G. Pullaiah College of Engineering and Technology
(Autonomous)

Department of Computer Science and Engineering

Web Technologies

Lecture Notes

Dr. S. Prem Kumar, M.Tech., Ph.D.,

Compiled from text book

INTERNET PROGRAMMING

A.A.PUNTAMBEKAR, Technical Publications

Unit - 2

Client Side Programming Java Script: An introduction to JavaScript–JavaScript DOM Model-Date and Objects,-Regular Expressions- Exception Handling-Validation-Built-in objects-Event Handling DHTML with JavaScript- JSON introduction – Syntax – Function Files – Http Request –SQL.

4 Client Side Programming

Syllabus

Java Script : An introduction to JavaScript – JavaScript DOM Model–Date and Objects, –Regular Expressions – Exception Handling – Validation – Built-in objects – Event Handling – DHTML with JavaScript – JSON introduction – Syntax – Function Files – Http Request – SQL.

Contents

Part I : Introduction to JavaScript

4.1 Features of JavaScript

Following are some features of JavaScript

1. **Browser Support :** For running the JavaScript in the browser there is no need to use some plug-in. Almost all the popular browsers support JavaScripting.
2. **Structure Programming Syntax:** The Javascript supports much commonly the structured language type syntax. Similar to C-style the programming blocks can be written.
3. It automatically inserts the semicolon at the end of the statement, hence there is no need to write semicolon at the end of the statement in JavaScript.
4. **Dynamic Typing :** It supports dynamic typing, that means the data type is bound to the value and not to the variable. For example one can assign integer value to a variable say 'a' and later on can assign some strg the ing value to the same variable in JavaScript.
5. **Run Time Evaluation :** Using the **eval** function the expression can be evaluated at run time.
6. **Support for Object :** JavaScript is object oriented scripting language. However handling of objects in JavaScript is somewhat different that the conventional object oriented programming languages. JavaScript has a small number of in-built objects.
7. **Regular Expression :** JavaScript supports use of regular expressions using which the text-pattern matching can be done. This feature can be used to validate the data on the web page before submitting it to the server.
8. **Function Programming :** In JavaScript functions are used. One function can accept another function as a parameter. Even, one function can be assigned to a variable just like some data type. The function can be run without giving the name.

4.2 Writing First JavaScript

The JavaScript can be directly embedded within the HTML document or it can be stored as external file.

Directly embedded JavaScript

The syntax of directly embedding the JavaScript in the HTML is

```
<script type="text/javascript">
        …
        …
        …
</script>
```

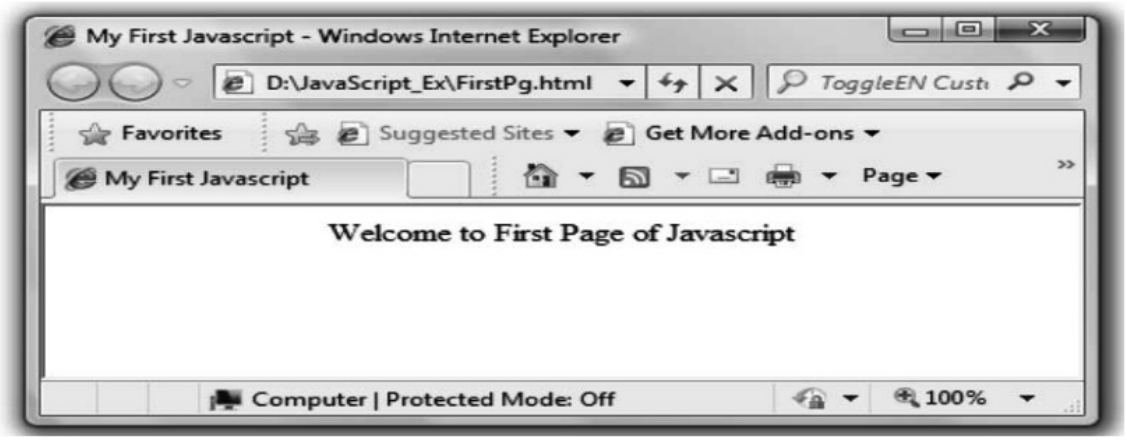
Ex. 4.2.1 : Write a JavaScript to display Welcome message in JavaScript

Sol. :

```
<!DOCTYPE html >
<html >
```

```
    <head>
        <title> My First Javascript </title>
    </head>
    <body>
        <center>
        <script type="text/javascript">
        /*This is the First JavaScript*/
        document.write(" Welcome to First Page of Javascript");
        </script>
        </center>
    </body>
</html>
```

Output



Script Explanation :

In above script

(1) we have embedded the javaScript within

```
<script type="text/javascript">
…
</script>
```

(2) a comment statement using /* and */. Note that this type of comment will be recognized only within the <script> tag. Because, JavaScript supports this kind of comment statement and not the XHTML document.

(3) Then we have written document.write statement, using which we can display the desired message on the web browser.

Indirectly Embedding JavaScript

If we want to embed the JavaScript indirectly, that means if the script is written in some another file and we want to embed that script in the HTML document then we must write the script tag as follows -

```
<script type="text/javascript" src="MyPage.js">
        …
        …
        …
</script>
```

Javascript is which is to be embedded is in the file **MyPage.js**

We will follow the following steps to use the external JavaScript file.

Step 1 : Create an XHTML document as follows -

XHTML Document[FirstPg.html]

```
<!DOCTYPE html >
<html>
        <head>
                <title> My First Javascript </title>
        </head>
                <body>
                        <center>
                        <script type="text/javascript" src="my_script.js">
                        </script>
                        </center>
                </body>
        </html>
```

This is an external javascript file,it can be specified with the attribute **src**

Step 2 :

JavaScript[my_script.js]

```
/*This is the First JavaScript*/
        document.write(" Welcome to First Page of Javascript");
```

Step 3 : Open the HTML document in Internet Explorer and same above mentioned output can be obtained.

Advantages of indirectly embedding of JavaScript

1. Script can be hidden from the browser user.
2. The layout and presentation of web document can be separated out from the user interaction through the JavaScript.

Disadvantages of indirectly embedding of JavaScript

1. If small amount of JavaScript code has to be embedded in XHTML document then making a separate JavaScript file is meaningless.
2. If the JavaScript code has to be embedded at several places in XHTML document then it is

complicated to make separate JavaScript file and each time invoking the code for it from the XHTML document.

4.3 Identifier, Keywords and Comments

1. Identifiers

Identifiers are the names given to the variables. These variables hold the data value. Following are some conventions used in JavaScript for handling the identifiers -

1. Identifiers must begin with either letter or underscore or dollar sign. It is then followed by any number of letters, underscores, dollars or digits.
2. There is no limit on the length of identifiers.
3. The letters in the identifiers are case-sensitive. That means the identifier INDEX, Index, index, inDex are considered to be distinct.
4. Programmer defined variable names must not have upper case letters.

2. Reserved words

Reserved words are the special words associated with some meaning. Various reserve words that are used in JavaScript are enlisted as below –

break	Continue	delete	for	in	return	throw	var	with
case	default	else	function	instanceof	switch	try	void	
catch	do	finally	if	new	this	typeof	while	

3. Comments

JavaScript supports following comments

1. The // i.e a single line comment can be used in JavaScript.
2. The /* and */ can be used as a multi-line comment.
3. The XHTML <!--> and <--> is also allowed in JavaScript

4. Semicolon

While writing the JavaScript the web programmer must give semicolon at the end of the statements.

4.4 Data Types

JavaScript defines two entities primitives and objects. The primitives are for storing the values whereas the object is for storing the reference to the actual value.

There are following primitive types used in JavaScript

1. Number 2. String 3. Boolean
4. Undefined 5. Null

There are three type of predefined objects in JavaScript

1. Number 2. String 3. Boolean

These objects are called **wrapper objects**. These wrapper objects provide properties and methods which can be used by primitive types.

Fig. 4.4.1 (a) Representation of primitive type

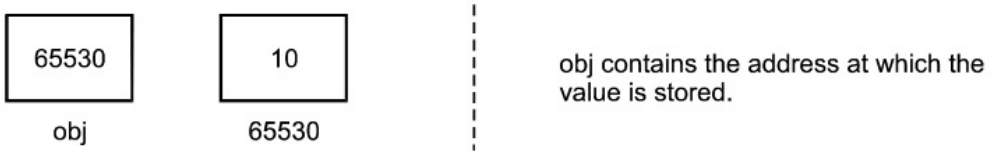


Fig. 4.4.1 (b) Representation of object

4.5 Variable

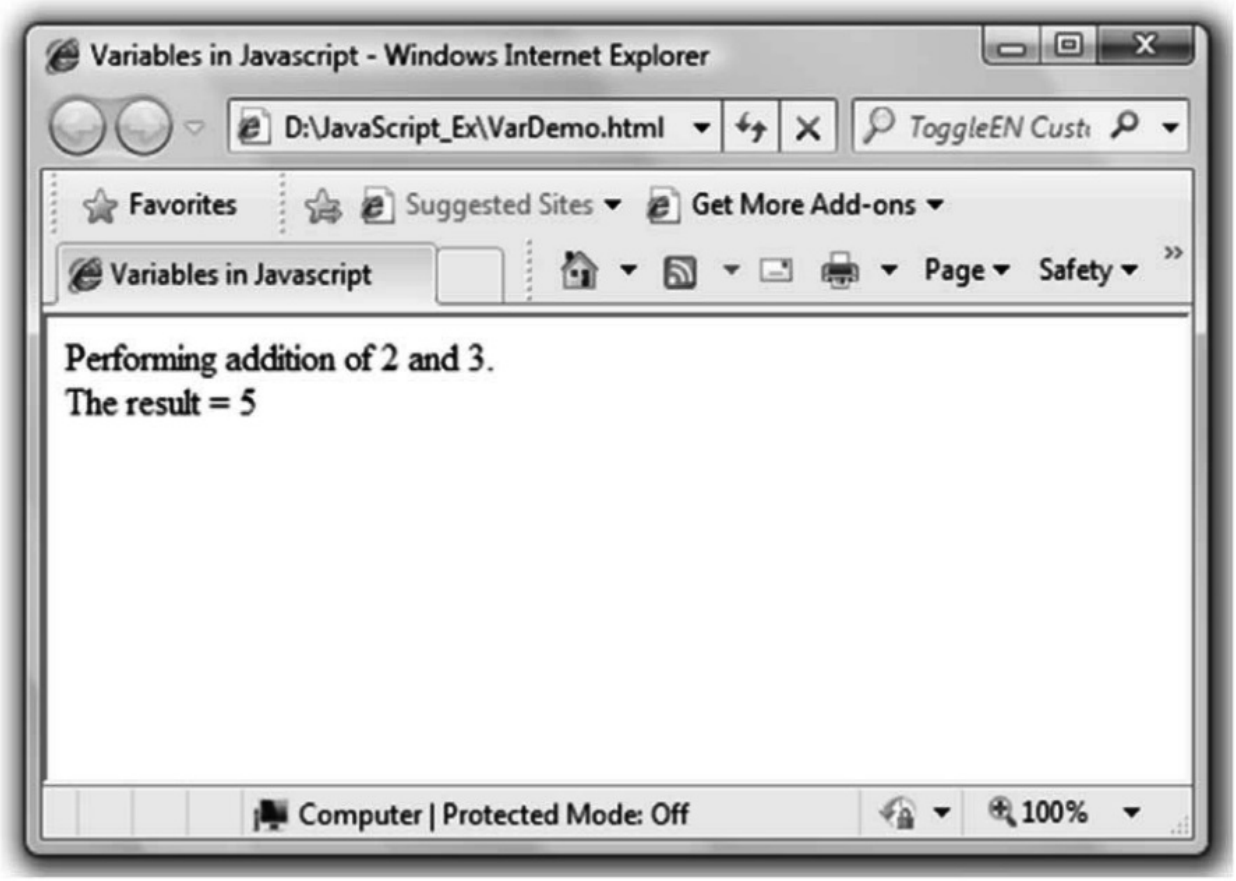
In JavaScript we can declare the variable using the reserved word var. The value of this variable can be any thing; it can be numeric or it can be string or it can be a Boolean value.

JavaScript[VarDemo.html]

```
<!DOCTYPE html >
<html >
 <head>
 <title> Variables in Javascript </title>
</head>
 <body>
  <script type="text/javascript">
   var a,b,c;
   var string;
   a=2;
   b=3;
   c=a+b;
   string ="The result = ";
   document.write("Performing addition of 2 and 3.  "+"<br/>");
   document.write(string);
   document.write(c);
  </script>
</body>
</html>
```

Variable declaration is done using **var**. Note that there is no data type required for handling variables.

Output



Note that using **var** we can define the variable which is of type numbers(2 , 3 or 5) as well as the string "The result".

4.6 Operators

Various operators used by JavaScript are as shown in following table -

Type	Operator	Meaning	Example
Arithmetic	+	Addition or unary plus	c = a+b
	–	Subtraction or unary minus	d = –a
	*	Multiplication	c=a*b
	/	Division	c=a/b
	%	Mod	c=a%b
Relational	<	Less than	a<4
	>	Greater than	b>10

	<=	Less than equal to	b<=10
	>=	Greater than equal to	a>=5
	==	Equal to	x==100
	!=	Not equal to	m!=8
Logical	&&	And operator	0&&1
	||	Or operator	0 ||1
Assignment	=	Is assigned to	a=5
Increment	++	Increment by one	++i or i++
Decrement	--	Decrement by one	– – k or k– –

The conditional operator is ? The syntax of conditional operator is

Condition?expression1:expression 2

Where expression1 denotes the true condition and expression2 denotes false condition.

For example :

a>b?true:false

This means that if a is greater than b then the expression will return the value true otherwise it will return false.

4.6.1 String Concatenation Operator

Two string can be concatenated using the + operator. A variable can be concatenated with the string using + operator also.

JavaScript[StrOper.html]

```
<!DOCTYPE html >
<html >
 <head>
  <title>String Concatenation Demo</title>
</head>
<body>
  <center>
   <script type="text/javascript">
    var first_string;
    first_string="Programming";
    document.write("<h3>"+first_string+" the web"+"</h3>");
   </script>
  </center>
</body>
</html>
```

Output

4.7 Input and Output

4.7.1 The document.write

- For displaying the message on the web browser the basic method being used is **document.write**. To print the desired message on the web browser we write the message in double quotes inside the document.write method.
- **For example**

```
document.write("Great India");
```

- We can pass the variable instead of a string as a parameter to the document.write.

For example

```
var my_msg="Great India";
document.write(my_msg);
```

Note that the variable my_msg should not be passed in double quotes to document.write otherwise the result the string "my_msg" will be printed on the browser instead of the string "Great India" .

- We can combine some HTML tags with the variable names in document.write so that the formatted display can be possible on the web browser.

For example – if we want to print the message "Great India" on the web browser in bold font then we can write following statements-

```
var my_msg="Great_India";
document.write("<strong>"+my_msg+"</strong>");
```

4.7.2 Popup Box

- One of the important features of JavaScript is its **interactivity** with the user.
- There are **three types of popup boxes** used in JavaScript by which user can interact with the browser.

alert box : In this type of popup box some message will be displayed.	
confirm box: In this type of popup box in which the message about confirmation will be displayed. Hence it should have two buttons **Ok** and **Cancel.**	
Prompt box is a type of popup box which displays a text window in which the user can enter something. Hence it has two buttons **Ok** and **Cancel**.	

JavaScript[PopupBox.html]

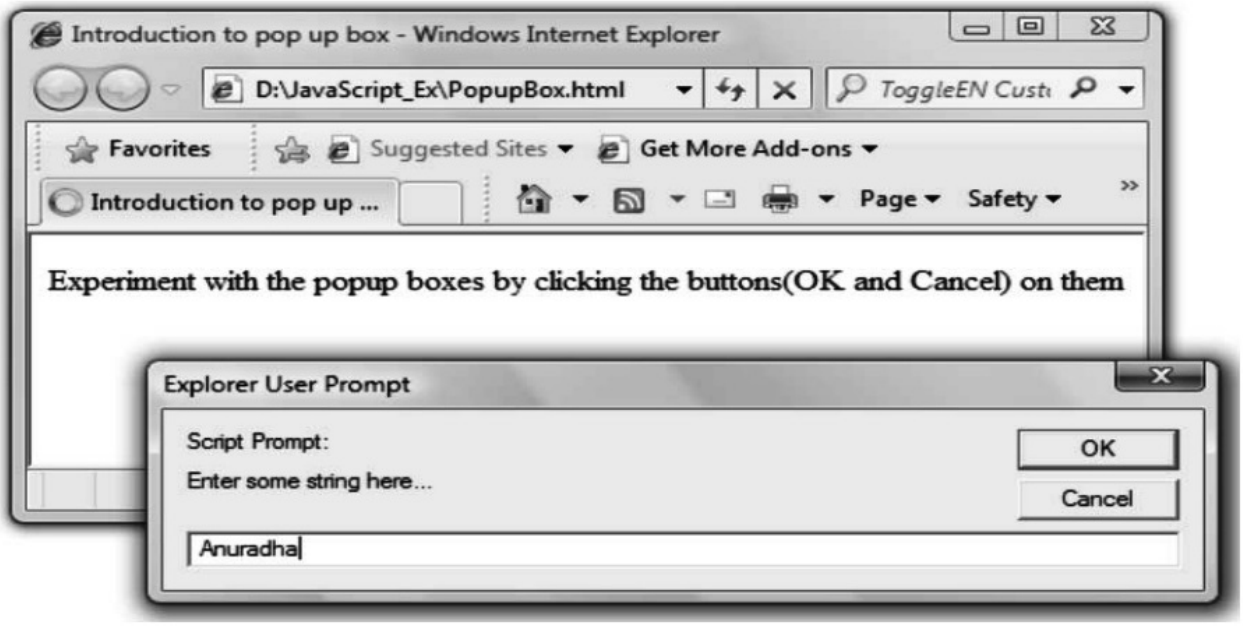
```
<!DOCTYPE html>
<html >
 <head>
  <title>Introduction to pop up box</title>
</head>
<body>
<p>Experiment with the popup boxes by clicking the buttons(OK and Cancel) on them</p>

   <script type="text/javascript">
    if(confirm("do you agree?"))
      alert("You have agreed");
    else
      input_text=prompt("Enter some string here..."," ");
    /*the value entered in prompt box is returned
    and stored in the variable text */
      alert("Hi "+input_text);
```

```
    </script>
</body>
</html>
```

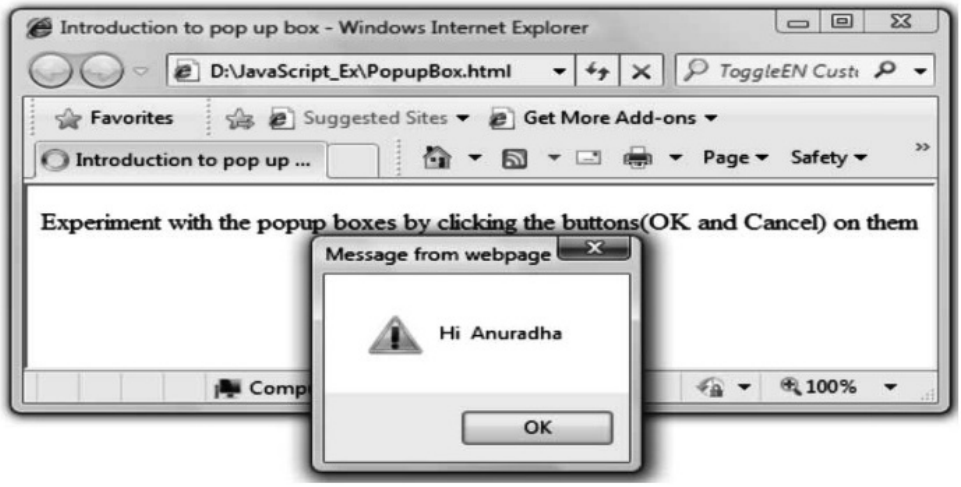
Output

If we click on OK button then an alert box will appear.

On load, the confirm box appears and if we click on Cancel button then the prompt box will appear. We can type some string within it.



Click OK button and we will get the alert box as follows -



Thus alert, confirm and prompt boxes cause the browser to wait for user response. The user responds by clicking the button on these popup boxes.

Script Explanation

On loading this script using web browser first of all a confirm box will be displayed. If we click on OK button an alert box will appear. Otherwise a prompt box will be displayed. Again if we click on the OK button of prompt box again an alert box will appear which will display the string which you have entered on prompt box. If you run the above script you will get all these effects.

4.8 Control Structures

AU : Dec.-08,18, May- 10, 11, Marks 9

Various control structures used in JavaScript are

Statement	syntax	Example

if-else	if(condition)
	statement
	else
	statement
		if(a>b)
		document.write(" a is greater than b");
		else
		document.write("b is greater than a");

while	while(condition)
	{
	statements
	}
		while(i<5)
		{
		i=i+1;
		document.write("value of i"+i)
		}

do..while	do
	{
	}while(condition);
		do
		{
		i=i+1;
		document.write("value of i"+i)
		}while(i<5);

for	for(initialization;test condition;stepcount)
	{
	statements
	}
		for(i=0;i<5;i++)
		{
		document.write(i);
		}

switch…case	switch(expression)
	{
	case 1: statements
	break;
	case 2: statements
	break;
	…
	default:
	statements
	}
		switch(choice)
		{
		case 1: c=a+b;
		break;
		case 2:c=a-b;
		break;
		}

break :
Similar to C or C++, the break statement is used to break the loop.
	break;	for(i=10;i>=0;i--)
		{
		if(i==5)
		break;
		}

continue:
The continue statement is used in a loop in order to continue(skip). The keyword **continue** is used to make use of continue statement in a loop.
	continue;	for(i=10;i>=0;i--)
		{
		if(i==5)
		{
		x=i;
		continue;
		}

Control structure is essential part of JavaScript. The Control Structures in JavaScript are just similar to that of C or C++. Various control structures are -

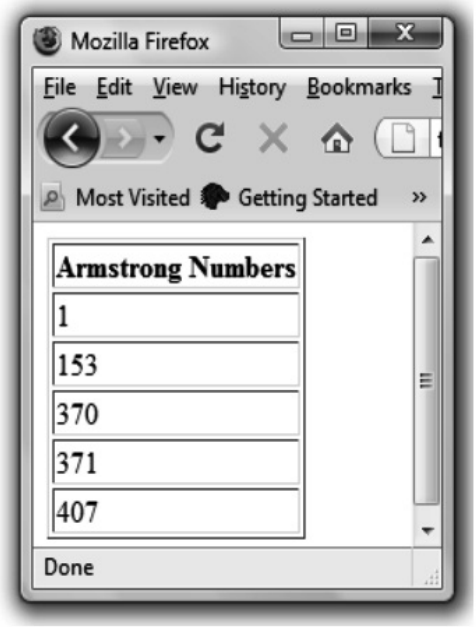
1. if statement 2. While 3. Do-while 4. for
5. switch case 6. break 7. Continue

Let us discuss various examples that make use of control structures.

Ex. 4.8.1 : Develop a javascript to generate 'ARMSTRONG NUMBERS' between the range 1 to 100. [Eg : 153 is an Armstrong number, since sum of the cube of the digits is equal to the number i.e. [13+53+33=153]

Sol. :

```
<html>
<body>
<table border="1" allign="center">
<th>Armstrong Numbers</th>
<script type="text/javascript">
var num,i,temp,sum;
var n=0;
i=1;
do
{
 num=i;
 sum=0;
 while(num>0)
 {
  n=num%10;
  n=parseInt(n);
  num=num/10;
  num=parseInt(num);
  sum=sum+(n*n*n);
  }
 if(sum==i)
 {
  document.write("<tr><td>"+i+"</td></tr>");
 }
 i++;
}while(i<=1000);
</script>
</table>
</body>
</html>
```



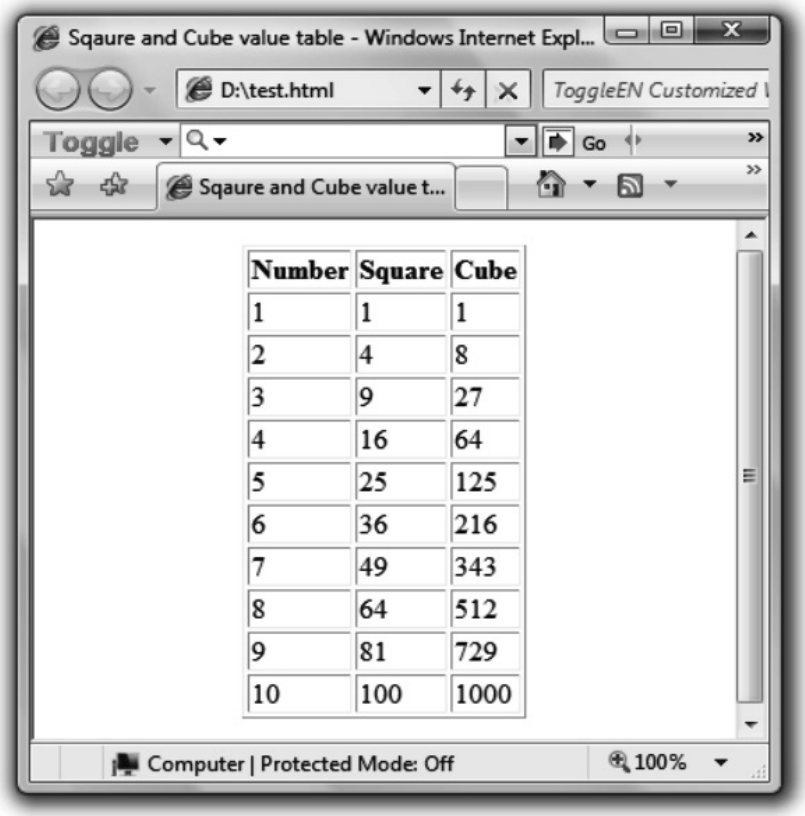
Ex. 4.8.2 : Write a javascript that displays the as per following :

(Calculate the squares and cubes of the numbers from 0 to 10)

Number	Square	Cube
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

Sol. :

```
<html>
<head>
<title>Sqaure and Cube value table</title>
</head>
<body>
<table border=1 align="center">
<th>Number</th><th>Square</th><th>Cube</th>
<script type="text/javascript">
for (i=1; i<=10; i++)
{
document.write("<tr><td>"+i+"</td><td>"+(i*i)+"</td><td>"+(i*i*i)+"</td></tr>");
}
</script>
</table>
</body>
</html>
```



Number	Square	Cube
1	1	1
2	4	8
3	9	27
4	16	64
5	25	125
6	36	216
7	49	343
8	64	512
9	81	729
10	100	1000

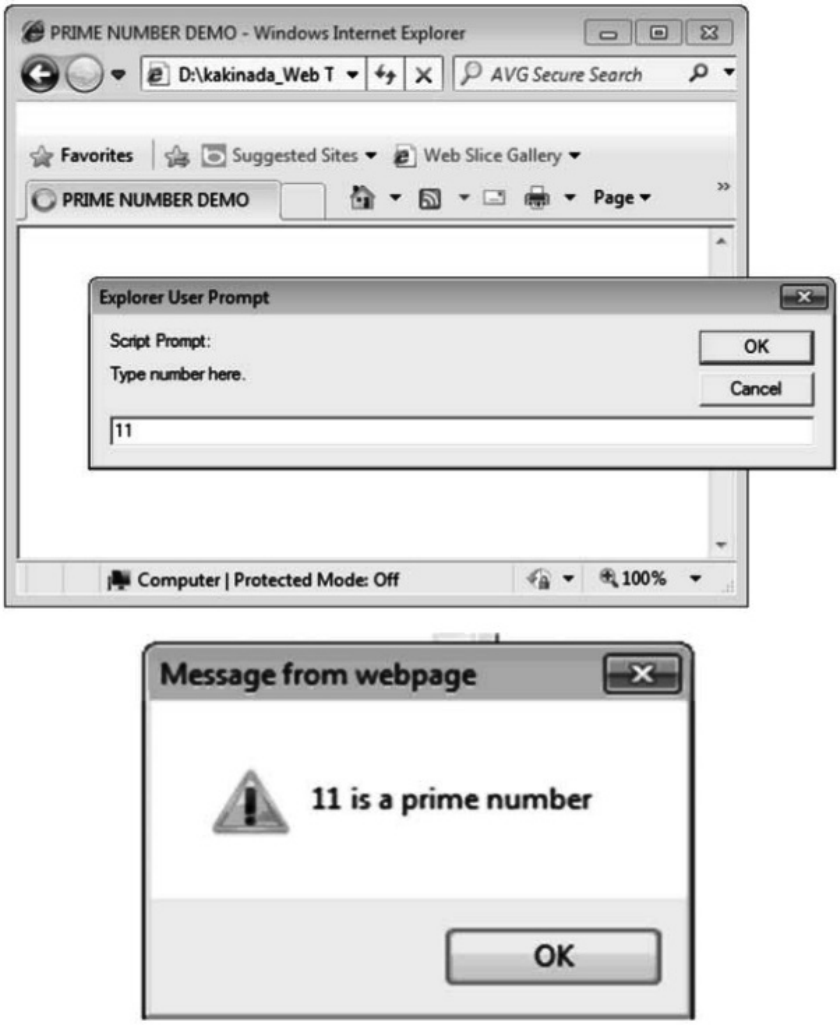
Ex. 4.8.3 : Write a script that reads an integer and displays whether it is a prime number or not.

Sol. :

```
<html>
 <head>
  <title>PRIME NUMBER DEMO</title>
 </head>
 <body>
  <script type="text/javascript">
  var num=prompt("Type number here.","");
  var b;
  var flag=1;
  for(i=2;i<num;i++)
  {
    b=num%i;
    if(b==0)
    {
     flag=0;

    break;
    }
  }
```

```
    if(flag==0)
      alert(num+" is not a prime number");
    else
      alert(num+" is a prime number");
</script>
</body>
</html>
```

Output



Ex. 4.8.4 : Write a JAVAScript to print characters of a string at odd positions.

