

Lab Guide for JavaScript (JS)

Contents

Background	1
JavaScript Assignments.....	1
Assignment 1: Understanding Client Side JavaScript	1
Assignment 2: Using Variables and Operators.....	4
Assignment 3: Using Control Structures and Loops	6
Assignment 4: Functions and Dialog Boxes	9
Assignment 5: External File and Built in Objects.....	12
Assignment 6:Working with DOM and Important objects	15
Assignment 7: Working with Form and Elements.....	19
Assignment 8: Reusable functions	25

Background

This document contains the assignments to be completed as part of the hands on for the subject JavaScript



Note: All assignments in this document must be completed in the sequence in this document in order to complete the course.

JavaScript Assignments

All the assignments in this section must be completed on Day 2&3 of your HJSX course.

Assignment 1: Understanding Client Side JavaScript

Objective: To understand the usage of Client Side JavaScript.

Background: You have learned HTML and Cascading Style Sheet (CSS). Using HTML and CSS you can create only static pages. Now you can create interactive pages using client side JavaScript.



Note: JavaScript is embedded as a small program in a web page that is interpreted and executed by the Web client. JavaScript is a scripting language - a scripting language is different from programming language. JavaScript is an interpreted language (means that scripts execute without preliminary compilation). JavaScript is supported by all major browsers like Netscape, Internet Explorer, Mozilla etc.

Estimated time: 10 minutes

Step 1: Create a folder named “assign1” under your working directory

Step 2: Type the following code into a text editor and save it as “firstjs.html”

```
<html>
<head>
<!-- Comments : To include a client side script use <script> tag -->
<script language="JavaScript">
<!--
    document.write("Displaying using JavaScript");
//-->
</script>
</head>
<body>
    <h1> First JavaScript Program</h1>
</body>
</html>
```



Note: To hide the script from the browsers, which are not supporting JavaScript use HTML comments (`<!-- , //-->`). `//` is the JavaScript single line comment.

Step 3: Modify the “firstjs.html” remove the language attribute of the `<script>` tag. And execute the program.

Eg. `<script>`



Note: If you are getting the proper output which is indicating that, the default scripting language for your browser is JavaScript. But it is good practice that always use the language attribute.

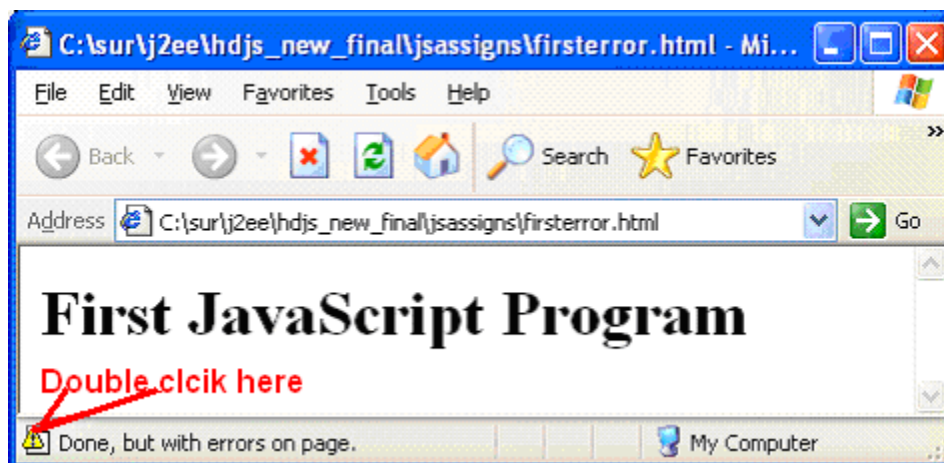
Step 4: Modify the “firstjs.html” and change the `<script>` tag like the following

```
<script language="Java Script">
```

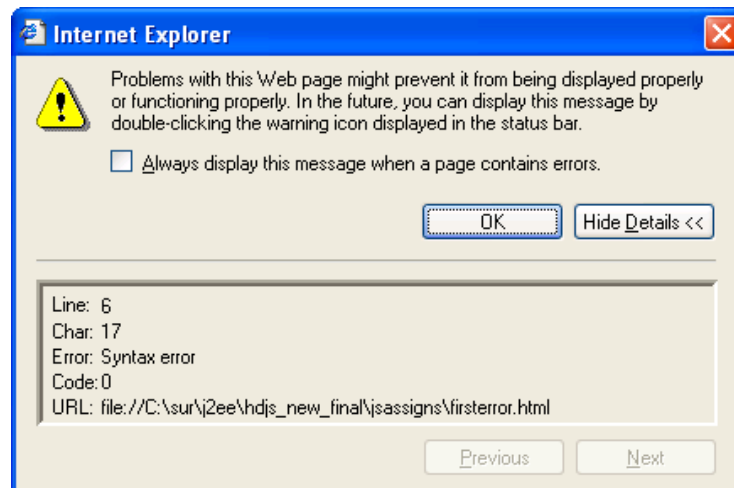
The script will not execute because the browser does not understand the script language. Note down the space between Java and Script and remove it.

Step 5: Modify the “firstjs.html” like the following and save as “firsterror.html”. Open the page in the browser.

```
document.write(<h1>"Displaying using JavaScript"</h1>);
```



Step 6: The page is displayed with errors. To find out the error in JavaScript double-click on the icon on the status bar. A small Dialog window will display the error messages with the line number. (if the error message is not displaying properly select the address(URL) from the address bar (**the URL should be selected like for copying**) and press enter, check the status bar).



Step 7: The Dialog box is showing that the error is at line number 6 and the message is “Syntax error”. Let us examine the code at line number 6

```
document.write(<h1>"Displaying using JavaScript"</h1>)
```

We can find that the syntax is wrong. The correct syntax is

```
document.write("<h1>Displaying using JavaScript</h1>")
```

Step8: Correct the error and save the file as “errorfree.html”

Summary of this exercise:

You have just learnt

- How to use Javascript in HTML documents.
- Debug and fix the error.

Deliverables of the exercise:

- 1) firstjs.html
- 2) firsterror.html
- 3) errorfree.html

Assignment 2: Using Variables and Operators.

