# JavaScript is a client-side scripting language. This means the web surfer's browser will be running the script.The name JavaScript is owned by Netscape. Microsoft calls its version of the language JScript. The generic name of the language is EcmaScript.

Basic Rules:

1. JavaScript statements end with semi-colons.
2. JavaScript is case sensitive.
3. JavaScript has two forms of comments:
   * Single-line comments begin with a double slash (//).
   * Multi-line comments begin with "/\*" and end with "\*/".

The HTML <script> tag is used to insert a JavaScript into an HTML page.

The JavaScript syntax is loosely based on the Java syntax. Java is a full blown programming environment and JavaScript could be seen as a sub-set of the Java syntax. Having said this, that is where the similarities end - Java and JavaScript are two totally different things.

In learning JavaScript you will become familiar with terms such as variables, functions, statements, operators, data types, objects etc.

It will take most of this tutorial to show you the complete JavaScript syntax. For now, I'll give you a quick intro by showing you an example and explanation.

Code:

<script type="text/javascript">

document.write("JavaScript is not Java");

</script>

This results in:

JavaScript is not Java

The<script> tags tell the browser to expect a script in between them. You specify the language using the type attribute. The most popular scripting language on the web is JavaScript.

 The bits that look like HTML comments tag (<-- -->) are just that - HTML comment tags. These are optional but recommended. They tell browsers that don't support JavaScript (or with JavaScript disabled) to ignore the code in between. This prevents the code from being written out to your website users.

 The part that writes the actual text is only 1 line (document.write("JavaScript is not Java");). This is how you write text to a web page in JavaScript. This is an example of using a JavaScript function (also known as *method*).

**Where to put your scripts?**

You can place your scripts in any of the following locations:

* Between the HTML document'shead tags.
* Within the HTML document's body (i.e. between the body tags).
* In an external file (and link to it from your HTML document).

How to add HTML tags to the JavaScript:

<html>  
<body>  
<script type="text/javascript">  
document.write("<h1>Welcome To Impinge!!!! </h1>");  
</script>  
</body>  
</html>

This Results in:

# JavaScript Operator with Variables

# Performing operations on variables that contain values is very common and easy to do. Below is a simple script that performs all the basic arithmetic operations. Variables are "containers" for storing information.

Example:

<body>

<script type="text/JavaScript">

var two = 2

var ten = 10

varlinebreak = "<br />"

document.write("two plus ten = ")

var result = two + ten

document.write(result)

document.write(linebreak)

document.write("ten \* ten = ")

result = ten \* ten

document.write(result)

document.write(linebreak)

document.write("ten / two = ")

result = ten / two

document.write(result)

</script>

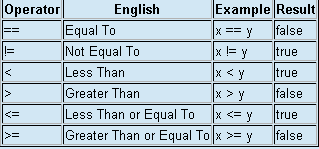
</body>

This Results in:

two plus ten = 12  
ten \* ten = 100  
ten / two = 5

# Comparison Operators

Comparisons are used to check the relationship between variables and/or values. A single equal sign sets a value while a double equal sign (==) compares two values. Comparison operators are used inside conditional statements and evaluate to either *true* or *false*.



Example:

<body>

<script type="text/JavaScript">

varlinebreak = "<br />"

varmy\_var = "Hello World!"

document.write(my\_var)

document.write(linebreak)

my\_var = "I am learning JavaScript!"

document.write(my\_var)

document.write(linebreak)

my\_var = "Script is Finishing up..."

document.write(my\_var)

</script>

</body>

## Conditional Statements

In JavaScript we have the following conditional statements:

* **if statement** - use this statement to execute some code only if a specified condition is true
* **if...else statement** - use this statement to execute some code if the condition is true and another code if the condition is false
* **if...else if....else statement** - use this statement to select one of many blocks of code to be executed
* **switch statement** - use this statement to select one of many blocks of code to be executed

There are two key parts to a JavaScript ***while*** *loop*:

1. The conditional statement which must be *True* for the *while loop's* code to be executed.
2. The *while loop's* code that is contained in curly braces "{ and }" will be executed if the condition is *True*.

