What is JavaScript?

 JavaScript was designed to add interactivity to HTML pages.

 JavaScript is a scripting language.

 A scripting language is a lightweight programming language

 JavaScript is usually embedded directly into HTML pages.

 JavaScript is an interpreted language. (means that scripts execute without preliminary compilation)

 Everyone can use JavaScript without purchasing a license.

What Can JavaScript do?

 JavaScript gives HTML designers a programming tool –
 HTML authors are normally not programmers, but JavaScript is a scripting language with a very simple syntax! Almost anyone can put small "snippets" of code into their HTML pages.
 JavaScript can react to events –
 A JavaScript can be set to execute when something happens, like when a page has finished loading or when a user clicks on an HTML element.
 JavaScript can manipulate HTML elements –
 A JavaScript can read and change the content of an HTML element.
 JavaScript can be used to validate data –
 A JavaScript can be used to validate form input
 JavaScript can be used to detect the visitor's browser –
 A JavaScript can be used to detect the visitor's browser, and - depending on the browser - load another page specifically designed for that browser.
 JavaScript can be used to create cookies –

 A JavaScript can be used to store and retrieve information on the visitor's computer.

<script> tag

 The HTML <script> tag is used to insert a JavaScript into an HTML document.
 Inside the <script> tag use the type attribute to define the scripting language.
 The <script> and </script> tells where the JavaScript starts and ends.
 The lines between the <script> and </script> contain the JavaScript and are executed by the browser.

Manipulating HTML Elements

 JavaScript is typically used to manipulate existing HTML elements.
 The HTML "id" attribute is used to identify HTML elements.
 To access an HTML element from a JavaScript, use the document.getElementById() method.
 The document.getElementById() method will access the HTML element with the specified id.

Example 1

Access the HTML element with the specified id, and change its content:

```
<html>
<body>
<h1>My Web Page</h1>
<p id="demo">A Paragraph.</p>
<script type="text/javascript">
document.getElementById("demo").innerHTML="My First JavaScript";
</script>
```

</body>
</html>

JavaScript Functions and Events

 The JavaScript statement in the example above, is executed when the page loads, but that is not always what we want.

 Sometimes we want to execute a JavaScript when an event occurs, such as when a user clicks a button.

 Then we put the script inside a **function**.

 Functions are normally used in combination with **events**.

JavaScript Functions in <head>

Example 2

```
<html>
<head>
<script type="text/javascript">
 function myFunction()
{
document.getElementById("demo").innerHTML="My First JavaScript Function";
}
 </script>
</head>
<body>
<h1>My Web Page</h1>
<p id="demo">A Paragraph</p>
<button type="button" onclick="myFunction()">Try it</button>
</body> </html>
```

3 Places to put JavaScript code

1. Between the body tag of html(As in Example 1)

2. Between the head tag of html(As in Example 2)

3. In .js file (external JavaScript)

Using an External JavaScript

 JavaScript can also be placed in external files.

 External JavaScript files often contain code to be used on several different web pages.

 External JavaScript files have the file extension .js.

 To use an external script, point to the .js file in the "src" attribute of the <script> tag:

Eg:

```
<html>
 <body>
<script type="text/javascript" src="myScript.js">
</script>
 </body>
 </html>
```

myScript.js

```
function msg(){
alert("Hello ");
}
```

JavaScript Variables

 JavaScript variables are used to hold values or expressions.

Rules for JavaScript variable names:

 Variable names are case sensitive (y and Y are two different variables)

 Variable names must begin with a letter, the $ character, or the underscore character.

Declaring (Creating) JavaScript Variables

```
var carname;
carname="Volvo";
```

Browser Object Model

The **Browser Object Model** (BOM) is used to interact with the browser.

 The default object of browser is window means you can call all the functions of window by specifying window or directly. For example:
window.alert("hello"); is same as: alert("hello");

 You can use a lot of properties (other objects) defined underneath the window object like document, history, screen, navigator, location, innerHeight, innerWidth,

Window Object

 The window object represents a window in browser.

 An object of window is created automatically by the browser.

 Window is the object of browser, it is not the object of javascript.

 The javascript objects are string, array, date etc.

Methods of window object

Method	**Description**
alert()	displays the alert box containing message with ok button.
confirm()	displays the confirm dialog box containing message with ok and cancel button.
prompt()	displays a dialog box to get input from the user.
open()	opens the new window.

close()	closes the current window.
setTimeout()	performs action after specified time like calling function, evaluating expressions etc.

Example of alert() in javascript

It displays alert dialog box. It has message and ok button.

```
<script type="text/javascript">
function msg(){
alert("Hello Alert Box");
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

Example of confirm() in javascript

It displays the confirm dialog box. It has message with ok and cancel buttons.

```
<script type="text/javascript">
function msg(){
var v= confirm("Are u sure?");
if(v==true){
alert("ok");
}
else{
alert("cancel");
}
}
</script>
```

<input type="button" value="delete record" onclick="msg()"/>

Example of prompt() in javascript

It displays prompt dialog box for input. It has message and textfield.