Objective: To work with JavaScript we need to know the different types of variables and operators.

Estimated time: 15 minutes

Step 1: Type the following code in an editor and save it as “varjs.html”.

```
<html>
<head>
<script language="JavaScript">
/* Comments : Variables declaration */
/* Comments : Inline scripting */

    var num1=10;
    var num2=20;
    var num3=num1+num2;
/* Comments : document.write is a method to write the contents on
the browser */
    document.write("The sum is "+num3);
</script>
</head>
<body>
    <h3> JavaScript variable demo</h3>
</body>
</html>
```



Note: Variable names are case-sensitive and they must begin with a letter or the underscore character.

Step 2: Modify the “varjs.html”, place the entire Javascript part inside the <body> tag and save it as “varjsbody.html”. Open the page in the browser.

Step 3: Remove the **var** keyword from “varjs.html” and save it as “remvar.html”. Open the page in the browser and check the output.



Note: var is used to declare a variable. In JavaScript you can use variables without declaring it. If a variable is prefixed by the keyword “var” within a function then it is a local variable, otherwise it is a global variable.

Step 4: Modify the “remvar.html”, change the variable name “num3” as “3num”. Save it as “varerror.html” Verify the output.

Step 5: Debug the error in “varerror.html” and correct it.

Step 6: Include the following code inside <head> and save it as “assign.html”.

```
<script language="JavaScript">
    inum=55;
    msg="4";
    sum=msg+inum;

    alert(sum);
/* Comments : The above statement displays a message dialog box */
</script>
```

Step 7: Write a program to find out $10.5 \% 2.5$, Save it as “mod.html”, what is the output?

Step 8: Include the following code inside <head> and save it as “unary.html”.

```
<script language="JavaScript">
    num1 = 10; num2 = 10;
    num3 = num1++;
    num4 = ++num2;
</script>
```

Display the values of num1, num2, num3 and num4.

Step 9: Find out what is the error in the following code, save it as “varerror2.html”

```
<script language="JavaScript">
    num1=100;
    Num1+=20;
</script>
```

Debug the error and display the value of “num1”.

Step 10: Find out what is the value of num1 in the following code and save it as “local.html”.

```
<script language="JavaScript">
/* Comments : Defining a function by name „fun” */

    function fun()
    {
/* Comments : Declaring a local variable „num1” */

        var num1=100;
    }
    fun()
/* Comments : calling function */
    document.write("the value of num1 is "+num1);
</script>
```

Step 11: Getting Error? Why? Debug it. (Refer the note in page-no 4)

Step 12: Modify the “local.html”, remove the “var” keyword and execute it.

Step 13: Open javascript1.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying only one alert box and displaying an error message. Debug the code and correct it.

Summary of this exercise:

You have just learnt

Declare and how to use variables.

Different operators in JavaScript

Difference between local and global variables.

Deliverables of the exercise:

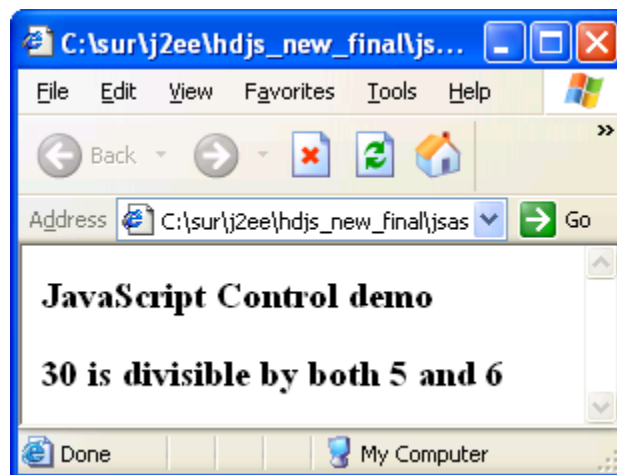
- 1) varjs.html
- 2) varjsbody.html
- 3) remvar.html
- 4) varerror.html
- 5) assign.html
- 6) mod.html
- 7) unary.html
- 8) varerror2.html
- 9) local.html

Assignment 3: Using Control Structures and Loops

Objective: Any logic can be created using control structures and loops. Here we are learning about the usage of control structures and loops.

Estimated time: 20 minutes

Step 1: Write a program to find whether a given number is divisible by 5 and 6 And save it as “control1.html”



Step 2: Write a program to find the largest of three numbers using if-else control structure and save it as “control2.html”.

Step 3: Write a program to find, whether the trainee has cleared the “Generic Comprehensive” exam or not with the following condition. To pass the exam the total mark should be greater than 65% and Sec A, Sec B, Sec C mark should be greater than 50%. Save the file as “control3.html”.


```
le,          Total >= 65
             /100 Sec A >= 25
             / 50
Sec B  >= 15 / 30
Sec C  >= 10 / 20
```

Step 4: Find out what is the output of the following code. Save the file as “control4.html”

```
<script language="JavaScript">
    var num=100;
    var str="100";
    /* Comments : Equality operator used for comparison */

    if(num==str){
        document.write("<h1> Equal</h1>");
    }
    else{
        document.write("<h1> Not Equal</h1>");
    }
</script>
```

Step 5: Modify the above code, replace “==” with “===”. What will be the output?

Step 6: Write a program to find whether the given character is vowel or not. Use Switch case control structure. Save the file as “switch.html”

Remember that JavaScript does not support character data type. Use String data type for doing the same. Eg letter="A".

Step 7: Write a program to display numbers from 1 to 10 using for loop in a table format and save it as “for1.html”.