When a *while loop* begins, the JavaScript interpreter checks if the condition statement is true. If it is, the code between the curly braces is executed. At the end of the code segment "}", the *while loop* loops back to the condition statement and begins again. Example:

<script type="text/javascript">

varmyCounter = 0;

varlinebreak = "<br />";

document.write("While loop is beginning");

document.write(linebreak);

while(myCounter< 10){

document.write("myCounter = " + myCounter);

document.write(linebreak);

myCounter++;

}

document.write("While loop is finished!");

</script>

Results in:

While loop is beginning  
myCounter = 0  
myCounter = 1  
myCounter = 2  
myCounter = 3  
myCounter = 4  
myCounter = 5  
myCounter = 6  
myCounter = 7  
myCounter = 8  
myCounter = 9  
While loop is finished!

The JavaScript ***For*** *Loop* resembles the for loop you may have seen in many other programming languages. It is used when you need to do a set of operations many times, with an increment of some kind after each run through the block of code.

Example:

<script type="text/javascript">

varlinebreak = "<br />";

document.write("For loop code is beginning");

document.write(linebreak);

for(i = 0; i < 5; i++){

document.write("Counter i = " + i);

document.write(linebreak);

}

document.write("For loop code is finished!");

</script>

Results in:

For loop code is beginning  
Counter i = 0  
Counter i = 1  
Counter i = 2  
Counter i = 3  
Counter i = 4  
For loop code is finished!

## JavaScript For...In Statement

The for...in statement loops through the elements of an array or through the properties of an object.

<script type="text/javascript">

var x;

var fur = new Array();

fur[0] = "Chair";

fur[1] = "Table";

fur[2] = "Phone";

for (x in fur)

{

document.write(fur[x] + "<br />");

}

</script>

# JavaScript Array

An array is a variable that can store many variables within it. Many programmers have seen arrays in other languages, and they aren't that different in JavaScript.

Creating an array is slightly different from creating a normal variable. Because JavaScript has variables and properties associated with arrays, you have to use a special function to create a new array.

Example:<script type="text/javascript">

varmyArray = new Array();

myArray[0] = "Computer";

myArray[1] = "Laptop";

myArray[2] = "Notebook";

document.write(myArray[0] + myArray[1] + myArray[2]);</script>

# JavaScript Array Sorting

<script type="text/javascript">

var myArray2= new Array();

myArray2[0] = "Computer";

myArray2[1] = "Laptop";

myArray2[2] = "Notebook ";

myArray2.sort();

document.write(myArray2[0] + myArray2[1] + myArray2[2]);

</script>

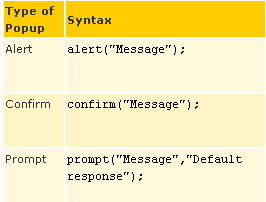
# Functions:

A function is a piece of code that sits dormant until it is referenced or called upon to do its "function". In addition to controllable execution, functions are also a great time saver for doing repetitive tasks.

Instead of having to type out the code every time you want something done, you can simply call the function multiple times to get the same effect. This benefit is also known as "code reusability".

A function that does not execute when a page loads should be placed inside the *head* of your HTML document.

## Types of Popups



### Alert

Displays a message to the user.The user will need to click "OK" to proceed.   
Typical use is when you want to make sure information comes through to the user.Example:

<html>

<head>

<script type="text/javascript">

**function popup() {**

**alert("Hi students!!!")**

**}**

</script>

</head>

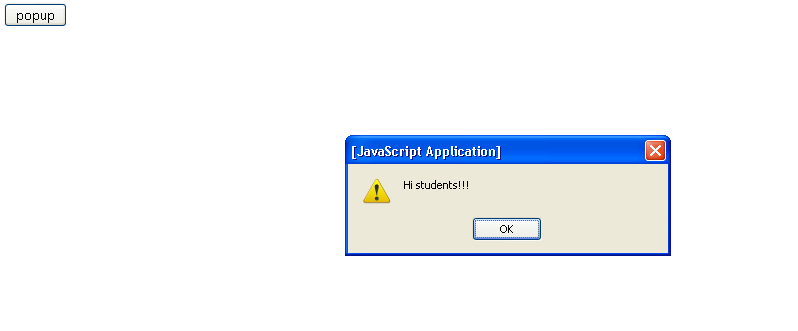
<body>

<input type="button" onclick="popup()" value="popup">

</body>

</html>

Results in:



