust dynamically items =
[0,0,0,0,0,0,0,0,0,0]; // can assign size & values []
```

To access an array element, use [] (as in C++/Java)

```
for (i = 0; i < 10; i++) {
    items[i] = 0; // stores 0 at each index
}
```

The length property stores the number of items in the array.

The Date object can be used to access the date and time. To create a Date object, use new & supply year/month/day/… as desired

```
today = new Date(); // sets to current date & time
newYear = new Date(2002,0,1); //sets to Jan 1, 2002 12:00AM
```

Methods can access individual components of a date includes:

```
newYear.getYear()
newYear.getMonth()
newYear.getDay()
newYear.getHours()
newYear.getMinutes()
newYear.getSeconds()
newYear.getMilliseconds()
```

Document Object Model(DOM):

DOM Objects can be referenced using JavaScript

- by their id or name (this is the easiest way, but you need to make sure a name is unique in the hierarchy)

● by their numerical position in the hierarchy, by walking the array that contains them ● by their relation to parent, child, or sibling (parentNode, previousSibling, nextSibling, firstChild, lastChild or the childNodes

JavaScript and DOM:

JavaScript is used to manipulate the objects. For this id of an element is needed to be passed to method getElementById() of document object, which returns the element with the given id. And then we can alter its property.

For example,
if you want to find a <p> with the id of "cool", use:

```
getElementById("cool")
document.getElementById(item).style.backgroundColor =color;
```

to access the elements nested in <p> tag we can use,

```
document.getElementById(item).childNodes[1].style.backgroundColo
```

r =color; **Document Object:**

innerHTML is a property of any document element that contains all of the html source and text within that element.

```
getElementById("cool").innerHTML ="new text string";
```

Methods:

document.write(…) : method that displays text in the page

document.URL : property that gives the location of the HTML document

document.lastModified :property that gives the date & time the HTML document was last changed

Activity:

1. Explore different methods of in-built JavaScript objects date, string, math, array etc. 2. Include at least two significant methods of some of these objects in your script 3. Extract elements of document using DOM and manipulate same using methods

Results: (Program printout with output)

```
<!DOCTYPE html>

<html lang="en">
```

```
    <head>

        <meta charset="UTF-8" />
        <meta http-equiv="X-UA-Compatible" content="IE=edge" />

        <meta name="viewport" content="width=device-width,
initial-scale=1.0" />

        <title>Document</title>

    </head>

    <body>

        <p id="demo"></p>

        <p id="demo1"></p>

        <p id="demo2"></p>

        <p id="demo3"></p>

        <p id="demo4"></p>

        <p id="demo5"></p>

        <p id="demo6"></p>

        <p id="demo7"></p>

        <p id="demo8"></p>

        <p id="demo9"></p>

        <p id="demo10">hellp</p>

        <input type="text" placeholder="Enter text" onblur="getVal()" />

        <script>

            const d = new Date();

            let text = d.getDay();

            const weekday = [
```

```
    "Sunday",
    "Monday",

    "Tuesday",

    "Wednesday",

    "Thursday",

    "Friday",

    "Saturday",

];

const month = [

    "January",

    "February",

    "March",

    "April",

    "May",

    "June",

    "July",

    "August",

    "September",

    "October",

    "November",

    "December",

];

document.getElementById("demo").innerHTML = weekday[text];
```

```
            document.getElementById("demo1").innerHTML = d.getDate();

            document.getElementById("demo2").innerHTML = d.getMinutes();

            document.getElementById("demo3").innerHTML =
month[d.getMonth()];

            document.getElementById("demo4").innerHTML =
d.toUTCString();

            document.getElementById("demo5").innerHTML = d.toJSON();



            const name = "Arya";

            document.getElementById("demo6").innerHTML = name.length;

            document.getElementById("demo7").innerHTML =
name.toUpperCase();

            document.getElementById("demo8").innerHTML =
name.toLowerCase();

            document.getElementById("demo9").innerHTML = Math.random();

            function getVal() {

                const val = document.querySelector("input").value;

                console.log(val);

            }

        </script>

    </body>

</html>
```

Questions:

Q1) Explain with examples on how Javascripts help in creating dynamic HTML page.

JavaScript can help create dynamic HTML pages by:

Changing the content and layout of a page dynamically

Handling user events, such as mouse clicks and key presses

Loading dynamic content without reloading the entire page

For example, JavaScript can change the text of a button when it's clicked, display an alert message when a user submits a form, or load new data into a table when the user scrolls down the page. These interactions make the web page more dynamic and engaging for the user.

Q2) What is DOM? Explain.

DOM stands for Document Object Model. It is a programming interface for web documents, which provides a structured representation of an HTML or XML document. The DOM represents the document as nodes and objects, which can be manipulated with programming languages like JavaScript.

In other words, the DOM is a tree-like structure that represents the elements and content of an HTML or XML document. Each element, attribute, and text node in the document is represented as an object in the DOM tree. These objects can be accessed and manipulated using JavaScript to change the content and behavior of a web page dynamically.

For example, with the DOM, you can access and modify the content of an HTML element, change the style of an element, add or remove elements dynamically, and respond to user events like mouse clicks or key presses.

The DOM is important for web developers because it provides a standardized way to access and manipulate web page content across different browsers and platforms. This makes it possible to create dynamic and interactive web pages that work well on different devices and web browsers.

Outcomes: CO 3 Apply JavaScript and JSON for Web Application development

Conclusion: (Conclusion to be based on the outcomes achieved) SUccessfully implemented various DOM and javascript methods

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References:

Books/ Journals/ Websites:

- "Web technologies: Black Book", Dreamtech Publications
- http://www.w3schools.com