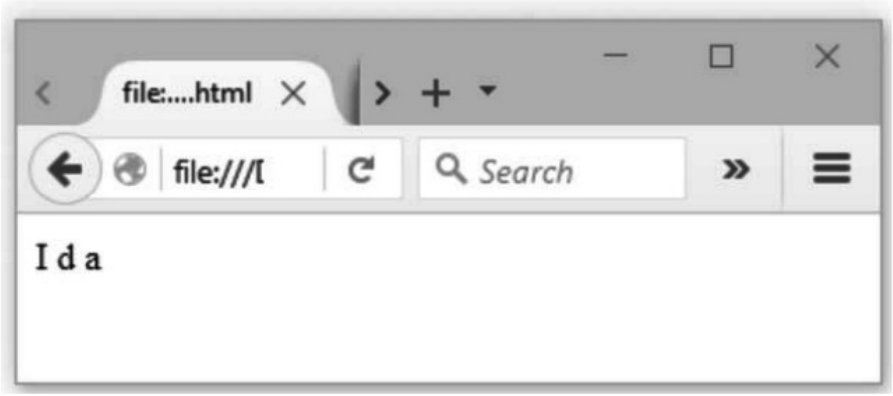
(for example for the string India, I, d and a should get printed).

Sol. :

```
<html>
<body>
<script>
var str="India";
var k=str.length;
```

```
for(i=0;i<=k;i=i+2)
{
 n=str.charAt(i);
 document.write(n);
 document.write(" ");
}
</script>
</body>
</html>
```

Output



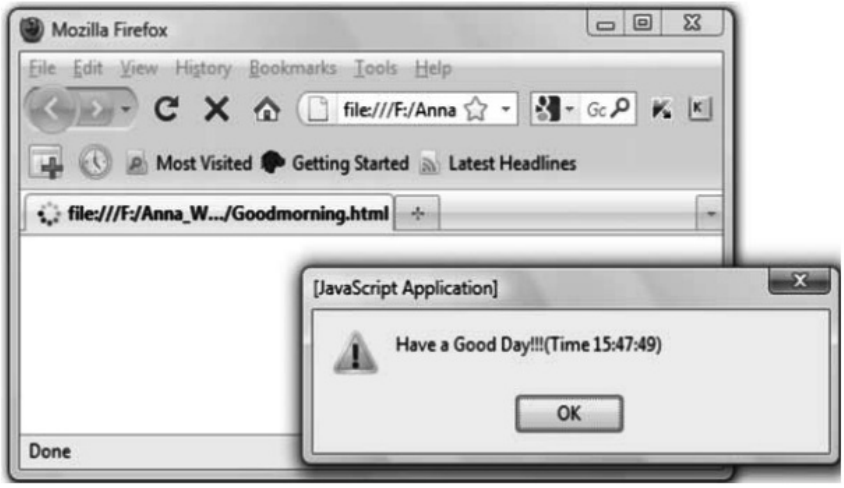
Ex. 4.8.5 : Develop a JavaScript page to demonstrate an If condition by which the time on your browser is less than 10; you will get a "Good morning" greeting.

AU : Dec.-08, Marks 2

Sol. : Goodmorning.html

```
<html>
<head>
<script type="text/javascript">
function GreetMsg()
{
 var today=new Date();
 var h=today.getHours();
 var m=today.getMinutes();
 var s=today.getSeconds();
 if(h<10)
  alert("Good Morning!!!(Time "+h+":"+m+":"+s+")");
 else
  alert("Have a Good Day!!!(Time "+h+":"+m+":"+s+")");
}
</script>
</head>
<body onload="GreetMsg()">
</body>
</html>
```

Output



Ex. 4.8.6 : Write a JavaScript to find and print the largest and smallest values among 10 elements of an array.

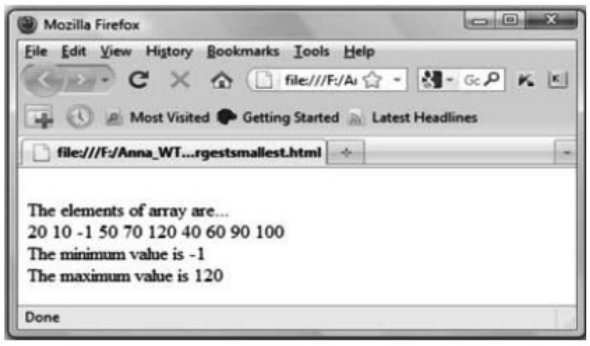
AU : May-10, Marks 8

Sol. : largestsmallest.html

```
<html>
  <head>
  <script type="text/javascript">
   function fun()
   {
    a=new Array(10);
    a[0]=20;
    a[1]=10;
    a[2]=-1;
    a[3]=50;
    a[4]=70;
    a[5]=120;
    a[6]=40;
    a[7]=60;
    a[8]=90;
    a[9]=100;
    document.write("<br/>The elements of array are...<br/>");
    max_val=a[0];
    for(i=0;i<10;i++)
     document.write(" "+a[i]);
    min_val=a[0];
    max_val=a[0];
    for(i=0;i<10;i++)

    {
```

```
      if(a[i]<min_val)
        min_val=a[i];
      if(a[i]>max_val)
        max_val=a[i];
    }
    document.write("<br/>The minimum value is "+min_val);
    document.write("<br/>The maximum value is "+max_val);
  }
  </script>
 <title>Finding largest and smallest Element</title>
 </head>
  <body onload=fun()>
  </body>
</html>
```

Output



Ex. 4.8.7 : Write a JavaScript to find the product of first 15 even numbers.

Sol. :

```
<html>
<body>
<script type="text/javascript">
prod=1;
for(i=1;i<15;i++)
{
    if(i%2)
    {
     prod=prod*i;
    }
}
document.write("Product of odd numbers between 1 to 15 is "+prod);
</script>
</body>
</html>
```

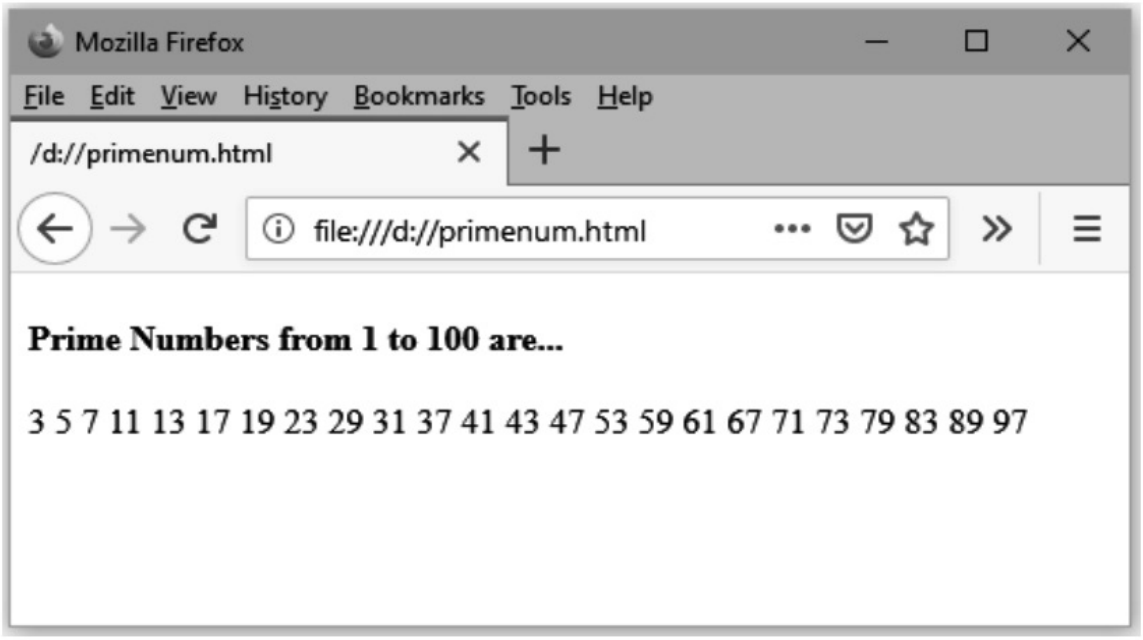
Ex. 4.8.8 : Write a JavaScript program to print Prime Numbers from 1 to 100.

AU : Dec.-18, Marks 9

Sol. :

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">

function primeNumber(from, to){
            var flag=false;
            for(i = from; i <= to; i++)
          {
            for( j = 2; j < i; j++)
            {
              if(i % j == 0)
               {
                    flag = false;
                    break;
               }
              else
               {
                    flag = true;
               }
            }
            if(flag)
            {
              document.write(" "+i);
            }
          }
      }
</script>
</head>
<body>
<h4> Prime Numbers from 1 to 100 are...</h4>
<script type="text/javascript">
primeNumber(1, 100)
</script>
</body>
</html>
```

Output



University Question

1. *State and explain the types of statements in JavaScript.* **AU : May-11, Marks 8**

4.9 Functions

AU : Dec.-07, 08, May-11,12,13, Marks 16

- We can write the functions in the JavaScript for bringing the modularity in the script.
- Separate functions can be created for each separate task. This ultimately helps in finding the bug from the program efficiently.
- We can define the function anywhere in the script either in head or body section or in both. But it is a standard practice to define the function in the head section and call that function from the body section.
- The keyword **function** is used while defining the function.
- The syntax for defining the function is

```
function name_of _function (arg1,arg2,…argn)
{
 …
 Statements
}
```

Here is a simple illustration in which we have written a function **my_fun()**

JavaScript[FunDemo.html]

```
<!DOCTYPE html>
<html>
```

```
<head>
 <title>Function Demo</title>
 <script type="text/javascript">
  function my_fun()
  {
    document.write("This statement is within the function");
  }
</script>
</head>
<body>
<center>
  <script type="text/javascript">
    document.write("This statement is before a function call");
    document.write("<br>");
    my_fun();
  call to the function
  </script>
</center>
</body>
</html>
```

Definition of function

Output

Script Explanation

The above code is pretty simple. We have defined one function named **my_fun** in the **head** section of the HTML document and called it from the **body** section. The corresponding **write** statements are used to display the messages on the browser window.

4.9.1 Returning Value from the Function

- We can return some value from the function using a keyword **return**.
- This return value is either stored in variable or directly displayed on the browser window.
- Following is a simple illustration in which the value is returned from the function and is directly displayed on the browser window using **document.write.**

JavaScript[FunRetDemo.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Function Demo</title>
 <script type="text/javascript">
  function my_fun()
  {
    str="I am function returned value";
    return str;
  }
</script>
</head>
<body>
<center>
  <script type="text/javascript">
    document.write("This statement is before a function call");
    document.write("<br>");
    document.write("<h3>"+my_fun()+"<h3>");
  </script>
</center>
</body>
</html>
```

Output

4.9.2 Passing the Parameters to the Function

- Similarly we can pass some arguments to the function.
- The syntax is

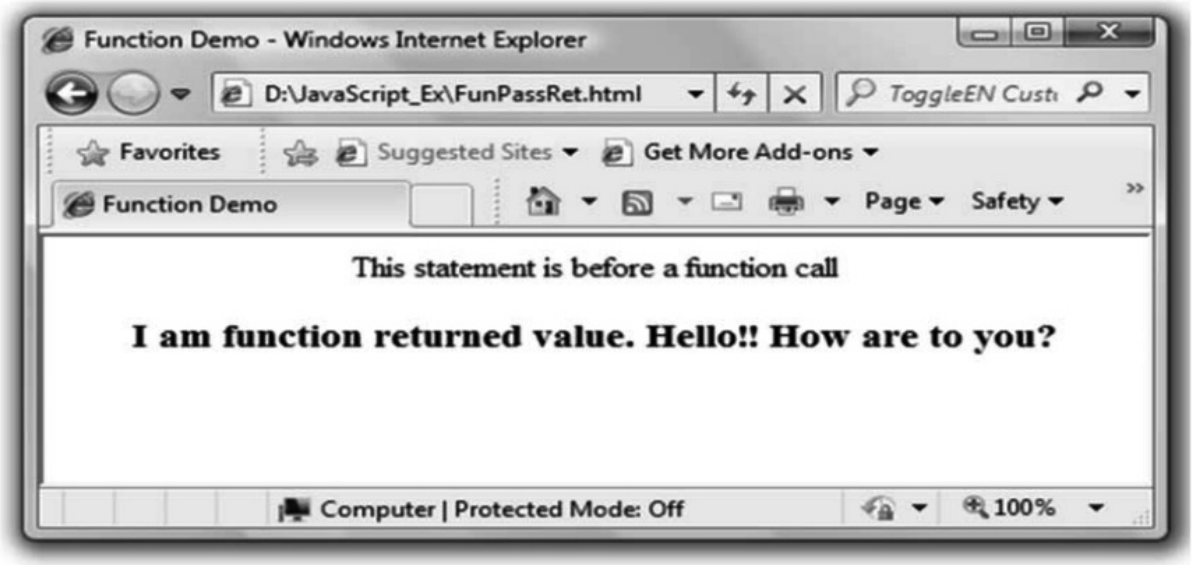
```
function function_name(aregument1, argument2…,argumentn)
{
//body of function
}
```

- In the following program, we have passed two arguments to the function and returning some value from the function.

JavaScript[FunPassRet.html]

```
<!DOCTYPE html >
<html>
 <head>
 <title>Function Demo</title>
 <script type="text/javascript">
  function my_fun(str1,str2)//two atrguments str1 and str2 are passed
  {
    str="I am function returned value."+" "+str1+" "+str2 ;
    return str; //returning one string value from function
  }
</script>
</head>
<body>
<center>
  <script type="text/javascript">
    document.write("This statement is before a function call");
    document.write("<br>");
    document.write("<h3>"+my_fun("Hello!!","How are to you?")+"<h3>");
  </script>
</center>
</body>
</html>
```

Output



4.9.3 Passing an Array to the Function

- Similar to C or C++ we can pass an entire array as a parameter to the function.
- This method of array passing is called as **call by reference.**
- When an array is passed to the function the array name is simply passed.
- In the following JavaScript we have initialized an array in the body part and to display those contents of an array we have called a **display()** function. And to this function an array is passed as an argument. Let see the JavaScript.

JavaScript[FunPassArr.html]

```
<!DOCTYPE html>
<html>
<head>
 <title>Passing an Array to the function</title>
 <script type="text/javascript">
  function display(a)
  {
   document.write("The contents of the array are..."+"<br>");
   i=0;
   for(i in a)
   {
    document.write(a[i]+"<br>");
    i++;
   }
  }
</script>
</head>
<body>
```

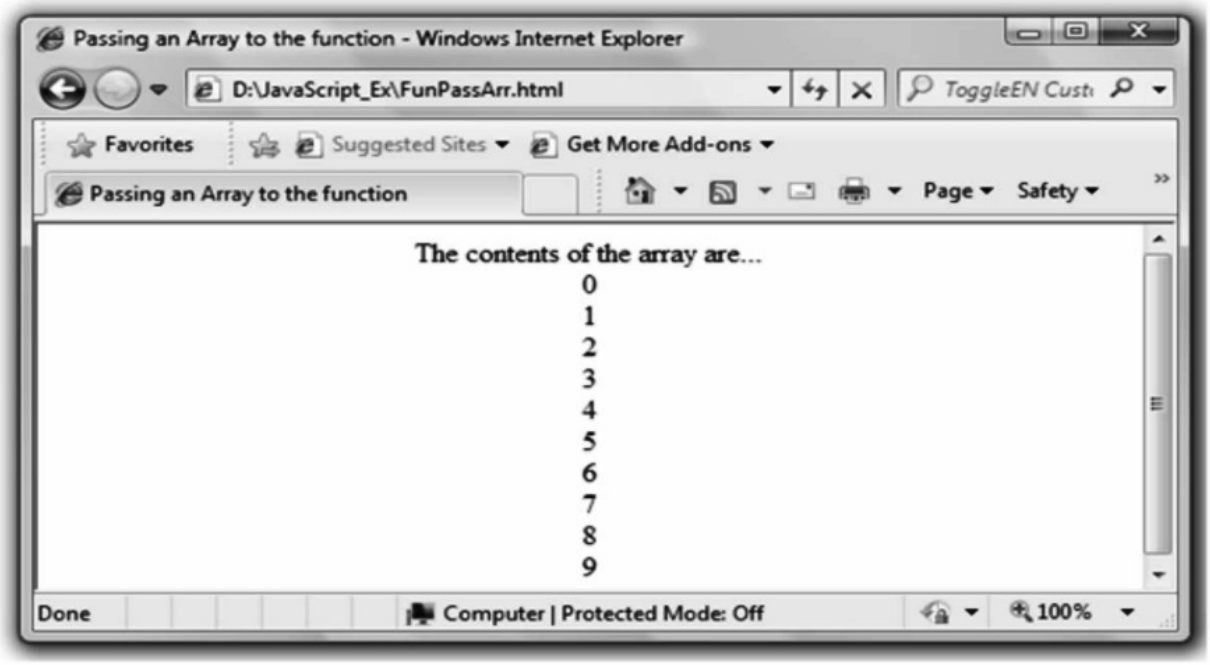
Function definition with array as an argument

```
<center>
 <script type="text/javascript">
   var ar=new Array(10);
   for(i=0;i<=9;i++)
   {
     ar[i]=i;
   }
   display(ar);
 </script>
</center>
</body>
</html>
```

Function call

In above given script we have called a **display** function by passing an array as a parameter to it. Hence we will get following output.

Output



Ex. 4.9.1 : Write a JavaScript to sort the elements of an array in ascending order

Sol. :

```
<!DOCTYPE html>
<html>
<head>
 <title>Sorting the Numbers</title>
 <script type="text/javascript">
 function display(a)
 {
  i=0;
  for(i in a)
```

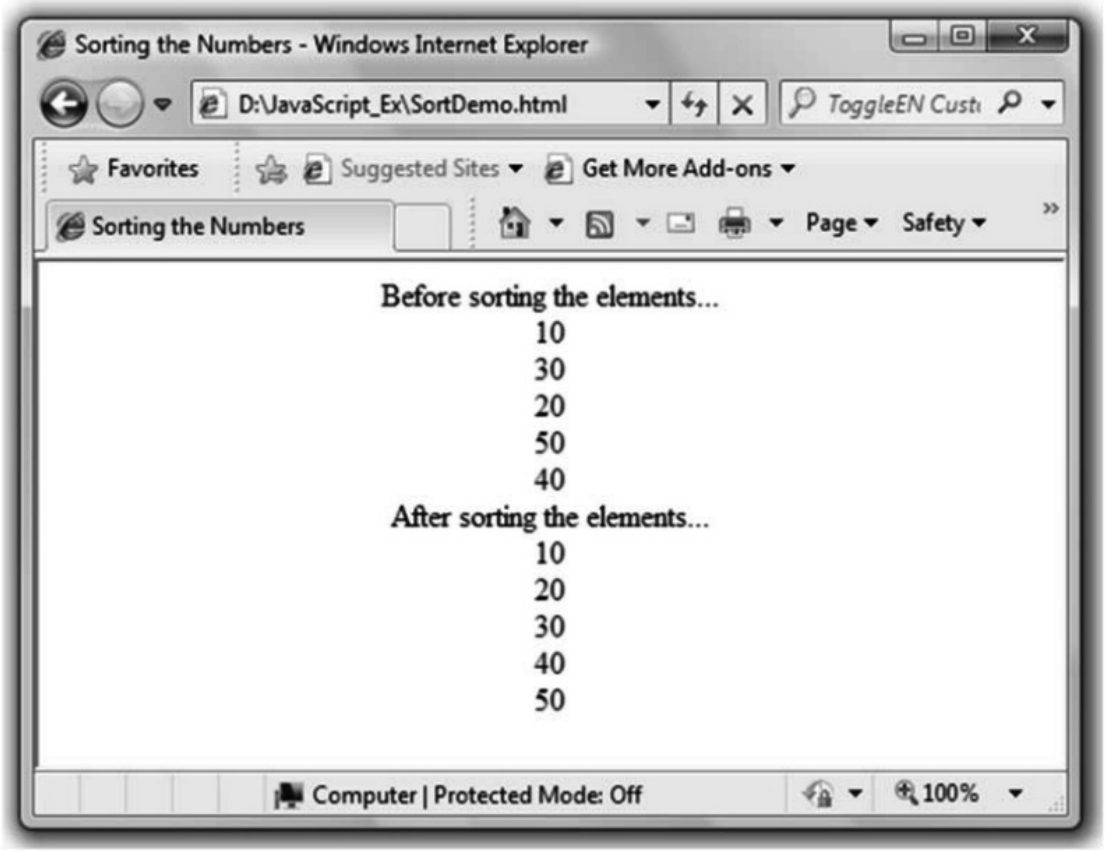
```
 {
  document.write(a[i]+"<br>");
  i++;
 }
}
function sort(a)
{
 for(i=0;i<a.length-1;i++)
 {
  for(j=i+1;j<a.length;j++)
  {
   if(a[i]>a[j])
   {
    temp=a[i];
    a[i]=a[j];
    a[j]=temp;
   }
  }
 }
}
</script>
</head>
<body>
<center>
 <script type="text/javascript">
  var ar=new Array(10,30,20,50,40);
  document.write("Before sorting the elements..."+"<br>");
  display(ar);
  sort(ar);
  document.write("After sorting the elements..."+"<br>");
  display(ar);
 </script>
</center>
</body>
</html>
```

Script Explanation

1. In the above program we have stored some values in the array in the body part.
2. Then in order to display the contents of an array we have called **display** function.
3. Then we have called one more function called **sort** in which the elements of an array are sorted.
4. Note that in the sort function we have used *length* property which returns total number of elements in an array.
5. The simple bubble sort method is used to sort the array in an ascending order.

6. Again the display function is called in order to display the sorted array.

Output



Ex. 4.9.2 : Write an HTML and JavaScript program which accepts N as input and displays first N Fibonacci numbers as list.

Sol. :

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function fun(str)
{
 var num=Number(str);
 var i,j,k,count;
 document.write("<h3>"+" The Fibonacci series is as follows..."+"</h3>");
 i=0;
 j=1;
 document.write(j);
 for(count=0;count<=num;count++)
 {
  k=i+j;
```

```
 i=j;
 j=k;
 document.write(","+k);
 }
}
</script>
</head>
<body>
<script type="text/javascript">
 var input_str=prompt("Enter some number","");
 fun(input_str);
</script>
</body>
</html>
```

Ex. 4.9.3 : Write a javascript that uses a loop, that searches a word in sentence held in an array, returning the index of the word.

Sol. :

```
<!DOCTYPE html>
<html >
 <head>
<script type="text/javascript">
function Search(form1)
{
 var Sentence= form1.input.value;
 var pattern=prompt("Enter the word to be searched for: ","");
 alert(pattern);
 temp=Sentence.split(" ");
 for(i=0;i<temp.length;i++)
 {
  if(temp[i]==pattern)
  {
    alert("The word is at position: "+(i+1));
 }
 }
}
</script>
</head>
<body>
<form name="form1">
 <b> Enter the sentence </b>
 <input type="text" name="input"><br/>
 <input type="button" value="Click Me" onclick="Search(form1)">
</form>
</body>
</html>
```

Output

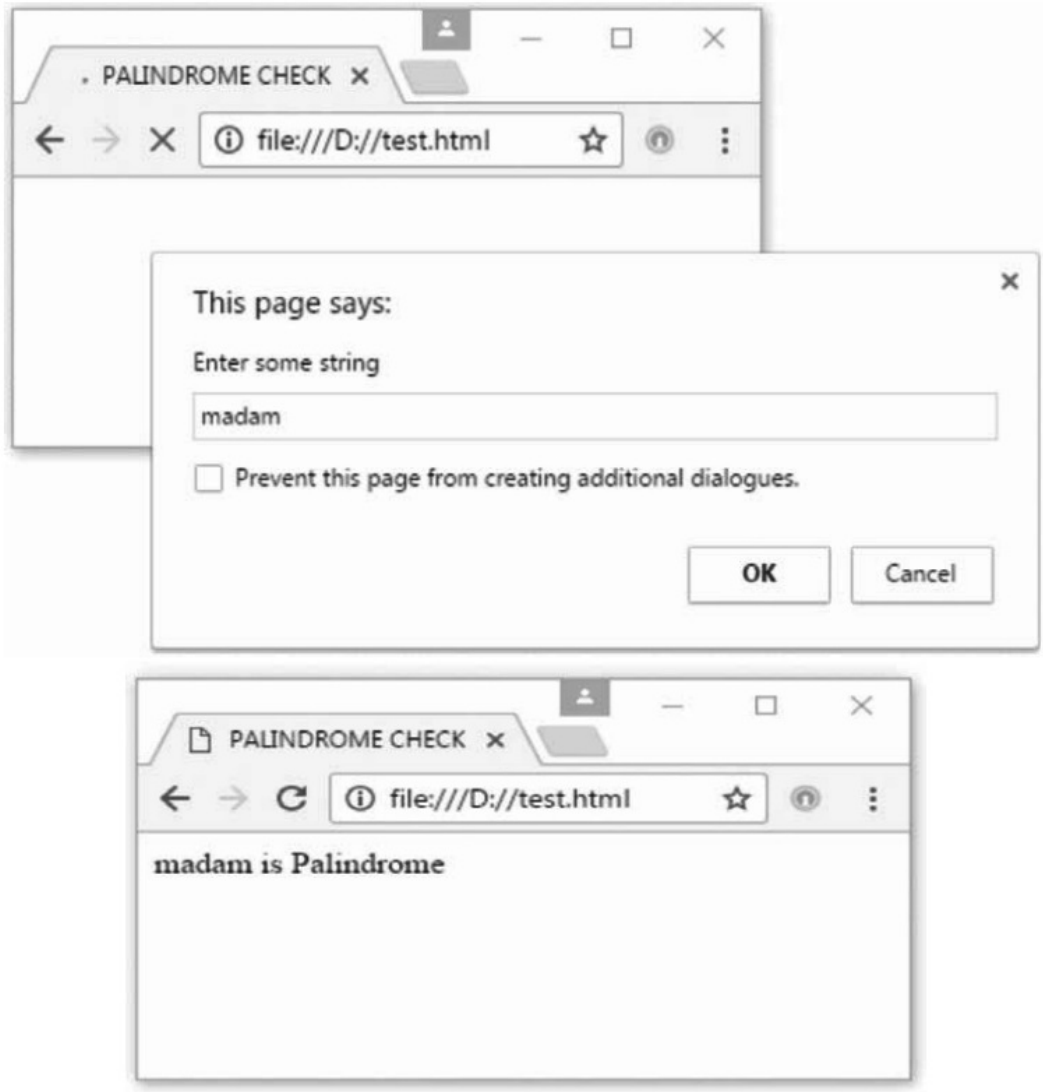

file:///D:...arch.html
file:///[
Search
Enter the sentence
My country is India
Click Me
Enter the word to be searched for:
is
OK
Cancel
The word is at position: 3
Prevent this page from creating additional dialogs
OK

Ex. 4.9.4 : Write a JavaScript to check input string is palindrome or not.

Sol. :

```
<html>
<head>
<title>PALINDROME CHECK</title>
<script type="text/javascript">
function palindrome(str)
{
    var len = str.length;

    for ( var i = 0; i < Math.floor(len/2); i++ )
    {
        if (str[i] !== str[len - 1 - i])
        {
           return false;
    }
        }
    return true;
}
</script>
</head>
<body>
<script type="text/javascript">
str=prompt("Enter some string","");
if(palindrome(str))
 document.write(str+" is Palindrome");
else
 document.write(str+"Not Palindrome");
</script>
</body>
</html>
```

Output



Ex. 4.9.5 : Develop and demonstrate a HTML file that includes JavaScript that uses a function for the following problems :

(a) Parameter : A string Output : The position of the string of the left-most vowel

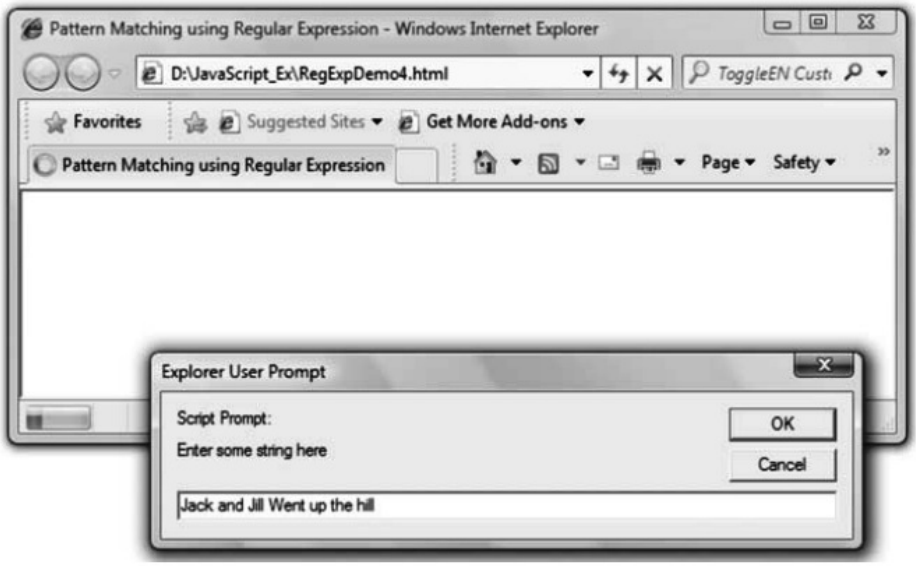
(b) Parameter : A number Output : The number with digits in the reverse order.

Sol. : (a) [RegExpDemo4.html]

```
<!DOCTYPE html>
<html >
 <head>
<title>Pattern Matching using Regular Expression </title>
<script type="text/javascript">
function TestString(str)
{
  document.write("The given string is ");
  document.write("<em>"+str+"</em>"+"<br/>");
```

