Course Code: SE-3003	Course : Web Engineering Lab

Spring 2023, Lab Manual – 06

LLO 02: Design and Implement a web application using JavaScript.

Contents:

- Intro to Web Engineering
- Technologies
- Tools
- Introduction to JavaScript
- JavaScript Functions
- Events
- Validation

Introduction to Web Engineering

Web Engineering is the application of systematic and quantifiable approaches (concepts methods, techniques tools) to cost - effective requirements analysis, design, implementation, testing, operation, and maintenance of **high quality Web applications.**

Technologies to be studied

- HTML
- CSS
- JavaScript
- Bootstrap
- JQuery
- PHP
- MySQL [Database]
- Laravel [PHP FRAMEWORK]

Tools – IDEs

- Visual Studio Code
- Adobe Dreamweaver
- Visual Studio

6.1 Introduction to JavaScript:

JavaScript often abbreviated as JS, is a programming language that is one of the core technologies of the World Wide Web, alongside HTML and CSS. As of 2022, 98% of websites use JavaScript on the client side for webpage behavior, often incorporating third-party libraries. All major web browsers have dedicated JavaScript engine to execute the code on users' devices. JavaScript is a high-level, often just-in-time compiled language that conforms to the ECMAScript standard. It has dynamic typing, prototype-based object-orientation, and first-class functions. It is multi-paradigm, supporting event-driven, functional, and imperative programming styles. It has application programming, interfaces (APIs) for working with text, dates, regular expressions, standard data structures, and the Document Object Model (DOM).

The ECMAScript standard does not include any input/output (I/O), such as networking, storage, or graphics facilities. In practice, the web browser or other runtime system provides JavaScript APIs for I/O. JavaScript engines were originally used only in web browsers, but are now core components of some servers and a variety of applications. The most popular runtime system for this usage is Node.js. Although Java and JavaScript are similar in name, syntax, and respective standard libraries, the two languages are distinct and differ greatly in design.

Scripting language vs programming language

Programming Language	Scripting Language
A programming language is an organized way of communicating with a computer.	A scripting language is a programming language that supports scripts.
Traditional programming is based on low level languages.	Scripting prefers high level languages.
The traditional programming languages such as C, C++, and Java are compiled.	Perl, Python, JavaScript, and other languages used for scripting are interpreted and do not require the compilation step.
General programming leads to closed software applications.	Scripting promotes open projects and is used for web applications.
More code needs to be written.	Less coding is required in scripting.

JavaScript Programs

A computer program is a list of "instructions" to be "executed" by a computer. In a programming language, these programming instructions are called statements. A JavaScript program is a list of programming statements.

In HTML, JavaScript programs are executed by the web browser.

JavaScript Statements

JavaScript statements are composed of: Values, Operators, Expressions, Keywords, and Comments. This statement tells the browser to write "Hello Dolly." inside an HTML element with id="demo":

Variable declaration using VAR, LET and CONST:

	let	var	const
Scope	Block	Function	Block
Declaring same name	Not possible	Possible	Not possible
Updating value	Possible	Possible	Not possible (Property is updatable)
Use before declaration	Error	Undefined	Error

Types of JavaScript Operators

There are different types of JavaScript operators:

- ## Arithmetic Operators

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
**	Exponentiation (ES2016)
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

Example:

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

<script>

let a = 3;

let x = (100 + 50) * a;

document.getElementById("demo").innerHTML = x;

</script>
```

- ## Assignment Operators

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

Assignment operators assign values to JavaScript variables. The Addition Assignment Operator (+=) adds a value to a variable.

Example:

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

<script>

var x = 10;

x += 5;

document.getElementById("demo").innerHTML = x;

</script>
```

- **Comparison Operators**

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

- **Logical Operators**

Operator	Description
&&	logical and
||	logical or
!	logical not

- **Type Operators**

Operator	Description
typeof	Returns the type of a variable
instanceof	Returns true if an object is an instance of an object type

Conditional Statements:

IF ELSE FAMILY:

Conditional statements are used to perform different actions based on different conditions. Conditional Statements Very often when you write code, you want to perform different actions for different decisions. You can use conditional statements in your code to do this.In JavaScript we have the following conditional statements:

Use if to specify a block of code to be executed, if a specified condition is true Use else to specify a block of code to be executed, if the same condition is false Use else if to specify a new condition to test, if the first condition is false Use switch to specify many alternative blocks of code to be executed

The if Statement

Use the `if` statement to specify a block of JavaScript code to be executed if a condition is true.