```
<script type="text/javascript">
function msg(){
var v= prompt("Who are you?");
alert("I am "+v);
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

Example of open() in javascript

It displays the content in a new window.

```
<script type="text/javascript">
function msg(){
open("http://www.softentec.com");
}
</script>
<input type="button" value="Soften Technologies" onclick="msg()"/>
```

Example of setTimeout() in javascript

It performs its task after the given milliseconds.

```
<script type="text/javascript">
function msg(){
setTimeout(
function(){
alert("Welcome after 2 seconds") },2000);
```

```
}
</script>
<input type="button" value="click" onclick="msg()"/>
```

Document Object Model

 The **document object** represents the whole html document.

 When html document is loaded in the browser, it becomes a document object.

 It is the **root element** that represents the html document.

 As mentioned earlier, it is the object of window.

 So window.document Is same as document

 According to W3C - *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*

Methods of document object

 We can access and change the contents of document by its methods.

 The important methods of document object are as follows:

Method	Description
write("string")	writes the given string on the doucment.
writeln("string")	writes the given string on the doucment with newline character at the end.
getElementById()	returns the element having the

	given id value.
getElementsByName()	returns all the elements having the given name value.
getElementsByTagName()	returns all the elements having the given tag name.
getElementsByClassName()	returns all the elements having the given class name.

Javascript - document.getElementsByName() method

 The document.getElementsByName() method returns all the element of specified name.

The syntax of the getElementsByName() method is given below:
document.getElementsByName("name")
Here, name is required.

Example of document.getElementsByName() method

 In this example, we going to count total number of genders.

 Here, we are using getElementsByName() method to get all the genders.

```
<script type="text/javascript">
function totalelements()
{
var allgenders=document.getElementsByName("gender");
alert("Total Genders:"+allgenders.length);
}
</script>
<form>
Male:<input type="radio" name="gender" value="male">
```

Female:<input type="radio" name="gender" value="female">

<input type="button" onclick="totalelements()" value="Total Genders">
</form>

The document.getElementsByTagName() method returns all the element of specified tag name.

The syntax of the getElementsByTagName() method is given below:

document.getElementsByTagName("name")
Here, name is required.

Example of document.getElementsByTagName() method

In this example, we going to count total number of paragraphs used in the document. To do this, we have called the document.getElementsByTagName("p") method that returns the total paragraphs.

```
<script type="text/javascript">
function countpara(){
var totalpara=document.getElementsByTagName("p");
alert("total p tags are: "+totalpara.length);
}
</script>
<p>This is a pragraph</p>
<p>Here we are going to count total number of paragraphs by getElementByTagName() method.
</p>
<p>Let's see the simple example</p>
<button onclick="countpara()">count paragraph</button>
```

JavaScript Form Validation

 It is important to validate the form submitted by the user because it can have inappropriate values.

 So validation is must.

 The JavaScript provides you the facility the validate the form on the client side so processing will be fast than server-side validation.

 So, most of the web developers prefer JavaScript form validation.

 Through JavaScript, we can validate name, password, email, date, mobile number etc
fields.

JavaScript form validation example

 In this example, we are going to validate the name and password.

 The name can't be empty and password can't be less than 6 characters long.
 Here, we are validating the form on form submit.

 The user will not be forwarded to the next page until given values are correct.

```
<script>
function validateform(){
var name=document.myform.name.value;
var password=document.myform.password.value;
if (name==null || name==""){
```

```
alert("Name can't be blank");
return false;
}else if(password.length<6){
alert("Password must be at least 6 characters long.");
return false;
}
}
</script>
<body>
<form name="myform" method="post" action="abc.jsp" onsubmit="return validateform()" >
Name: <input type="text" name="name"><br/>
Password: <input type="password" name="password"><br/>
<input type="submit" value="register">
</form>
```

JavaScript email validation

 We can validate the email by the help of JavaScript.

 There are many criteria that need to be follow to validate the email id such as:

 email id must contain the @ and . character

 There must be at least one character before and after the @.

 There must be at least two characters after . (dot).

Let's see the simple example to validate the email field.

```
<script>
```

```
function validateemail()
{
var x=document.myform.email.value;
var atposition=x.indexOf("@");
var dotposition=x.lastIndexOf(".");
if (atposition<1 || dotposition<atposition+2 || dotposition+2>=x.length){
alert("Please enter a valid e-mail address \n atpostion:"+atposition+"\n dotposition:"+dotposition);
return false;
}
}
</script>
<body>
<form name="myform" method="post" action="#" onsubmit="return validateemail();">
Email: <input type="text" name="email"><br/>
<input type="submit" value="register">
</form>
```

JavaScript Retype Password Validation

```
<script type="text/javascript">
function matchpass(){
var firstpassword=document.f1.password.value;
var secondpassword=document.f1.password2.value;
if(firstpassword==secondpassword){return true;

}
else{
alert("password must be same!");
return false;
}
```

```
}
</script>
<form name="f1" action="register.jsp" onsubmit="return matchpass()">
Password:<input type="password" name="password" /><br/>
Re-enter Password:<input type="password" name="password2"/><br/>
<input type="submit">
</form>
```