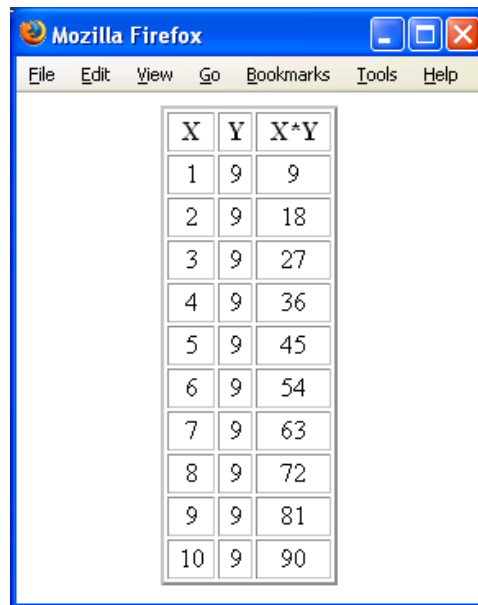
Note: When you are using attribute values inside the write method, it is better to use inside single codes.

```
document.write("<table border='1'>");
```

Otherwise use escape character \" like

```
document.write("<table border='\"1\"'>");
```

Step 8: Write a program to display the multiplication table of the given number using a loop. Display the output in a table format. Save the file as “multtable.html”.

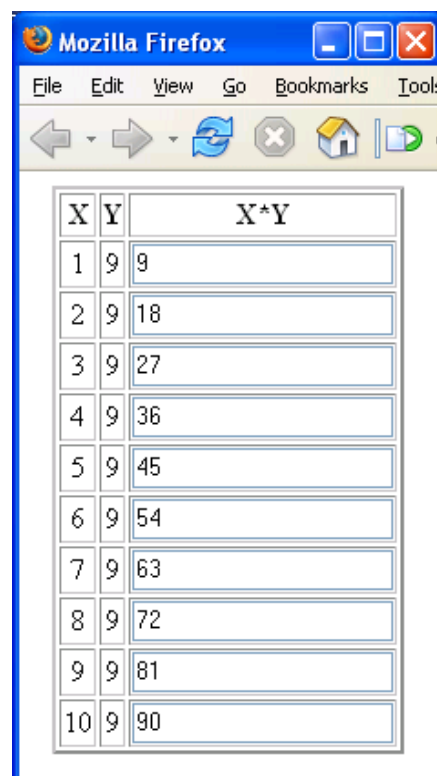


A screenshot of the Mozilla Firefox browser window. The address bar is empty. The main content area displays a static table with three columns: 'X', 'Y', and 'X*Y'. The 'Y' column contains the value '9' for all rows. The 'X' column contains values from 1 to 10. The 'X*Y' column contains the corresponding multiplication results from 9 to 90.

X	Y	X*Y
1	9	9
2	9	18
3	9	27
4	9	36
5	9	45
6	9	54
7	9	63
8	9	72
9	9	81
10	9	90

Step 9: Modify the above program and display the result inside a textbox. Save the file as “multtabtext.html”. Verify the output.

Place the <table> tag inside a <form> tag and use “text” input type. The value of the input type should be assigned dynamically.



A screenshot of the Mozilla Firefox browser window. The address bar shows a file path. The main content area displays a table with three columns: 'X', 'Y', and 'X*Y'. The 'Y' column contains the value '9' for all rows. The 'X' column contains values from 1 to 10. The 'X*Y' column contains input fields with the corresponding multiplication results (9, 18, 27, 36, 45, 54, 63, 72, 81, 90) entered into them.

X	Y	X*Y
1	9	9
2	9	18
3	9	27
4	9	36
5	9	45
6	9	54
7	9	63
8	9	72
9	9	81
10	9	90

Step 10: Open javascript2.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying name instead of printing its value “lmcs”. Debug the code and correct it.

Step 11: Open javascript3.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying a syntax error. Debug the code and correct it.

Summary of this exercise:

You have just learnt

- Using control structure if-else and switch
- Using for loop, while and do-while loop.
- Dynamic table creation.

Deliverables of the exercise:

- 1) control1.html
- 2) control2.html
- 3) control3.html
- 4) control4.html
- 5) switch.html
- 6) for1.html
- 7) multtable.html
- 8) multtabtext.html

Assignment 4: Functions and Dialog Boxes

Objective: Functions are one of the fundamental building blocks in JavaScript. A function is a JavaScript procedure—a set of statements that performs a specific task. Here we will learn how to create user defined functions and make use of in built functions.

Estimated time: 15 minutes

Step 1: Type the following code in to a text editor and save it as “fun1.html”.

```
<script language="JavaScript">
/* Comments : function definition */
function addFun(num1,num2)
{
    document.write("the sum is "+ (num1+num2));
}
/* Comments: Calling function */
addFun(10,20);
</script>
```

Step 2: Modify the above program, add a function for multiplying two numbers.



Note: The main advantage of a function is reusability. So instead of displaying the output inside the function return the result to the caller using the **return** keyword.

Step 3: Type the following code into a text editor and save it as “funreturn.html”.

```
<html><head>
  <script language="JavaScript">
    function addFun(num1,num2)
    {
      /* Comments : Returning value to the calling environment */

      return num1+num2;
    }
  </script>
</head><body>
  <h1> Function Demo</h1>
  /* Comments : More than one script tag can be used in a document */
  <script language="Javascript">
    document.write("<h1>Sum is "+addFun(20,50)+"</h1>");
  </script>
</body></html>
```

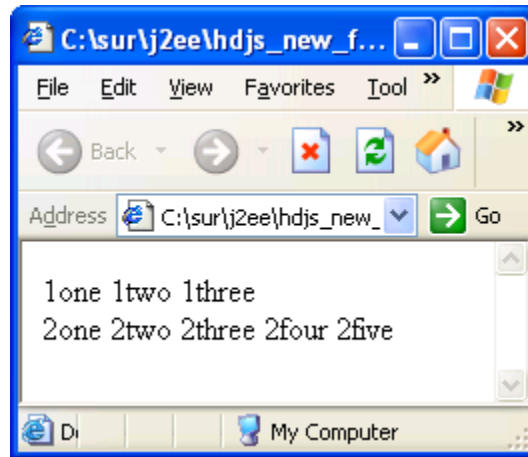
Step 4: Write a function called `power(arg1, arg2)` to find the power of the given number and save it as “power.html”.



Note: The arguments of a function are maintained in an array. Within a function, you can address the parameters passed to it as follows: `arguments[i]`. The total number of arguments is indicated by `arguments.length`.