```
  var i=str.match(/[aeiouAEIOU][a-zA-Z]*/);
  return i;
}
</script>
</head>
<body>
  <h3>
   <script type="text/javascript">
     var input_str=prompt("Enter some string here","");
     p=TestString(input_str);
     document.write("The leftmost vowel is found at index "+p.index);
   </script>
   </h3>
</body>
</html>
```

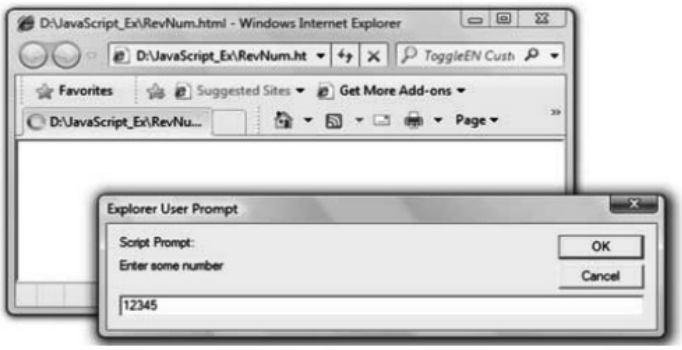
Output



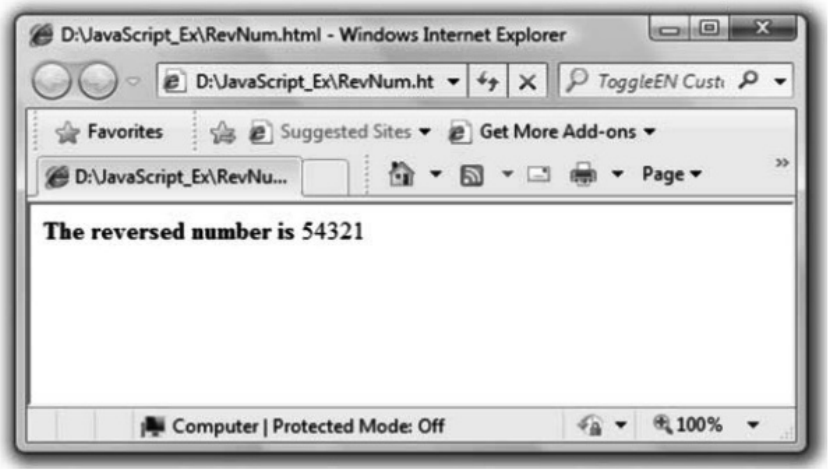
b) [RevNum.html]

```
<!DOCTYPE html>
<html >
 <head>
<script type="text/javascript">
function fun(str)
{
 var num=Number(str);
 var n=0;
 document.write("<b>"+"The reversed number is "+"</b>");
 while(num>0)
 {
  n=num%10;
  n=parseInt(n);
  num=num/10;
  num=parseInt(num);
  document.write(n);
 }
}
</script>
</head>
<body>
<script type="text/javascript">
 var input_str=prompt("Enter some number","");
 fun(input_str);
</script>
</body>
</html>
```

Output



Click OK button and you will get the reversed number as follows -



Ex. 4.9.6 : Write a JavaScript function to find the product of two arguments and return the result.

AU : Dec.-08, Marks 5

Sol. :

```
<!DOCTYPE html>
<html>
 <head>
 <title>Function Demo</title>
 <script type="text/javascript">
  function my_fun(num1,num2)
  {
   return num1*num2;
  }
</script>
</head>
<body>
<center>
  <script type="text/javascript">
    document.write("The product of 10 and 20 is ");
    document.write("<br>");
    document.write("<h3>"+my_fun(10,20)+"</h3>");
  </script>
</center>
</body>
</html>
```

Ex. 4.9.7 : Write a JavaScript function named drawstripes that will draw a number of colored strips by generating an HTML table in which each row has (possibly) different color. Your function must have 3 parameters :

- **Width, the width of each strip**
- **Height, the height of each strip**
- **Colors an array in which each element is the name of color. Each color name is a valid HTML color (can be used as the value of BGCOLOR attribute or CSS color property)**

AU : Dec.-07, CSE, Marks 16

Sol. : [stripeDemo.html]

```
<html>
<head>
<title></title>
<script>
var colors = new Array('#ff0000','#00ff00','#0000ff','#ff00ff');
function drawstrips(id)
{
    if(document.getElementById)
    {
     var table = document.getElementById(id);//get the table
     var rows = table.getElementsByTagName("tr");       //get the row of table
     var wd,ht;
     for(i = 0; i < rows.length; i++)
     {
        wd=prompt("Enter the width of the strip"+i,"");
        ht=prompt("Enter the height of the strip"+i,"");
           w=Number(wd);
           h=Number(ht);
        display(rows[i],i,w,h);
```

```
      }
     }
}
function display(row,i,w,h)
{
     row.style.backgroundColor = colors[i%colors.length]; //using CSS style
     row.style.width = w;
     row.style.height = h;
}
</script></head>
<body onLoad="drawstrips('MyTable')">
<table id="MyTable" cellspacing="10" cellpadding="10">
      <tr><td>Strip 1</td></tr>
      <tr><td>Strip 2</td></tr>
      <tr><td>Strip 3</td></tr>
     </table>
</body>
```

Varying color, width and height of each

Note : For the sake of understanding following output is given for the above application]

Ex. 4.9.8 : Write a JavaScript to find the sum of first n even numbers and display the result. Get the value of n from the user.

AU : May-13, Marks 8

Sol. :

```
<html>
<head>
<script type="text/javascript">
function EvenNumSum(str)
{
```

```
    var n=Number(str);
    sum=0;
    for(i=1;i<n;i++)
    {
    if(i%2==0)
     sum=sum+i;
    }
    document.write("<br>"+" Sum of first "+n+" even numbers is "+sum+"<br>");
}
</script>
</head>
<body>
<script type="text/javascript">
    var n=prompt("Enter the value of n","");
    EvenNumSum(n);
</script>
</body>
</html>
```

4.9.4 Global Functions of JavaScript

Global functions are the top-level functions in JavaScript that are independent of any specific object. These functions use the built in objects of the JavaScript. Following are some global functions -

1. encodeURI(URI)

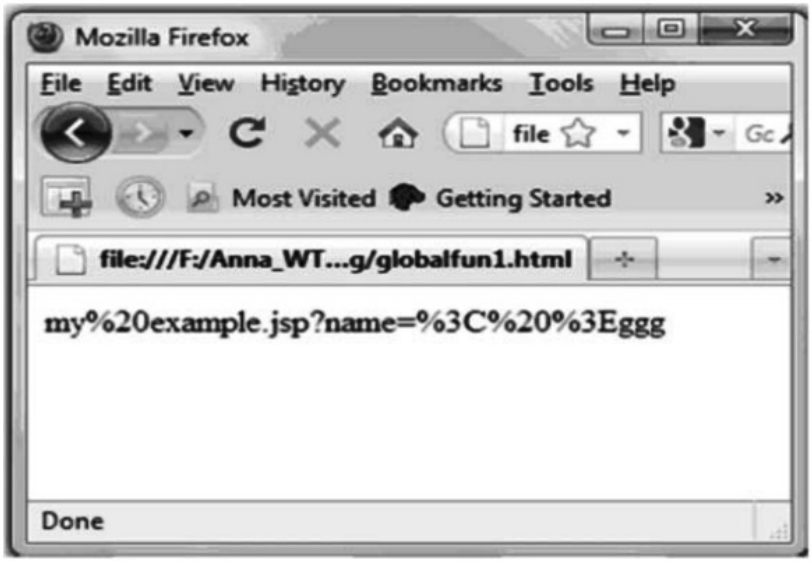
This function is used to encode the URI.This function encodes special characters, except: , / ? : @ & = + $ #.

For example

globalfun1.html

```
<html>
  <body>
   <script type="text/javascript">
    var uri="my example.jsp?name=< >ggg"
    document.write(encodeURI(uri)+ "<br />");
   </script>
  </body>
</html>
```

Output



2. decodeURI

This function decodes the encoded URI.

For example

globalfun2.html

```
<html>
  <body>
    <script type="text/javascript">
      var uri="my example.jsp?name=< >ggg"
      document.write(encodeURI(uri)+ "<br />");
      document.write(decodeURI(uri)+ "<br />");
    </script>
  </body>
</html>
```

Output

3. parseInt

This function parses a string and returns the integer value.

The syntax is -

```
parseInt(string,radix)
```

String is parsed to obtain integer value	It represents the numerical system such as octal, decimal or hexadecimal. It is optional parameter.

Example

globalfun3.html

```
<html>
  <body>
    <script type="text/javascript">
      document.write(parseInt("100"));
    </script>
  </body>
</html>
```

The output of above code will be 100.

Similar to the **parseInt** function the **parseFloat** function is used to obtain the real value from the string.

4. eval

The eval function is used to evaluate the expression.

The syntax is

```
eval(string);
```

Example

globlafun4.html

```
<html>
  <body>
    <script type="text/javascript">
      document.write(eval("2+3*5"));
    </script>
  </body>
</html>
```

The output of above code will be **17.**

University Questions

1. *Explain how functions can be written in JavaScript with an example.* **AU : May-11, Marks 8**
2. *Explain how local and global functions can be written using JavaScript.* **AU : May-12, Marks 8**

4.10 Arrays

AU : May- 08,12, Dec.-13, Marks 8

- Arrays is a collection of similar type of elements which can be referred by a common name.
- Any element in an array is referred by an array name followed by "[" followed by position of the element followed by].
- The particular position of element in an array is called array index or subscript.

For example –

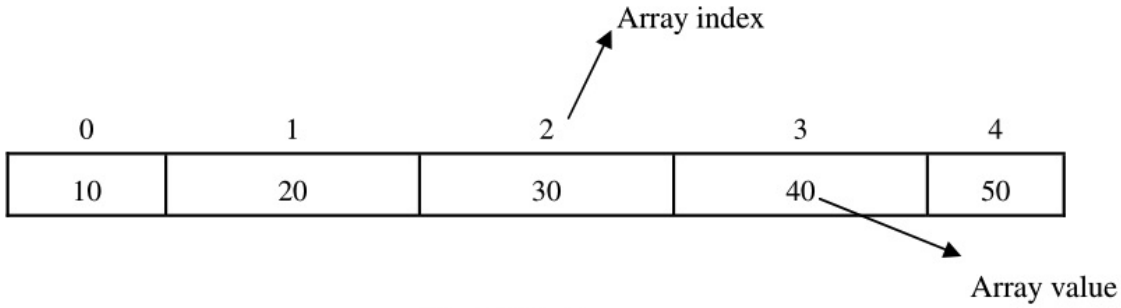


Fig. 4.10.1 Arrays

- Normally the first element in an array is stored at 0th location, however we can start storing the element from any position.

4.10.1 Array Declaration

- In JavaScript the array can be created using **Array** object.
- Suppose, we want to create an array of 10 elements then we can write,

```
var ar = new Array(10);
```

- Using new operator we can allocate the memory dynamically for the arrays.
- In the brackets the size of an array is mentioned and the var ar denotes the name of the array. Thus by the above sentence an array ar will be created in which we can store 10 elements at the most. Sometimes the above statement can be written like this

```
var ar;
 ar=new Array(10);
```

4.10.2 Array Initialization

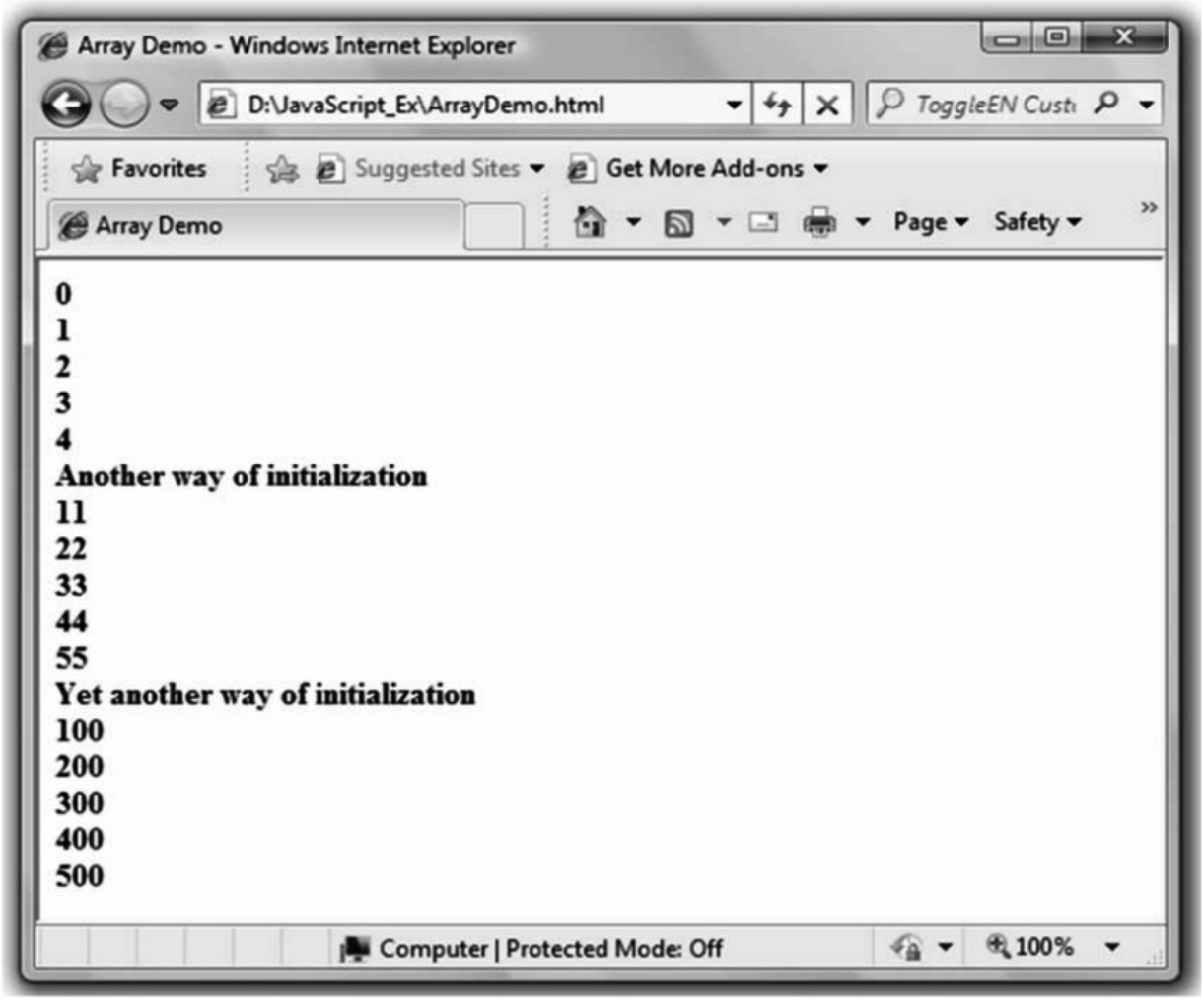
Let us see how to store some elements in an array.

- **JavaScript Program [ArrayDemo.html]**

```
<!DOCTYPE html>
<html>
<head>
  <title>Array Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
    a=new Array(5);//creation of array
    for(i=0;i<5;i++)
    {
      a[i]=i;
      document.write(a[i]+"<br>");//displaying array
    }
    document.write("Another way of initialization"+"<br>");
    b=new Array(11,22,33,44,55);//creation of array
    for(i=0;i<5;i++)
    {
      document.write(b[i]+"<br>");//displaying array
    }
    document.write("Yet another way of initialization"+"<br>");
    var c=[100,200,300,400,500];//creation of array
    for(i=0;i<5;i++)
    {
      document.write(c[i]+"<br>");//displaying array
    }
  </script>
 </strong>
</body>
</html>
```

As you can notice that, in above JavaScript an array can be initialized in three different ways which is shown by boldface. Hence an output of above script will be

Output



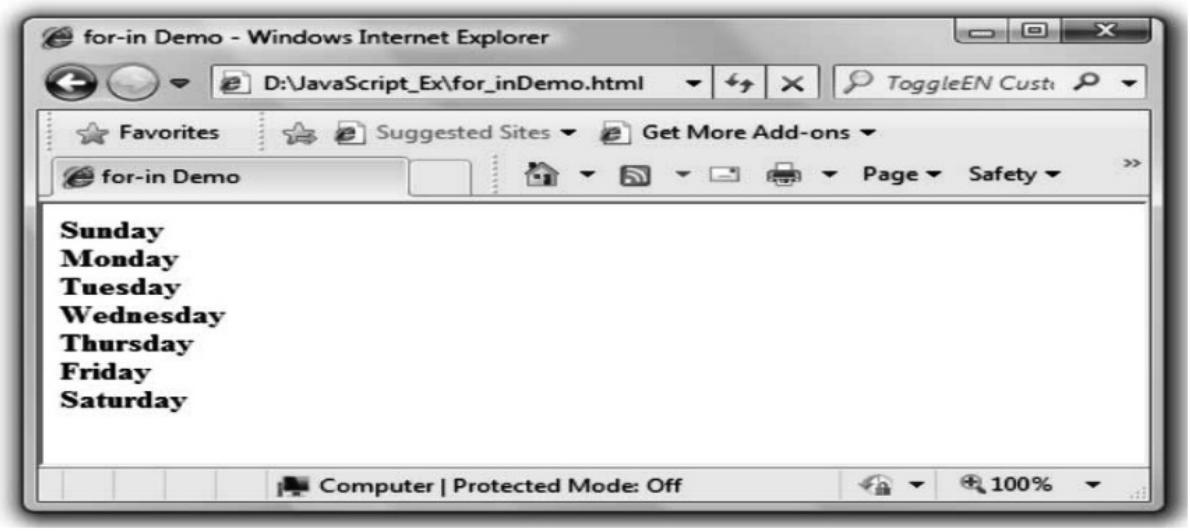
There is one control structure in JavaScript which is closely associated with array elements and such a control structure is **for…in**. Let us see a simple JavaScript which makes use of **for-in** control structure to display elements of an array.

JavaScript [for_inDemo.html]

```
<!DOCTYPE>
<html>
<head>
  <title>for-in Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
   Days=new Array();
   Days[0]="Sunday";
   Days[1]="Monday";
   Days[2]="Tuesday";
```

```
    Days[3]="Wednesday";
    Days[4]="Thursday";
    Days[5]="Friday";
    Days[6]="Saturday";
    for(i in Days)
    {
     document.write(Days[i]+"<br>");
    }
  </script>
 </strong>
</body>
</html>
```

Output



Ex. 4.10.1 : Write a javascript that reads ten numbers and displays the count of negative numbers, the count of positive numbers and the count of zero from the list.

Sol. :

```
<html>
    <body>
    <script type="text/javascript">
    ar=new Array();
    ar[0]=10;
    ar[1]=0;
    ar[2]=-9;
    ar[3]=0;
    ar[4]=20;
    ar[5]=30;
    ar[6]=3;
    ar[7]=88;
    i=0;
    for(i in ar)
```

Reading ten numbers

```
    {
     document.write(ar[i]+"<br/>");
     i++;
    }
    positive_count=0;
    negative_count=0;
    zero_count=0;
    i=0;
    for(i in ar)
    {
     if(ar[i]==0)
        zero_count++;
     else if(ar[i]>0)
        positive_count++;
     else
        negative_count++;
    i++;
    }
    document.write("<br>"+" Total Number of positive elements are
    document.write(" Total Number of negative elements are
    document.write(" Total Number of zero elements are "+zero_count+"<br/>");
    </script>
    </body>
</html>
```

Displaying the ten numbers

Computing the count for zero, positive and negative numbers

4.10.3 Two Dimensional Array

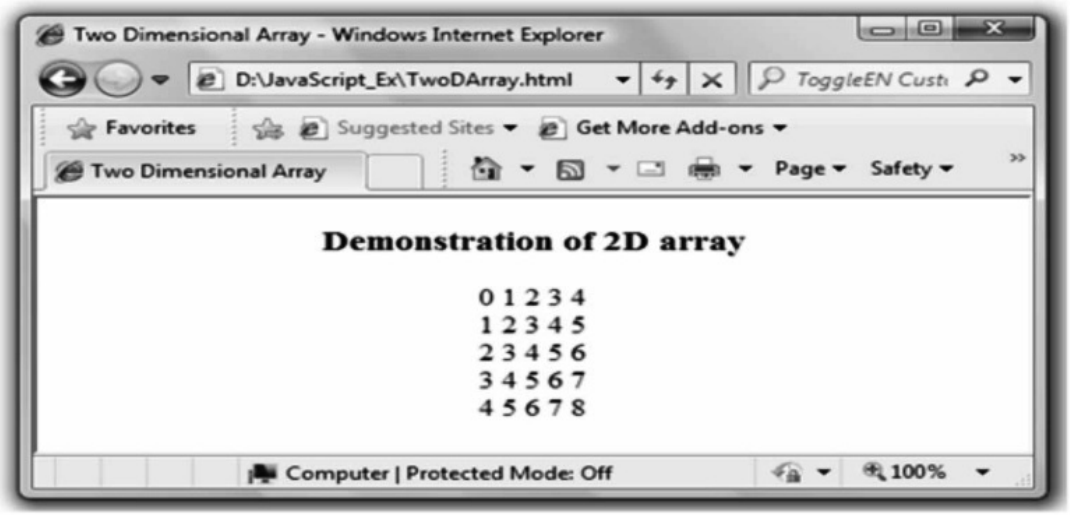
Actually JavaScript does not support the multidimensional arrays. Hence for defining the 2D array we make use of single dimensional array, how? Here it is -

JavaScript[TwoDArray.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Two Dimensional Array</title>
</head>
<body>
 <center>
 <h3> Demonstration of 2D array</h3>
 <script type="text/javascript">
   a=new Array();//creation of rows of array
   for(i=0;i<5;i++)
   a[i]=new Array();//creating columns of array
   for(i=0;i<5;i++)
   {
```

```
    for(j=0;j<5;j++)
    {
     a[i][j]=i+j;
     document.write(a[i][j] +"  ");
    }
    document.write("<br>");
   }
  </script>
</center>
</body>
</html>
```

Output



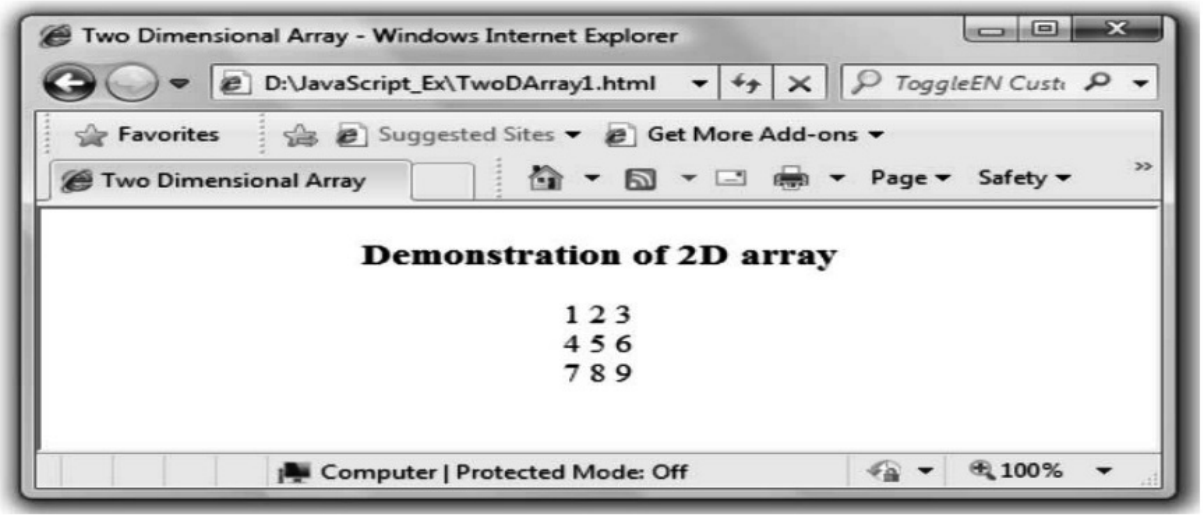
There is another way by which the two dimensional array can be initialized. This method is represented in the following Script

JavaScript[TwoDArray1.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Two Dimensional Array</title>
</head>
<body>
 <center>
 <h3> Demonstration of 2D array</h3>
 <script type="text/javascript">
  var a=[ [1,2,3],
          [4,5,6],
          [7,8,9]
        ]; //creation of array
  for(i=0;i<3;i++)
```

```
  {
  for(j=0;j<3;j++)
  {
   document.write(a[i][j] +"  ");
  }
   document.write("<br>");
  }
 </script>
</center>
</body>
</html>
```

Output



Ex. 4.10.2 : Use a one-dimensional array and write a script to solve following problem . Read in 20 numbers, each of which is between 1 and 100. As each number is read, print it only if it is not a duplicate of a number that has already been read.

AU : May-08, Marks 8

Sol. : duplicate.html

```
<html>
 <head>
 <script type="text/javascript">
 function change()
 {
  a=new Array(20);
  for(i=0;i<=19;i++)
  {
   var input_str=prompt("Enter the any number between 1 to 100","");
   var val=Number(input_str);
   a[i]=val;
  }
  for(i=0;i<=18;i++)
  {
```

```
    for(j=i+1;j<=19;j++)
    {
     if(a[i]>a[j])//before checking duplicate elements
     {
      temp=a[i]; //arrange them in sorted order
      a[i]=a[j];
      a[j]=temp;
     }
    }
   }
   document.write("The elements (without considering duplicate elements)are...");
   for(i=0;i<=18;i++)
   {
    flag=0;
    for(j=i+1;j<=19;j++)
    {
     if(a[i]==a[j])//checking if two adjacent elements are duplicate
     {
      a[j]=-1;//-1 is assigned to duplicate elements
      flag=1;//flag indicates a[i] and a[j] are duplicate
     }
    }
    if(flag==1)//flag=1 represents that a[i] element
     a[i]=-1;//needs to be marked as duplicate
   }
   for(i=0;i<=19;i++)
   {
    if(a[i]>0) //-1 means duplicate elements
     document.write(a[i]+",");//elements that are not -1 are unique elements
   }
  }
  </script>
 <title>Eliminating Duplicate Elements</title>
 </head>
 <body onload=change()>
 </body>
</html>
```

University Questions

1. *Explain the way in which javaScript handles arrays with example.* **AU : May-12, Marks 8**
2. *Explain the JavaScript array handling and array methods.* **AU : Dec.-13, Marks 8**

Part II : Introduction to DOM Model

4.11 Definition of DOM

- The **D**ocument **O**bject **M**odeling (DOM) is for defining the standard for accessing and manipulating HTML,XML and other scripting languages.
- It is the **W3C recommendation** for handling the structured documents.
- Normally the structured information is provided in XML, HTML and many other documents.
- Hence DOM provides the standard set of programming interfaces for working with XML and XHTML and JavaScript.

What is DOM?

Document Object Model (DOM) is a set of platform independent and language neutral Application Programming Interface (API) which describes how to access and manipulate the information stored in XML,HTML and JavaScript documents.

4.12 The Document Tree

AU : May-11,12, Dec.-13, Marks 8

- Basically DOM is an **Application Programming Interface (API)** that defines the interface between HTML document and application program. That means, suppose application program is written in Java and this Java program wants to access the elements of HTML web document then it is possible by using a set of Application Programming Interfaces (API) which belongs to the DOM.
- The DOM contains the **set of interfaces** for the document tree node type. These interfaces are similar to the Java or C++ interfaces. They have **objects, properties and methods** which are useful for respected node type of the web document.
- The documents in DOM are represented using a tree like structure in which every element is represented as a node. Hence the tree structure is also referred as DOM tree.
- **For example :** Consider following XHTML document.

```
<html>
    <head>
      <title>This is My Web Page </title>
    </head>
    <body>
      <h1>Hello Friends </h1>
      <h2>How are you?</h2>
      </h3>See you</h3>
    </body>
    </html>
```

- The DOM tree will be

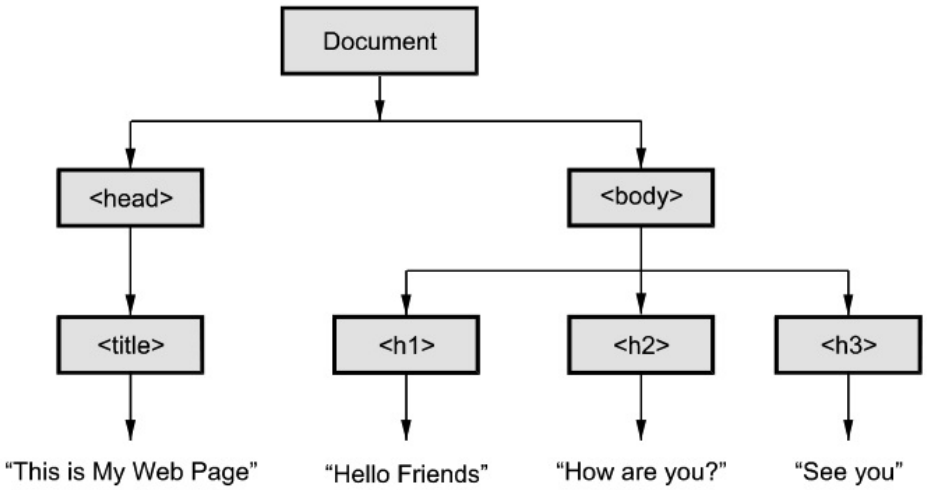


Fig. 4.12.1 DOM structure for simple web document

- We can describe some basic terminologies used in DOM tree as follows -
  1. Every element in the DOM tree is called **node**.
  2. The topmost single node in the DOM tree is called the **root**.
  3. Every child node must have a parent **node**.
  4. The bottommost nodes that have no children are called **leaf nodes**.
  5. The nodes that have the common parent are called **siblings**.

University Questions

1. *Explain about document tree in detail.* AU : May-11,12
2. *List and explain the various types of document nodes* AU : Dec.-13, Marks 8

4.13 Modifying Element Style

AU : Dec.-12, 15, Marks 16

We can access or change the contents of document by using various methods. Some commonly used properties and methods of DOM are as follows :

Methods

Method	Meaning
getElementById	This method is used to obtain the specific element which is specified by some id within the script.
createElement	This method is used to create an element node.
createTextNode	Useful for creating a text node.
createAttribute	Useful for creating attribute.
appendChild	For adding a new child to specified node, this method is used.

removeChild	For removing a child node of a specific node, this method is used.
getAttribute	This method is useful for returning the specified attribute value.
setAttribute	This method is useful for setting or changing the specified attribute to the specified value.

Properties

Property	Meaning
attributes	This property is used to get the attribute nodes of the node.
parentNode	This DOM property is useful for obtaining the parent node of the specific node.
childNodes	This DOM property is useful for obtaining the child nodes of the specific node.
innerHTML	It is useful for getting the text value of a node.

4.13.1 Accessing Elements using DOM

- There are several ways by which we can access the elements of the web document.
- To understand these methods of accessing we will consider one simple web document as follows.