### Confirm

Asks the user to confirm something.Example:

<html>

<head>

<script type="text/javascript">

functionshowconfirm(){

var t=confirm("Press any button!");

if (t==true)

{

alert("You pressed OK!");

}

else

{

alert("You pressed Cancel!");

}}</script>

</head>

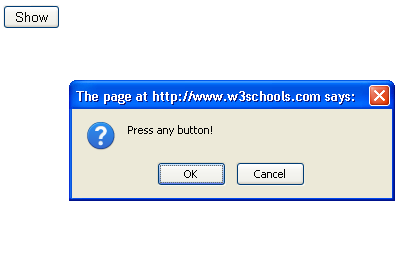
<body>

<input type="button" onclick="showconfirm()" value="Show" />

</body>

</html>

Results in:



### Prompt

Prompts the user for information.A small dialogue box pops up and appears in front of the web page currently in focus. The confirm box is different from the alert box. It supplies the user with a choice; they can either press OK to confirm the popup's message or they can press cancel and not agree to the popup's request.

<head>

<script type="text/javascript">

functionshow\_prompt() {

var reply = prompt("Hey! What's your name?", "")

alert ( "Nice to see you " + reply + "!")

}</script>

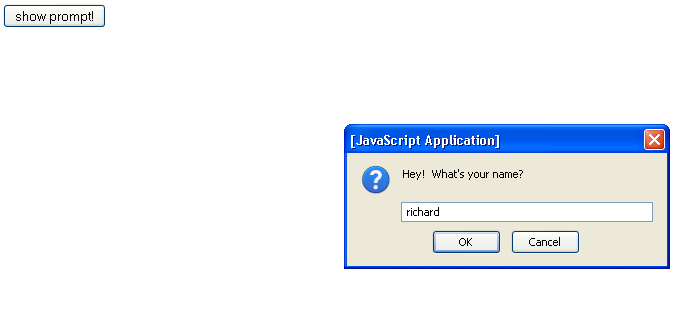
</head>

<body>

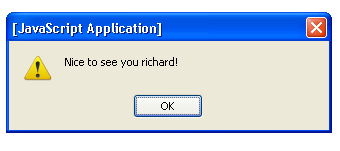
<input type="button" onclick="show\_prompt()" value="show prompt!">

</body>

Results in:



After clicking OK, alert box will get open.



# JavaScript Print Script

## The JavaScript print function *window.print()* will print the current webpage when executed.

<form>

<input type="button" value="Print This Page" onClick="window.print()" />

</form>

Results in:

## 

# JavaScript Window.Location

Control over what page is loaded into the browser rests in the JavaScript property *window.location*. By setting *window.location* equal to a new URL, you will in turn change the current webpage to the one that is specified.

## Example:

<script type="text/javascript">

window.location = "http://www.google.com/"

</script>

# JavaScript String Length

Advanced scripters will often need to know how long a JavaScript string is. For example, if a webdeveloper was creating a submission form that required the username to be no longer than 20 characters, then she would need to check the length of the string before allowing the user to submit data.

## The *length* property returns the number of characters that are in a string, using an integer.

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

## var t = "Richard";

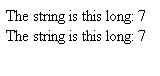
## var length = t.length;

## document.write("The string is this long: " + length);

## document.write("<br />The string is this long: " + t.length);

## </script>

## Results in:



**Split Function:**

<script type="text/javascript">

varmyString = "123456789";

varmySplitResult = myString.split("5");

document.write("The first element is " + mySplitResult[0]);

document.write("<br /> The second element is " + mySplitResult[1]);

</script>

Results in:

The first element is 1234  
The second element is 6789

# String Search Function

This string function takes a regular expression and then examines that string to see if there are any matches for that expression. If there is a match , it will return the position in the string where the match was found. If there isn't a match, it will return -1.