Step 5: Type the following code and save it as “funarg.html”.

```
<html><head>
<script language="JavaScript">
  /* Comments : function does not have any arguments, but arguments
  array will store the arguments */
  function dispStr()
  {
    for(i=0;i<arguments.length;i++)
    {
      document.write(arguments[i]+" ");
    }
  }
  /* Comments : Calling function with three arguments */
  dispStr("1one", "1two", "1three");
  document.write("<br>");
  /* Comments : Calling function with five arguments */
  dispStr("2one", "2two", "2three", "2four", "2five");
</script>
</head><body></body></html>
```



Step 6: Write a function which takes n number of arguments and return the average of n numbers and save it as “numarg.html”



Note: There are three different types of dialog boxes (alert, prompt and confirm) which are the methods of “window” object. All these three dialog boxes are modal i.e. The user must close it before continuing.



Note: Event handlers are very powerful and useful in client side scripting. They are JavaScript code that are not added inside the <script> tags, but rather, inside the html tags, that execute JavaScript when an event happens, such as pressing a button, moving your mouse over a link, submitting a form ,etc. For Example

```
<input type="button" value="Close" name="but2"
onClick="closeWin()">
```

Step 7: type the following code and save it as “dialog.html”.

```
<html><head>
<script language="JavaScript">
function dispMsg()
{
  /* Comments : Using prompt dialog box data can be accepted through
  data entry field from the user */

  num1=parseInt(prompt("enter the first  number","0"));

  /* Comments : parseInt method used to convert string to a numeric
  value */

  num2=parseInt(prompt("enter the second  number"));
  alert("Sum is \n"+(num1+num2));
}
function closeWin()
{
```

```
/* Comments : confirm dialog box used to get user confirmation about
closing the window or not */

    res=confirm("do you want to close the window?");
    if(res==true) window.close();
}
</script>
</head><body>
<form name="f1">
/* Comments : Event Onclick used with „Add“ button . Hence on
clicking the „Add“ button „dispMsg()“ function is invoked */

<input type="button" value="Add" name="but1" onClick="dispMsg()">
<input type="button" value="Close" name="but2" onClick="closeWin()">
</form>
</body></html>
```

Step 8: Open javascript4.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying a syntax error. Debug the code and correct it.

Step 9: Open javascript5.html using Textpad. Predict the output and verify the same by running the .html file (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG).

Summary of this exercise:

You have just learnt

- Create and invoke user defined functions.
- The arguments array and its usage
- Use Dialog boxes.
- Use onClik event handler.

Deliverables of the exercise:

- 1) fun1.html
- 2) funreturn.html
- 3) power.html
- 4) funarg.html
- 5) numarg.html
- 6) dilog.html

Assignment 5: External File and Built in Objects

Objective: To understand the usage of External JavaScript file (.js) and learn the usage of some important built in objects.

Estimated time: 25 minutes

Step 1: Type the following code in to a text editor and save it as “ext.js”.

```
/* Comments : no need to use script tag here */
function extMsg()
{
```

```
        alert("Message from ext.js");  
    }
```

Step 2: Type the following code into a text editor and save it as “ext.html”

```
<html><head>  
  <!-- Comments : Using external javascript file -->  
  <script language="JavaScript" src="ext.js"></script>  
  <!-- Comments : Normal mistake, forget to close the script tag -->  
</head><body>  
  <input type="button" value=" Add " name="but1"  
    onClick="extMsg()" "><br>  
</form>  
</body></html>
```

Step 3: Write a program to display the current date, month and year and save it as “date1.html”. (display in a format “dd-mm-yyyy”)
Use date object and its methods.

```
d=new Date();  
/* Date object created with current date */  
year=d.getFullYear();  
/* getYear() returns the current year */
```

Step 4: Modify the above program and display the date in the following format.
“dd-mon-yyyy”.

Step 5: Write a program to find the length of the following string and save it as “strllen.html”.

```
str = "Krishna Java Training". Use the length property of the string object.  
len=str.length
```

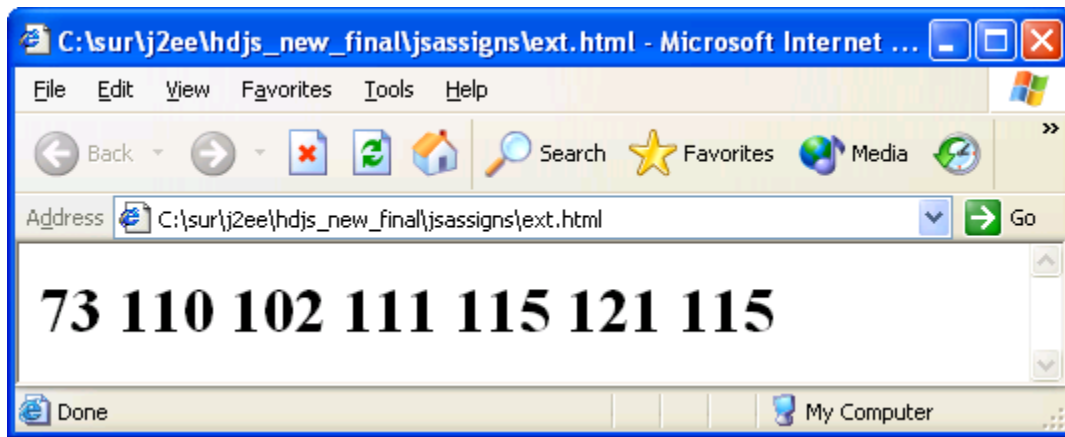
Step 6: Write a program to display the ASCII value of the following string. Save the file as “strascii.html”

```
str="Imcs".
```

Use the `charCodeAt()` function of the string object.

`str.charCodeAt(0)` will return the ASCII value of the first character “I”.

Use a loop for displaying the entire string.



Note: In Javascript String is an object not an array of characters, so if you want to access individual characters you need to use the methods associated with the string object.

Step 7: Write a program to find the occurrence of “@” and “.” in the following email address. Make sure that there is only one occurrence of “@” character. Save the file as “email.html”

email=“ jack@lmcs.com” Use the following String methods

- 1) indexOf(searchValue[, fromIndex])
- 2) lastIndexOf(searchValue[, fromIndex])

Eg. Here num= email.indexOf(“@”) , the num value is “4”.

