Unit-3

Introduction to JavaScript

Content

- ✓JavaScript Basics
- ✓Strings, Arrays, Functions
- ✓Objects in JavaScript
- ✓building simple applications using JavaScript and HTML

Introduction

→ JavaScript is:

→ An object-oriented scripting language that is designed primarily for people who are building web pages using HTML

→ JavaScript programs are embedded within HTML documents in source code form.

→ The script is interpreted by the browser.

→ Platform independence.

Contd…

→Javascript is object-oriented.

→It allows interaction with the properties of object that it recognizes.

→Internal built-in objects(e.g. document object)

→Browser objects (e.g. window object).

→Javascript program can run both on the client and the server sides

→Client side:

-- By the browser after downloading the HTML document

→Server side:

--requires Netscape Livewire environment

Javascript Objects and Methods

→In the context of Javascript:

→An object is a collection of properties and methods which can be viewed, modified and interacted.

→ A simple example of a property is color, which is rather easy to visualize.

→They can be directly manipulated by referring to the object and the property by name and then setting its value.

→Example: The background color of a page can be changed as:

document.bgcolor="blue"

Contd.

→In the object-oriented paradigm, methods refer to functions that can be used to manipulate objects and their properties.

→Example: The method write(), which when invoked in the document object, causes a specific string of characters to be outputted.

```
document.write("Hello , Welcome to Javascript world!");
```

Contd.

→ Methods which are defined on an object give the range of choices available for interacting with the object.

→ Examples:

- → A *window* object can be opened or closed using the *open()* and *close()* methods respectively.
- → A *form* object has a *submit()* method which transmits the contents of the form to the web server.
- → The sequential list of a user's path through a member of URLs is represented by the *history* object, which has *forward()* and *backward()* methods to move through the list.

Contd.

→A part from the pre-defined methods, it is also possible to create user-defined methods.

→Control the rate with which a line of text scrolls across the screen.

→Determine the path of an animated object across the display.

→Event handlers can be coded in Javascript.

→They are triggered in response to certain conditions in some objects.

→Can be used to control the sequence of activities in response to some object state.

Running Scripts/Embedding script

→Scripts written in Javascript must either be embedded within an HTML document or be referenced as an external file that is loaded with the HTML document.

→ All recent browsers can detect and interpret inline Javascript code directly.

→ The <SCRIPT> tag is used.

→ Syntax:

```
<script language="javascript">
        //JavaScript code goes here
</script>
```

Including external JavaScript files

How to include external JS filel

→Can be done using the src attribute in the <script> tag.

Example:

```
<script src="../../codes/test.js">
</script>
```

→ This behaves exactly as if the contents of the specified javascript file appeared directly between the tags.

Example: Referring external JavaScript file

HTML file: MyWebpage.htm

```
<html>
<script src="MyJavascript.js"
  language="javascript"></script>
<body>
</body>
  <form>
        <input type="button"
  name="btMsg" value="ClickHere"
  onClick="DisplayMsg();">
   </form>
</html>
```

JavaScript file: MyJavascript.js

```
//Add your script
function DisplayMsg()
{
   alert("Text messge displayed on the Pop-
   up window is taken from external
   javascript file");
}
```

Advantages:

→Less code has to be written and stored.

→Commonly used javascript code can be shared among several pages.

→Only a single copy needs to be stored on the web server.

→Such javascript files can be cached by the browser, allowing faster page loading.

→The URL specified in the src attribute can also refer to another web server.

Contd.

→Language="JavaScript" is optional.

→For browsers that do not understand Javascript, it is possible to use the HTML comment tag "<!--" & "-->" to bracket out the Javascript code.

→ The marked block is treated as hidden by the browsers that do not understand Javascript.

→These tags are ignored by browsers that can interpret Javascript.

Contd.

language: This attribute specifies what scripting language you are using. Typically, its value will be *javascript*

type: This attribute is what is now recommended to indicate the scripting language in use and its value should be set to *"text/javascript"*.