Example:

<script type="text/javascript">

varmyRegExp ="John";

var string1 = "Today John went to the store and talked with Alex.";

var matchPos1 = string1.search(myRegExp);

if(matchPos1 != -1)

document.write("There was a match at position " + matchPos1);

else

document.write("There was no match in the first string");

</script>

Results in:



**String indexOf Function**

*indexOf* has two arguments, with the second one being optional:

1. SearchString - What you would like to search for.
2. Offset (optional) - How far into the string you would like the search to begin. If you want to search the whole string, omit this argument.

<script type="text/javascript">

var a = "He is playing";

varaPosition = a.indexOf("is");

document.write("The position = " + aPosition);

</script>

Results in:



# JavaScript getElementById:

If you want to quickly access the value of an HTML input give it an *id* to make your life a lot easier. This small script below will check to see if there is any text in the text field "myText". The argument that *getElementById* requires is the *id* of the HTML element you wish to utilize.

<script type="text/javascript">

functionnotEmpty(){

varmyTextField = document.getElementById('myText');

if(myTextField.value != "")

alert("You entered: " + myTextField.value)

else

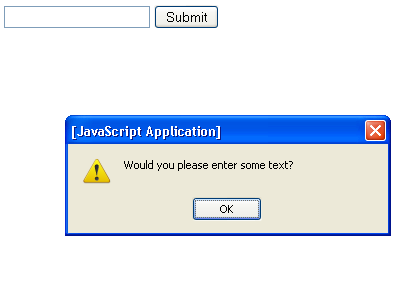
alert("Would you please enter some text?")

}

</script>

<input type='text' id='myText' />

<input type='button' onclick='notEmpty()' value='Submit’/>

Results in:

# JavaScript innerHTML

Each HTML element has an *innerHTML* property that defines both the HTML code and the text that occurs between that element's opening and closing tag. By changing an element's *innerHTML* after some user interaction, you can make much more interactive pages.

However, using *innerHTML* requires some preparation if you want to be able to use it easily and reliably. First, you must give the element you wish to change an [id](http://www.tizag.com/cssT/cssid.php). With that *id* in place you will be able to use the *getElementById* function, which works on all browsers.

<script type="text/javascript">

functionchangeText(){

document.getElementById('boldStuff').innerHTML = 'Priya';

}

</script>

<p>Welcome <b id='boldStuff'>People!!!</b></p>

<input type='button' onclick='changeText()' value='Change Text'/>

# JavaScript Date Today

<h4>It is now

<script type="text/javascript">

varcurrentTime = new Date()

</script>

</h4>

# Get the JavaScript Time

The Date object has been created, and now we have a variable that holds the current date! To get the information we need to print out, we have to utilize some or all of the following functions:

* **getTime()** - Number of milliseconds since 1/1/1970 @ 12:00 AM
* **getSeconds()** - Number of seconds (0-59)
* **getMinutes()** - Number of minutes (0-59)
* **getHours()** - Number of hours (0-23)
* **getDay()** - Day of the week(0-6). 0 = Sunday, ... , 6 = Saturday
* **getDate()** - Day of the month (0-31)
* **getMonth()** - Number of month (0-11)
* **getFullYear()** - The four digit year (1970-9999)

Example 1:

<h4>It is now

<script type="text/javascript">

varcurrentTime = new Date()

var month = currentTime.getMonth() + 1

var day = currentTime.getDate()

var year = currentTime.getFullYear()

document.write(month + "/" + day + "/" + year)

</script>

</h4>

Example 2:

<h4>It is now

<script type="text/javascript">

varcurrentTime = new Date()

var hours = currentTime.getHours()

var minutes = currentTime.getMinutes()

if (minutes < 10){

minutes = "0" + minutes

}

document.write(hours + ":" + minutes + " ")

if(hours > 11){

document.write("PM")

} else {

document.write("AM")

}

</script>

</h4>

## Events

By using JavaScript, we have the ability to create dynamic web pages. Events are actions that can be detected by JavaScript.

Every element on a web page has certain events which can trigger a JavaScript. For example, we can use the onClick event of a button element to indicate that a function will run when a user clicks on the button. We define the events in the HTML tags.