Step 8: Write the program to find the length of the following Array. Save the file as “array.html”.

```
var arr =new Array(1,2,55,66,33,22,1);
```

Use the length property of the Array.

Step 9: Modify the above program and display individual elements.

Step 10: Modify the above program and reverse the “Array”. Use the reverse() method of the Array object. Save the file as “arrayrev.html”.

Step 11: Modify the above program and find the biggest and smallest elements in the Array and save it as “bigsmall.html”.

Step 12: Divide the following sentence into words and display in the ascending order of words. Save the file as “split.html”.

```
sent="javascript is very easy to learn".
```

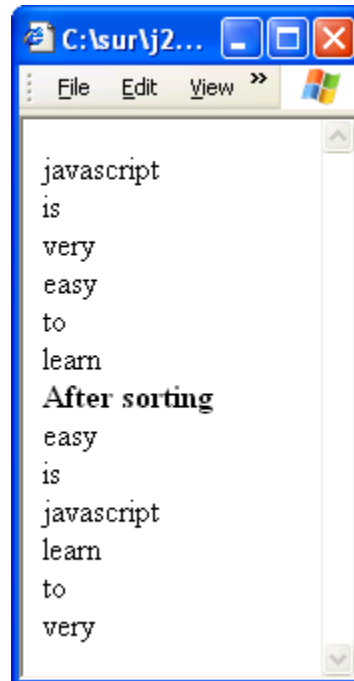
You can use the split() method of the String object to split the sentence to an Array of Strings.

```
wordarr=sent.split(" ");
```

/ splitting the sentence stored in „sent“ upon each space */*

wordarr is become an array of string.

Use the sort() method of the array for sorting the string



Step 13: Open javascript6.html using browser.(supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). Debug and fix the errors.

Summary of this exercise:

You have just learnt

Using external Javascript file

Working with Date, Math, String and Array Object.

Deliverables of the exercise:

- 1) ext.js
- 2) ext.html
- 3) date.html
- 4) math.html
- 5) math1.html
- 6) strlen.html
- 7) strascii.html
- 8) email.html
- 9) array.html
- 10) bigsmall.html
- 11) split.html

Assignment 6: Working with DOM and Important objects

Objective: DOM defines the document object hierarchy, here we are learning about the usage of different objects and its usage.

Estimated time: 20 minutes

Step 1: Type the following code and save it as “navigator.html”.

```
<script language="JavaScript">
    document.write("the browser name is "+navigator.appName);
```

```
</script>
```



Note: “navigator” object contains information about the browser and it’s version.

Step 2: Modify the above program and find the operating system of the system and version of the browser.

Use the `appVersion` and `platform` properties of the navigator object.

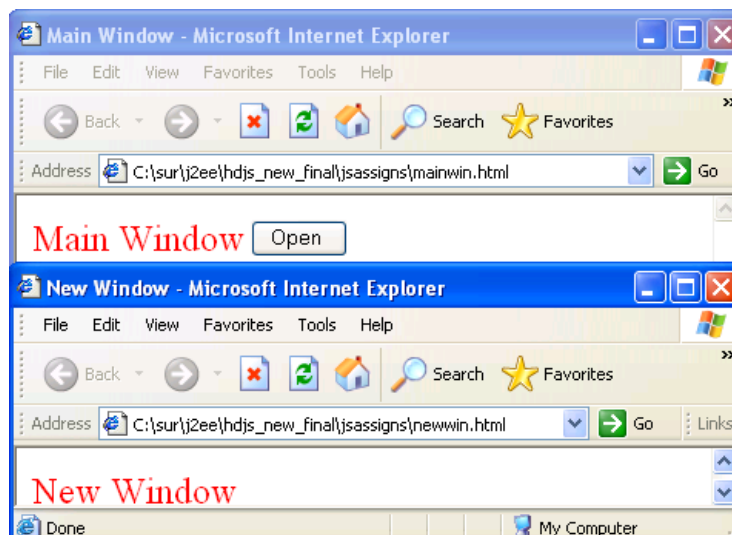
Step 3: Type the following code and save it as “newwin.html”.

```
<html>
<head><title> New Window</title></head>
</body>
    <font color="red" size="5"> New Window</font>
</body>
</html>
```

Step 4: Type the following code and save it as “mainwin.html”.

```
<html>
<head><title> Main Window</title>
<script language="JavaScript">
function openNew() {
/* Comments : To open a file „newwin.html” in a window by name
„newone” */

    newwin=window.open("newwin.html","newone");
}
</script>
</head><body>
    <font color="red" size="5"> Main Window</font>
    <input type="button" value=" Open  " onClick="openNew()">
</body></html>
```





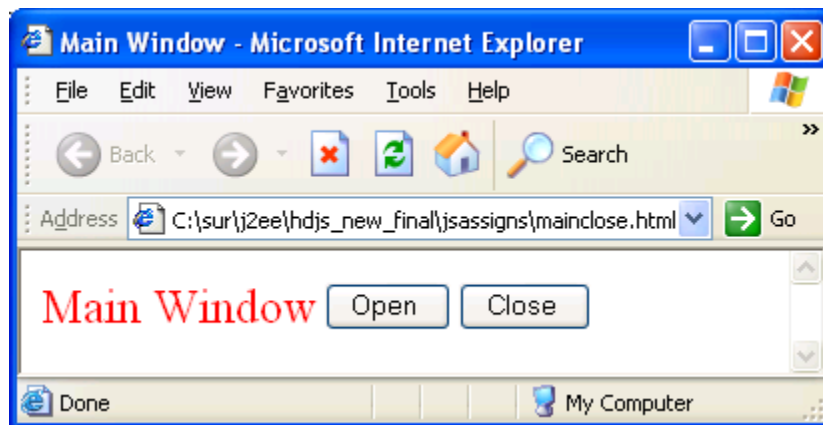
Note: Consider `newwin=window.open("newwin.html","newone")`, “newwin” is the object reference to the new window, “newwin.html” is the html file you want to open and “newone” is the name of the new window.