Example:

```
<script language="javascript" type="text/javascript">
        JavaScript code
</script>
```

General HTML page with JS

→ Javascript code snippet can be inserted anywhere in the HTML document.

```
<HTML>
    <HEAD>
        <TITLE>……..</TITLE>
        <SCRIPT LANGUAGE = "Javascript">
            <!-- java script code-->
        </SCRIPT>
    </HEAD>
     <BODY>……..</BODY>
</HTML>
```

HTML Code

JavaScript code

JavaScript Data Types

→JavaScript variables can hold many data types: numbers, strings, arrays, objects and more

Example:

```
→var length = 16;                                  // Number
→var lastName = "Johnson“;                         // String
→var cars = ["Saab", "Volvo", "BMW"];              // Array
→var x = {firstName:"John", lastName:"Doe"};    // Object
```

→JavaScript has dynamic types. This means that the same variable can be used as different types:

Example:

```
var x;               // Now x is undefined
var x = 5;           // Now x is a Number
var x = "John";      // Now x is a String
```

Javascript program:Example1

```
<html>
   <head>
         <title>javascript program demo</title>
   </head>
   <body>
         <script>
               document.writeln("<H1>Hello Good Day </H1>");
               document.writeln("<H1>Welcome to Javascript code</H1>");
         </script>
   </body>
</html>
```

Example2

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <pre>
        <script language="javascript">
            var num1 = 10;
            var num2 = 20;
            var result = num1 + num2;
            document.writeln("Sum of" + "  " + num1 + "  " + num2 + " "
                    + "  " + "is" + "  " + result);
        </script>
    </pre>
</body>
</html>
```

JavaScript function

→A function is a group of reusable code which can be called anywhere in your program.

→ function keyword, followed by a unique function name, a list of parameters (that might be empty), and a statement block surrounded by curly braces.

Syntax:

```
<html>
<head>
    <title>Function Demo</title>
</head>
<body>
    <script type="text/javascript">
        function functionname(parameter-list)
        {
            statements
        }
    </script>
</body>
</html>
```

Calling function

```
<html>
<head>
<script type="text/javascript">
         function sayHello()
         {
            document.write ("Hello there!");
         }
      </script>

</head>
<body>
    <p>Click the following button to call the function</p>
    <form>
        <input type="button" onclick="sayHello()" value="Say Hello">
    </form>
    <p>Use different text in write method and then try...</p>
</body>
</html>
```

Exercise: Javascript to Add two numbers using the function

```
<!DOCTYPE html>
<html>
<head>
<title>Addition of two numbers</title>
<script type="text/javascript">
    function Addnumbers() {
        var num1 = parseInt(document.getElementById('txtnumber').value);
        var num2 = parseInt(document.getElementById('txtsecnumber').value);
        var result = num1 + num2;
        document.getElementById('result').innerHTML = result;
    }
</script>
</head>
<body>

    <form>
        <label>Enter first Number:</label> <input type="number" id="txtnumber">
        <label>Enter second Number:</label> <input type="number"
            id="txtsecnumber">
        <button type="button" id="btncal" onclick="Addnumbers()">Calculate</button>
        <p id="result"></p>
    </form>
</body>
</html>
```

Example2

```
<html>
    <body>
        <h1>My First JavaScript</h1>
        <button type="button“ onclick="document.getElementById('demo').innerHTML
          = Date()">
        Click me to display Date and Time.</button>
        <p id="demo"></p>
    </body>
</html>
```

Property	Content Type	HTML Tags	Use Case
innerText	Plain text	Ignored	Get or display raw text content
innerHTML	HTML content	Included	Modify element content with formatting

Example3:Use of functions & events

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<title>Insert title here</title>
<script language="javascript">
    function HelloMessage()
    {
        var txt_name = document.getElementById('txtname').value;
        document.write("<h1>Hello!"+" "+ txt_name + " "+ "welcome to Javascript programming");
    }
</script>
</head>
<body>
<form>
    <label>Name:</label>
    <input type="text" id="txtname">
    <input type="button" id="btnsubmit" value="Submit" onclick="HelloMessage();">
</form>
</body>
</html>
```

Types of operators in JavaScript.

→ Arithmetic Operators

Operator	Description	Example
+	Addition	10+20 = 30
-	Subtraction	20-10 = 10
*	Multiplication	10*20 = 200
/	Division	20/10 = 2
%	Modulus (Remainder)	20%10 = 0
++	Increment	var a=10; a++; Now a = 11
--	Decrement	var a=10; a--; Now a = 9

Comparison Operators

Operator	Description	Example
==	Is equal to	10==20 = false
===	Identical (equal and of same type)	10==20 = false
!=	Not equal to	10!=20 = true
!==	Not Identical	20!==20 = false
>	Greater than	20>10 = true
>=	Greater than or equal to	20>=10 = true
<	Less than	20<10 = false
<=	Less than or equal to	20<=10 = false

