JavaScript is used to create interactive websites. It is mainly used for:

* Client-side validation,
* Dynamic drop-down menus,
* Displaying date and time,
* Displaying pop-up windows and dialog boxes (like an alert dialog box, confirm dialog box and prompt dialog box),
* Displaying clocks etc.

**<script>**

document.write("Hello JavaScript by JavaScript");

**</script>**

Javascript example is easy to code. JavaScript provides **3 places** to put the JavaScript code: within body tag, within head tag and external JavaScript file.

3 Places to put JavaScript code

* Between the body tag of html
* Between the head tag of html
* In .js file (external javaScript)

The **script** tag specifies that we are using JavaScript.

The **text/javascript** is the content type that provides information to the browser about the data.

The **document.write()** function is used to display dynamic content through JavaScript. We will learn about document object in detail later.

**<script** type="text/javascript"**>**

 alert("Hello Blue Pearl Computer Education");

**</script>**

**External**

**<script** type="text/javascript" src="message.js"**></script>**

Types of JavaScript Comments

There are two types of comments in JavaScript.

* Single-line Comment
* Multi-line Comment

**<script>**

// It is single line comment

document.write("hello javascript");

**</script>**

**<script>**

/\* It is multi line comment.

It will not be displayed \*/

document.write("example of javascript multiline comment");

**</script>**

**JavaScript Variable**

A **JavaScript variable** is simply a name of storage location. There are two types of variables in JavaScript : local variable and global variable.

There are some rules while declaring a JavaScript variable (also known as identifiers).

* Name must start with a letter (a to z or A to Z), underscore( \_ ), or dollar( $ ) sign.
* After first letter we can use digits (0 to 9), for example value1.
* JavaScript variables are case sensitive, for example x and X are different variables.

Correct JavaScript variables

var x = 10;

var \_value="sonoo";

**<script>**

var x = 10;

var y = 20;

var z=x+y;

document.write(z);

**</script>**

JavaScript local variable

A JavaScript local variable is declared inside block or function. It is accessible within the function or block only. For example:

**<script>**

function abc(){

var x=10;//local variable

}

**</script>**

JavaScript global variable

A **JavaScript global variable** is accessible from any function. A variable i.e. declared outside the function or declared with window object is known as global variable. For example:

**<script>**

var data=200;//gloabal variable

function a(){

document.writeln(data);

}

function b(){

document.writeln(data);

}

a();//calling JavaScript function

b();

**</script>**

**Javascript Data Types**

JavaScript provides different **data types** to hold different types of values. There are two types of data types in JavaScript.

* Primitive data type
* Non-primitive (reference) data type

JavaScript is a **dynamic type language**, means you don't need to specify type of the variable because it is dynamically used by JavaScript engine. You need to use **var** here to specify the data type. It can hold any type of values such as numbers, strings etc. For example:

var a=40;//holding number

var b="Rahul";//holding string

JavaScript primitive data types

There are five types of primitive data types in JavaScript. They are as follows:

|  |  |
| --- | --- |
| Data Type | Description |
| String | represents sequence of characters e.g. "hello" |
| Number | represents numeric values e.g. 100 |
| Boolean | represents boolean value either false or true |
| Undefined | represents undefined value |
| Null | represents null i.e. no value at all |

JavaScript non-primitive data types

The non-primitive data types are as follows:

|  |  |
| --- | --- |
| Data Type | Description |
| Object | represents instance through which we can access members |
| Array | represents group of similar values |
| RegExp | represents regular expression |

**JavaScript Operators**

JavaScript operators are symbols that are used to perform operations on operands. For example:

var sum=10+20;

Here, + is the arithmetic operator and = is the assignment operator.

There are following types of operators in JavaScript.