Step 5: Modify the above program, replace the function with the following code.

```
function openNew(){
    newwin=window.open("newwin.html","newone","width=300;
    height=400;toolbar=no");
}
```

Step 6: Save the “mainwin.html” as “mainclose.html”. And modify the following Insert a button with a value “close” and onClick of the button call the following function.

```
function closeWin()
{
    /* Comments: newwin is the reference of the new window */
    newwin.close();
}
```



Step 7: Type the following code and save it as “status.html”

```
<script language="JavaScript">
    /*Comments: set a string in the browser status bar */
    window.status="Check the status bar";
    /*Comments: Status is a property of window */
</script>
```



Note: If the status bar is not visible, select menu option, view->status bar.

Step 8: Type the following code and save it as “history.html”.

```
<html><body>
<h1>History demo</h1>
<!-- Commnets : Using Inline JavaScript -->
```

```
<input type="button" value="back" onClick="history.go(-1)">
</body></html>
```



Note: If you want work with history object more than one page should be loaded in to the browser.

- a) Open “status.html”
- b) In the same browser open “history.html”

Step 9: Type the following code and save it as “location.html”.

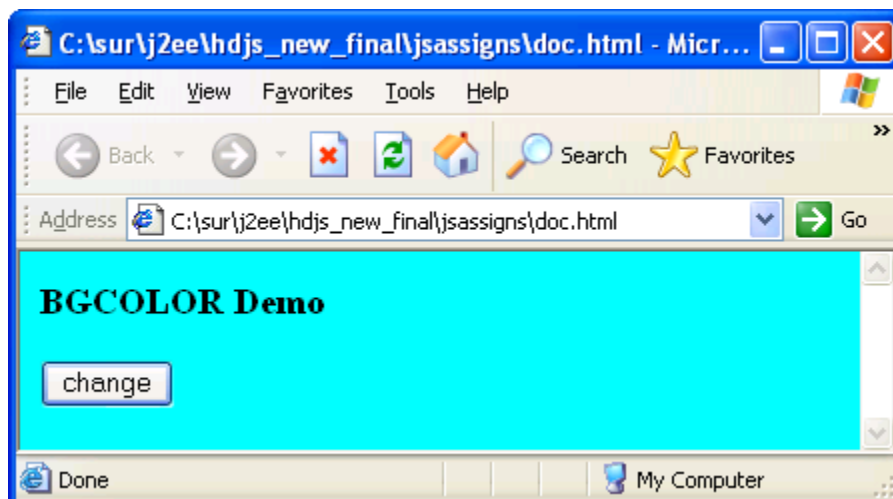
```
<html><head>
<script language="JavaScript">
    function fun() {
        /*Comments: href property of Location object used to specify the complete URL */

        location.href="status.html";
    }
</script>
</head><body>
    <h3> Location Demo</h3>
    <input type="button" value="change" onClick="fun()">
</body></html>
```

Step 10: Modify the above program, use location.replace(URL) method for replacing the page. Display the hostname and protocol. (Use protocol and hostname properties)

Step 11: Type the following code and save it as “doc.html”

```
<body>
<h3> BGCOLOR Demo</h3>
<input type="button" value="change"
        nClick="document.bgColor='cyan';">
</body>
```



Summary of this exercise:

You have just learnt

How to use the following objects.

navigator
window
document
history
location

Deliverables of the exercise:

- 1) navigator.html
- 2) newwin.html
- 3) mainwin.html
- 4) mainclose.html
- 5) status.html
- 6) history.html
- 7) location.html
- 8) doc.html

Assignment 7: Working with Form and Elements

Objective: One of the Web's most interesting possibilities is the creation of interactive features. You can interact to the website using form elements. Here we are going to learn about how to access the form elements.



Note: In “DOM” you can access each element in a form using the following hierarchy.
window -> document -> form -> element.

Estimated time: 35 minutes

Step 1: Type the following code and save it as “actionpage.html”.

```
<html><body>  
  <h3> Action page demo</h3>  
</body></html>
```

Step 2: Type the following code and save it as “mainform.html”. Open the page in a browser, click on the submit button verify the URL.

```
<html>  
<body>  
<h1> Form Demo</h1>  
  /*Comments: Creating a form with name „frm1” */  
  
  <form name="frm1" action="actionpage.html" method="get">  
    <input type="text" name="txt1" value="Imcs"><br>  
    /*Comments: onClick of the following submit button the action attribute of form  
reflects the functionality to be carried out further */  
  
    <input type="submit" value="submit">  
</form></body>
```

```
</html>
```

Step 3: Modify the above program, set the method attribute value as “post”. Open the page in a browser, click on the submit button verify the URL.

```
<form name="frm1" action="actionpage.html" method="post">
```

Step 4: Modify the above program, change the “submit” to “button” and save the file as “mainform1.html”. Click on the button. What is the Output?

```
<input type="button" value="submit">
```



Note: Button can not submit the form to the server unless and until you incorporate client side scripting. Use the submit method of the form for doing the same.

Modify the above program using the following code.

```
<input type="button" value="Submit"
onClick="document.frm1.submit()">
```



Note: Submission of the form is required only for the server side programs. We are dealing with client side programs. So at the point of time we are not dealing with the submission of the form.

Step 5: Type the following code and save it as “formelement.html”

```
<html><head>
<script language="JavaScript">
/*Comments:  only three elements in the form, to access that we can
use elements array */
function funDisp()
{
    alert("textbox val is "+document.forms[0].elements[0].value);
    alert("textbox val is "+document.forms[0].elements[1].value);
    alert("textbox val is "+document.forms[0].elements[2].value);
}
</script></head><body>
<form name="frm1">
    <input type="text" name="txt1" value="Imcs"><br>
    <input type="radio" name="rad1" value="1" checked>Ok<br>
    <input type="button" value="Submit" name="but1"
onClick="funDisp()">
</form></body></html>
```

Step 6: Modify the above program, replace the function with the following function. Save the file as “formnames.html”.