Bitwise & Logical Operators

Operator	Description	Example
&	Bitwise AND	(10==20 & 20==33) = false
|	Bitwise OR	(10==20 | 20==33) = false
^	Bitwise XOR	(10==20 ^ 20==33) = false
~	Bitwise NOT	(~10) = -10
<<	Bitwise Left Shift	(10<<2) = 40
>>	Bitwise Right Shift	(10>>2) = 2
>>>	Bitwise Right Shift with Zero	(10>>>2) = 2
Operator	Description	Example
&&	Logical AND	(10==20 && 20==33) = false
||	Logical OR	(10==20 || 20==33) = false
!	Logical Not	!(10==20) = true

Assignment Operators

Operator	Description	Example
=	Assign	10+10 = 20
+=	Add and assign	var a=10; a+=20; Now a = 30
-=	Subtract and assign	var a=20; a+=10; Now a = 10
*=	Multiply and assign	var a=10; a*=20; Now a = 200
/=	Divide and assign	var a=10; a/=2; Now a = 5
%=	Modulus and assign	var a=10; a%=2; Now a = 0

JavaScript: Decision & looping statements

→JavaScript supports conditional statements which are used to perform different actions based on different conditions.

1. If Statement, If.. else statement, if.. else.. if statement
2. switch statement
3. Four types of loops in JavaScript.
    1. for loop
    2. While loop
    3. do-while loop
    4. for-in loop

Examples: if, if..else, if..else if

```
<html>
<head>
<title>IF Statement demo</title>
</head>

<body>
    Enter Value:
    <input type="number" id="txtnumber">
    <input type="button" id="btnSubmit" value="Submit"
        onclick="CompareValue();">
</body>
<script type="text/javascript">
    function CompareValue() {
        var txtinput = parseInt(document.getElementById("txtnumber").value);
        if (txtinput >= 35)
            alert("Entered value is:"
                    + parseInt(document.getElementById("txtnumber").value));
        else
            alert("Input value is less than passing score!!!!!!!!!!");
    }
</script>
</html>
```

Contd…

```
<script  type="text/javascript">
var a=20;
if(a%2==0){
document.write("a is even number");
}
else{
document.write("a is odd number");
}
</script>
```

```
<script  type="text/javascript">
var a=20;
if(a==10){
document.write("a is equal to 10");
}
else if(a==15){
document.write("a is equal to 15");
}
else if(a==20){
document.write("a is equal to 20");
}
else{
document.write("a is not equal to 10, 15 or 20
");
}
</script>
```

Contd…

```
<script  type="text/javascript">
      var grade='B';
      var result;
      switch(grade){
            case 'A': result="A Grade";
                  break;
            case 'B':  result="B Grade";
                  break;
            case 'C': result="C Grade";
                  break;
            default: result="No Grade";
      }
      document.write(result);
</script>
```

JavaScript: Loops

```
<script  type="text/javascript">
for (i=1; i<=5; i++)
{
            document.write(i + "<br/>")
}
</script>
```

```
<script  type="text/javascript">
var i=11;
while (i<=15)
 {
      document.write(i + "<br/>");
       i ++;
  }
</script>
```

```
<script  type="text/javascript">
var i=21;
do
   {
            document.write(i + "<br/>");
            i++;
   }while (i<=25);
</script>
```

The purpose of loop is to execute piece of code repeatedly until specified condition is satisfied.

for-in loop

Syntax:

```
for (variableName in Object) {
    statement(s);
}
```

Event Handling

→ An *event* is a notification that something specific has occurred, either in the browser, such as the completion of the loading of a document, or a browser user action, such as a mouse click on a form button.

→ An event handler is a script that is implicitly executed in response to the appearance of an event.

→ Event handlers enable a Web document to be responsive to browser and user activities.

→ Example: Check for simple errors and omissions in user input to the elements of a form

Example: Onclick event

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
<script type="text/javascript">
    function MyJSfuntion() {
        alert("Hello! This function is called when user clicks button");
    }
</script>
</head>
<body>
    <button type="button" onclick="MyJSfuntion();">TriggerJSFun</button>
</body>
</html>
```

Math Object Methods

Method	Description
abs(x)	Returns the absolute value of x
acos(x)	Returns the arccosine of x, in radians
asin(x)	Returns the arcsine of x, in radians
atan(x)	Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians
atan2(y,x)	Returns the arctangent of the quotient of its arguments
ceil(x)	Returns x, rounded upwards to the nearest integer
cos(x)	Returns the cosine of x (x is in radians)
exp(x)	Returns the value of E^x
floor(x)	Returns x, rounded downwards to the nearest integer
log(x)	Returns the natural logarithm (base E) of x
max(x,y,z,...,n)	Returns the number with the highest value
min(x,y,z,...,n)	Returns the number with the lowest value
pow(x,y)	Returns the value of x to the power of y
random()	Returns a random number between 0 and 1
round(x)	Rounds x to the nearest integer
sin(x)	Returns the sine of x (x is in radians)
sqrt(x)	Returns the square root of x
tan(x)	Returns the tangent of an angle

Confirm Box