```
<html >
<head>
    <title>This is My Web Page </title>
</head>
<body>
<form name="form1">
     <input type="text" name="myinput"/>
</form>
</body>
</html>
```

Method 1

- Every XHTML document element is associated with some address. This address is called **DOM address**.
- The document has the collection of **forms** and **elements**. Hence we can refer the text box element as

```
var Dom_Obj=document.forms[0].elements[0];
```

- But this is not the suitable method of addressing the elements. Because if we change the above script as

…

```
<form name="form1">
    <input type="button" name= "mybutton"/>
    <input type="text" name="myinput"/>
</form>
…
```

then index reference gets changed. Hence another approach of accessing the elements is developed.

Method 2

- We can access the desired element from the web document using JavaScript method **getElementById**. The element access can be made as follows -

```
var Dom_Obj=document.getElementById("myinput");
```

- But if the element is in particular group, that means if there are certain elements on the form such as radio buttons or check boxes then they normally appear in the groups. Hence to access these elements we make use of its index. Consider the following code sample

```
<form id="Food">
    <input type="checkbox" name="vegetables" value="Spinach" />Spinach
    <input type="checkbox" name="vegetables" value="FenuGreek" />FenuGreek
    <input type="checkbox" name="vegetables" value="Cabbage" />Cabbage
</form>
```

- For getting the values of these checkboxes we can write following code.

```
var Dom_obj=document.getElementById("Food");
for(i = 0 ; i < Dom_Obj.vegetables.length ; i++)
    document.write(Dom_Obj.vegetables[i]+"<br/>");
```

4.13.2 Modifying Elements using DOM

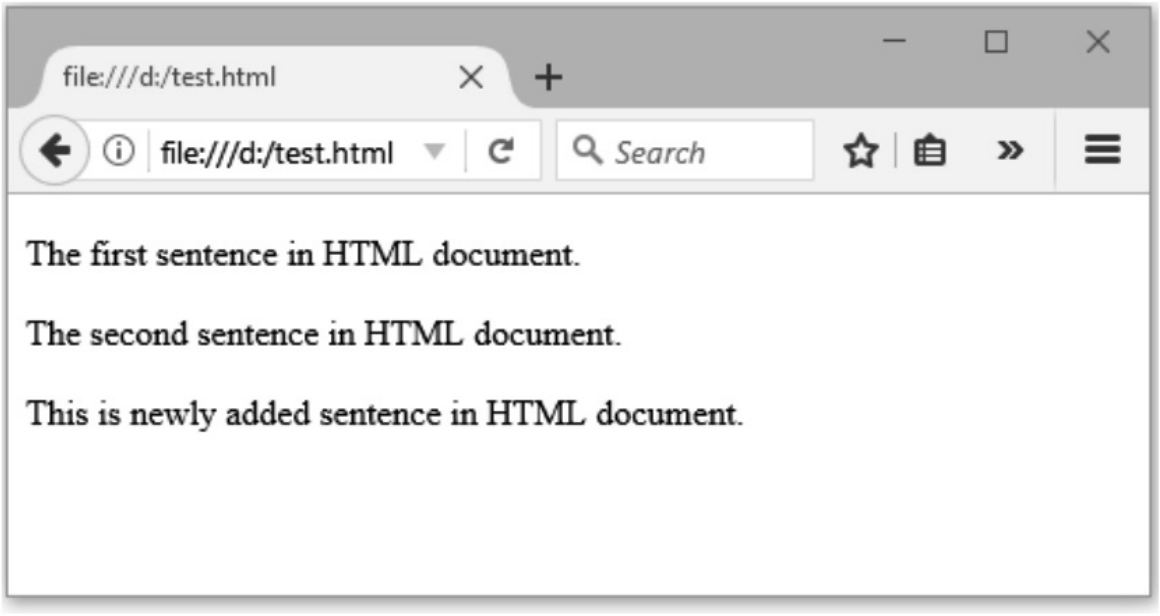
Appending an element Using DOM

We can add new node in the HTML document using the DOM method **appendChild** method

Following example illustrates the idea

```
<!DOCTYPE html>
<html>
<body>

<div id="div1">
<p id="p1">The first sentence in HTML document.</p>
<p id="p2">The second sentence in HTML document.</p>
</div>
```

```
<script>
var pnode = document.createElement("p");
var node = document.createTextNode("This is newly added sentence in HTML document.");
pnode.appendChild(node);
var element = document.getElementById("div1");
element.appendChild(pnode);
</script>
</body>
</html>
```



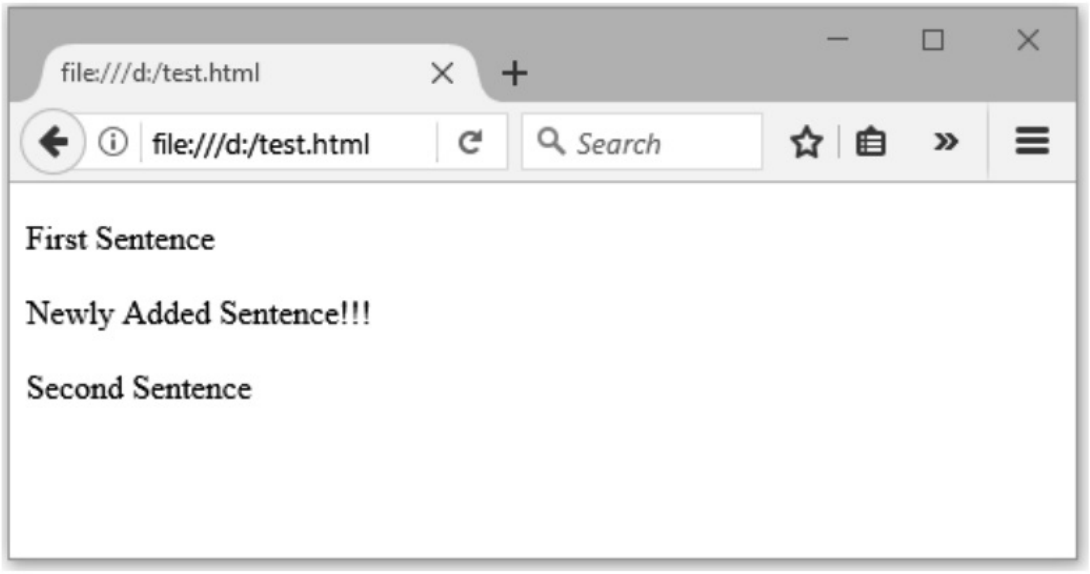
Inserting an element Using DOM

We can insert a node in between the nodes by using appendChild and insertBefore method. For example

```
<!DOCTYPE html>
<html>
<body>

<div id="div1">
<p id="p1">First Sentence</p>
<p id="p2">Second Sentence</p>
</div>

<script>
var pnode = document.createElement("p");
var node = document.createTextNode("Newly Added Sentence!!!");
pnode.appendChild(node);
var element = document.getElementById("div1");
```

```
var nextnode = document.getElementById("p2");
element.insertBefore(pnode,nextnode);
</script>

</body>
</html>
```

Output



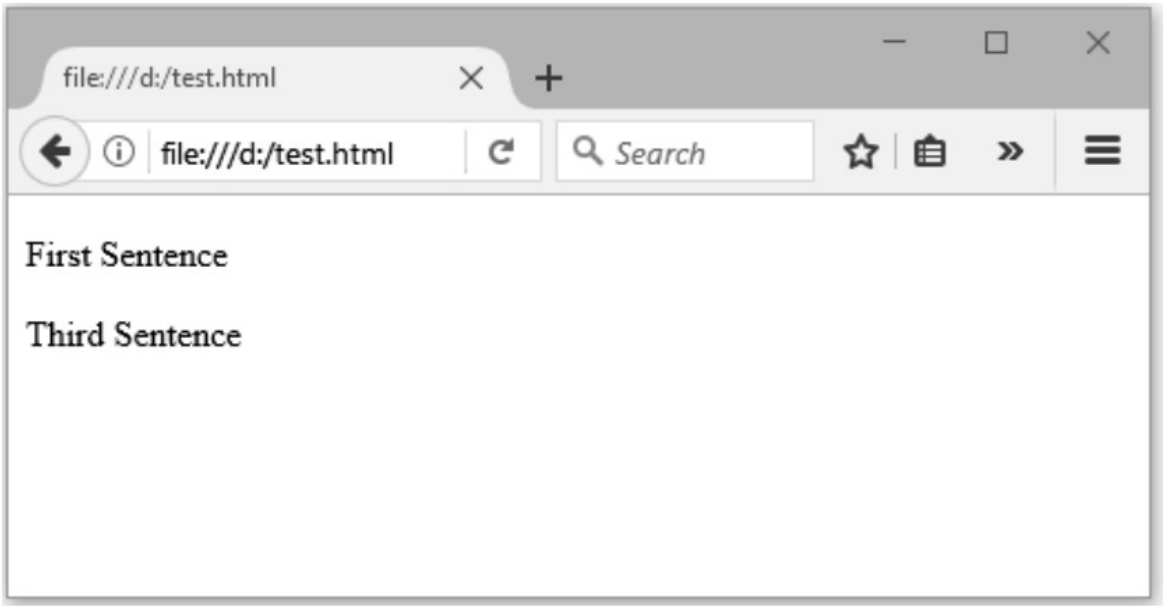
Removing an element Using DOM

We can remove or delete particular from HTML document using DOM's **removeChild** method.

Here is the illustration

```
<!DOCTYPE html>
<html>
<body>

<div id="div1">
<p id="p1">First Sentence</p>
<p id="p2">Second Sentence</p>
<p id="p3">Third Sentence</p>
</div>

<script>
var parentnode = document.getElementById("div1");
var node = document.getElementById("p2");
parentnode.removeChild(node);
</script>
</body>
</html>
```

Output

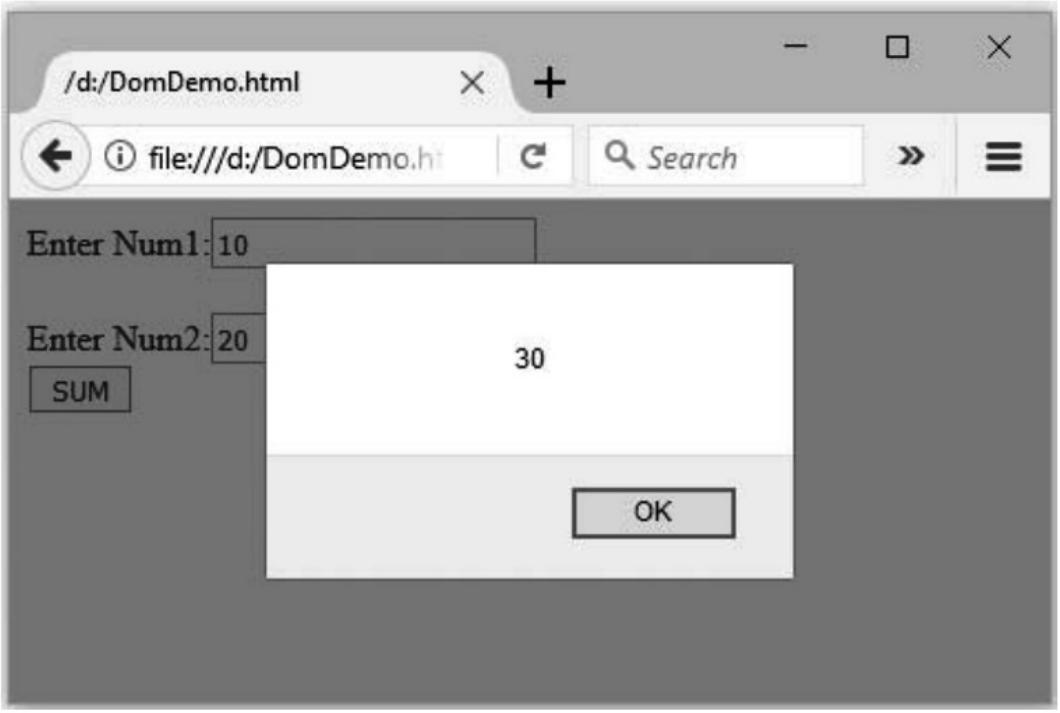


Ex. 4.13.1 : Write a JavaScript to display sum of two elements. Make use of appropriate DOM method.

Sol. :

```
<!Doctype html>
<html>
<head>
<script type="text/javascript">
function getSum()
{
    var number1=document.getElementById("number1").value;
    var number2=document.getElementById("number2").value;
    var num1=Number(number1);
var num2=Number(number2);
    alert(num1+num2);
}
</script>
</head>
<body>
<form>
Enter Num1:<input type="text" id="number1" /><br/> <br/>
Enter Num2:<input type="text" id="number2" /><br/>
<input type="button" value="SUM" onclick="getSum()"/>
</form>
</body>
</html>
```

Output



Ex. 4.13.2 : Write a JavaScript to welcome the user by his/her name when he enters his/her name in textbox. Make use of appropriate DOM method.

Sol. :

```
<!Doctype html>
<html>
<head>
<script type="text/javascript">
function display()
{
    var name=document.getElementsByName("user");
alert("Welcome "+name[0].value);
}
</script>
</head>
<body>
<form>
Enter Name:<input type="text" name="user" /><br/> <br/>
<input type="button" value="Greet" onclick="display()"/>
</form>
</body>
</html>
```

Output

University Questions

1. *With a simple example illustrate how the elements of the HTML document tree structure can be accessed using Javascript.* **AU : Dec.-12, Marks 16**
2. *Explain DOM model.* **AU : Dec.-15, Marks 8**

4.14 Objects in JavaScript

AU : Dec.-09, 11, 13, Marks 16

In JavaScript object is a collection of properties. These properties are nothing but the members of the classes from Java or C++. For instance - in JavaScript the object Date() is used which happens to the member of the class in Java.

4.14.1 Math Objects

- For performing the mathematical computations there are some useful methods available from **math** object.
- For example if we want to find out minimum of two numbers then we can write -

```
document.write(math.min(4.5,7.8));
```

The above statement will result in 4.5. Thus using various useful methods we can perform many mathematical computations.

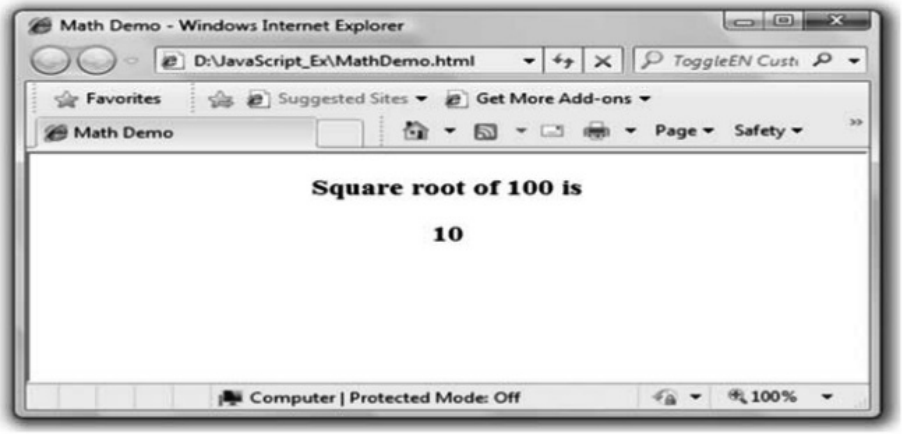
- Here are some commonly used methods from **math** object.

Method	Meaning
sqrt(num)	This method finds the square root of num.
abs(num)	This method returns absolute value of num.
ceil(num)	This method returns the ceil value of num. For example ceil(10.3) will return 11.
floor(num)	This method returns the floor value of num. For example floor(10.3) will return 10.
log(num)	This method returns the natural logarithmic value of num. For example log(7.9) will return 2.
pow(a,b)	This method will compute the ab. For example pow(2,5) will return 32.
min(a,b)	Returns the minimum value of a and b.
max(a,b)	Returns the maximum value of a and b.
sin(num)	Returns the sine of num.
cos(num)	Returns the cosine of num.
tan(num)	Returns the tangent of num.
exp(num)	Returns the exponential value i.e. enum .

JavaScript Program[MathDemo.html]

```
<!DOCTYPE html>
<html >
 <head>
  <title>Math Demo</title>
</head>
<body>
  <center>
   <h3>Square root of 100 is </h3>
   <script type="text/javascript">
    var num=100;
    document.write("<h3>"+Math.sqrt(num)+"</h3>");
   </script>
  </center>
</body>
</html>
```

Output



4.14.2 Number Objects

Various properties of number object are -

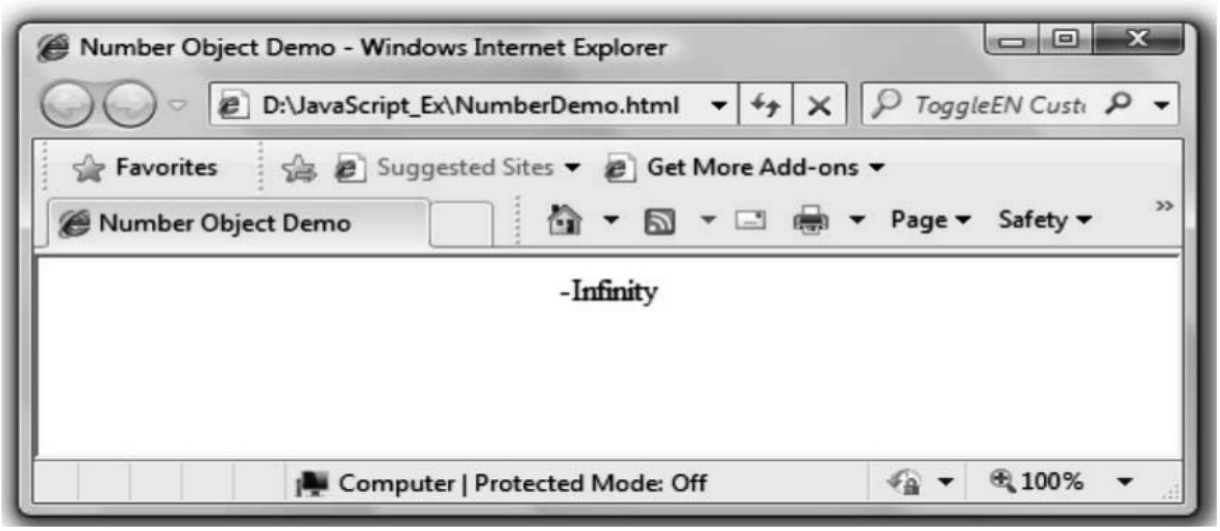
Property	Meaning
MAX_VALUE	Largest possible number gets displayed.
MIN_VALUE	Smallest possible number gets displayed.
NaN	When not a number then NaN is displayed.
PI	The value of PI gets displayed.
POSITIVE_INFINITY	The positive infinity gets displayed.
NEGATIVE_INFINITY	The negative infinity gets displayed.

Using **Number**.*property_name* we can display the property value. Following JavaScript uses the property of negative infinity.

JavaScript[NumberDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Number Object Demo</title>
</head>
<body>
 <center>
  <script type="text/javascript">
  document.write(Number.NEGATIVE_INFINITY);
  </script>
 </center>
</body>
</html>
```

Output

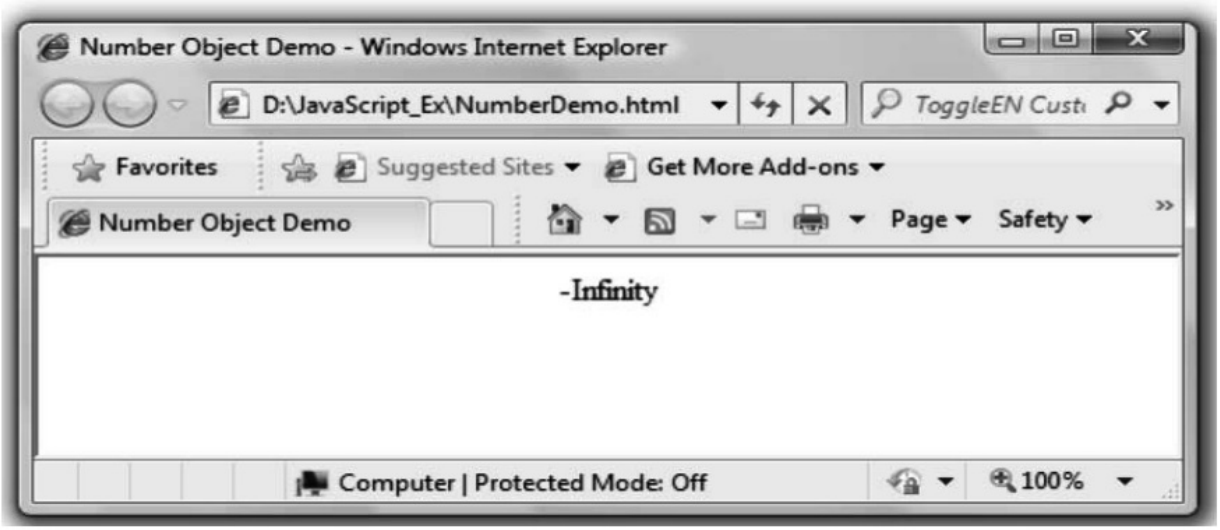


4.14.3 Date Objects

- This object is used for obtaining the date and time.
- This date and time value is based on computer's local time (system's time) or it can be based on GMT(**G**reenwich **M**ean **T**ime).
- Nowadays this GMT is also known as UTC i.e. **U**niversal **C**o-ordinated **T**ime. This is basically a world time standard.
- Following are **t**he commonly used methods of Date object.

Method	Meaning
getTime()	It returns the number of milliseconds. This value is the difference between the current time and the time value from 1st January 1970.
getDate()	Returns the current date based on computers local time.
getUTCDate()	Returns the current date obtained from UTC.
getDay()	Returns the current day. The day number is from 0 to 6 i.e. from Sunday to Saturday.
getUTCDay()	Returns the current day based on UTC. The day number is from 0 to 6 i.e. from Sunday to Saturday.
getHours()	Returns the hour value ranging from 0 to 23, based on local time.
getUTCHours()	Returns the hour value ranging from 0 to 23, based on UTC timing zone.
getMilliseconds()	Returns the milliseconds value ranging from 0 to 999, based on local time.

Output



4.14.3 Date Objects

- This object is used for obtaining the date and time.
- This date and time value is based on computer's local time (system's time) or it can be based on GMT(**G**reenwich **M**ean **T**ime).
- Nowadays this GMT is also known as UTC i.e. **U**niversal **C**o-ordinated **T**ime. This is basically a world time standard.
- Following are **t**he commonly used methods of Date object.

Method	Meaning
getTime()	It returns the number of milliseconds. This value is the difference between the current time and the time value from 1st January 1970.
getDate()	Returns the current date based on computers local time.
getUTCDate()	Returns the current date obtained from UTC.
getDay()	Returns the current day. The day number is from 0 to 6 i.e. from Sunday to Saturday.
getUTCDay()	Returns the current day based on UTC. The day number is from 0 to 6 i.e. from Sunday to Saturday.
getHours()	Returns the hour value ranging from 0 to 23, based on local time.
getUTCHours()	Returns the hour value ranging from 0 to 23, based on UTC timing zone.
getMilliseconds()	Returns the milliseconds value ranging from 0 to 999, based on local time.

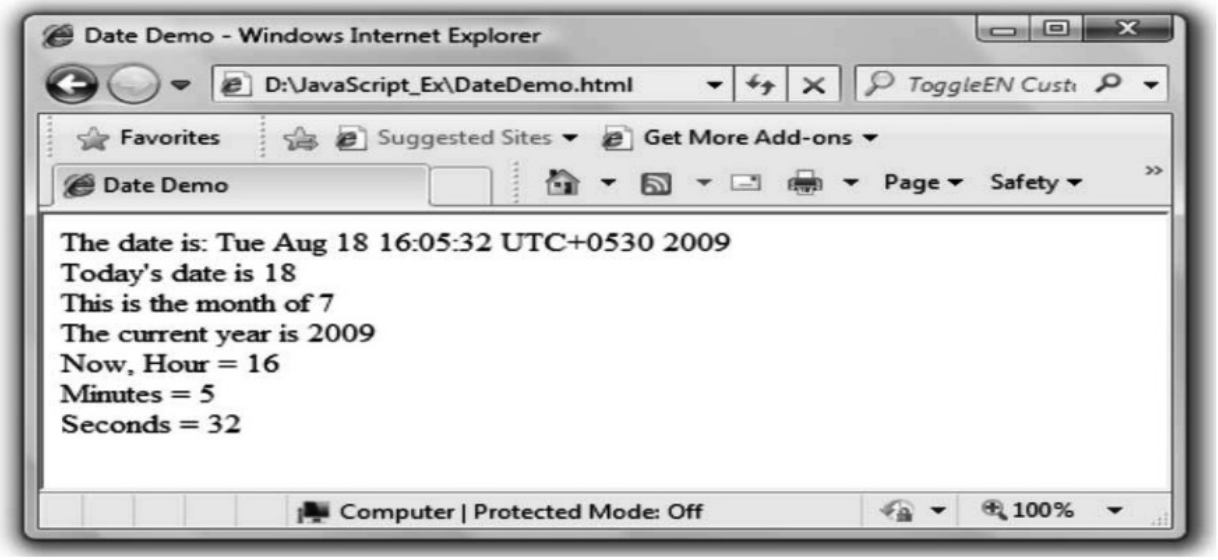
getUTCMilliseconds()	Returns the milliseconds value ranging from 0 to 999, based on UTC timing zone.
getMinutes()	Returns the minute value ranging from 0 to 59, based on local time.
getUTCMinutes()	Returns the minute value ranging from 0 to 59, based on UTC timing zone.
getSeconds()	Returns the second value ranging from 0 to 59, based on local time.
getUTCSeconds()	Returns the second value ranging from 0 to 59, based on UTC timing zone.
setDate(value)	This function helps to set the desired date using local timing or UTC timing zone.
setHour(hr,minute,second,ms)	This function helps to set the desired time using local or UTC timing zone. The parameters that can be passed to this function are hour,minute,seconds and milliseconds. Only hour parameter is compulsory and rest all are the optional parameters.

In the following web document we are making use of **Date()** object and some useful methods of it.

JavaScript[DateDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Date Demo</title>
</head>
<body>
   <script type="text/javascript">
    var my_date=new Date();
    document.write("The date is: "+my_date.toString()+"<br>");
    document.write("Today's date is "+my_date.getDate()+"<br>");
    document.write("This is the month of  "+my_date.getMonth()+"<br>");
    document.write("The current year is "+my_date.getFullYear()+"<br>");
    document.write("Now, Hour = "+my_date.getHours()+"<br>");
    document.write("Minutes = "+my_date.getMinutes()+"<br>");
    document.write("Seconds = "+my_date.getSeconds()+"<br>");
   </script>
</body>
</html>
```

Output



Script Explanation :

1. In the above script we have created an instance *my_date* of the object **Date().**
2. Then using **my_date.toString()** method we can display the current date along with the time.
3. Then using the functions like getDate(), getMonth(), getFullYear() we are displaying the date, month and year separately.
4. Similarly using the functions getHours(), getMinutes() and getSeconds() we can display the current hour, minute and seconds separately.

4.14.4 Boolean Objects

- This object is the simplest kind of object which is used especially when we want to represent **true** and **false** values.
- Here is a simple javascript in which the Boolean type variable is used -

Javascript Program

```
<html>
<body>
<script type="text/javascript">
var temp=new Boolean(false);
document.write("<b>"+"The boolean value is:   ");
document.write(temp.toString());
</script>
</body>
</html>
```

Output

4.14.5 String Objects

- String is a collection of characters.
- In JavaScript using **string** object many useful string related functionalities can be exposed off.
- Some commonly used methods of string object are concatenating two strings, converting the string to upper case or lower case, finding the substring of a given string and so on.
- Here is a listing of some methods of string.

Method	Meaning
concat(str)	This method concatenates the two strings. For example s1.concat(s2) will result in concatenation of string s1 with s2.
charAt(index_val)	This method will return the character specified by value index_val.
substring(begin,end)	This method returns the substring specified by begin and end character.
toLowerCase()	This function is used to convert all the uppercase letters to lower case.
toUpperCase()	This function is used to convert all the lowercase letters to upper case.
valueOf()	This method returns the value of the string.

There is one important property of string object and that is **length**. For example

```
var my_str="Hello";
 var len;
 len=my_str.length;
```

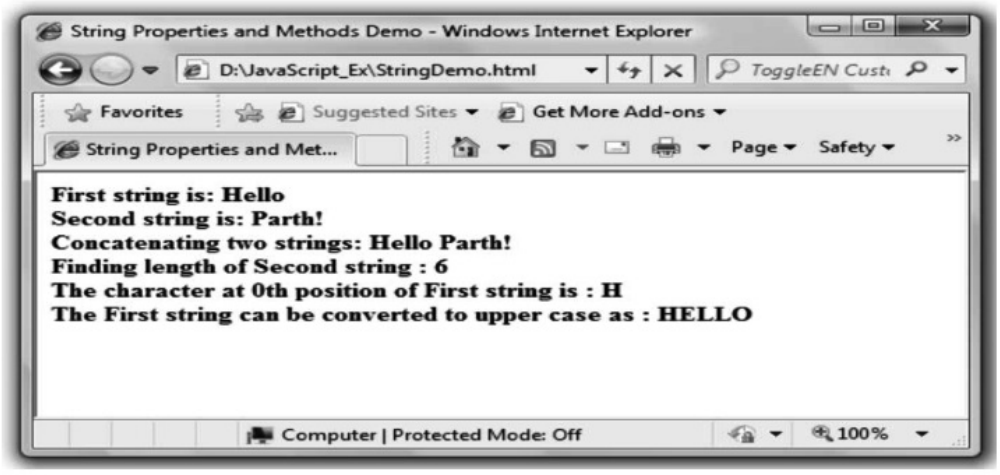
Length of the string "Hello" will be stored in the variable len

Here is sample program in which some methods of string object are used.

JavaScript[StringDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>String Properties and Methods Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
    var s1="Hello ";
    var s2="Parth!";
    document.write("First string is: "+s1+"<br>");
    document.write("Second string is: "+s2+"<br>");
    document.write("Concatenating two strings: "+s1.concat(s2)+"<br>");
    document.write("Finding length of Second string : "+s2.length+"<br>");
    document.write("The character at 0th position of First string is : "+s1.charAt(0)+"<br>");
    document.write("The First string can be converted to upper case as : "+s1.toUpperCase());
  </script>
 </strong>
</body>
</html>
```

Output



Ex. 4.14.1 : Write a script that inputs a line of text, tokenizes it with String method split and displays the tokens in reverse order.