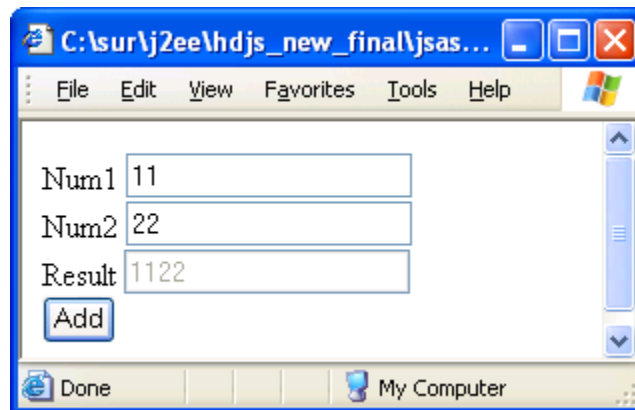
```
/* If we know the name of the form and elements, we can use it
directly, "frm1" is the name of the form and "txt1" is the name of
the textbox */
function funDisp()
{
    alert("textbox val is "+document.frm1.txt1.value);
```

```

    alert("textbox val is "+document.frm1.rad1.value);
    alert("textbox val is "+document.frm1.but1.value);
}

```

Step 7: Write a program to accept two numbers from two text boxes and on click of Add button the resultant value should be placed in the third text box as shown below and save it as “addnum.html”.



Step 8: The sum of “11” and “22” is giving as “1122”. Who is responsible for this mistake? Yes it is our mistake, here two strings are getting concatenated and the result is a string. So before adding the numbers convert them to a number.



Note: Any value from a form element will treat as a String, so before doing mathematical calculation convert them to numbers.

Modify the above program using the parseInt method.

Step 9: Save the above file as “simplecalc.html”. Add three more buttons and functionality for subtracting, multiplying and dividing two numbers.

Step 10: If the user enters only numbers it works fine, but if the user enters a string then the result will be “NaN”. How can we avoid this. Why can’t we check the validity for the data entered by the user. i.e check if the input is correct or not.

We can use the Top Level function “isFinite()” and “isNaN()” for checking if the given string is a number or not.

Consider the following code for checking the first textbox value is a number or not.

*/*Comments : Not isFinite() will return true if num1 is a string */*

```

if(!isFinite(num1))
{
    alert("Please enter a number!");
    return false;
}

```

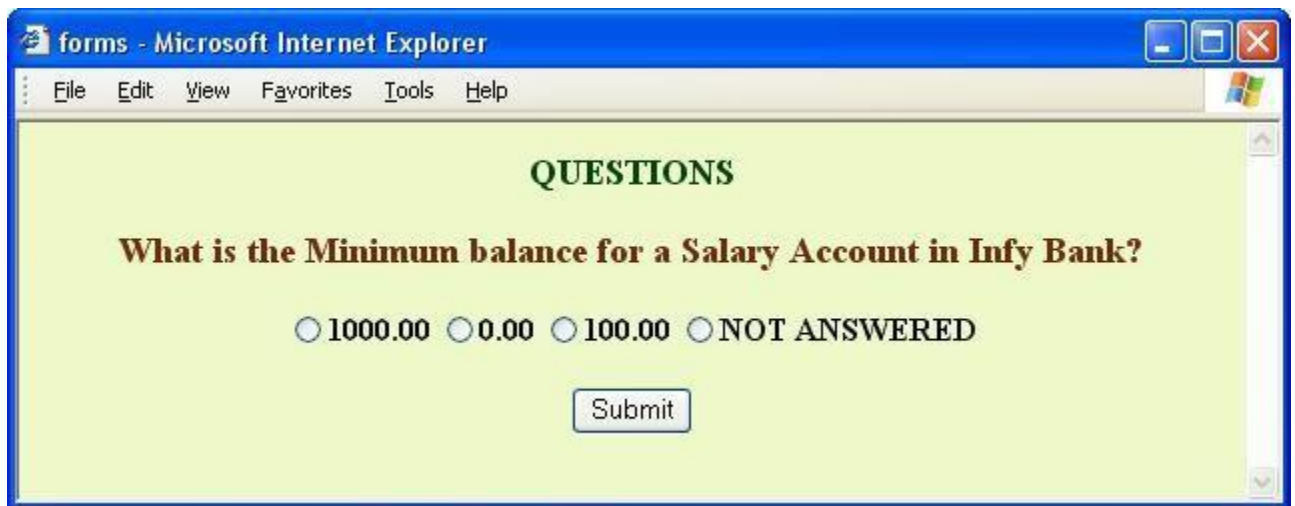
Modify the “simplecalc.html” save as “simplecalcval.html”.

Step 11: Create a form with a username field and a password field (A textbox and password type) and save it as “user.html”. Make sure that user is entering a minimum 8 characters and a maximum of 15 characters.

Use maxlength=15 attribute of <input> for checking the maximum length.

Step 12: Display a multiple choice question with four choices (Use a radio group with the name “rad”). When the user chooses the answer and click on the button it should display the answer is correct or not. Save the file as “apti.html”.

All radio buttons should have the same name(“rad”), so use array of objects and access individual elements using rad[0], rad[1], rad[2], and rad[3]. Use the “checked” property of the radio button to check the user is selected the radio or not.



Step 13: Type the following code and save it as “select.html”.

```
<html><head>
<script language="JavaScript">
function disp()
{
    ind = document.f1.webs.selectedIndex;
    /*Comments : index of the selected Item */
    alert("the value is  \n"+document.f1.webs.options[ind].value);
    alert("the text is  \n"+document.f1.webs.options[ind].text);
}
</script>
</head><body>
<form name="f1">
<select name="webs" onchange="disp()">
    <option value="1"> http://sparsh
    <option value="2"> http://kshop
    <option value="3"> http://172.21.20.155/
    <option value="4">http://enr.ad.Imcs.com/
</select>
```