* Arithmetic Operators
* Comparison (Relational) Operators
* Logical Operators
* Assignment Operators
* Special Operators

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations on the operands. The following operators are known as 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 |

JavaScript Comparison Operators

The JavaScript comparison operator compares the two operands. The comparison operators are as follows:

|  |  |  |
| --- | --- | --- |
| 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 |

JavaScript Bitwise Operators

The bitwise operators perform bitwise operations on operands. The bitwise operators are as follows:

|  |  |  |
| --- | --- | --- |
| 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 |

JavaScript Logical Operators

The following operators are known as JavaScript logical operators.

|  |  |  |
| --- | --- | --- |
| Operator | Description | Example |
| && | Logical AND | (10==20 && 20==33) = false |
| || | Logical OR | (10==20 || 20==33) = false |
| ! | Logical Not | !(10==20) = true |

JavaScript Assignment Operators

The following operators are known as JavaScript 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 |

**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 world");

is same as:

alert("hello world");

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

7



Methods of window object

The important methods of window object are as follows:

|  |  |
| --- | --- |
| 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.bluepearl.co.in");

}

**</script>**

**<input** type="button" value="bluepearl computer education" 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 to Blue pearl after 2 seconds")

},2000);

}

**</script>**

**<input** type="button" value="click" onclick="msg()"**/>**

**JavaScript History Object**

The **JavaScript history object** represents an array of URLs visited by the user. By using this object, you can load previous, forward or any particular page.

The history object is the window property, so it can be accessed by:

window.history  Or history

Property of JavaScript history object

There are only 1 property of history object.

|  |  |  |
| --- | --- | --- |
| No. | Property | Description |
| 1 | length | returns the length of the history URLs. |

Methods of JavaScript history object

There are only 3 methods of history object.

|  |  |  |
| --- | --- | --- |
| No. | Method | Description |
| 1 | forward() | loads the next page. |
| 2 | back() | loads the previous page. |
| 3 | go() | loads the given page number. |

Example of history object

Let’s see the different usage of history object.

* history.back();//for previous page
* history.forward();//for next page
* history.go(2);//for next 2nd page
* history.go(-2);//for previous 2nd page

**JavaScript Navigator Object**

The **JavaScript navigator object** is used for browser detection. It can be used to get browser information such as appName, appCodeName, userAgent etc.

The navigator object is the window property, so it can be accessed by:

window.navigator   (or) navigator

Property of JavaScript navigator object

There are many properties of navigator object that returns information of the browser.

|  |  |  |
| --- | --- | --- |
| No. | Property | Description |
| 1 | appName | returns the name |
| 2 | appVersion | returns the version |
| 3 | appCodeName | returns the code name |
| 4 | cookieEnabled | returns true if cookie is enabled otherwise false |
| 5 | userAgent | returns the user agent |
| 6 | language | returns the language. It is supported in Netscape and Firefox only. |
| 7 | userLanguage | returns the user language. It is supported in IE only. |
| 8 | plugins | returns the plugins. It is supported in Netscape and Firefox only. |
| 9 | systemLanguage | returns the system language. It is supported in IE only. |
| 10 | mimeTypes[] | returns the array of mime type. It is supported in Netscape and Firefox only. |
| 11 | platform | returns the platform e.g. Win32. |
| 12 | online | returns true if browser is online otherwise false. |

Methods of JavaScript navigator object

The methods of navigator object are given below.

|  |  |  |
| --- | --- | --- |
| No. | Method | Description |
| 1 | javaEnabled() | checks if java is enabled. |
| 2 | taintEnabled() | checks if taint is enabled. It is deprecated since JavaScript 1.2. |

***Example of navigator object***

Let’s see the different usage of history object.

**<script>**

document.writeln("**<br/>**navigator.appCodeName: "+navigator.appCodeName);

document.writeln("**<br/>**navigator.appName: "+navigator.appName);

document.writeln("**<br/>**navigator.appVersion: "+navigator.appVersion);

document.writeln("**<br/>**navigator.cookieEnabled: "+navigator.cookieEnabled);

document.writeln("**<br/>**navigator.language: "+navigator.language);

document.writeln("**<br/>**navigator.userAgent: "+navigator.userAgent);

document.writeln("**<br/>**navigator.platform: "+navigator.platform);

document.writeln("**<br/>**navigator.onLine: "+navigator.onLine);

**</script>**

**JavaScript Screen Object**

The **JavaScript screen object** holds information of browser screen. It can be used to display screen width, height, colorDepth, pixelDepth etc.