Sol. :

```
<html>
<body>
<script type="text/javascript">
```

```
var str=new String("I like writing in javascript");
var a=new Array();
a=str.split(' ');
document.write("<strong>The original string is</strong>"+"<br/>");
for(i=0;i<a.length;i++)
document.write(a[i]+" ");
document.write("<br/>");
document.write("<strong>The reversed string is</strong>"+"<br/>");
for(i=a.length-1;i>=0;i--)
document.write(a[i]+" ");
</script>
</body>
</html>
```

4.14.6 Object Creation and Modification

- The web designer can create an object and can set its properties as per his requirements.
- The object can be created using new expression.
- Initially the object with no properties can be set using following statements

  Myobj=new Object();
- Then using dot operator we can set the properties for that object.

- The object can then be modified by assigning the values to this object.

JavaScript[ObjectDemo.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Object Creation Demo</title>
</head>
<body>
<strong>
  <script type="text/javascript">
    var Myobj;
    var n,i;
    //creating an object
    Myobj=new Object();
    //setting the propertis for newly created object
    Myobj.id=10;
    Myobj.name="Anuradha";
    document.write("The ID is "+Myobj.id+"<br/>");
    document.write("The Name is "+Myobj.name);
  </script>
 </strong>
</body>
</html>
```

Output

Alternative method of creating an object and modifying the properties is as given below -

```
var Myobj={name:"Anuradha",id:10};
```

Then using **document.write** statement we can display the property values as follows -

```
document.write("The ID is "+Myobj.id+"<br/>");
document.write("The Name is "+Myobj.name);
```

The property of created object can be deleted using an expression **delete**. Normally the property of an object is deleted in order to free the allocated memory so that this memory can be reused by other process.

University Questions

1. *Discuss javascript array object in detail.* AU : Dec.-11, Marks 8
2. *Explain the javascript object Math.* AU : Dec.-13, Marks 8
3. *What are the methods associated with array object in JavaScript ? Explain each one with example.* AU : Dec.-09, Marks 16

4.15 Regular Expressions

AU : May-08, Marks 8

- **Definition :** Regular Expression is a special text string that defines the search pattern. It is a logical expression.
- **For example** – For counting specific characters in a string or to replace some substring by another substring we need to create a regular expression.
- We can create a regular expression pattern using forward slash /. For instance -

  re = /abc/
- Regular expression is a powerful way for searching and replacing the characters in the string.
- The words of regular expression are called **special characters**.
- Various special characters that can be used in Regular expression along with their meanings are written in the following table :

Special Character	Meaning
.	Any character except newline
A	The character a
ab	The string ab
a|b	a or b
a*	0 or more a's
\	Escapes a special character
[ab-d]	One character of: a, b, c, d
[^ab-d]	One character except: a, b, c, d

[\b]	Backspace character
\d	One digit
\D	One non-digit
\s	One whitespace
\S	One non-whitespace
\w	One word character
\W	One non-word character
*	0 or more
+	1 or more
?	0 or 1
{2}	Exactly 2
{2, 5}	Between 2 and 5
{2,}	2 or more
(…)	Group of pattern
^	Start of string
$	End of string
\b	Word boundary
\n	Newline
\r	Carriage return
\t	Tab
\0	Null character

Methods that use regular expressions

Method	Description
exec	A RegExp method that executes a search for a match in a string. It returns an array of information or null on a mismatch.
test	A RegExp method that tests for a match in a string. It returns true or false.

match	A String method that executes a search for a match in a string. It returns an array of information or null on a mismatch.
matchAll	A String method that returns an iterator containing all of the matches, including capturing groups.
search	A String method that tests for a match in a string. It returns the index of the match, or –1 if the search fails.
replace	A String method that executes a search for a match in a string, and replaces the matched substring with a replacement substring.
split	A String method that uses a regular expression or a fixed string to break a string into an array of substrings.

4.15.1 Finding Non Matching Characters

We can find the non matching characters from the given text by placing ^ as the first character within a square [ ].

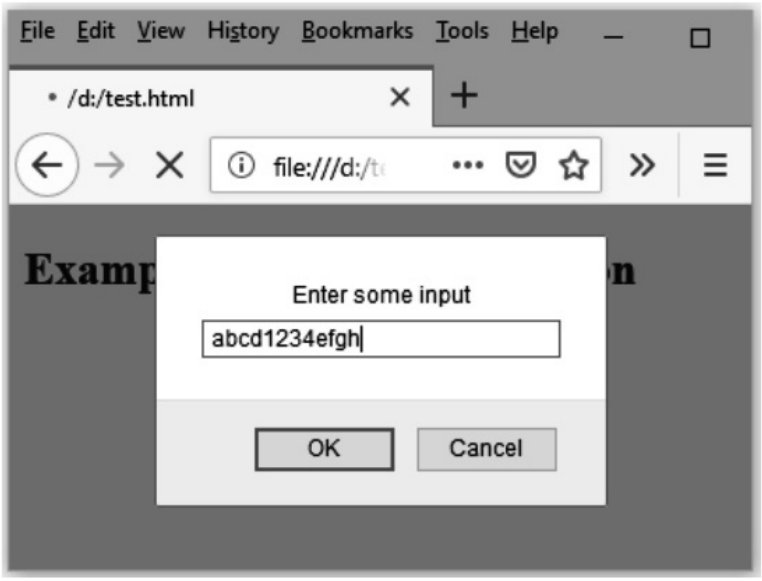
Ex. 4.15.1 : Write a Java program that checks whether the string entered by the user contains digit or not.

Sol. :

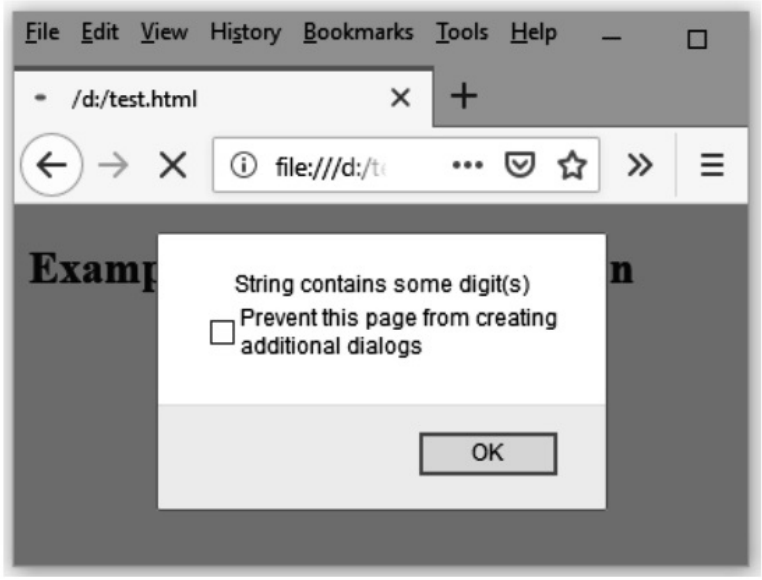
```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
        function TestString(str)
        {
           re=/[^[0-9]]/; // [0-9] indicates any digit
           if(re.test(str))
            {
              alert("The string does not contain
             any digit");
           }
              else
           {
              alert("String contains some digit(s)");
```

```
        }
    }
    </script>
</head>
<body>
<h2>Example of Regular Expression</h2>
<script type="text/javascript">
    var input_str=prompt("Enter some input","");
    TestString(input_str);
</script>
</body>
</html>
```

Output



Click OK button and you will get



4.15.2 Entering the Range of Characters

- For matching any digit we need not have to enter every digit right from 0 to 9. Similarly for matching letters we need not have to test with every single alphabet. We can achieve this by entering range of characters.
- For example – To match a digit we must have a regular expression as [0-9]. Thus placing the range within a square bracket helps us to evaluate a complete range of set of characters.
- Suppose we enter [j-s] then that means match the characters j, k, l, m, n, o, p, q, r and s.

4.15.3 Matching Digits and Non Digits

- Determining whether the string contains digits or non digits is a common task in any search pattern. For instance – in the application of validating telephone number this is the most wanted task.
- This can be simplified by JavaScript by writing the regular expression.
- If we write \d then that means search the text for digits and if we write \D then that means search the text for non-digits.

4.15.4 Matching Punctuations and Symbols

- The \w special symbol tells the browser to determine whether the text contains a letter, number, or an underscore.
- The \W special symbol tells the browser to determine whether the text contains other than a letter, number, or an underscore.
- Using \W is equivalent to using [a-zA-Z0-9_]. The last _ indicates space character.

4.15.5 Matching Words

- Any word in the text is defined as set of characters. A word is determined by a word boundary that is the space between two words.
- The boundary can be defined by using special symbol \b
- For example – From the string 'Boycott' we can get a match for 'cot' by using regular expression /\bcot\b/

4.15.6 Replacing a Text using Regular Expressions

Using **replace** function we can replace the desired pattern. The first parameter in the replace function is the string which is to be replaced and the second parameter is the replacing string.

For example- Consider following JavaScript in which the word 'country' is replaced by 'India'

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
      function TestString(str)
      {
        new_str=str.replace("country","India");
        document.write(new_str);
      }
    </script>
</head>
<body>
    <script type="text/javascript">
    str="I love my country";
    document.write(str);
    </script>
    <form name="form1">
      <input type="button" value="Replace"
      onclick="TestString(str)">
    </form>
</body>
</html>
```

Output

4.15.7 Returning a Matched Character

The **exec()** method searches string for text that matches with the regular expression. If it finds a match , it returns an array of results, otherwise it returns null.

If we want to search for particular pattern from a text then exec() method can be used as follows -

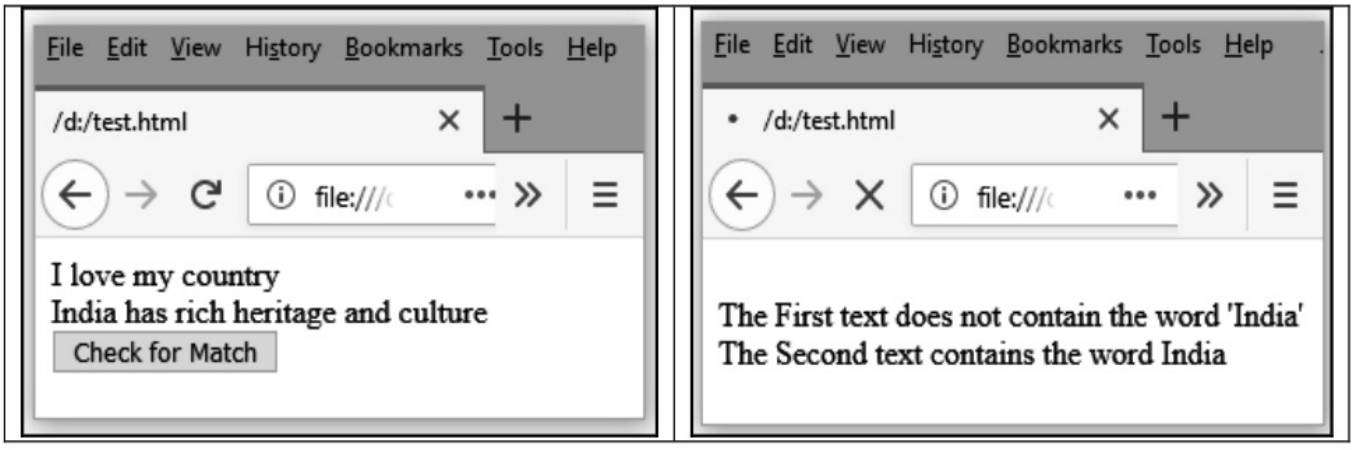
pattern.**exec(text)**

This function returns an array of matched result.

Example Program

```
<!DOCTYPE html>
<html>
<head>
    <script type="text/javascript">
      function TestString(str1,str2)
      {
        re=/India/g;
        result1=re.exec(str1);
        result2=re.exec(str2);
        if(result1)
        document.write("<br/>The First text contains the word "+result1);
        else
        document.write("<br/>The First text does not contain the word 'India' ");

        if(result2)
        document.write("<br/>The Second text contains the word "+result2);
        else
        document.write("<br/>The Second text does not contain the word 'India' ");
      }
</script>
</head>
<body>
<script type="text/javascript">
str1="I love my country";
str2="India has rich heritage and culture";
document.write(str1);
document.write("<br/>"+str2);
</script>
<form name="form1">
  <input type="button" value="Check for Match" onclick="TestString(str1,str2)">
</form>
</body>
</html>
```

Output



4.15.8 Regular Expression Object Properties

There are various regular expression object properties that help in matching particular word, character, last character, index at which to start the next match and so on. These properties are enlisted in the following table -

Regular Expression Object	Properties
$1 (through $9)	Parenthesized substring matches
$_	Same an input
$*	Same as multiline
$&	Same as lastMatch
$+	Same as lastParen
$`	Same as leftContent
$'	Same as rightContext
global	Search globally (g modifier in use)
ignoreCase	Search case-insensitive (i modifier in use)
input	The string to search if no string is passed
lastIndex	The index at which to start the next match
lastMatch	The last match characters
lastParen	The last parenthesized substring match
leftContext	The substring to the left of the most recent match
multiline	Whether strings are searched across multiple lines
prototype	Allows the addition of properties to all objects
rightContext	The substring to the right of the most recent match
source	The regular expression pattern itself

Example Program

```
<!DOCTYPE html>
<html>
<head>
    <title>Pattern Matching using Regular Expression </title>
    <script type="text/javascript">
      function TestString(str)
      {
      // for each Sunil - replace it with Mr. and then Sunil
      alert(str.replace(/Sunil/g, 'Mr.$&'));
      }
    </script>
</head>
<body>
```

```
    <script type="text/javascript">
      str= "Sunil kumar, Sunil Shetty,Sunil Kale";
      document.write(str);
    </script>
    <button onclick="TestString(str)">Change</button>
</body>
</html>
```

Output

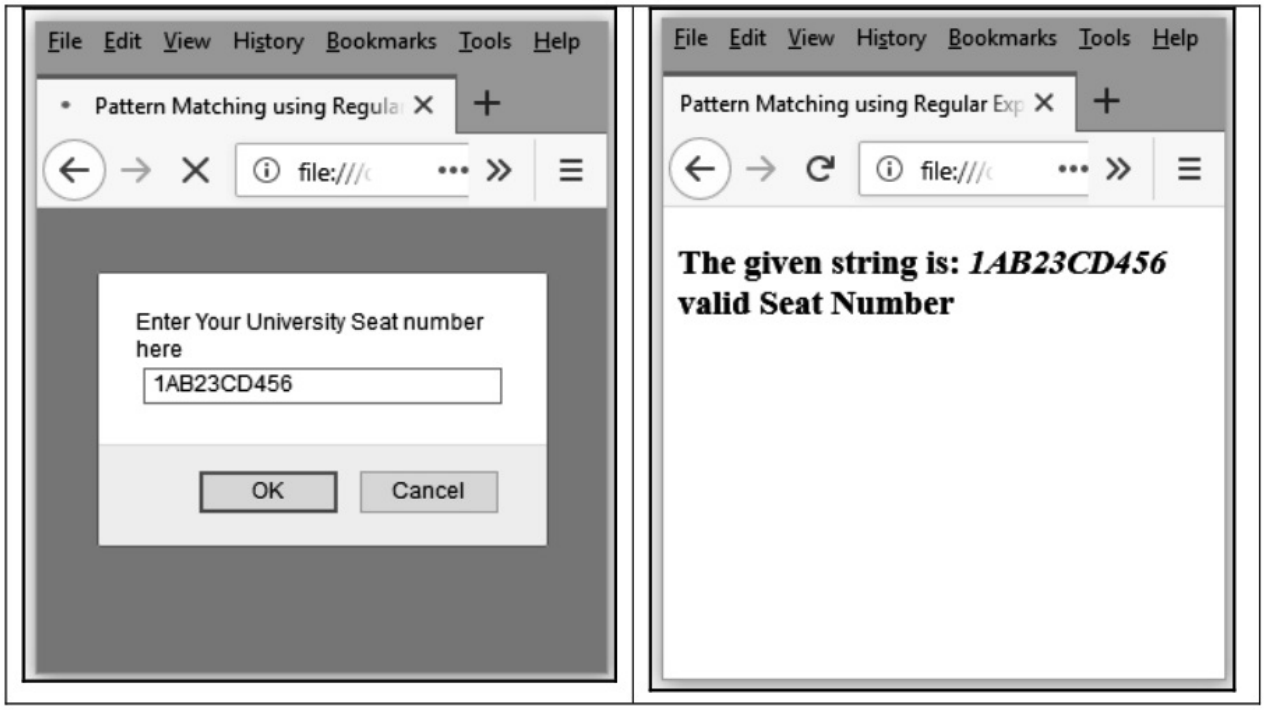
Ex. 4.15.2 : Develop and demonstrate, using Javascript document that collects USN(the valid format is : A digit from 1 to 4 followed by two upper-case characters followed by two digits followed by two upper case characters followed by three digits:no embedded spaces allowed) of the user. Event handler must be included for the form element that collects this information to validate the input. Message in the alert windows must be produced when errors are detected.

Sol. :

```
<!DOCTYPE html>
<html>
<head>
<title>Pattern Matching using Regular Expression </title>
<script type="text/javascript">
function TestString(str)
{
  document.write("The given string is: ");
  document.write("<em>"+str+"</em>"+"<br/>");
  var i=str.match(/[1234][A-Z]{2}\d{2}[A-Z]{2}\d{3}/);
```

```
  if(i==null)
   return false;
  else
  return true;
 }
</script>
</head>
<body>
  <h3>
   <script type="text/javascript">
     var input_str=prompt("Enter Your University Seat number here","");
     if(TestString(input_str))
       document.write("valid Seat Number");
     else
       document.write("Invalid Seat Number");
   </script>
   </h3>
</body>
</html>
```

Output



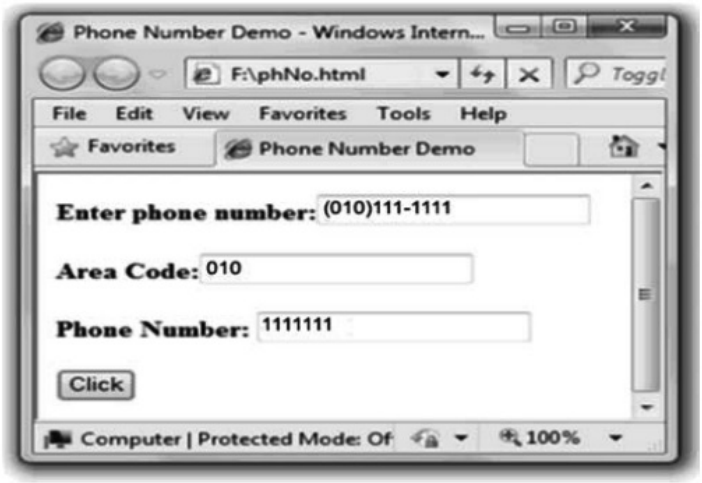
Ex. 4.15.3 : Write a Javascript that inputs a telephone number as a string in the form of (555) 5555555. The script should use string method split to extract the area code as a token, the first 3 digits of the phone number as a token, the last 4 digits of the phone number as a token. Display the area code in one text field and seven digit phone number in another text field.

AU : May-08, CSE, Marks 8

Sol. : HTML Document[phNo.html]

```
<html>
<head>
  <title>Phone Number Demo</title>
  <script type="text/javascript">
   function TestString()
   {
    var str=document.form1.input.value;
    var i=str.match(/"("\d{3}")"\" "\d{3}"-"\d{4}/);
    var temp1=str.split("(");
    var temp2=temp1[1].split(")")
    document.form1.area.value=temp2[0];
    var temp3=temp2[1].split(" ");
        var temp4=temp3[1].split("-")
    document.form1.phnum.value=temp4[0]+temp4[1];
    }
  </script>
</head>
 <body>
   <form name="form1">
    <b>Enter phone number:</b><input type="text" name="input" value=""><br/><br/>
    <b>Area Code:</b><input type="text" name="area" value=""><br/><br/>
    <b>Phone Number:</b> <input type="text" name="phnum" value=""><br/><br/>
    <input type="button" value="Click"onclick=TestString(input)>
 </body>
</html>
```

Output



4.16 Exception Handling

- **Definition:** Exception handling is a mechanism that handles the runtime errors so that the normal flow of the application can be maintained.

- Basically an exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.
- The exception handling in JavaScript can be performed with the help of following types of statements –
  1. The try statement lets you test a block of code for errors.
  2. The catch statement lets you handle the error.
  3. The throw statement lets you create custom errors.
  4. The finally statement lets you execute code, after try and catch, regardless of the result.
- Basic syntax of try-catch block is

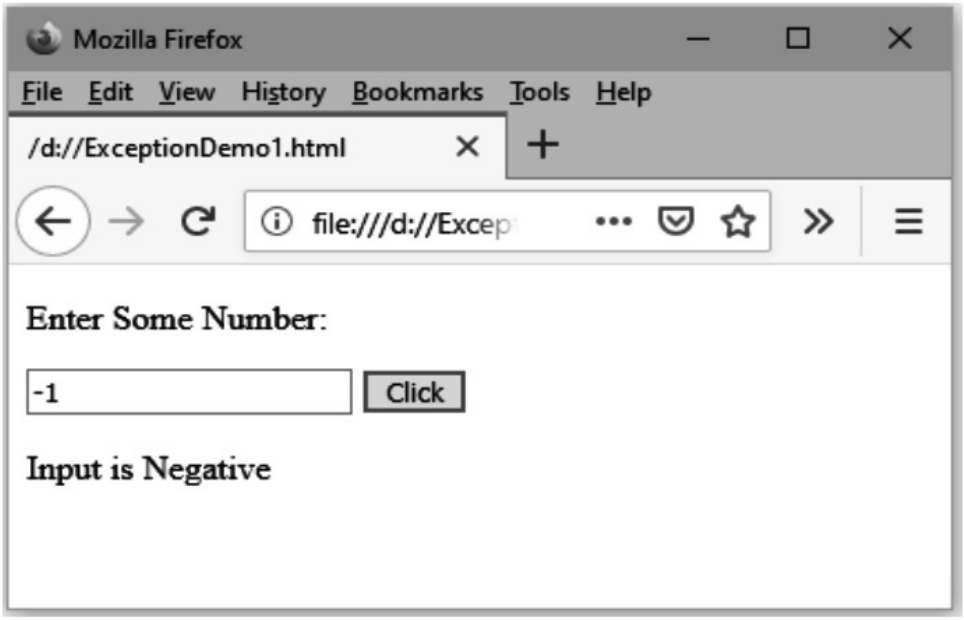
```
try {
 Block of code to try
}
catch(err) {
Block of code to handle errors
}
```

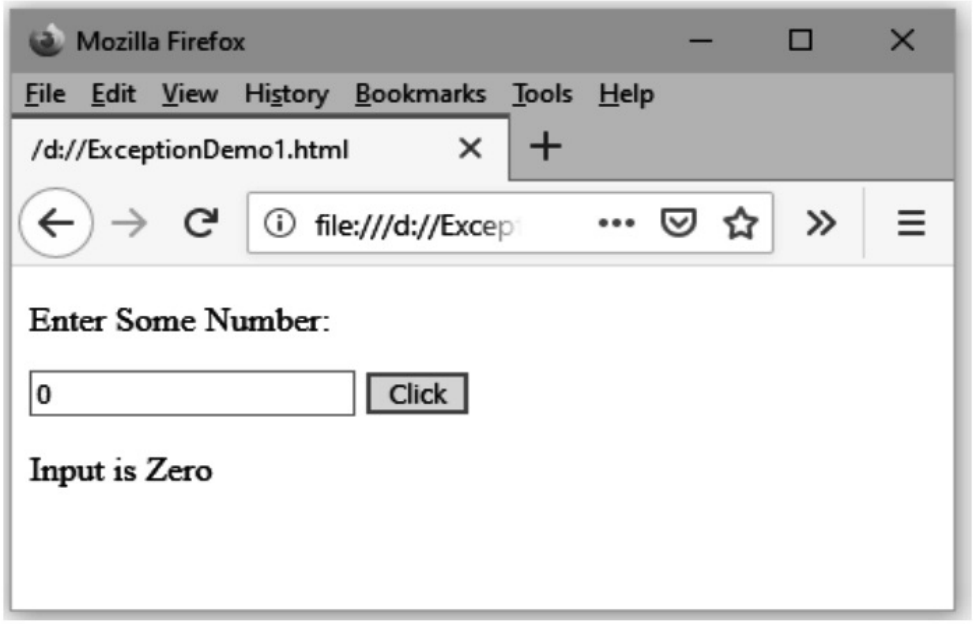
- When error occurs, JavaScript will normally throw an exception so that appropriate error message can be displayed and it will avoid a crash of a program.
- Following example illustrates the exception handling mechanism in JavaScript

```
<!DOCTYPE html>
<html>
<body>

<p>Enter Some Number:</p>

<input id="demo" type="text">
<button type="button" onclick="GetPositiveNumber()">Click</button>
<p id="myID"></p>

<script>
    function GetPositiveNumber() {
    var message, num;
    message = document.getElementById("myID");
    message.innerHTML = "";
    num = document.getElementById("demo").value;
    try {
          if(num == "")
            throw "empty";
          if(isNaN(num))
            throw "not a number";
          num = Number(num);
          if(num < 0)
            throw "Negative";
```

```
            if(num == 0)
              throw "Zero";
        }
        catch(err) {
            message.innerHTML = "Input is " + err;
        }
}
</script>
</body>
</html>
```

Output



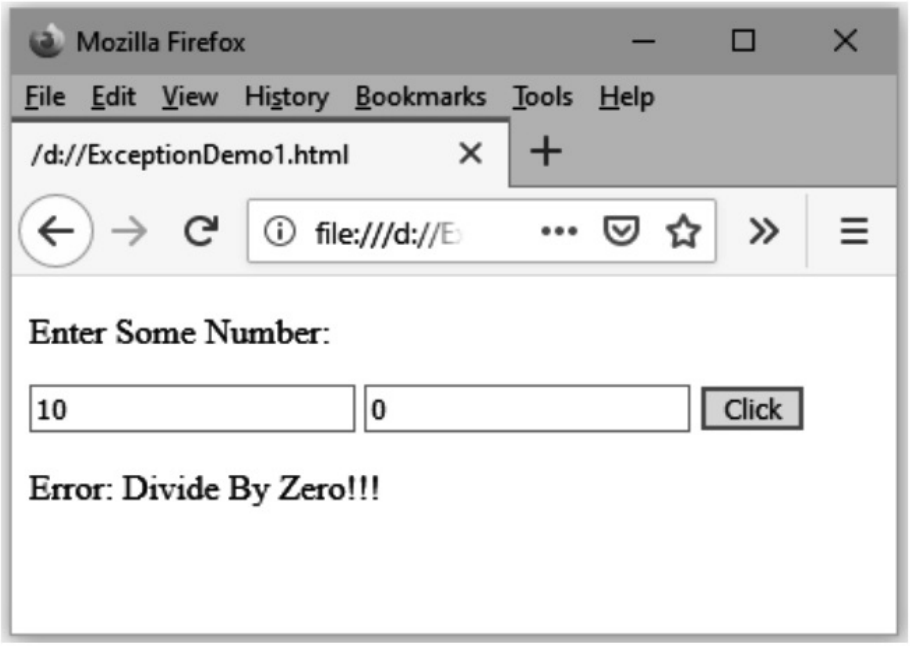


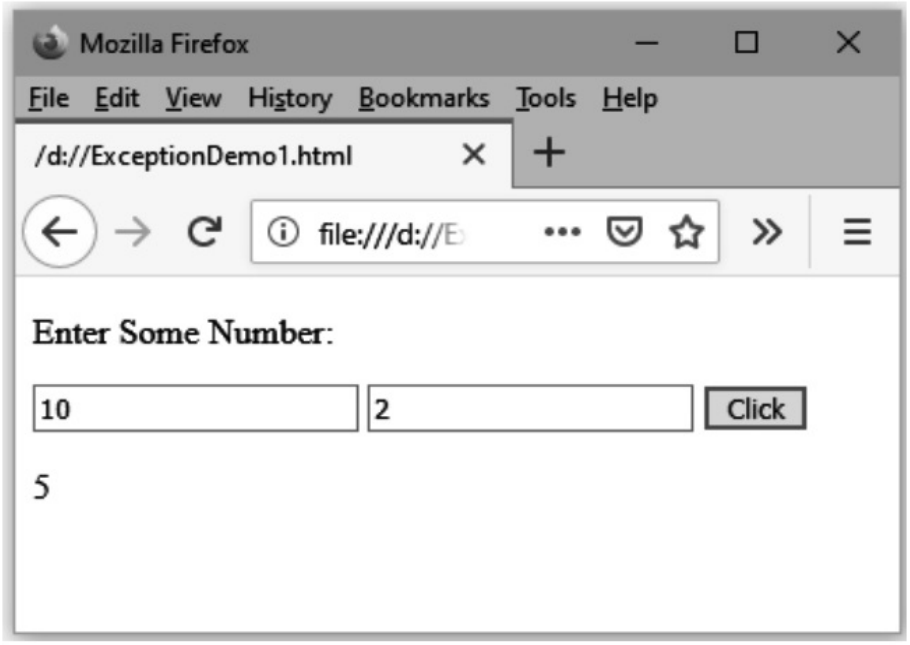
Ex. 4.16.1 : Write a JavaScript to handle divide by zero error using exception handling mechanism.

Sol. :

```
<!DOCTYPE html>
<html>
<body>

<p>Enter Some Number:</p>
<input id="numerator" type="text">
<input id="denominator" type="text">
<button type="button" onclick="divide()">Click</button>
<p id="myID"></p>

<script>
    function divide() {
    var n,result, message, num;
    message = document.getElementById("myID");
    message.innerHTML = "";
    d = document.getElementById("denominator").value;
    try {
        if(d == "")
           throw "empty";
        if(isNaN(d))
           throw "not a number";
        num = Number(d);
        if(num == 0)
           throw "Divide By Zero!!!";
        else
        {
           n = document.getElementById("numerator").value;//obtaining numerator
           n=Number(n);//converting text string to number
           ans=n/num;//performing division
           message.innerHTML = ""+ans;
        }

     }
     catch(err) {
        message.innerHTML = "Error: " + err;
    }
}
</script>
</body>
</html>
```

Output





4.17 Validation

AU : May-14, Marks 8

- Various control objects are placed on the form. These control objects are called **widgets.**
- These widgets used in JavaScript are – Textbox, Push button, Radio button, Checkbox and so on.
- In JavaScript the **validation of these widgets** is an important task.