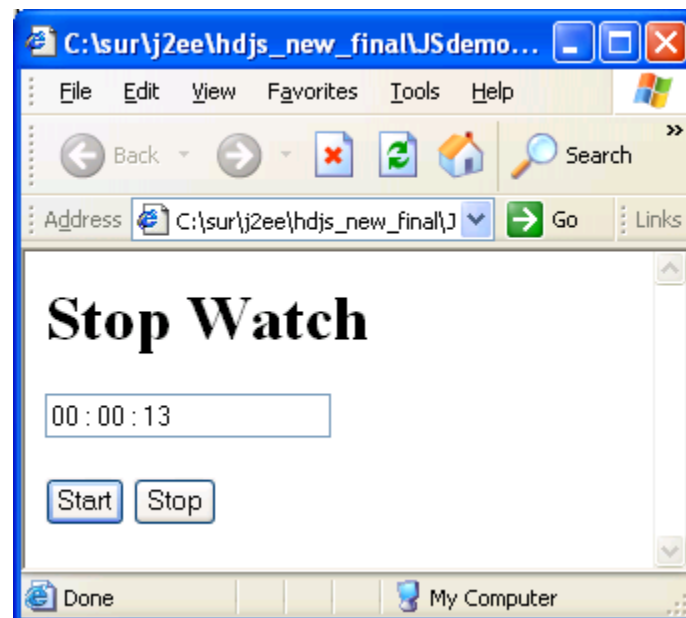
```
</form></body></html>
```

Step 14: Save the above program as “selectwebs.html”. When the user selects a particular website it should redirect to the particular site.

Step 15: Create a textarea for accepting feedback from the user, do not allow the user to enter more than 200 characters. Save the file as “textarea.html”.

Step 16: Write a program to display a stop watch. When user clicks on the start button it should start display the number of seconds and when user clicks on the stop button it should stop counting.

Use `id = setTimeout(funName, time in MS)` method of window object for calling a function recursively with a time delay. Use `clearTimeout(id)` for stop the function call. Save the file as “stopwatch.html”



Step 17: Create the following form, do the validations. Save the file as “val.html”.



- All the fields are mandatory.
- Name should be a string with minimum 3 Character.
- Date of birth should not contain alphabets.
- Email should be a string with and “@” and “.” Character.

Step 18: Open javascript7.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying a syntax error. Debug the code and correct it.

Step 19: Open javascript8.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). It is displaying a syntax error. Debug the code and correct it.

Step 20: Open javascript9.html in the browser (supplied with lab guide, check the folder HTML_CSS_JS_CODE_FOR_DEBUG). Click on the submit button without entering any values in to the textboxes. It is invoking the action page even if it is displaying an error message. Debug and fix the issue.

Summary of this exercise:

You have just learnt

Working with form and its properties.

Working with the following form elements

Text

Password

Textarea

Radio

Select

Form Validation

Deliverables of the exercise:

- 1) actionpage.html
- 2) mainform.html
- 3) mainform1.html

- 4) formelement.html
- 5) formnames.html
- 6) addnum.html
- 7) simplecalc.html
- 8) simplecalcval.html
- 9) user.html
- 10) apti.html
- 11) select.html
- 12) selectwebs.html
- 13) textarea.html
- 14) stopwatch.html
- 15) val.html.

Assignment 8: Reusable functions

Objective: To create reusable functions .This could be used for the validation of form elements.

Estimated time:35 minutes

Step 1: Create a text box for accepting Employee Number, a submit button by name „Submit“ and save it as “empvalidate.html”

Step 2: Write functions to validate the following.

1. To check whether Empno field is empty ,isEmpty() returns 0 if empty else 1.
2. To check whether Empno field has ONLY spaces , onlySpaces() returns 0 if all the characters are spaces else 1
3. To check whether Empno field has ONLY digits, onlyDigits() returns 0 if all the characters are digits else 1

Step 3: Write a function(validateEmpDetails()) to invoke the functions defined previously for the validation of Emp Number.

Step 3.1: Invoke function isEmpty(). If it returns 0 then display a suitable error message and return false from the function.

Step 3.2: If isEmpty() returns 1 then invoke the next function onlySpaces(). If the function returns 0 then display a suitable error message and return false from the function.

Step 3.3: if onlySpaces() returns 1 then invoke the next function onlyDigits(). If the function returns 0 then display a suitable error message and return false else return true from the function.

Step 4: if true is returned then action performed should be a message display “successful validation” being written in another html i.e display.html.

Hint: Use action attribute for the above

Use Onsubmit event handler as mentioned below.

onSubmit= "return validateEmpDetails()"

Step 5: Write a separate function to check whether the name has ONLY alphabets[onlyAlphabets()].

Step 6: Modify the above HTML file by adding another text field for accepting Employee Name.

Step 7: Append the function(validateEmpDetails()) with the following functionalities. Invoke the functions defined previously for the validation of EmpName.

Step 7.1: Invoke function isEmpty(). If it returns 0 then display a suitable error message and return false from the function.

Step 7.2: If isEmpty() returns 1 then invoke the next function onlySpaces(). If the function returns 0 then display a suitable error message and return false from the function.

Step 7.3: if onlySpaces() returns 1 then invoke the next function onlyAlphabets (). If the function returns 0 then display a suitable error message and return false else true from the function validateEmpDetails().

Step 8: The action performed should invoke the html file display.html in case true is returned from validateEmpDetails() function.

Step 9: Include the given js file dateValidation.js in the script tag.

Step 10: Modify the above code by adding another text field for accepting DateOfJoining(dd-mon-yyyy).

Step 11: Append the function(validateEmpDetails()) with the following functionalities. Invoke the functions defined previously for the validation of DateOfJoining.

Step 11.1: Invoke function isEmpty(). If it returns 0 then display a suitable error message and return false from the function.

Step 11.2: If isEmpty() returns 1 then invoke the next function onlySpaces(). If the function returns 0 then display a suitable error message and return false from the function.

Step 11.3: if onlySpaces() returns 1 then invoke the next function dateValidate () given in dateValidation.js. If the function returns 1 then display a suitable error message and return false else true from the function validateEmpDetails().

Step 12: The action performed should invoke the html file display.html in case true is returned from validateEmpDetails() function.

Summary of this exercise:

You have learnt how to have reusable functions for client side validation

Also have learnt how to use external js function.

Deliverables of the exercise:

1. empvalidate.html
2. display.html