**Example containing all the events:**

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

function handler(t,c,s)

{

t.value += s;

c.checked = true;

return 0

}

</script>

<form name="formA" action="javascript:void(0)"

onreset= "handler(formB.t1, formB.reset, 'reset')"

onsubmit="handler(formB.t1, formB.submit, 'submit')">

<textarea rows="4" cols="45" wrap="soft"

onclick= "handler(formB.t1, formB.click, 'click ')"

ondblclick= "handler(formB.t1, formB.dblclick, 'dblclick ')"

onmouseover="handler(formB.t1, formB.over, 'over ')"

onmouseout= "handler(formB.t1, formB.out, 'out ')"

onmousemove="handler(formB.t2, formB.move, '+')"

onfocus= "handler(formB.t1, formB.focus, 'focus ')"

onblur= "handler(formB.t1, formB.blur, 'blur ')"

onkeydown= "handler(formB.t1, formB.keydown, 'keydown ')"

onkeyup= "handler(formB.t1, formB.keyup, 'keyup ')"

onkeypress= "handler(formB.t1, formB.keypress, 'keypress ')"

onchange= "handler(formB.t1, formB.change, 'change ')"

onselect= "handler(formB.t1, formB.select, 'select ')">

Move the mouse, click and type here. Try selecting text.

</textarea>

<input type="Reset"><input type="Submit" value="Submit">

</form>

<form name="formB">

<!-- Event Handler Output: -->

<textarea name="t1" rows="8" cols="45" wrap="soft"></textarea>

<textarea name="t2" rows="8" cols="10" wrap="soft"></textarea>

click:<input type="checkbox" name="click">

dblclick:<input type="checkbox" name="dblclick">

over:<input type="checkbox" name="over">

out:<input type="checkbox" name="out">

move:<input type="checkbox" name="move">

focus:<input type="checkbox" name="focus">

keydown:<input type="checkbox" name="keydown">

keyup:<input type="checkbox" name="keyup">

select:<input type="checkbox" name="select">

keypress:<input type="checkbox" name="keypress">

<!-- additional checkboxes not shown... -->

</form>

# JavaScript Void(0)

Sometimes, you may need to call some JavaSript from within a link. Normally, when you click a link, the browser loads a new page (or refreshes the same page).

This might not always be desirable. For example, you might only want to dynamically update a form field when the user clicks a link.

To prevent the load from refreshing, you could use the JavaScript void() function and pass a parameter of 0 (zero).

**Math Object:**

Example:

<html>

<script type="text/javascript">

document.write(Math.E);

document.write("<br/>");

document.write(Math.PI)

document.write("<br/>");

document.write(Math.SQRT2)

document.write("<br/>");

document.write(Math.SQRT1\_2)

document.write("<br/>");

document.write(Math.LN2)

document.write("<br/>");

document.write(Math.LN10)

document.write("<br/>");

document.write(Math.LOG2E)

document.write("<br/>");

document.write(Math.LOG10E);

document.write("<br/>");

document.write(Math.pow(10,2));

document.write("<br/>");

document.write(Math.sqrt(169));

document.write("<br/>");

with (Math)

{

var x= sin(3.5)

var y=tan(5)

var result=max(x,y)

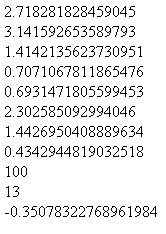
document.write(result);

}

</script>

</html>

This Results in:



[round()](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_round)  
How to use round().

[random()](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_random)  
How to use random() to return a random number between 0 and 1.

[max()](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_max)  
How to use max() to return the number with the highest value of two specified numbers.

[min()](http://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_min)  
How to use min() to return the number with the lowest value of two specified numbers.

<html>

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

document.write(Math.round(0.60) + "<br />");

document.write(Math.random() + "<br />");

document.write(Math.max(5,10) + "<br />");

document.write(Math.min(5,10) + "<br />");

</script></body>

</html>

# JavaScript Browser Detection

The Navigator object contains information about the visitor's browser.

There are some things that just don't work on certain browsers - especially on older browsers.