  Let us understand the validation of form elements with the help of examples –

Ex. 4.17.1 : Write a JavaScript for password verification.

Sol. :

```
<!DOCTYPE html>
<html>
<head>
  <title>Demo of onlclick Tag Attibute</title>
<script type="text/javascript">
 function my_fun()
 {
  var mypwd=document.getElementById("pwd");
  var my_re_pwd=document.getElementById("re_pwd");
  if(mypwd.value=="")
  {
    alert("You have not entered the password");
    mypwd.focus();
    return false;
  }
  if(mypwd.value!=my_re_pwd.value)
  {
    alert("Password is not verified, Re-enter both the passwords");
    mypwd.focus();
    mypwd.select();
    return false;
  }
  else
  {
    alert("Congratulations!!!");
    return true;
  }
 }
</script>
</head>
<body>
  <center>
  <form id="form1">
    <label> Enter your password
    <input type="password" value="" id="pwd" />
    </label>
    <br/><br/>
    <label> Re-Enter the password
    <input type="password" value="" id="re_pwd" onblur="my_fun();"/>
    </label><br/>
    <input type="submit" value="Submit" name="submit" onsubmit="my_fun();"/>
     <input type="reset" value="Reset" name="reset"/><br/>
```

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

Output(Run1)

Output(Run2)

Output(Run3)

Ex. 4.17.2 : Write JavaScript to validate a for consisting of Name, Age, Address, EmailID, hobby (check box), Gender(radio box), country(Drop down menu).

Sol. : ApplicationForm.html

```
<html>
<head>
<title>The Student Registration Form</title>
<script type=text/javascript>
function validate()
{


      var i;
      var name_str=document.my_form.name;
      var phoneID=document.my_form.ph_txt;
      var ph_str=document.my_form.ph_txt.value;
      var str=document.my_form.Email_txt.value;
      if((name_str.value==null)||(name_str.value==""))
      {
      alert("Enter some name")
      return false
}
        if(document.my_form.Age_txt.value=="")// validating age
      {
      alert("Enter Some Age")
      return false
      }
            if((document.my_form.Age_txt.value<"5")&&(document.my_form.Age_txt.value>"21"))
      {
      alert("Invalid Age")
```

Validating Name

```
return false
}

if(ph_str.length<1 ||ph_str.length>11)
{
alert("Invalid length of Phone Number")
return false
}
for (i = 0; i < ph_str.length; i++)
{
      var ch = ph_str.charAt(i);
   if (((ch < "0") || (ch > "9"))){
alert("Invalid Phone Number")

phoneID.focus()
 return false
}
}

var index_at=str.indexOf("@")
var len=str.length
var index_dot=str.indexOf(".")
var emailID=document.my_form.Email_txt
if ((emailID.value==null)||(emailID.value==""))
{
alert("Please Enter your Email ID")
emailID.focus()
return false
}
if (str.indexOf("@")==-1)
{
  alert("Invalid E-mail ID")
  return false
}
  if (str.indexOf(".")==-1 || str.indexOf(".")==0
          || str.indexOf(".")==index_at)
{
   alert("Invalid E-mail ID")
   return false
}

 if (str.indexOf("@",(index_at+1))!=-1)
 {
   alert("Invalid E-mail ID")
   return false
 }
 if (str.indexOf(" ")!=-1)
```

Validating Phone Number

Validating Email ID

Validating Email ID

```
    {
        alert("Invalid E-mail ID")
        return false
     }
    if (!document.my_form.group1[0].checked && !document.my_form.group1[0].checked)
     {
        alert("Please Select Sex");
        return false;
     }
    if (!document.my_form.group1[0].checked && !document.my_form.group1[0].checked)
     {
        alert("Please Select Sex");
        return false;
     }
      return true

}
</script>
</head>
<body bgcolor=aqua>
<center><h3>Application Form</h3></center>
<form name=my_form onsubmit=validate()>
<strong>Name: </strong>
<input type=text name=name><br/>

<strong>Age: </strong>
<input type=text name=Age_txt><br/>

<strong>Phone No:</strong>
<input type=text name=ph_txt><br/>
<strong>Email: </strong>
<input type=text name=Email_txt><br/><br/>
<strong>Sex:  </strong>
<input type="radio" name="group1" value="Male">Male
<input type="radio" name="group1" value="Female">Female<br/><br/><br/>
<strong>Hoby: </strong>
<input type="checkbox" name ="option1" value="Singing">Singing<br/>
<input type="checkbox" name ="option1" value="Reading">Reading<br/>
<input type="checkbox" name ="option1" value="T.V.">Watching T.V<br/>
<br/><br/>
<strong>Country:</strong>
<select name="My_Menu">
<option value="India">India</option>
<option value="China">China</option>
```

```
<option value="Shrilanka">Shrilanka</option>
</select>
<center>
<input type=submit value=Submit></br>
</center>

</body>
</html>
```

Output

Ex. 4.17.3 : Design a HTML form for validation the users with fields user name and password and ok button which should receive the input from the user and responses as authorized or invalid user name or invalid password. AU : May-14, Marks 8

Sol. :

```
<head>
  <title>HTML FORM</title>
  <script type=text/javascript>
   function validate()
    {
     var user=document.form1.username;
     var pass=document.form1.password;
    if((user.value==null)||(user.value==""))
    {
     alert("Enter some user name");
```

```
        }
      if((pass.value==null)||(pass.value==""))
       {
        alert("Enter some password");

        }
        if(user.value=="admin")
        {
        alert("Valid user name");

        }
        if(pass.value=="password")
        {
        alert("Valid password");
        }
        else
        alert("Invalid username/password");
      }
   </script>
  </head>
  <body>
   <form id="form1" name="form1" onsubmit=validate()>
   <table width="510" border="0" align="center">
   <tr>   <td>Username:</td>
   <td><input type="text" name="username"/></td>
   </tr>
   <tr>    <td>Password</td>
   <td><input type="password" name="password"/></td>
   </tr>
   <tr>
   <td> </td>
   <td><input type="submit" name="button" value="OK"  /></td>
   </tr>
   </table>
   </form>
  </body>
 </html>
```

Ex. 4.17.4 : Write a Javascript to validate radio button, operator field and check box.

Sol. :

JavaScript Program

```
<html>
<head>
<script LANGUAGE="JavaScript">

function ValidateForm(form)
{

if ((form.gender[0].checked == false ) && ( form.gender[1].checked == false ))
 { alert ( "Please choose your Gender: Male or Female" );
  return false;
 }
 if ((form.status[0].checked == false ) && ( form.status[1].checked == false ))
 { alert ( "Please choose your choice: AC or non AC" );
  return false;
 }
 if ( form.course.selectedIndex == 0 )
 { alert ( "Please select Some Course." );
  return false;
 }
 return true;
}
</script>
</head>
<body>
<br>
<form>
 Your Gender: <input type="radio" name="gender" value="Male"> Male
 <input type="radio" name="gender" value="Female"> Female
 <br/>
 Choice: <input type="checkbox" name="status">AC
 <input type="checkbox" name="status"> Non AC
 <br/>
 Course:
 <select name="course">
  <option value="">Select an Option:</option>
  <option value="Computer">Computer</option>
  <option value="Mechanical">Mechanical</option>
  <option value="E&Tc">E&Tc</option>
 </select>
 <br/><br/>
<input type="button" name="SubmitButton" value="Submit"
 onClick="ValidateForm(this.form)">
 <input type="reset" value="Reset">
</form>
</body>
</html>
```

Validating Radio Button

Validating Checkbox

Validating Option

4.18 Event Handling

AU : Dec-09,12, May-10,11, Marks 8

- **Definition of Event :** Event is an activity that represents a change in the environment. For example mouse clicks, pressing a particular key of keyboard represent the events. Such events are called intrinsic events.
- **Definition of Event Handler :** Event handler is a script that gets executed in response to these events. Thus event handler enables the web document to respond the user activities through the browser window.
- JavaScript support this special type of programming in which events may occur and these events get responded by executing the event handlers. This type of programming is called event-driven programming.
- Events are specified in lowercase letters and these are case-sensitive.
- **Event Registration :** The process of connecting event handler to an event is called event registration. The event handler registration can be done using two methods -
  - Assigning the tag attributes
  - Assigning the handler address to object properties.

Internet Programming

On occurrence of events the tag attribute must be assigned with some user defined functionalities. This will help to execute certain action on occurrence of particular event.

Commonly used events and tag attributes are enlisted in the following table -

Events	Intrinsic event attribute	Meaning	Associated tags
blur	onblur	Losing the focus.	<button> <input> <a> <textarea> <select>
change	onchange	On occurrence of some change.	<input> <textarea> <select>
click	onclick	When user clicks the mouse button.	<a> <input>
dblclick	ondblclick	When user double clicks the mouse button.	<a> <input> <button>

focus	onfocus	When user acquires the input focus.	<a> <input> <select> <textarea>
keyup	onkeyup	When user releases the key from the keyboard.	Form elements such as input,button,text,textarea and so on.
keydown	onkeydown	When user presses the key down	Form elements such as input,button,text,textarea and so on.
keypress	onkeypress	When user presses the key.	Form elements such as input,button,text,textarea and so on.
mousedown	onmousedown	When user clicks the left mouse button.	Form elements such as input,button,text,textarea and so on.
mouseup	onmouseup	When user releases the left mouse button.	Form elements such as input,button,text,textarea and so on.
mousemove	onmousemove	When user moves the mouse.	Form elements such as input,button,text,textarea and so on.
mouseout	onmouseout	When the user moves the mouse away from some element.	form elements such as input,button,text,textarea and so on.
mouseover	onmouseover	When the user moves the mouse away over some element.	Form elements such as input,button,text,textarea and so on.
load	onload	After getting the document loaded.	<body>
reset	onreset	When the reset button is clicked.	<form>
submit	onsubmit	When the submit button is clicked.	<form>
select	onselect	On selection.	<input> <textarea>
unload	onunload	When user exits the document.	<body>

The use of these tag attributes for handling the events is illustrated by following code sample

```
<input type =“button” name=“My_button”
onclick=“My_fun()”; />
```

Tag attribute **Event handler**

That means when the user clicks the **button** then as an event handler a user defined function **My_fun()** gets called. Basically in this function user writes the instructions that need to be executed on the button click event.

Ex. 4.18.1 : Discuss the properties of mouse events associated with DOM2 with an example.

AU : Dec-12, Marks 8

Sol. : Various types of mouse events associated with DOM2 are click, mousedown, mouseup, mouseout and mouseover.

Events	Intrinsic event attribute	Meaning	Associated tags
mousedown	onmousedown	When user clicks the left mouse button.	Form elements such as input,button,text,textarea and so on.
mouseup	onmouseup	When user releases the left mouse button.	Form elements such as input,button,text,textarea and so on.
mousemove	onmousemove	When user moves the mouse.	Form elements such as input,button,text,textarea and so on.
mouseout	onmouseout	When the user moves the mouse away from some element	form elements such as input,button,text,textarea and so on.
mouseover	onmouseover	When the user moves the mouse away over some element	Form elements such as input,button,text,textarea and so on.
click	onclick	When user clicks the mouse button.	<a> <input>
dblclick	ondblclick	When user double clicks the mouse button.	<a> <input> <button>

Following is an example in which image can be dragged and dropped anywhere in the HTML document. Accordingly the co-ordinates will be displayed.

```
<html>
<head>
  <title> Drag and Drop Demo</title>
  <script type = "text/javascript">
   var X_offset, Y_offset, Item;
   function Catch_function(e)
   {
      Item = e.currentTarget;
      var posX = parseInt(Item.style.left);
      var posY = parseInt(Item.style.top);
      X_offset = e.clientX - posX;
      Y_offset = e.clientY - posY;
      document.addEventListener("mousemove", Move_function, true);
      document.addEventListener("mouseup", Drop_function, true);
      e.stopPropagation();
      e.preventDefault();

   }
   function Move_function(e)
   {
      Item.style.left = (e.clientX - X_offset) + "px";
      Item.style.top = (e.clientY - Y_offset) + "px";
      e.stopPropagation();
   }
   function Drop_function(e)
   {
      document.removeEventListener("mouseup", Drop_function, true);
      document.removeEventListener("mousemove", Move_function, true);
      e.stopPropagation();
   }
  </script>
</head>
<body>
  <h3>
   Click on image below-drag it, move it and then drop anywhere!!!
   <br /><br />
   <img src="lamp1.jpg"
    style = "position: absolute;
```

Registering the mouse event with the help of **addEventListener** function.

```
    top: 100px;
    left: 100px;
    border:2px solid;"
    onmousedown = "Catch_function(event);" />
  </h3>
</body>
</html>
```

Output

4.18.1 Handling Events from the Body Elements

To understand how events works in JavaScript let us put some Form components on the JavaScript. The **onload** event gets activated as soon as the web page gets loaded in the browser's window. Following script along with its output illustrate the **onload** tag attribute.

JavaScript[OnloadDemo.html]

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
  <title>Demo of onload Tag Attibute</title>
  <script type="text/javascript">
   function my_fun()
   {
   //This message will be displayed on page loading


  alert("Welcome");
   }
  </script>
</head>
<body onload="my_fun()">
```

When web document gets loaded on the browser window then **my_fun()** will be called

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

Output

University Questions

1. *Explain the process of event registration.*
2. *Explain DOM event handling in detail.* — AU : May-11, Marks 8
3. *Describe the event object properties.* — AU : May-10, Marks 8
4. *Explain any four intrinsic event attributes.*
5. *Write a javascript to demonstrate the onload event.* — AU : Dec.-09, Marks 6

Part III : DHTML

4.19 DHTML with JavaScript

AU : Dec.-16, Marks 8

- The DHTML stands for **D**ynamic **H**ypertext **M**arkup **L**anguage.
- Basically with DHTML a web developer can control how to display and position HTML elements in a browser window.
- The first important difference between HTML and DHTML is that HTML is used to create **static web pages** and DHTML is used to create **dynamic web pages.**

- The HTML consists of simple HTML tags, on the other hand DHTML is made up of **HTML tags + Cascading Style Sheets(CSS) + JavaScripts.**
- The HTML does not allow to alter the text and graphics on the web page unless web page gets changed. But in DHTML you can alter the text and graphics of the web page and for that matter you need not have to change the entire web page.
- Creation of HTML web pages is **simplest** but less interactive. Creation of DHTML is **complex but more interactive.**

Difference between HTML and DHTML

Sr. No.	HTML	DHTML
1.	HTML stands for Hypertext Markup Language. HTML is used to create **static web pages.**	DHTML stands for Dynamic Hypertext Markup Language. DHTML is used to create **dynamic web pages**.
2.	HTML sites work slowly upon the client server technology.	DHTML sites work fast upon the client server technology.
3.	HTML does not make use of the technology for making the pages dynamic.	DHTML makes use of CSS, events, methods and so on to make the page dynamic.
4.	HTML does not require any processing from the browser.	DHTML requires processing from the browser.

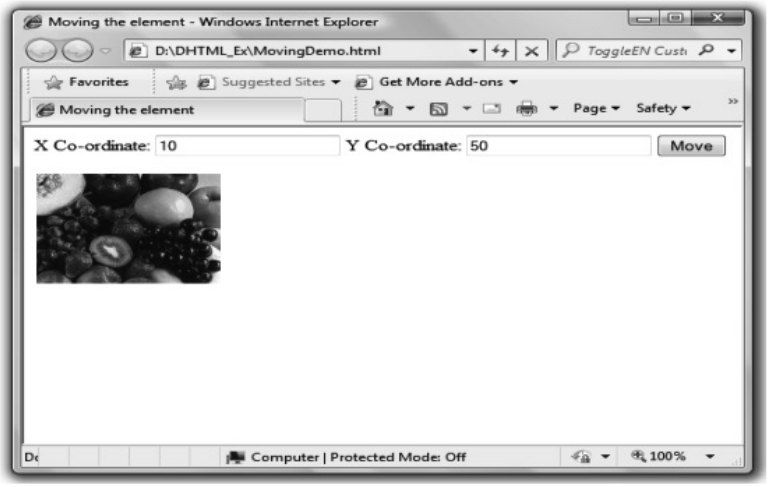
4.19.1 Moving Elements

- The position attribute decide the position of an element on the web browser , we can change this position by simply changing its **left** and **top** values.
- Note that only **absolute** or **relative** elements can be moved because static elements just neglect the top and left values.
- In the following example, we have placed two input boxes and one button element on the form. With the help of these elements we want to move the image **"fruit.jpg"** which is placed on the web document.
- For moving this image we have invoked one function "my_fun()". This function takes the values of X and Y co-ordinates through two text boxes that are supplied with and accordingly change the left and top values of the positioned element.
- The output shown along with this script itself is illustrative.

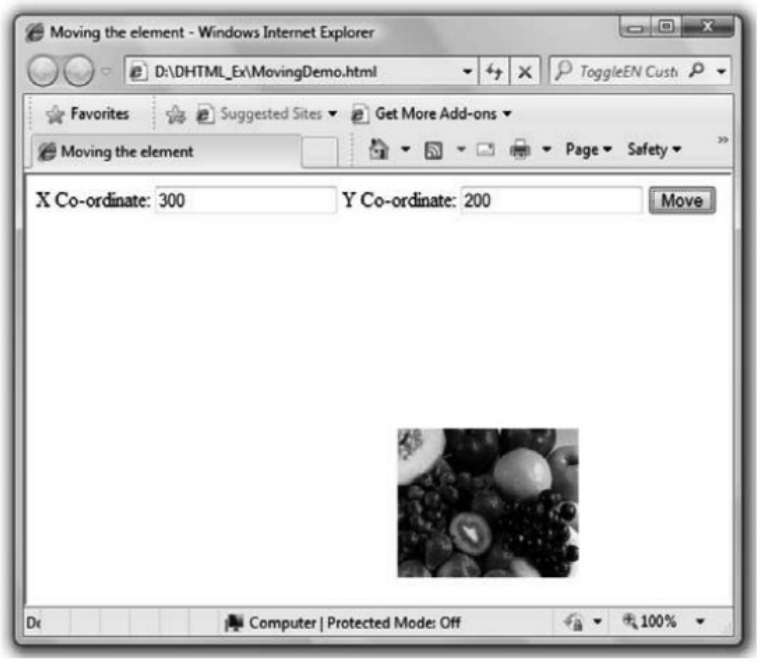
DHTML Document[MovingDemo.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Moving the element</title>
<script type="text/javascript">
 function my_fun(x_pos,y_pos)
 {
  var Dom_obj=document.getElementById("my_img").style;
  Dom_obj.top=y_pos+"px";;
  Dom_obj.left=x_pos+"px";
 }
</script>
</head>
<body>
 <form>
  <label>X Co-ordinate:
  <input type="text" value="10" id="x"/>
  </label>
  <label>Y Co-ordinate:
  <input type="text" value="50" id="y"/>
  </label>
  <input type="button" value="Move" onclick="my_fun(x.value,y.value);"/>
 </form>
 <div id="my_img" style="position:absolute;top:50px;left:10px;">
  <img src="fruit.jpg" />
 </div>
</body>
</html>
```

Output



Just change the values in two text boxes meant for X and Y co-ordinates, then click the Move button and you can see following sort of output.



4.19.2 Elements Visibility

We can have control over the visibility of particular element on the web document. This can be done using the property **visibility.** There are two values that can be set to the "visibility" and those are **visible** for getting the element displayed and **hidden** for hiding the element. Following is a simple DHTML document which makes use of this property.

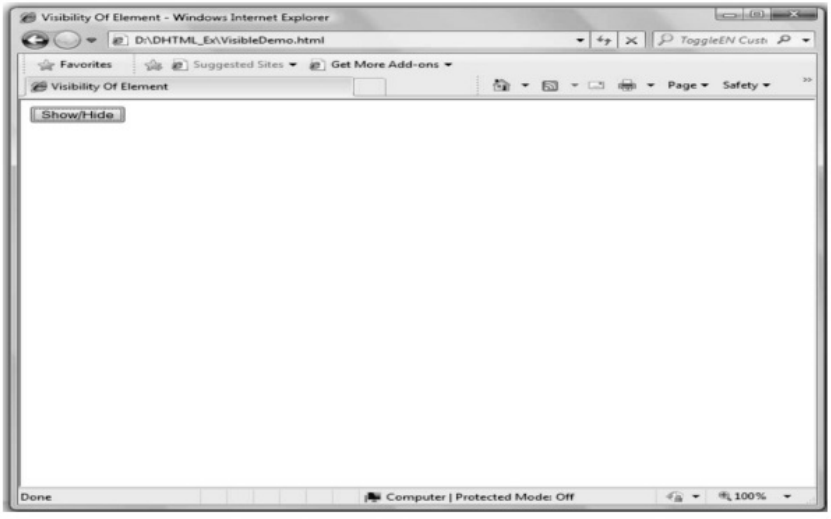
DHTML Document[VisibilityDemo.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Visibility Of Element</title>
<script type="text/javascript">
 function my_fun()
 {
  var Dom_obj=document.getElementById("my_img").style;
  if(Dom_obj.visibility=="visible")
   Dom_obj.visibility="hidden";//Image will be as hidden
  else
   Dom_obj.visibility="visible";//image will be displayed
 }
</script>
</head>
```

```
<body>
  <form>
    <input type="button" value="Show/Hide" onclick="my_fun();"/>
  </form>
  <div id="my_img" style="position:absolute;top:10px;left:100px;">
    <img src="bird.jpg" />
  </div>
</body>
</html>
```

Output

Just click Show/Hide button repeatedly and get the effect of visibility demo.



4.19.3 Changing Colors

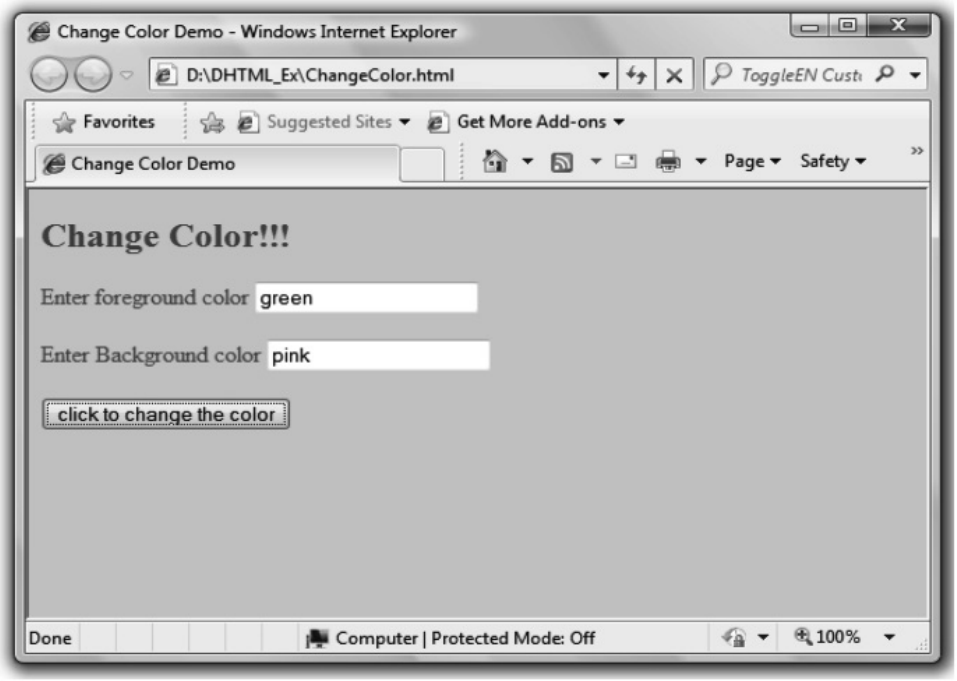
There are two important properties – **backgroundColor** and **color** using which the background and foreground color can be changed.

In the following script we have put two text boxes in which the name of desired color can be written *(note that this name should be strictly in lower case letters)* and then user must click the button in order to get the effect. Illustrative output is given along with this script.

DHTML Document[ChangeColor.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Change Color Demo</title>
<script type="text/javascript">
 function my_fun()
 {
  var F_color=document.getElementById("fore_color").value;
  var B_color=document.getElementById("back_color").value;
  document.body.style.backgroundColor=B_color;
  document.body.style.color=F_color;
 }
</script>
</head>
<body>
 <h2>Change Color!!!</h2>
  <form>
   <label>Enter foreground color
   <input type="text" value="" id="fore_color"/>
   </label>
   <br/><br/>
   <label>Enter Background color
    <input type="text" value="" id="back_color"/>
   </label>
   <br/><br/>
    <input type="button" value="click to change the color"   onclick="my_fun();"/>
  </form>
</body>
</html>
```

Output



4.19.4 Locating the Mouse Cursor

We can locate the position of the mouse with the help some mouse event. The mouse click event is implemented by **MouseEvent** interface. The mouse position can be denoted by two co-ordinates i.e. x and y co-ordinates.

There are two ways of specifying the position of the mouse.

- **Co-ordinates with respect to browser window**

  Using the clientX and clientY the x and y positions of mouse event can be obtained. These positions are calculated from the upper left corner of browser window.

- **Co-ordinates with respect to computer screen**

  Using the screenX and screenY the x and y positions of mouse event can be obtained. These positions are calculated related to clients computer screen.

  But normally used of clientX and clientY is preferred for obtaining the x and y positions when some mouse event occurs on the browser window.

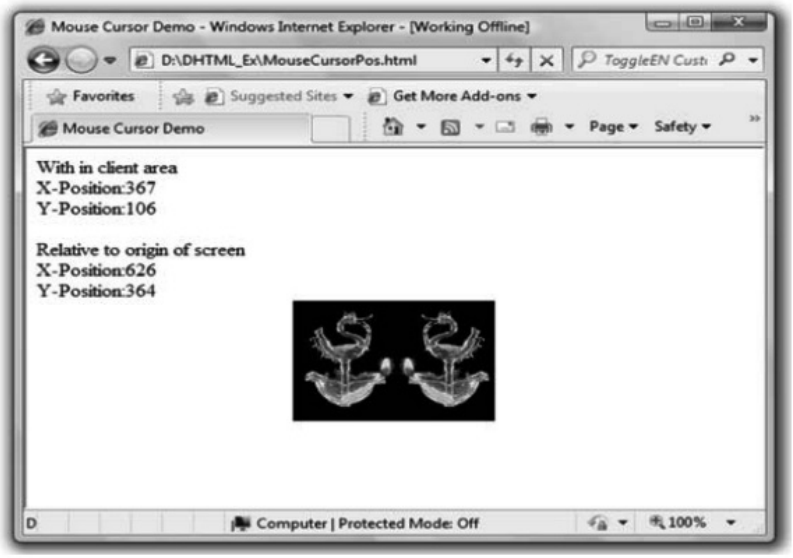
In the following document we are displaying both the types of co-ordinates.