The navigator object is the window property, so it can be accessed by:

window.screen  Or, screen

Property of JavaScript Screen Object

There are many properties of screen object that returns information of the browser.

|  |  |  |
| --- | --- | --- |
| No. | Property | Description |
| 1 | width | returns the width of the screen |
| 2 | height | returns the height of the screen |
| 3 | availWidth | returns the available width |
| 4 | availHeight | returns the available height |
| 5 | colorDepth | returns the color depth |
| 6 | pixelDepth | returns the pixel depth. |

***Example of JavaScript Screen Object***

Let’s see the different usage of screen object.

**<script>**

document.writeln("**<br/>**screen.width: "+screen.width);

document.writeln("**<br/>**screen.height: "+screen.height);

document.writeln("**<br/>**screen.availWidth: "+screen.availWidth);

document.writeln("**<br/>**screen.availHeight: "+screen.availHeight);

document.writeln("**<br/>**screen.colorDepth: "+screen.colorDepth);

document.writeln("**<br/>**screen.pixelDepth: "+screen.pixelDepth);

**</script>**

**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. It has properties and methods. By the help of document object, we can add dynamic content to our web page.

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."*

Properties of document object

Let's see the properties of document object that can be accessed and modified by the document object.



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. |

**Accessing field value by document object**

In this example, we are going to get the value of input text by user. Here, we are using **document.form1.name.value** to get the value of name field.

Here, **document** is the root element that represents the html document.

**form1** is the name of the form.

**name** is the attribute name of the input text.

**value** is the property, that returns the value of the input text.

Let's see the simple example of document object that prints name with welcome message.

**<script** type="text/javascript"**>**

function printvalue(){

var name=document.form1.name.value;

alert("Welcome: "+name);

}

**</script>**

**<form** name="form1"**>**

Enter Name:**<input** type="text" name="name"**/>**

**<input** type="button" onclick="printvalue()" value="print name"**/>**

**</form>**

**Javascript - document.getElementById() method**

The **document.getElementById()** method returns the element of specified id.

In the previous page, we have used **document.form1.name.value** to get the value of the input value. Instead of this, we can use document.getElementById() method to get value of the input text. But we need to define id for the input field.

Let's see the simple example of document.getElementById() method that prints cube of the given number.

**<script** type="text/javascript"**>**

function getcube(){

var number=document.getElementById("number").value;

alert(number\*number\*number);

}

**</script>**

**<form>**

Enter No:**<input** type="text" id="number" name="number"**/><br/>**

**<input** type="button" value="cube" onclick="getcube()"**/>**

**</form>**

**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>**

**Javascript - document.getElementsByTagName() method**

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>**

**Another example of document.getElementsByTagName() method**

In this example, we going to count total number of h2 and h3 tags used in the document.

**<script** type="text/javascript"**>**

function counth2(){

var totalh2=document.getElementsByTagName("h2");

alert("total h2 tags are: "+totalh2.length);

}

function counth3(){

var totalh3=document.getElementsByTagName("h3");

alert("total h3 tags are: "+totalh3.length);

}

**</script>**

**<h2>**This is h2 tag**</h2>**

**<h2>**This is h2 tag**</h2>**

**<h3>**This is h3 tag**</h3>**

**<h3>**This is h3 tag**</h3>**

**<h3>**This is h3 tag**</h3>**

**<button** onclick="counth2()"**>**count h2**</button>**

**<button** onclick="counth3()"**>**count h3**</button>**

**Javascript - innerHTML**

The **innerHTML** property can be used to write the dynamic html on the html document.

It is used mostly in the web pages to generate the dynamic html such as registration form, comment form, links etc.

**Example of innerHTML property**

In this example, we are going to create the html form when user clicks on the button.

In this example, we are dynamically writing the html form inside the div name having the id mylocation. We are identifing this position by calling the document.getElementById() method.