Sometimes it can be useful to detect the visitor's browser, and then serve the appropriate information.

The best way to do this is to make your web pages smart enough to look one way to some browsers and another way to other browsers.

The Navigator object contains information about the visitor's browser name, version, and more

<html>  
<body>  
<script type="text/javascript">  
document.write("Browser CodeName: " + navigator.appCodeName);  
document.write("<br /><br />");  
document.write("Browser Name: " + navigator.appName);  
document.write("<br /><br />");  
document.write("Browser Version: " + navigator.appVersion);  
document.write("<br /><br />");  
document.write("Cookies Enabled: " + navigator.cookieEnabled);  
document.write("<br /><br />");  
document.write("Platform: " + navigator.platform);  
document.write("<br /><br />");  
document.write("User-agent header: " + navigator.userAgent);  
</script>  
  
</body>  
</html>

# JavaScript Try...Catch Statement

The try...catch statement allows you to test a block of code for errors.

The try block contains the code to be run, and the catch block contains the code to be executed if an error occurs.

Example:

<html>  
<head>  
<script type="text/javascript">  
var txt="";  
function givemessage()  
{  
try  
  {  
  adddlert("Welcome to sebiz!");  
  }  
catch(err)  
  {  
  txt="There was an error on this page.\n\n";  
  txt+="Error description: " + err.description + "\n\n";  
  txt+="Click OK to continue.\n\n";  
  alert(txt);  
  }  
}  
</script>  
</head>  
  
<body>  
<input type="button" value="View" onclick="givemessage()" />  
</body>  
  
</html>

## The Throw Statement

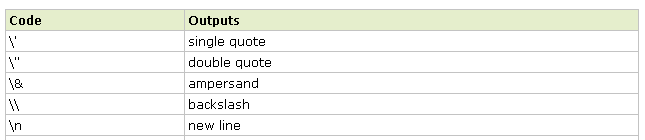
The throw statement allows you to create an exception. If you use this statement together with the try...catch statement, you can control program flow and generate accurate error messages.

<html>  
<body>  
<script type="text/javascript">  
var x=prompt("Enter a number between 0 and 10:","");  
try  
  {   
  if(x>10)  
    {  
    throw "Err1";  
    }  
  else if(x<0)  
    {  
    throw "Err2";  
    }  
  else if(isNaN(x))  
    {  
    throw "Err3";  
    }  
  }  
catch(er)  
  {  
  if(er=="Err1")  
    {  
    alert("Error! The value is too high");  
    }  
  if(er=="Err2")  
    {  
    alert("Error! The value is too low");  
    }  
  if(er=="Err3")  
    {  
    alert("Error! The value is not a number");  
    }  
  }  
</script>  
</body>  
</html>

In JavaScript you can add special characters to a text string by using the backslash sign.

## Insert Special Characters

The backslash (\) is used to insert apostrophes, new lines, quotes, and other special characters into a text string.





# JavaScript Cookies

Cookies are small text files that sit on your hard disk. Cookies are created when you visit websites that use cookies to store information that they need (or prefer). Websites often use cookies to personalise the user experience - such as remembering your name (assuming you supplied it previously) or remembering the items in your shopping cart from previous visits.

Despite the many misconceptions about cookies being malicious, they are quite harmless. Cookies can't give your personal details to other websites, or transmit a virus or anything like that. A cookie can only be read by the server that created it. Websites normally use cookies to make its users' lives easier, not harder.

## Creating Cookies in JavaScript

document.cookie =

"myContents=Quackit JavaScript cookie experiment;

expires=Fri, 19 Oct 2007 12:00:00 UTC; path=/";

Now check your cookies folder to see if the cookie was created. Alternatively, write code to read the cookie.

Note: If the cookie wasn't created, check the expiry date - it needs to be a date in the future.

You can update this value by using the same code with a different value. If you want to add a second value, simply use a different variable name (for example "myContents2=").