DHTML Document[MouseCursorPos.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Mouse Cursor Demo</title>
```

```
 <script type="text/javascript">
 function my_fun(e)
 {
  points1.innerText="With in client area"+"\n"+"X-Position:"+e.clientX+"\n"+
  "Y-Position:"+e.clientY;
  points2.innerText="\n"+"Relative to origin of screen"+"\n"+"X-Position:"
 +e.screenX+"\n"+"Y-Position:"+e.screenY;
 }
 </script>
</head>
<body onclick="my_fun(event);">
 <span id="points1">(0,0)</span>
 <br/>
 <span id="points2">(0,0)</span>
 <div align="center">
  <img src="lamp1.jpg"/>
 </div>
</body>
</html>
```

Output



In above script we have used **innerText** attribute. If certain block of text gets displayed and may get changed repeatedly then the innerText property is used. Onclick event we are passing the object **event** to the event handler function. Hence we can obtain the current positions.

4.19.5 Reacting to a Mouse Click

AU : Dec.-16, Marks 8

When we click the mouse button it goes down and then comes back. These activities can be caught using the **onmousedown** and **onmouseup** events respectively.

Following is an example in which we will display one image when user presses mouse button and another image gets displayed when user releases the mouse button.

DHTML Document[MouseClick.html]

```
<!DOCTYPE html>
<html>
<head>
  <title>Response to mouse click</title>
<script type="text/javascript">
function my_fun()
{
 my_img.src="waterfall.jpg";
}
function my_fun1()
{
 my_img.src="TajMahal.jpg";
}
</script>
</head>
<body>
 <center>
  <h1> Two Wonders of the World</h1>
  <img id="my_img" onmousedown="my_fun()" onmouseup="my_fun1()"
    src="TajMahal.jpg" width="200" height="150"/>
  <p> Click this image to change it</p>
 </center>
</body>
</html>
```

Output

If click on the image we will get another image being displayed.

4.19.6 Slow Movements of Elements

- The element can be moved by changing the left and top properties.
- For moving an element slowly we have to move it by small amount repetitively.
- Each move must be separated by some amount of time.
- There are two methods defined by an object window which are responsible for moving an element. These methods are setTimeout and setInterval.
- The setTimeout function takes two parameters. The first one is the function which is responsible for moving an element. And the second parameter is the delay time given in milliseconds. The function which is responsible for moving an element can be called with the specified amount of delay. For example

  ```
  setTimeout("Move1();",10)
  ```

  means the function **Move1()** can be called after 10 milliseconds repeatedly.
- Following is a simple DHTML document in which the text is moved in forward and backward direction using setTimeout function.

DHTML Document[TxtMove.html]

```
<!DOCTYPE html>
<html>
<head>
<title>Moving Text Example</title>
<script language="javascript">
```

```
var x=0;
var direction=1;
function Move1()
{
  Dom_obj=document.getElementById("txt");
  x += dir;
  if (x >= 300 || x <= – 300)
     dir = 0 – dir;
  Dom_obj.style.left=x;
  window.setTimeout("Move1();",10);
 }
</script>
</head>
<body onload="Move1();">
<h3 align="center" style="position: relative;" ID="txt">
 King is on the move!!!
</h3>
</body>
</html>
```

Due DOM_obj.style.top=x the element moves horizontally.

If we change to DOM_obj.style.top=x then the text will move in vertical direction.

Output

- In above script, we have called a **Move1()** function in which we are accessing the <h3> elements whose id is "txt".
- Then an object **Dom_obj** is created. The value of x will be incremented by one each time and is assigned to the left property of the object **Dom_obj.**
- Then **setTimeout** function will be called. This method calls the function **Move1()** again with the time delay of 10 milliseconds.

- Thus on each call of **Move1()** function the value of x will be incremented and is assigned to the **left** property of **Dom_obj.** This continues until the value being reached is 300.
- On reaching the value to 300, the value of x gets decremented. This causes this to move the element in backward direction.
- Hence as an output we get the text is moving forward and backward.

Review Question

1. *Write a DHTML program to handle the user click event.* **AU : Dec.-16, Marks 8**

4.20 Programming Examples

AU : May-08,16, June-09, Dec.-08,09, Marks 16

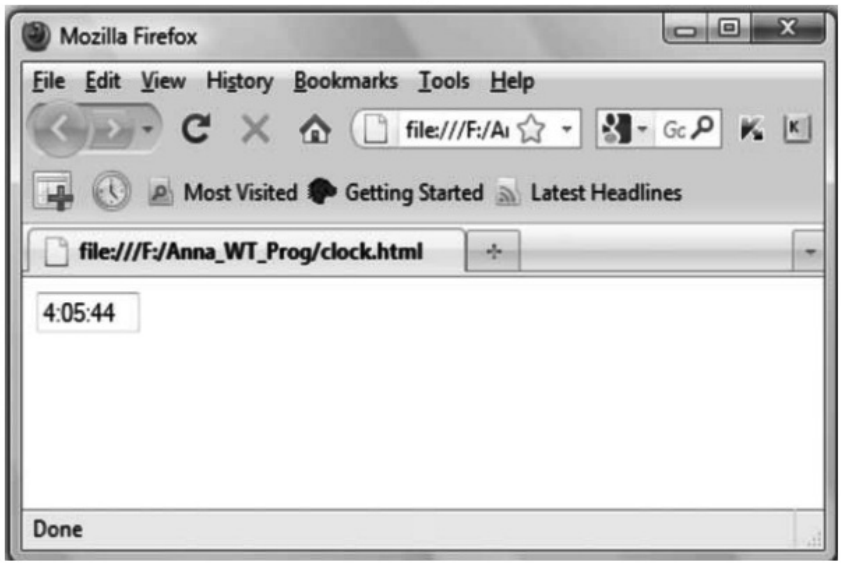
Ex. 4.20.1 : Design web page to create a clock with a timing event. **AU : June- 09, Marks 4**

Sol. : clock.html

```
<html>
<head>
<script type="text/javascript">
function Display()
{
 var today=new Date();
 var h=today.getHours();
 var m=today.getMinutes();
 var s=today.getSeconds();
 h=ConvertTwelve(h);
 m=SingleDigit(m);
 s=SingleDigit(s);
 document.getElementById("mytext").value=h+":"+m+":"+s;
 t=setTimeout('Display()',500);
}
 function ConvertTwelve(h)
 {
  if(h>=12)
  h=h-12;
  return h;
 }
 function SingleDigit(i)
 {
  if (i<10)
  {
   i="0" + i;//writing 0 before single digit number
  }
  return i;
```

```
}
</script>
</head>
<body onload="Display">
  <input type="text" value="" size="5" id="mytext"/>
</body>
</html>
```

Output



Ex. 4.20.2 : Write HTML program for the registration of new customer to the online banking system (Customer Data collected using a form, after submitting account number and type of account and username, password is displayed as output) AU : May-08, CSE, Marks 8

Sol. : HTML Document[bank.html]

```
<html>
<head>
  <title>Online Banking System</title>
  <script type="text/javascript">
  function fun()
  {
   alert("Name: "+form1.userName.value);
   alert("Password: "+form1.pwd.value);
   alert("Account Type: "+form1.MyMenu.value);
   alert("Account Number: "+form1.AccNo.value);
  }
  </script>
</head>
<body bgcolor="khaki">
  <form name="form1" onsubmit="fun()">
   <table>
```

Ex. 4.20.3 : Develop a HTML page which accepts

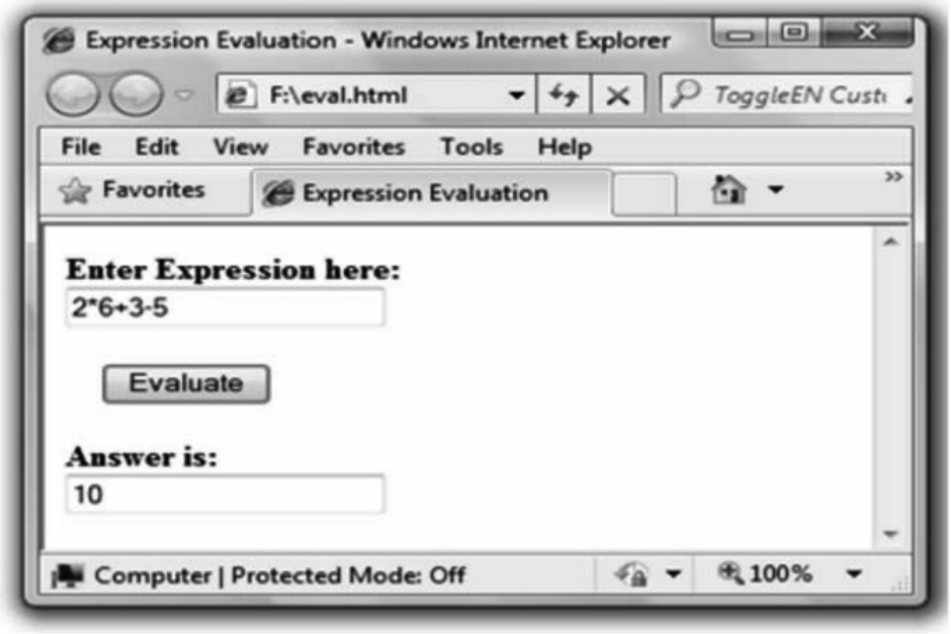
- Any mathematical expression
- Evaluate the expression
- Displays the result of the evaluation.

AU : Dec.-08, CSE, Marks 16

Sol. : HTML Document[eval.html]

```
<html>
<head>
 <title>Expression Evaluation</title>
 <script type="text/javascript">
  function Compute()
  {
   document.form1.ans.value=eval(document.form1.my_exp.value);
  }
  </script>
</head>
<body>
 <form name="form1">
 <b>Enter Expression here:</b><br/><input type="text" name="my_exp">
  <br/><br/>
      <input type="button" value="Evaluate" onclick=Compute()>
  <br/><br/>
  <b>Answer is: </b><br/><input type="text" name="ans">
 </form>
</body>
</html>
```

Output



Ex. 4.20.4 : Develop a simple online shopping application using JavaScript.
(Assume your own data) **AU : Dec.-09, CSE, Marks 16**

Sol. :

Step 1 : We will first write a simple JavaScript which will allow user to enter the personal details and items to be purchased.

HTML Document[shopping.html]

```
<!DOCTYPE html PUBLIC "-//W#C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<script language="JavaScript">
var cost;
function fun()
{
  cost=0;
 if(document.myform.Item1.checked){cost=cost+20.15;}
 if(document.myform.Item2.checked){cost=cost+10.10;}
 if(document.myform.Item3.checked){cost=cost+26;}
 if(document.myform.Item4.checked){cost=cost+29;}
 document.myform.total.value="$"+cost;
}
</script>
</head>
<body bgcolor="pink">
<form name="myform" action="http://localhost/cgi-bin/purchaseInfo.cgi" enctype="text/plain"
method="post">
<center><h2>Online shopping Application</h2></center>
<strong>Personal Details:</strong>
<table>
<tr><td>name:</td><td> <input type="text" name="customer"/></td></tr>
<tr><td>City:</td><td> <input type="text" name="city"/></td></tr>
<tr><td>Phone:</td><td> <input type="text" name="phone"/></td></tr>
<tr><td>Email_Id:</td><td> <input type="text" name="email"/></td></tr>
</table>
<hr/>
<strong>Description:</strong>
<p>Following are some items with their prices.</p>
<p>Choose the required items by checking the chekboxes.</p>
</div>
```

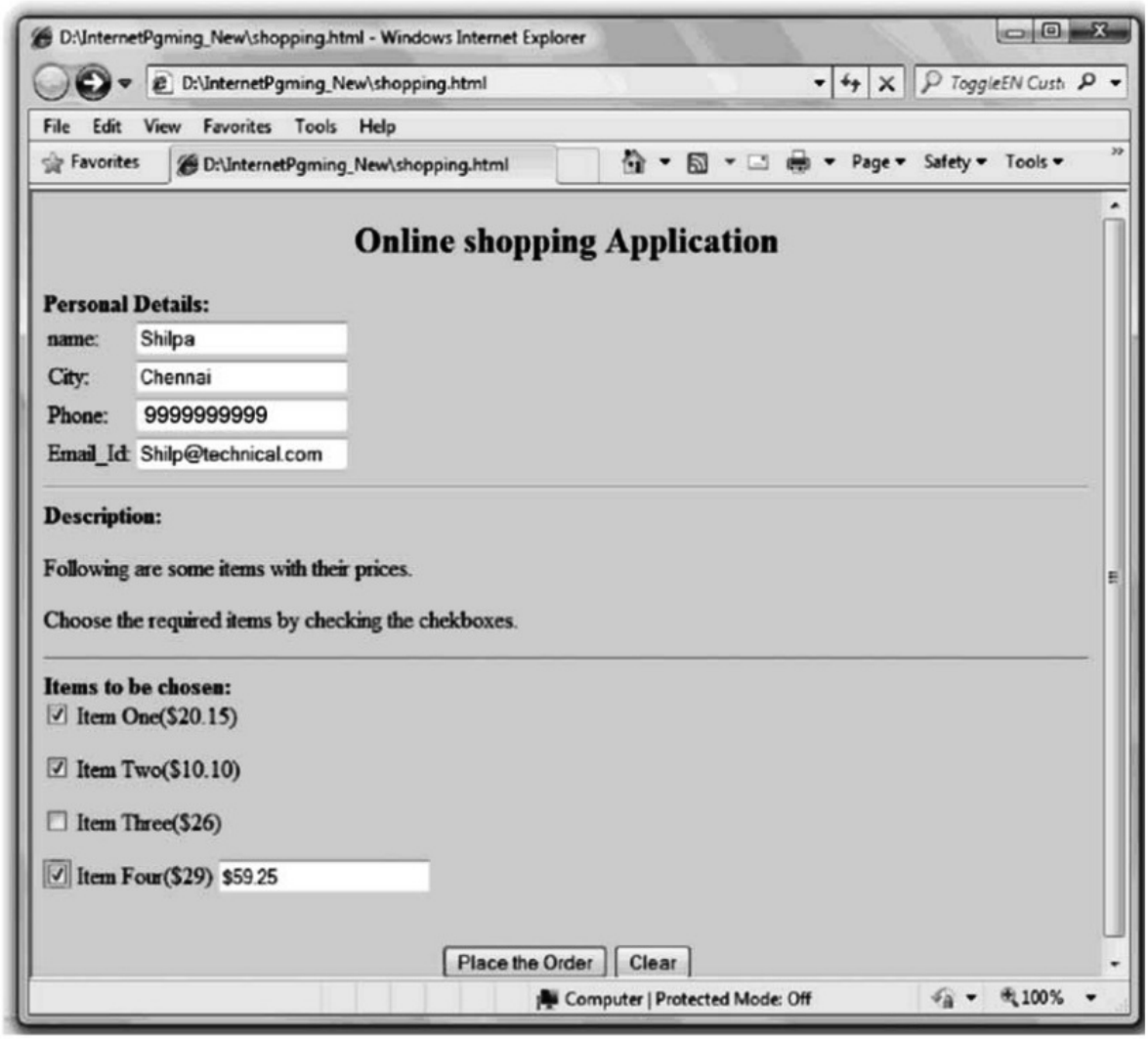
```
<hr/>
<strong>Items to be chosen:</strong>
<br/>
<input type="checkbox" name="Item1" value="Item1_chosen" onclick="fun()"> Item One($20.15)
<p><input type="checkbox" name="Item2" value="Item2_chosen" onclick="fun()"> Item Two($10.10)
<p><input type="checkbox" name="Item3" value="Item3_chosen" onclick="fun()"> Item Three($26)
<p><input type="checkbox" name="Item4" value="Item4_chosen" onclick="fun()"> Item Four($29)
<input type="text" name="total" value="0"/>
<br/><br/>
<center>
<input type="submit" value="Place the Order" />
<input type="reset" value="Clear" />
</center>
</form>
</body>
</html>
```

Step 2 : When user click the Place the order button the submitted input can be collected in a cgi file. The sample cgi file can be written as follows -

CGI Document[purchaseInfo.cgi]

```
#!/program files/perl/bin/perl
use CGI qw(:standard);
print header;
print start_html(-BGCOLOR =>"Khaki");
print h2("Thank you for spending your valuable time with us!!!<br/>");
my($name)=param("customer");
print"<b>Order submitted is...</b>$name";
print end_html;
```

Step 3 : The output can be viewed as follows -



Click the 'Place the order' button and following output can be seen

Ex. 4.20.5 : Write a Javascript for validating name and date.

Sol. : JavaScript

```
<!DOCTYPE html>
<html>
<head>
  <title>Demo of Event Registration</title>
<script type="text/javascript">
 function chkDt(event)
 {
 var inputstr =event.currentTarget;
 var str=inputstr.value;
 if(str.match(/\d{1,2}:\d{1,2}:\d{4}/))
 {
  a=str.split(":");
  var dd=Number(a[0]);
  var mm=Number(a[1]);
  var yy=Number(a[2]);
  if(mm==1||mm==01)
  {
   if(dd>=30)
   {
   alert("Invalid date,Please re-enter it(February have 28 or 29 days)");
   inputstr.focus();
   inputstr.select();
   }
   else
     alert("Valid Date");
  }
  else if(mm==0||mm==2||mm==4||mm==6||mm==7||mm==9||mm==11)
  {
   if(dd>31)
   {
   alert("Invalid date,Please re-enter it");
   inputstr.focus();
   inputstr.select();
   }
   else
     alert("Valid Date");
  }
  else if(mm==3||mm==5||mm==8||mm==10)
  {
   if(dd>30)
   {
   alert("Invalid date,Please re-enter it");
   inputstr.focus();
```

Accessing the text box created for entering the name using the **event.currentTarget**

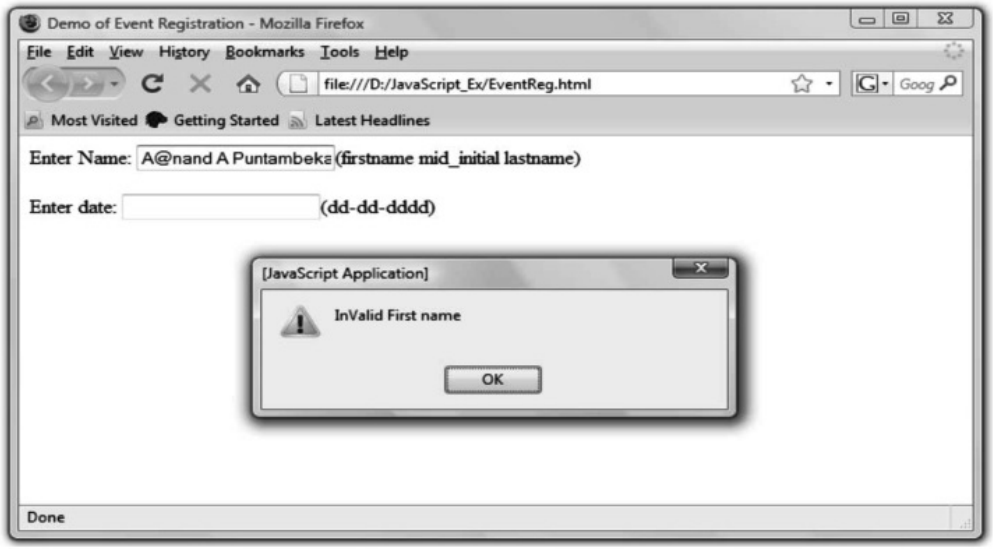
```
    inputstr.select();
    }
    else
      alert("Valid Date");
   }
  }
   else
   {
    alert("invalid Input format!!,Please try again");
    inputstr.focus();
    inputstr.select();
   }

}
function chkName(event)
{
 var inputstr =event.currentTarget;
 var str=inputstr.value;
 if(str.match(/[a-zA-Z]+\s[a-zA-Z]\s[a-zA-Z]+/))
 {
  a=str.split(" ");
  var First_name=a[0];
  var Mid_initial=a[1];
  var Last_name=a[2];
  if(First_name.match(/^[a-zA-Z]{1,15}$/))
  {
    if(Mid_initial.match(/[a-zA-Z]$/))
    {
     if(Last_name.match(/^[a-zA-Z]{1,15}$/))
      alert("Valid name, now enter date");
     else
      alert("InValid Last Name");
    }
    else
      alert("InValid Middle name initial");
   }
   else
     alert("InValid First name");
  }
  else
    alert("You have entered wrong input");
 }
 </script>
</head>
```

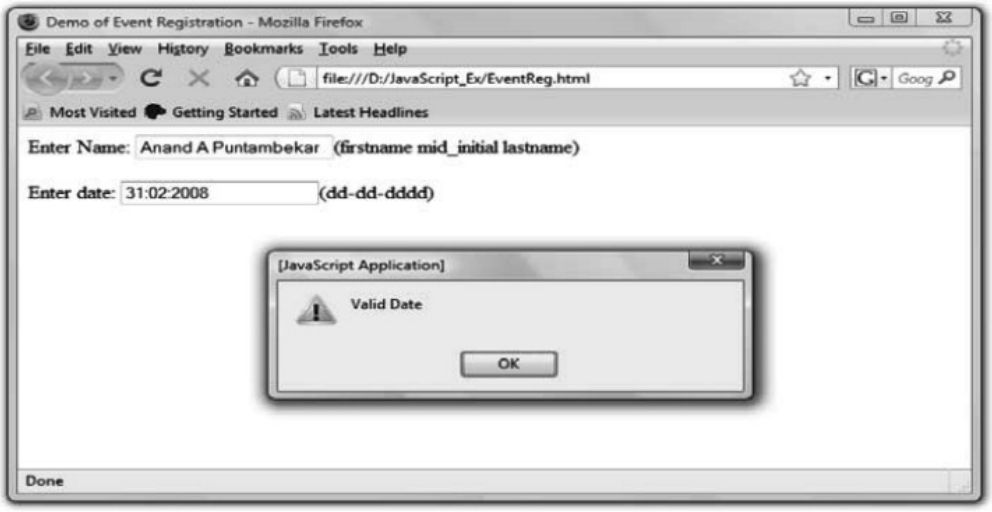
Accessing the text box created for entering the name using the **event.currentTarget**

```
<body>
  <form id="form1">
   <label>Enter Name:
     <input type="text" value="" id="Name_chk" />(firstname mid_initial lastname)
   </label>
   <br/><br/>
   <label>Enter date:
   <input type="text" value="" id="Dt_chk" />(dd-dd-dddd)
   </label>
   <br/>
  </form>
<script type="text/javascript">
  var name_node=document.getElementById("Name_chk");
  var dt_node=document.getElementById("Dt_chk");
 dt_node.addEventListener("change",chkDt,false);
  name_node.addEventListener("change",chkName,false);
</script>
 </body>
</html>
```

Getting the object for each element and then registering the event handlers

Output





Thus document object model is useful in handling objects and events.

Ex. 4.20.6 : Create a script that repeatedly flashes an image on the screen. Do this by changing the visibility of the image. Allow users to control the "blink speed". AU : May-08, Marks 8

Sol. : Blinking.html

```
<html>
<head>
<script type="text/javascript">
function GetSpeed()
{
var input_str=prompt("Enter the speed value here:","");
var i=Number(input_str);
blinkImage(i);
}
function blinkImage(i)
{

 if(!document.getElementById('blink').style.visibility)
  {
    document.getElementById('blink').style.visibility="visible"
  }
  if(document.getElementById('blink').style.visibility=="visible")
  {
    document.getElementById('blink').style.visibility="hidden";
  }
  else
  {
    document.getElementById('blink').style.visibility="visible";
  }
  j=i;
```

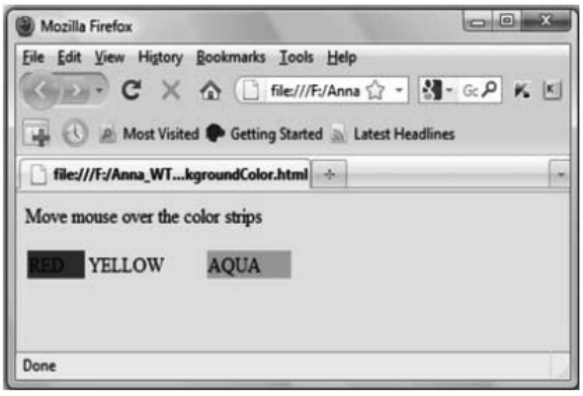
```
 timer=setTimeout("blinkImage(j)",i);
}
function stoptimer()
{
clearTimeout(timer);
}
</script>
<title>Image Blinking Demo</title>
</head>
<body onload="GetSpeed()">
<img src="flower1.gif" id="blink" style="visibility:hidden" >
</body>
</html>
```

Ex. 4.20.7 : Develop a DHTML page to change the background color using mouse over event on three squares containing different colors. AU : Dec.-08, Marks 2

Sol. : backgroundcolor.html

```
<html>
<head>
<script type="text/javascript">
function changeColor(color)
{
 document.getElementById('B').style.background=color;
}
</script>
</head>
<body id="B">
 <p>Move mouse over the color strips</p>
 <p> </p>
 <table width="50%">
  <tr>
   <td bgcolor="red" onmouseover="changeColor('red')"
              onmouseout="changeColor('transparent')">RED</td>
   <td bgcolor="yellow" onmouseover="changeColor('yellow')"
              onmouseout="changeColor('transparent')">YELLOW</td>
   <td bgcolor="aqua" onmouseover="changeColor('aqua')"
              onmouseout="changeColor('transparent')">AQUA</td>
  </tr>
 </table>
</body>
</html>
```

Output



Ex. 4.20.8 : Write a DHTML to shake the window of a button click. Make use of two buttons one button used to start the shaking and another button used to stop the shaking of window.

AU : Dec.-08, Marks 10

Sol. : shake button.html

```
<head>
<script  language="JavaScript1.2">
function shake(n)
{
 if(parent.moveBy)
 {
  for (i = 100; i > 0; i--)
  {
   for (j = n; j > 0; j--)
   {
    parent.moveBy(0,i);
    parent.moveBy(i,0);
    parent.moveBy(0,-i);
    parent.moveBy(-i,0);
   }
  }
 }
}
function Stop_Shake(obj)
{
 parent.moveBy=0;
 obj.style.left=0;
 obj.style.top=0;
}
</script>
</head>
<body>
<center>
<form>
<input type="button" onClick="Stop_Shake(this)" value="Stop">
```

```
<input type="button" onClick="shake(100)" value="Shake">
</form>
</center>
</ form>

</body>
```

Ex. 4.20.9 : Write a DHTML to change the background colour of a button, Mouse over three colored table cells and the background color will change. AU : Dec.-08, Marks 10

Sol. : buttoncolor.html

```
<html>
<head>
<script type="text/javascript">
function changeColor(color)
{
 document.getElementById('MyButton').style.background=color;
}
</script>
</head>
<body>
 <p>Move mouse over the color strips</p>
 <p> </p>
 <table width="50%">
  <tr>
   <td bgcolor="red" onmouseover="changeColor('red')">RED</td>
   <td bgcolor="yellow" onmouseover="changeColor('yellow')">YELLOW</td>
   <td bgcolor="aqua" onmouseover="changeColor('aqua')">AQUA</td>
  </tr>
 </table>
<form>
<input id="MyButton" type="button" value="Changing Color of Button">
</form>
</body>
</html>
```

Output

Ex. 4.20.10 : Write a JavaScript for displaying the context menu. The context menu is one that is shown when user right clicks anywhere in the document.

Sol. :

test1.html

```
<div id="ContxtMenu"
style="width:150px;border:1px solid black;background
color:yellow;visibility:hidden;position:absolute;line-height:30px; padding-left: 10px">
  <a href="http://www.google.com">Search Engine</a><br />
  <a href="http://www.facebook.com">Social Networking</a><br />
  <a href="http://www.gmail.com">Email Service</a><br / >
</div>

<script type="text/javascript">
var browserObj=document.getElementById && !document.all
var visibility=0

function Display(e)
{
 el=document.getElementById("ContxtMenu")
 visibility=1
 if (browserObj)
 {
  el.style.left=pageXOffset+e.clientX+"px"
  el.style.top=pageYOffset+e.clientY+"px"
  el.style.visibility="visible"
  e.preventDefault()
  return false
}

}

function Hide()
{
 if (typeof el!="undefined" && visibility)
 {
   el.style.visibility="hidden"
   visibility=0
 }
}
if (browserObj)
{
 document.addEventListener("contextmenu", Display, true)
 document.addEventListener("click", Hide, true)
}
</script>
```

Output

[Note that the Mozilla FireFox browser is used to get the following output]

Ex. 4.20.11 : Use Javascript and HTML to create a page with two panes. The first pane (on the left) should have a text area where HTML code can be typed by the user. The pane on the right side should display the preview of the HTML code typed by the user, as it would be seen on the browser. AU : May-16, Marks 16

Sol. :

Step 1 : Create the Main Window containing two frames. The code for this document is as follows -

MainWindow.html

```
<!DOCTYPE html>
<html>
<head>
   <title>Frameset Example</title>
</head>
   <frameset cols="50%,50%">
      <frame src="Left.html" name="left"/>
      <frame src="Right.html" name="right"/>
   </frameset>
<body>
</body>
</html>
```

Step 2 : Create the left frame containing the textarea in which we can write any HTML code. This document is named as Left.html. It is as follows -

Left.html

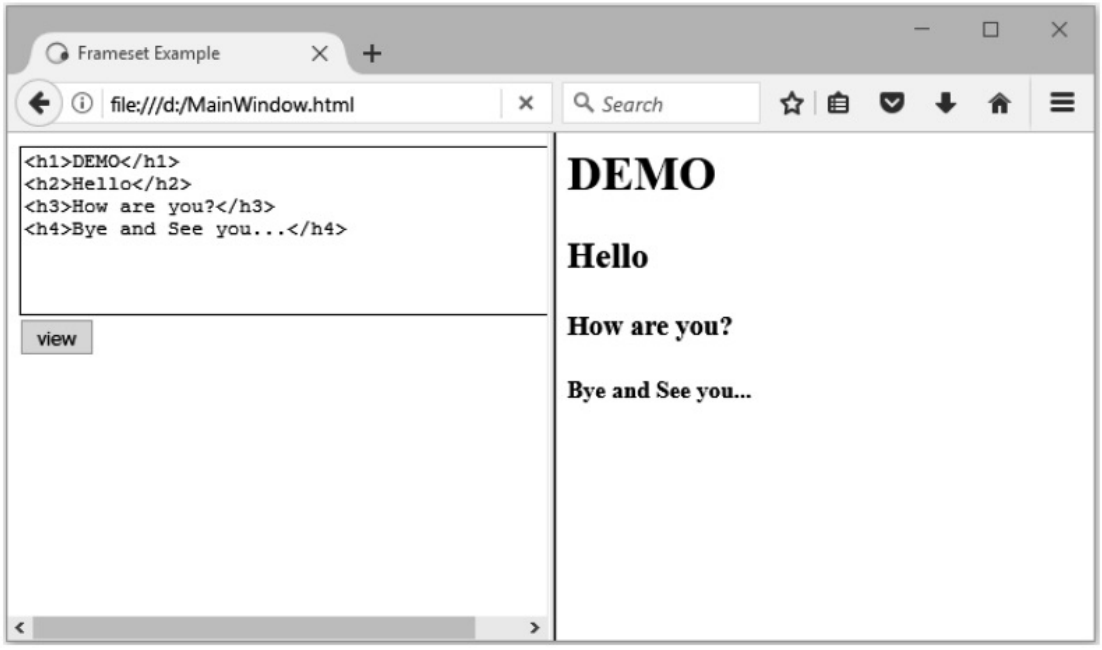
```
<!DOCTYPE html>
<html>
<head>
</head>
<body>
```

```
<form>
<textarea name="data" cols=45 rows=6></textarea>
     <br>
     <input type="button" value=" view "
onclick="javascript:parent.right.displayHTML(this.form)">
</form>
</body>
</html>
```

Step 3 :

```
<!DOCTYPE html>
<html>
<head>
<script type="text/javascript">
function displayHTML(form)
{
     var inf = form.data.value;
win = window.open('', '', 'right', 'toolbar = no, status = no');
      win.document.write("" + inf + "");
}
</script>
</head>
<body>
</body>
</html>
```

Step 4 : Open Web browser and open the MainWindow.html document(created in step1). The sample output will be as follows -



Part IV: Introduction to JSON

4.21 What is JSON ?

- JSON stands for JavaScript Object Notation.
- Using JSON we can store and retrieve data. This text based open standard format.
- It is extended from JavaScript language.

Features of JSON

1. It is **text based, lightweight data interchange format.**
2. It is **language independent.**
3. It is **easy to read and write.**
4. It is **easy** for machines **to parse and generate.**
5. It uses the conventions that are **familiar** to the languages like C, C++, Java, JavaScript, Perl, Python and so on.

4.21.1 Syntax

JSON is built on **two structures :**

1. A collection of name/value pairs. In various languages, this is realized as an object, record, struct, dictionary, hash table, keyed list, or associative array.
2. An ordered list of values. In most languages, this is realized as an array, vector, list, or sequence.

JSON Object

- JSON object holds the Key value pair.
- Each key is represented as string and value can be of any datatype.
- The key and value are separated by colon.
- **Syntax**

  { string : value, ...}
- **For example**

  "Age":38
- Each key value pair is separated by comma.
- The object is written within the { }curly brackets.

1) JSON object containing value of different data types

```
{
   "student":
```

```
    {
     "name":        AAA",        ← String value
     "roll_no": 10,              ← Numeric value
     "Indian": true         ←  Boolean value
    }
}
```

2) JSON Nested Object

```
{
    "student":
    {
     "name":  "AAA",
     "roll_no": 10,
     "address":
        {
        "Street": "Shivaji Nagar",
        "City": "Pune",
        "Pincode": 411005
        }
    }
}
```

3) Creation of JSON object with JavaScript

In JavaScript we can write the JSON object and store it in some variable. For example