```
<body>
        <script>
            function respond()
            {
                        confirm("Are you sure want to continue!");
            }
         </script>
           <H2>Click on the following button</h2>
         <form>
              <table>
                    <tr>
                          <td >
                     <input style="width:140; font-size:18pt;" type="button"  value="Click here" onClick="respond()">
                           </td>
                    </tr>
                </table>
           </form>
      </body>
```

Confirm Box: properties

→The confirm() method returns a boolean value.

1. “true” if “OK” is pressed
2. “false” if “Cancel” is pressed

→The return value can be used in decision logic.

Javascript Alerts

→A Javascript alert box, when clicked, displays a text, and waits until the visitor presses the “OK” button.

→Syntax:

alert(“Any Message to be displayed.”);

Example: Alert

```
<script>
        function GotoPage()
        {
                alert("Hello!");
                alert("Good Day!");
                 window.location = "http://www.sdmcet.ac.in";
        }
</script>
<form>
      <table>
            <tr>
                <td >
            <input style="width:140; font-size:18pt;" type="button"  value="Click Here" onClick="GotoPage()">
                 </td>
              </tr>
          </table>
</form>
```

Example 5:Alert

```
<script>
   function respond()
   {
            alert("Hello, Welcome to scripting language");
            alert("Your are watching pop up window");
            window.location="http://www.sdmcet.ac.in";
    }
</script>
 <form>
            <input type="button" value="Click Here" onClick="respond();">
 </form>
```

Example6: Mathematical Calculation

→This example will show how to:

→Extract data from a form field into a variable.

→Perform mathematical calculations on that variable.

→Store the result back into a form field.

Example: temperature conversion

```
<script>
function Change_Temp(myform)
{
var
cent=parseFloat(myform.option.
value);
fahr = (cent * 1.8) + 32;
myform.Fahr.value = fahr;
}
</script>
```

```
<form>
   <table>
         <tr>
          <td>
          Centigrade:<input style="width:140; font-size:18pt;"
type="text"  name="option">
          </td>
<td >
<input style="width:140; font-size:18pt;" type="button"
value="Convert" onClick="Change_Temp(this.form);">
</td>
<td>
Fahrenheit:<input style="width:140; font-size:18pt;"
type="text"  name="Fahr">
                                        </td>
                                  </tr>
                            </table>
                      </form>
```

Form Validation Using JavaScript

Form validation is done to check the accuracy of the user's entered information before they could submit the form.

Form/Control validation

→Javascript can be used for client-side validation

→Data entered can be extracted and accessed.

→Checks can be made on the data entered.

-- Non-alphabetic characters in a name.

-- Non-numeric characters in roll number, age, etc.

--Example:

-- To check the age limit for particular positions.

Primitive type values-'null' & 'undefined'

→JavaScript includes two additional primitive type values

1. 'null'
2. 'undefined'

→null - A null means the absence of a value.

→A null value evaluates to false in the conditional expression.

→Does not apply to.

→Unknown.

→The value is known but absent

→ undefined – undefined means lack of value or unknown value.

→An undefined evaluates to false when used in the conditional expression.

Example 1:Control validation

```
<html>
<body>
   <script>
   function checkvalue() {
   var mystring = document.getElementById('myString').value;
   if(mystring == null || mystring == "") {
      alert ('Empty value is not allowed');
      return false;
   }
```

Contd.

```
else {
        alert("correct input");
        return true;
    }
}
</script>
</body>
<form onsubmit="checkvalue()">
    <input name="myString" type="text" value='' id="myString">
    <input type="submit" value="ClickTosubmit" />
</form>
</html>
```

Example: Page redirection

```
<html>
 <head>
  <title>Page Redirection</title>
 </head>
 <body>
  <script>
        function RedirectTo()
        {
          window.location = "http://www.sdmcet.ac.in";
        }
        alert("You will be redirected in five seconds");
        setTimeout('RedirectTo()', 5000);
  </script>
  </body>
</html>
```

Example: new window

```
<html>
 <head>
  <title>Page Redirection</title>
 </head>
 <body>
 </body>
  <form>
  <input type="button" value="new window" onClick
  ="window.open('http://www.sdmcet.ac.in','mywindow','width=400','height=200','toolbar=yes'
  , 'status=yes')">
</form>
</html>
```