Syntax

```
if (condition) {
  //  block of code to be executed if the condition is true
}
```

```
if (hour < 18) {
  greeting = "Good day";
}
```

Switch Statement

The switch statement is used to perform different actions based on different conditions.

```
switch(expression) {
  case x:
    // code block
    break;
  case y:
    // code block
    break;
  default:
    // code block
}
```

```
switch (new Date().getDay()) {
  case 0:
    day = "Sunday";
    break;
  case 1:
    day = "Monday";
    break;
  case 2:
     day = "Tuesday";
    break;
  case 3:
    day = "Wednesday";
    break;
  case 4:
    day = "Thursday";
    break;
  case 5:
    day = "Friday";
    break;
  case 6:
    day = "Saturday";
}
```

JavaScript Loops

Loops are handy, if you want to run the same code over and over again, each time with a different value. Often this is the case when working with arrays:

- `for` - loops through a block of code a number of times
- `for/in` - loops through the properties of an object
- `for/of` - loops through the values of an iterable object
- `while` - loops through a block of code while a specified condition is true
- `do/while` - also loops through a block of code while a specified condition is true

The syntax of these loops are

```
for (expression 1; expression 2; expression 3) {
  // code block to be executed
}
```

```
for (key in object) {
  // code block to be executed
}
```

```
for (variable of iterable) {
  // code block to be executed
}

while (condition) {
  // code block to be executed
}
```

JavaScript Functions

A JavaScript function is a block of code designed to perform a particular task. A JavaScript function is executed when "something" invokes it (calls it).

```
// Function to compute the product of p1 and p2
function myFunction(p1, p2) {
  return p1 * p2;
}
```

JavaScript Function Syntax

A JavaScript function is defined with the function keyword, followed by a name, followed by parentheses (). Function names can contain letters, digits, underscores, and dollar signs (same rules as variables). The parentheses may include parameter names separated by commas:

(parameter1, parameter2, ...)

The code to be executed, by the function, is placed inside curly brackets: {}

Function Return

When JavaScript reaches a return statement, the function will stop executing. If the function was invoked from a statement, JavaScript will "return" to execute the code after the invoking statement.

Functions often compute a **return value**. The return value is "returned" back to the "caller":

```
let x = myFunction(4, 3);   // Function is called, return value will end
up in x

function myFunction(a, b) {
  return a * b;             // Function returns the product of a and b
```

```
}
```

DOM:

With the HTML DOM, JavaScript can access and change all the elements of an HTML document. The HTML DOM (Document Object Model, When a web page is loaded, the browser creates a Document Object Model of the page. The HTML DOM model is constructed as a tree of Objects:

With the object model, JavaScript gets all the power it needs to create dynamic HTML:

- JavaScript can change all the HTML elements in the page
- JavaScript can change all the HTML attributes in the page
- JavaScript can change all the CSS styles in the page
- JavaScript can remove existing HTML elements and attributes
- JavaScript can add new HTML elements and attributes
- JavaScript can react to all existing HTML events in the page
- JavaScript can create new HTML events in the page

Code Examples:

```
<div id="a"></div>
<div class="c"></div>
<h1> </h1>

<p class="intro">Hello World!.</p>
<h4 id="demo"> </h4>
<script>
document.getElementById("a").innerHTML = "Its is just DOM by ID";
document.getElementsByClassName("c").innerHTML = "Its is just DOM by Class";
function Dom()
{
var res = document.getElementsByName("naam");
alert("Tag value is "+res.length);
}
document.getElementsByTagName("h1").innerHTML = "Its is just DOM by ID";

const x = document.querySelectorAll("p.intro");
document.getElementById("demo").innerHTML =
'The first paragraph (index 0) with class="intro" is: ' + x[0].innerHTML;

</script>
<h3 type="text" name="naam">This is Tag Name</h3>
<h3 type="text" name="naam">This is Tag Name</h3>
```

<input type="button" onclick="Dom()" value="Click here" />

JavaScript Events

HTML events are "things" that happen to HTML elements. When JavaScript is used in HTML pages, JavaScript can "react" on these events.