```
var stud = { "name":"AAA", "roll_no":10, "city":"Pune" };
```

Here the object named stud is created.

Example

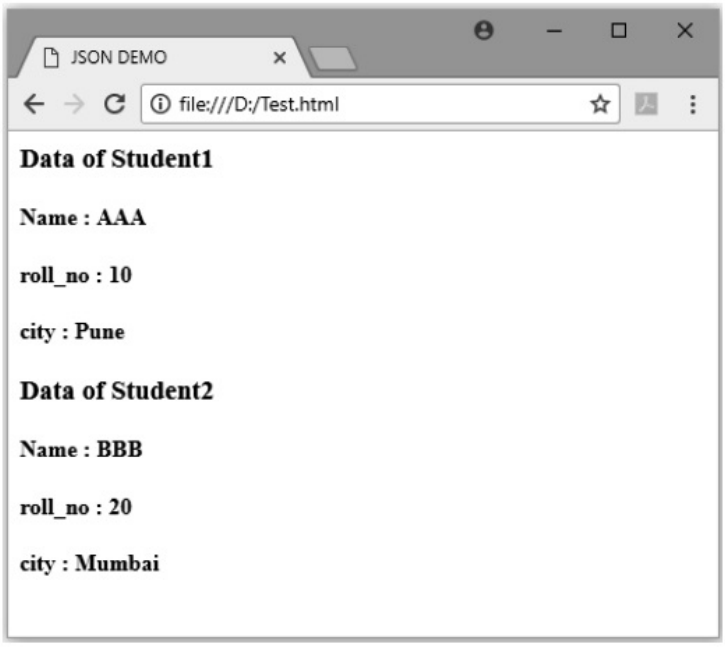
Let us understand how to create an JSON object and obtain its value, and get it displayed on the web browser using JavaScript. The code is as follows -

Test.html

```
<html>
  <head>
    <title>JSON DEMO</title>
    <script language = "javascript" >
      var stud1 = { "name":"AAA", "roll_no":10, "city":"Pune"};
      document.write("<h3>Data of Student1</h3>");
      document.write("<h4>Name : " + stud1.name+"</h4>");
      document.write("<h4>roll_no : " + stud1.roll_no+"</h4>");
      document.write("<h4>city : " + stud1.city+"</h4>");

      var stud2 = { "name":"BBB", "roll_no":20, "city":"Mumbai"}
      document.write("<h3>Data of Student2</h3>");
```

```
        document.write("<h4>Name : " + stud2.name+"</h4>");
        document.write("<h4>roll_no : " + stud2.roll_no+"</h4>");
        document.write("<h4>city : " + stud2.city+"</h4>");
      </script>
    </head>
    <body>
    </body>
</html>
```

Output



JSON with Arrays

JSON array represents the collection of values. It is denoted within [ ]. The elements of array are separated by comma.

Syntax

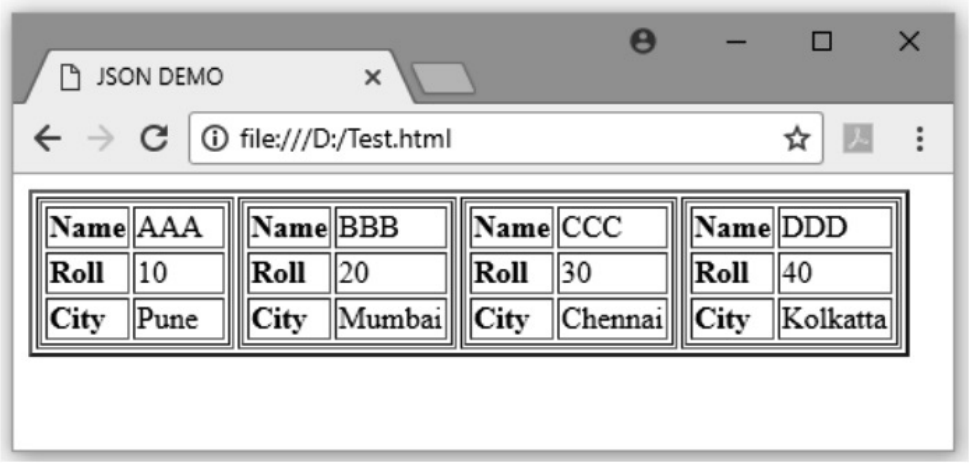
```
[ value, .....]
```

Example

```
<html>
  <head>
    <title>JSON DEMO</title>
    <script language = "javascript" >
      var obj = {"Student":[
        { "name":"AAA", "roll_no":10, "city":"Pune"},
        { "name":"BBB", "roll_no":20, "city":"Mumbai"},
        { "name":"CCC", "roll_no":30, "city":"Chennai"},
        { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
```

```
        ]};
var i = 0
        document.writeln("<table border = '2'><tr>");
    for(i = 0;i<obj.Student.length;i++)
    {
        document.writeln("<td>");
        document.writeln("<table border = '1' width = 100 >");
        document.writeln("<tr><td><b>Name</b></td><td width = 50>"+
        obj.Student[i].name+"</td></tr>");
        document.writeln("<tr><td><b>Roll</b></td><td width = 50>" + obj.Student[i].roll_no
        +"</td></tr>");
        document.writeln("<tr><td><b>City</b></td><td width = 50>" + obj.Student[i].city
       +"</td></tr>");
        document.writeln("</table>");
        document.writeln("</td>");
    }
   </script>
  </head>
  <body>
  </body>
</html>
```

Output



4.21.2 Function Files

Using JSON, it is possible to define a function in a separate external JS file and we can access this functionality from HTML document.

Following example illustrates this idea

Example Code

Step 1 : We can write a JavaScript file containing an array of Student's information. This file is saved with .js extension. The code for it is as follows -

myFile.js

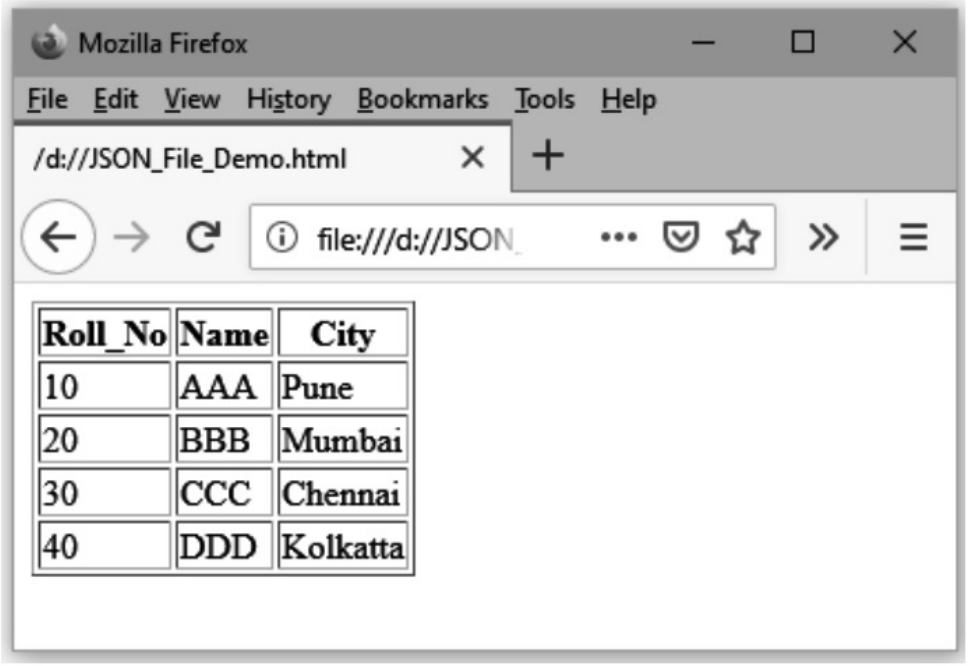
```
Student([
      { "name":"AAA", "roll_no":10, "city":"Pune"},
      { "name":"BBB", "roll_no":20, "city":"Mumbai"},
      { "name":"CCC", "roll_no":30, "city":"Chennai"},
      { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
])
```

Step 2 : Now we will invoke this file in our JSON script to access the elements of an array. The code is as follows -

JSON_File_Demo.html

```
<!DOCTYPE html>
<html>
<body>

<div id="myID"></div>
<script>
function Student(arr) {
  var out = "";
  var i;
  out+='<table border=1>'
  out+='<tr><th>Roll_No</th><th>Name</th><th>City</th></tr>'
  for(i = 0; i<arr.length; i++) {
  out += '<tr><td>'+arr[i].roll_no + '</td><td>' +arr[i].name + '</td><td>' +arr[i].city +
'</td></tr>';
  }
  out+='</table>'
  document.getElementById("myID").innerHTML = out;
}
</script>
<script src="myFile.js"></script> // Invoking external file

</body>
</html>
```

Output



Script Explanation :

In above script,

1. We have written the Student function in a file named **MyFile.js**
2. This function defines an array of three fields Roll_No, Name and City. There are four elements in that array
3. In our HTML file, the JSON code refers to the external JS file as shown below

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

4. Then the elements of array are accessed and displayed on the browser as follows

```
for(i = 0; i<arr.length; i++) {
    out += '<tr><td>'+arr[i].roll_no +
  '</td><td>' +arr[i].name +
  '</td><td>' +arr[i].city + '</td></tr>';
}
```

4.21.3 HTTP Request

- JSON is most commonly used in asynchronous HTTP requests. This is where an application pulls data from another application via an HTTP request on the web.
- **XMLHttpRequest** is an API that provides scripted client functionality for transferring data between a client and a server.
- It allows you to get data from an external URL without having to refresh the page

XMLHttpRequest includes a number of methods and attributes.

- We use two functions of XMLHttpRequest - **Open()** and **send().**
- The **open()** to initialize the request, and **send()** to send the request.
- The Syntax of method **open of XMLHttpRequest** is

  XMLHttpRequest.open(method, url[, async[, user[, password]]])

  where
  - **Method :** The HTTP request method to use, such as "GET", "POST", "PUT", "DELETE", etc. Ignored for non-HTTP(S) URLs.
  - **url :** A string representing the URL to send the request to.
  - **async (Optional) :** An optional Boolean parameter, defaulting to true, indicating whether or not to perform the operation asynchronously. If this value is false, the send() method does not return until the response is received. If true, notification of a completed transaction is provided using event listeners.
- Following code illustrates this idea

  Step 1 : Create a text file as

  myfile.txt

```
[
    { "name":"AAA", "roll_no":10, "city":"Pune"},
    { "name":"BBB", "roll_no":20, "city":"Mumbai"},
    { "name":"CCC", "roll_no":30, "city":"Chennai"},
    { "name":"DDD", "roll_no":40, "city":"Kolkatta"}
]
```

  Step 2 : The JSON code to invoke above text file using Http Request is as follows -

JSON_HttpReq_Demo.html

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
</head>
<body>

<div id="myID"></div>
<script>

var xmlhttp = new XMLHttpRequest();
var url = "file:myfile.txt";

xmlhttp.onreadystatechange = function() {
    if (this.readyState == 4 && this.status == 200) {
     var stud_arr = JSON.parse(this.responseText);
```

```
      Student(stud_arr);
     }
};

xmlhttp.open("GET", url, true);
xmlhttp.send();

function Student(arr) {
 var out = "";
 var i;
 out+='<table border=1>'
 out+='<tr><th>Roll_No</th><th>Name</th><th>City</th></tr>'
 for(i = 0; i<arr.length; i++) {
 out += '<tr><td>'+arr[i].roll_no + '</td><td>' +arr[i].name + '</td><td>' +arr[i].city +
   '</td></tr>';
 }
 out+='</table>'
 document.getElementById("myID").innerHTML = out;
}
</script>
</body>
</html>
```

The output as obtained in previous section is displayed on the browser.

4.21.4 SQL

It is possible to retrieve data from database table. For that purpose we can use following steps -

(i) PHP script to read the contents of database table.

(ii) This PHP file can be invoked and accessed using **XMLHttpRequest** API.

Let us understand this concept with following example

Step 1 : Create a Database table named Student_table in MySQL as follows

Roll_No	Name	City
10	AAA	Pune
20	BBB	Mumbai
30	CCC	Chennai
40	DDD	Kolkatta

Step 2 : Now write a PHP script as follows

DBSelect.php

```
<html>
<head>
<title>PHP-Database Connectivity</title>
</head>
<body>
<?php
    $servername = "localhost";
    $username = "root";
    $password = "";
    $dbname="Student_DB";

    // Create connection
    $conn = new mysqli($servername, $username, $password,$dbname);

    // Check connection
    if ($conn->connect_error) {
     die("Connection failed: " . $conn->connect_error);
    }
    echo "Connected successfully";
    $sql = "SELECT * FROM Student_table";
    $result = $conn->query($sql);
    if ($result->num_rows > 0)
    {
        $outp = "[";
     while($rs = $result->fetch_assoc())
     {
           if ($outp != "[") {$outp .= ",";}
           $outp .= '{"Roll_No":"'  . $rs["Roll_No"] . '",';
           $outp .= '"Name":"'   . $rs["Name"]       . '",';
           $outp .= '"City":"'. $rs["City"]     . '"}';
     }
    }
    else
    {
     echo "0 results";
    }
    $conn->close();
?>
```

Step 3 : Now write JSON code in the following file

JSON_SQL_Demo.html

```
<!DOCTYPE html>
<html>
```

```
<head>
  <meta charset="UTF-8">
</head>
<body>

<div id="myID"></div>
<script>

var xmlhttp = new XMLHttpRequest();
var url = "localhost:8089/php-examples/DBSelect.php";

xmlhttp.onreadystatechange = function() {
     if (this.readyState == 4 && this.status == 200) {
      var stud_arr = JSON.parse(this.responseText);
      Student(stud_arr);
     }
};

xmlhttp.open("POST", url, true);
xmlhttp.send();

function Student(arr) {
 var out = "";
 var i;
 out+='<table border=1>'
 out+='<tr><th>Roll_No</th><th>Name</th><th>City</th></tr>'
 for(i = 0; i<arr.length; i++) {
  out += '<tr><td>'+arr[i].Roll_No + '</td><td>' +arr[i].Name + '</td><td>' +arr[i].City +
'</td></tr>';
 }
 out+='</table>'
 document.getElementById("myID").innerHTML = out;
}
</script>
</body>
</html>
```

The output as obtained in previous section is displayed on the browser.

Two Marks Questions with Answers

Q.1 What is the benefit of using JavaScript code in an HTML document ? **AU : Dec.-07, CSE**

Ans. : Following are some significant benefits of using JavaScript code in an HTML document

1. The client side data validation can be possible using JavaScript.
2. We can access the HTML elements. That means we can have control over HTML, BODY, TITLE tags.
3. Since JavaScript is based on the programming constructs we can control the logic of a simple scripting language. For example if a mouse pointer is moving over a text then change its colour.
4. JavaScript can determine the visitor's browser and can load the page accordingly.

Q.2 What is the need for client side scripting ? **AU : May-09, CSE**

Ans. : Following are the issues that get handled using client side scripting -

1. Simple scripting language like HTML or XML represents the data on the web page in simplest manner.
2. The scripting languages like JavaScript and DHTML are useful for enhancing the interactivity of the user with the web browser.
3. Mathematical evaluation is also possible using the client side scripting.
4. Using client side scripting languages the web page which gets designed for the user provides a graphical user interface using which the user can input his data in an elegant manner. This data can then be submitted to database or any server side scripting language in order to process it further.
5. The input validation can also be possible using client side scripting language like JavaScript.
6. Using cascading stylesheets with the client scripts the look and feel of the web page can be enhanced.

Q.3 List out the objects used in JavaScript with its purpose. **AU : May-09, CSE, May-12**

Ans. : Various objects that can be used in JavaScript are -

1. **Math Object -** This object provides useful methods that are required for mathematical computations.

   For example : For computing square root of some number sqrt(num) method is used.

   For finding out minimum value of a and b min(a,b) method is used.

2. **Number Object -** This object is used has various numeric properties.

   For example : Largest possible number gets displayed using MAX_VALUE, for displaying the pi value PI property is used.

3. **Date Object -** This object is used for obtaining date and time. For example using getTime() function the number of milliseconds can be obtained.
4. **Boolean Object -** This is the simplest object used to obtain true or false values.
5. **String Object -** This object provides many useful string related functionalities.

   For example : For finding out the substring of a given string substring(begin,end) method can be used.

Q.4 Comment on the statement. "Each object of a class has its own instance of static member variable." **AU : Dec.-08**

Ans. : When we create an object for a class then each object has its own distinct copies of variables. Such variables are called instance variables. For example if declare a class MyCar as follows

```
class MyCar
 {
private String color;
private String model;
private double price;
 }
```

This class has instance variables color, model and price. Now, each object of class MyCar will have its own values for these variables. These values will be static in nature. These objects can be stored at different memory locations. Thus every instance of the class(object) shares a class variables. Hence it is said that every object of a class has its own instance of static member variable.

Q.5 What is JavaScript statement ? Give an example. **AU : Dec.-11**

Ans. : The assignment statement in JavaScript is very much similar to C. For example

```
Sum+=10
Sum=sum+10
```

Can be written in JavaScript.

Q.6 Give any three uses of JavaScript.

Ans. :

1) The JavaScript can be used to create some web applications such as Calculator, Calender, Paint like applications.
2) JavaScript can be used to detect the visitor's browser.
3) JavaScript can be used to validate the data.

Q.7 What kind of comments are supported by the JavaScript ?

Ans. : JavaScript supports following comments -

The // is used to specify the single line comment.

The /* and */ can be used to specify the multi-line comments.

The XHTML <!- -> and <- -> is used in JavaScript.

Q.8 Explain how the string literals are used in JavaScript ?

Ans. : The string is a collection of characters. The string literal is used within the single quote or double quotes.

Q.9 What is the use of the word 'var' in the JavaScript ?

Ans. The word var is used to define the variables of numeric or string type.

Q.10 What do you mean by Coercion ?

Ans. : JavaScript supports the automatic type conversion. The coercion is a type conversion method. This is implicit type of conversion.

Q.11 What is the use of parseInt() and parseFloat() methods ?

Ans. :

The parseInt() and parseFloat() methods are used to separate out the integer and float values respectively from the string. For example parseInt("12abc") will return 12 from the string.

Q.12 What is the use of pop up boxes in JavaScript ?

Ans. : There are three types of popup boxes used in JavaScript. Using these popup boxes the user can interact with the web application.

Q.13 What is the use of toString method with respect to arrays ?

Ans. : The toString method converts the array to string.

Q.14 What are the advantages of indirectly embedding the JavaScript in the Web document ?

Ans. : Following are the advantages of indirectly embedding the JavaScript in the web document -

1. Script can be hidden from the browser.
2. The layout and presentation of web document can be separated out from the user interaction through the JavaScript.

Q.15 What are the properties and methods object window for screen output ?

Ans. : Following are some properties and methods of object window -

1. The property **write** of object window is used to display something on the screen.
2. The methods **alert, prompt** and **confirm** are useful for handling screen output and keyboard input.

Q.16 Explain array creation in JavaScript with example. **AU : May-11**

Ans. : In JavaScript the array can be created using Array object. Suppose, we want to create an array of 10 elements then we can write,

```
var ar = new Array(10);
```

Using new operator we can allocate the memory dynamically for the arrays. In the brackets the size of an array is mentioned and the var ar denotes the name of the array. Thus by the above sentence an array ar will be created in which we can store 10 elements at the most. Sometimes the above statement can be written like this

```
var ar;
ar=new Array(10);
```

Q.17 What are global functions in JavaScript ?

Ans. : The global functions are the top level functions in JavaScript that are independent of any specific object. These functions are built in objects of JavaScript. Examples of global functions are –

Name of the Function	Purpose
encodeURI	This function is used to encode URI
decodeURI	This function is used to decode the encoded URI
parseInt	This function parses the string and returns the integer value.
eval	The eval function is used to evaluate the expression.

Q.18 What is the meaning of widgets ?

Ans. : Various graphical controls such as text box, check box,radio buttons, push buttons and so on can be placed on the form for user interactivity. These control objects are called the widgets.

Q.19 State the types of java script statements with examples. **AU : Dec.-13**

Ans. : Various types of javascript statements are -

1. Simple assignment statements
2. Conditional statements
3. Object manipulation statements
4. Comment statements
5. Exception handling statements

Q.20 Write a Java Script to print "Good Day" using IF-ELSE condition. **AU : May-14**

Ans. :

```
<html>
<head>
```

Ans. : In JavaScript the array can be created using Array object. Suppose, we want to create an array of 10 elements then we can write,

```
var ar = new Array(10);
```

Using new operator we can allocate the memory dynamically for the arrays. In the brackets the size of an array is mentioned and the var ar denotes the name of the array. Thus by the above sentence an array ar will be created in which we can store 10 elements at the most. Sometimes the above statement can be written like this

```
var ar;
ar=new Array(10);
```

Q.17 What are global functions in JavaScript ?

Ans. : The global functions are the top level functions in JavaScript that are independent of any specific object. These functions are built in objects of JavaScript. Examples of global functions are –

Name of the Function	Purpose
encodeURI	This function is used to encode URI
decodeURI	This function is used to decode the encoded URI
parseInt	This function parses the string and returns the integer value.
eval	The eval function is used to evaluate the expression.

Q.18 What is the meaning of widgets ?

Ans. : Various graphical controls such as text box, check box,radio buttons, push buttons and so on can be placed on the form for user interactivity. These control objects are called the widgets.

Q.19 State the types of java script statements with examples. **AU : Dec.-13**

Ans. : Various types of javascript statements are -

1. Simple assignment statements
2. Conditional statements
3. Object manipulation statements
4. Comment statements
5. Exception handling statements

Q.20 Write a Java Script to print "Good Day" using IF-ELSE condition. **AU : May-14**

Ans. :

```
<html>
<head>
```

```
<script>
function MyMessage()
{
   var today=new Date();
   var h=today.getHours();
   if(h<12)//After 12 O'clock Say Good Bye
        //Before 12 O'clock say Good Day
     document.write("Good Day");
   else
        document.write("Good Bye");
}
</script>
</head>
<body onload="MyMessage()">
</body>
</html>
```

Q.21 List any four methods of date object. **AU : Dec.-16**

Ans.

Method	Meaning
getTime()	It returns the number of milliseconds. This value is the difference between the current time and the time value from 1st January 1970.
getDate()	Returns the current date based on computers local time.
getDay()	Returns the current day. The day number is from 0 to 6 i.e. from sunday to saturday.
getHours()	Returns the hour value ranging from 0 to 23, based on local time.
getSeconds()	Returns the second value ranging from 0 to 59, based on local time.

Q.22 How exceptions are handled in JavaScript ? **AU : May 17**

Ans. : The exceptions are handled in Javascript using try… catch block.s

For example –

```
<html>
 <head>
   <script type="text/javascript">
     <!--
       function myFunc()
       {
```

```
        var a = 100;
        var b = 0;
        try{
          if ( b == 0 ){
            throw( "Divide by zero error." );
          }
          else
          {
            var c = a / b;
          }
        }
        catch ( e ) {
          alert("Error: " + e );
        }
      }
    //-->
  </script>
 </head>
```

Q.23 What are object literal in JavaScript. **AU : Dec.-18**

Ans. : The object literal in javascript is a name-value pair wrapped in curly braces.

For example -

```
Var myobj = {
              name : "AAA",
              rollno : 2,
              marks : 82
            };
```

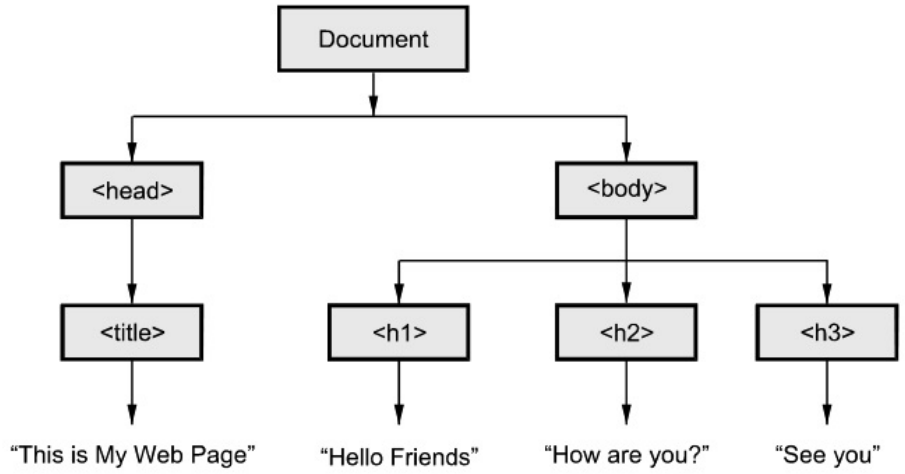
Q.24 What is DOM ?

Ans. : Document Object Model (DOM) is a set of platform independent and language neutral application programming interface (API) which describes how to access and manipulate the information stored in XML,XHTML and JavaScript documents.

Q.25 Define DOM Tree.

Ans. : The documents in DOM are represented using tree like structure in which every element is represented as a node. Such a tree like structure is called DOM Tree.

For example -



Q.26 List the different methods defined in document and window object of JavaScript.

AU : May-11, 13

Ans. : Methods in document object

Methods	Description
document.createAttribute()	Creates an attribute node.
document.createElement()	Creates an element node .
document.createTextNode()	Creates a text node.
document.getElementById()	Returns the element that has ID attribute with the specified value.
document.getElementByName()	Returns the element specified by its name.
document.write()	Writes HTML expression to a document.

Methods in window object

Methods	Description
alert()	Displays the alert box containing some message and OK button.
confirm()	Displays a dialog box with a message and OK and Cancel button.
createPopup()	Creates a pop up window.
moveTo()	Moves a window to some specific position.
open()	Opens a new window.
print()	Prints the contents in the window.

Q.27 What is the use of 'all' in DOM tree traversal ?

Ans. : The all is a object model collection which is used to refer all the HTML elements. For representing all the elements present on the HTML document the 'all' is used.

Q.28 List out various level of Document Object Modelling.

Ans. : Various levels of DOM are -

DOM0 DOM1 DOM2 DOM3

Q.29 What is getElementById method ?

Ans. : The getElementById is a method which is defined in DOM1. The element access can be made by using this method. For example -

```
var Dom_Obj=document.getElementById("myinput");
```

Q.30 What is the drawback of the DOM0 event Model ?

Ans. : The main drawback of DOM0 event model is that two different event handlers can not be assigned to the same event. This drawback is overcome in DOM2 event model.

Q.31 What do you mean by the term event registration ?

Ans. : The process of connecting event handler to an event is called event registration. The event handler registration can be done using two methods -

- Assigning the tag attributes
- Assigning the handler address to object properties.

Q.32 Define the term intrinsic events. **AU : May-12**

Ans. : Event is an activity that represents a change in the environment. For example mouse clicks, pressing a particular key of keyboard represents the events. Such events are called intrinsic events.

Q.33 List some HTML intrinsic attributes. **AU : May-11, 12**

Ans. : Some of the intrinsic attributes are listed in the following table

Intrinsic Attribute	Meaning
Onblur	This event is for losing the focus
onchange	On occurrence of some change this event occurs
onclick	When user clicks the mouse button this event occurs
ondblclick	When user double clicks the mouse button.

Q.34 Define event bubbling. **AU : May-10**

Ans. : Suppose, there is an element present inside another element. Then during the event handling, if the event which is present in the inner element is handled and then the event of the outer element is handled. This process of event handling is called event bubbling.

Q.35 List the types of event listeners in DOM2. **AU : Dec.-12**

Ans. : There are three types of event listeners in DOM2 -

1. **Capturing listeners :** The capturing listener is associated with the event that occurs at the root node in the document tree and propagates towards the target node. It is created with the call to addEventListener().
2. **Bubbling listeners :** The bubbling listener is associated with the event that starts at the target node and moves up the document tree to the root node. It is created with the call to addEventListener().
3. **Target listeners :** Target listener is the event listener which is added in the target node.

Q.36 What is meant by DHTML ? **AU : Dec.-13**

Ans. : The DHTML stands for Dynamic HTML. This is a markup language used to create interactive and animated web site.

Q.37 Define event programming. Name any two of its techniques. **AU : Dec.-15**

Ans. : • Event is an activity that represents a change in the environment. For example mouse clicks, pressing a particular key of keyboard represent the events. Event programming is technique in which such events are handled using the code.

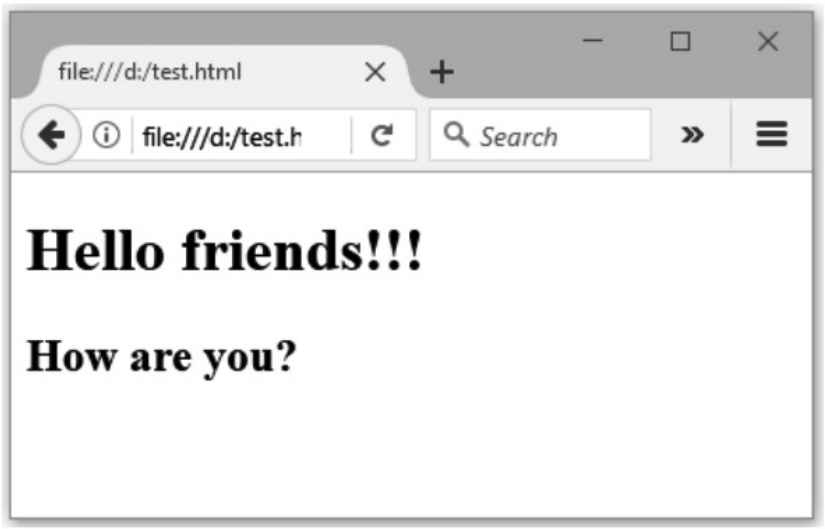
- Event capturing and event bubbling are the two techniques of event propagation.

Q.38 Write appropriate JavaScript code to remove an element(current element) from a DOM. **AU : May-16**

Ans. :

```
<!DOCTYPE html>
<html>
<body>
 <div id="headers">
<h1 id="s1">Hello friends!!!</h1>
<h2 id="s2">How are you?</h2>
<h3 id="s3"> See you...</h3>
</div>
<script>
 var parent = document.getElementById("headers");
 var child = document.getElementById("s3");
 parent.removeChild(child);
</script>
</body>
</html>
```

Output


file:///d:/test.html
file:///d:/test.h
Search
Hello friends!!!
How are you?

❑❑❑