**<script** type="text/javascript" **>**

function showcommentform() {

var data="Name:**<input** type='text' name='name'**><br>**Comment:**<br>**

**<textarea** rows='5' cols='80'**></textarea>**

**<br><input** type='submit' value='Post Comment'**>**";

document.getElementById('mylocation').innerHTML=data;

}

**</script>**

**<form** name="myForm"**>**

**<input** type="button" value="comment" onclick="showcommentform()"**>**

**<div** id="mylocation"**></div>**

**</form>**

**Show/Hide Comment Form Example using innerHTML**

<!DOCTYPE html**>**

**<html>**

**<head>**

**<title>**First JS**</title>**

**<script>**

var flag=true;

function commentform(){

var cform="**<form** action='Comment'**>**Enter Name:**<br><input** type='text' name='name'**/><br/>**

Enter Email:**<br><input** type='email' name='email'**/><br>**Enter Comment:**<br/>**

**<textarea** rows='5' cols='70'**></textarea><br><input** type='submit' value='Post Comment'**/></form>**";

if(flag){

document.getElementById("mylocation").innerHTML=cform;

flag=false;

}else{

document.getElementById("mylocation").innerHTML="";

flag=true;

}

}

**</script>**

**</head>**

**<body>**

**<button** onclick="commentform()"**>**Comment**</button>**

**<div** id="mylocation"**></div>**

**</body>**

**</html>**

**Javascript - innerText**

The **innerText** property can be used to write the dynamic text on the html document. Here, text will not be interpreted as html text but a normal text.

It is used mostly in the web pages to generate the dynamic content such as writing the validation message, password strength etc.

Javascript innerText Example

In this example, we are going to display the password strength when releases the key after press.

**<script** type="text/javascript" **>**

function validate() {

var msg;

if(document.myForm.userPass.value.length**>**5){

msg="good";

}

else{

msg="poor";

}

document.getElementById('mylocation').innerText=msg;

 }

**</script>**

**<form** name="myForm"**>**

**<input** type="password" value="" name="userPass" onkeyup="validate()"**>**

Strength:**<span** id="mylocation"**>** **</span>**

**</form>**

**JavaScript Form Validation**

It is important to validate the form submitted by the user because it can have inappropriate values. So, validation is must to authenticate user.

JavaScript provides facility to validate the form on the client-side so data processing will be faster than server-side validation. Most of the web developers prefer JavaScript form validation.

Through JavaScript, we can validate name, password, email, date, mobile numbers and more 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 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>**

JavaScript Number Validation

Let's validate the textfield for numeric value only. Here, we are using isNaN() function.

**<script>**

function validate(){

var num=document.myform.num.value;

if (isNaN(num)){

  document.getElementById("numloc").innerHTML="Enter Numeric value only";

  return false;

}else{

  return true;

  }

}

**</script>**

**<form** name="myform" onsubmit="return validate()" **>**

Number: **<input** type="text" name="num"**><span** id="numloc"**></span><br/>**

**<input** type="submit" value="submit"**>**

**</form>**

JavaScript validation with image

Let’s see an interactive JavaScript form validation example that displays correct and incorrect image if input is correct or incorrect.

**<script>**

function validate(){

var name=document.f1.name.value;

var password=document.f1.password.value;

var status=false;

if(name.length**<1**){

document.getElementById("nameloc").innerHTML=

" <img src='unchecked.gif'/> Please enter your name";

status=false;

}else{

document.getElementById("nameloc").innerHTML=" <img src='checked.gif'/>";

status=true;

}

if(password.length**<6**){

document.getElementById("passwordloc").innerHTML=

" <img src='unchecked.gif'/> Password must be at least 6 char long";

status=false;

}else{

document.getElementById("passwordloc").innerHTML=" <img src='checked.gif'/>";

}

return status;

}

**</script>**

**<form** name="f1" action="#" onsubmit="return validate()"**>**

**<table>**

**<tr><td>**Enter Name:**</td><td><input** type="text" name="name"**/>**

**<span** id="nameloc"**></span></td></tr>**

**<tr><td>**Enter Password:**</td><td><input** type="password" name="password"**/>**

**<span** id="passwordloc"**></span></td></tr>**

**<tr><td** colspan="2"**><input** type="submit" value="register"**/></td></tr>**

**</table>**

**</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>**