HTML Events
An HTML event can be something the browser does, or something a user does. Here are some examples of HTML events. An HTML web page has finished loading. An HTML input field was changed. An HTML button was clicked Often, when events happen, you may want to do something. JavaScript lets you execute code when events are detected. HTML allows event handler attributes, with JavaScript code, to be added to HTML elements. With single quotes:
<element event='some JavaScript'>
<element event="some JavaScript">
<button onclick="document.getElementById('demo').innerHTML = Date()">The time is?</button>

Event Validation Example:

```
<html>
<script>
function calValue() {
//fetch all gender radio button data
var male = document.getElementById('g1');
var female = document.getElementById('g2');
var otherg = document.getElementById('g3');

//fetch all language radio button data
var hindi = document.getElementById('l1');
var english = document.getElementById('l2');
var otherl = document.getElementById('l3');

var gender;
var language;
//check which gender is selected using radio button
```

```
if(male.checked == true) {
  gender = male;
} else if(female.checked == true) {
  gender = female;
} else if(otherg.checked == true) {
  gender = otherg
}

//check which language is selected using radio button
if(hindi.checked == true) {
  language = hindi;
} else if(english.checked == true) {
  language = english;
} else if(otherl.checked == true) {
  language = otherl
}
//return data to HTML form
return document.getElementById("result").innerHTML = "Your selected gender is: " +
gender.value + "</br> and </br> Selected language is: " + language.value;
}



function validateemail()
{
var x=document.myform.email.value;
var atposition=x.indexOf("@");
var dotposition=x.lastIndexOf(".");
if (atposition<1 || dotposition<atposition+2 || dotposition+2>=x.length){
  alert("Please enter a valid e-mail address \n atpostion:"+atposition+"\n
dotposition:"+dotposition);
  return false;
  }
}
</script>

<body>
<h2> Simple radio Buttons Example </h2>
  <!-- create radio button for gender selection -->
  <p> <b> Select your gender: </b> </p>
  <input type="radio" id="g1" name="gender" value="male">
```

```
  <label for="male"> Male </label> <br>
  <input type="radio" id="g2" name="gender" value="female">
  <label for="female"> Female </label> <br>
  <input type="radio" id="g3" name="gender" value="other">
  <label for="other"> Other </label>

  <br>
  <!-- create radio button for language selection -->
  <p> <b> Select your language: </b> </p>
  <input type ="radio" id="l1" name="language" value="hindi">
  <label for ="male"> Hindi </label> <br>
  <input type="radio" id="l2" name="language" value="english">
  <label for="female"> English </label> <br>
  <input type="radio" id="l3" name="language" value="other">
  <label for="other"> Other </label> <br> <br>

  <input type="submit" value="Submit" onclick="calValue()">
  <h3 id="result" style="color:blue"> </h3>


  <br/><br/><br/><br/><br/>
  <form name="myform"  method="post" action="#" onsubmit="return validateemail();">
    Email: <input type="text" name="email"><br/>

    <input type="submit" value="register">
</body>
</html>
```

Task

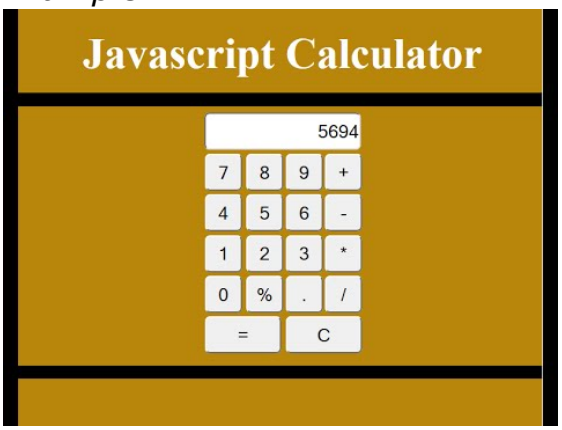
Task 01: Write a function in which all types of Data types of JS will be used using LET, VAR, CONST and NOTHING variable initialization.

Task 02: Write a function in JS which will reverse the String in ascending order.

Task 03: Write a JavaScript program to print the table of entered number in the web browser.

Task 04: Create a JavaScript program to display the simple Calculator which will perform basic arithmetic operations.

Example:



Task 05: Write a function to check if an input string is a palindrome and display the result of web page.

Task 06: Create a Sign up page and display the entered value after the submission of form on the webpage using DOM.

Task 07: Validation is one of the important concept in the web page, create a complete Sing up page having name, contact, email password, radio and checkbox button on it and validate using JS regular expression individually according to the user control.

Task 08: Using JS create an Image Slider that will move image automatically after some delay, you have to create your own logic to do so.