## Reading Cookies in JavaScript

document.write(document.cookie);

You simply reference the cookie using document.cookie. The only problem with the above code is that it outputs the equals sign and everything before it (i.e. "myContents="). To stop this from happening, try the following code:

document.write(document.cookie.split("=")[1])

## Deleting Cookies in JavaScript

To delete a cookie, you can use the same code you used to create it but this time, set the expiry date in the past:

document.cookie =

"myContents=Quackit JavaScript cookie experiment;

expires=Fri, 14 Oct 2005 12:00:00 UTC; path=/";

**//for creating a cookie**

<script type="text/javascript">

var today = new Date();

varnextMonth = new Date(today.getYear(), today.getMonth()+1, today.getDate());

setCookie("technofundoDemoCookie", "Manish", nextMonth);

</script>

# JavaScript Status Bar Messages

JavaScript can be used to display messages in the status bar using window.status. For example, you can display a javascript status bar message whenever your users hover over your hyperlinks.

<a href="a.html"

onMouseover="JavaScript:window.status='Status Bar Message goes here'; return true"

onMouseout="JavaScript:window.status=''; return true">Hover over me!</a>

Most (newer) major browsers disable status bar messages by default. If your status bar doesn't change when you hover over the link, it's probably because of this.

If you really want to see this example, you can enable status bar messages by changing your browser settings.

For example, in Firefox:

1. Go to *Tools > Options*
2. Click the *Content* tab
3. Ensure that the *JavaScript* option is checked
4. Click *Advanced* (next to the *Enable JavaScript* option)
5. Check the *Change status bar text* option
6. Click *OK* to save this screen
7. Click *OK* again

In Internet Explorer:

1. Go to *Tools > Internet Options*
2. Click the *Security* tab
3. Ensure that the *Internet* option is selected/highlighted
4. Click *Custom Level...* (this launches the security settings for the Internet zone)
5. Scroll down until you see *Allow status bar updates via script* (under the *Scripting* option). Click *Enable*
6. Click *OK* to save this screen
7. Click *OK* again

Note that, because this is the default setting in most browsers, there's a good chance that most of your users won't see your status bar message

# JavaScript Form Validation

The idea behind JavaScript form validation is to provide a method to check the user entered information before they can even submit it. JavaScript also lets you display helpful alerts to inform the user what information they have entered incorrectly and how they can fix it. In this lesson we will be reviewing some basic form validation, showing you how to check for the following:

* If a text input is empty or not
* If a text input is all numbers
* If a text input is all letters
* If a text input is all alphanumeric characters (numbers & letters)
* If a text input has the correct number of characters in it (useful when restricting the length of a username and/or password)
* If a selection has been made from an HTML select input (the drop down selector)
* If an email address is valid
* How to check all above when the user has completed filling out the form

# Form Validation - Checking for Non-Empty

<script type='text/javascript'>

functionnotEmpty(elem, helperMsg){

if(elem.value.length == 0){

alert(helperMsg);

elem.focus();

return false;

}

return true;

}

</script>

<form>

Required Field: <input type='text' id='req1'/>

<input type='button'

onclick="notEmpty(document.getElementById('req1'), 'Please Enter a Value')"

value='Check Field' />

</form>

# Form Validation - Checking for All Numbers

<script type='text/javascript'>

functionisNumeric(elem, helperMsg){

varnumericExpression = /^[0-9]+$/;

if(elem.value.match(numericExpression)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

</script>

<form>

Numbers Only: <input type='text' id='numbers'/>

<input type='button'

onclick="isNumeric(document.getElementById('numbers'), 'Numbers Only Please')"

value='Check Field' />

</form>

# Form Validation - Checking for All Letters

functionisAlphabet(elem, helperMsg){

varalphaExp = /^[a-zA-Z]+$/;

if(elem.value.match(alphaExp)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

# Form Validation - Restricting the Length

functionlengthRestriction(elem, min, max){

varuInput = elem.value;

if(uInput.length>= min &&uInput.length<= max){

return true;

}else{

alert("Please enter between " +min+ " and " +max+ " characters");

elem.focus();

return false;

}

}

# Form Validation - Selection Made

<script type='text/javascript'>

functionmadeSelection(elem, helperMsg){

if(elem.value == "Please Choose"){

alert(helperMsg);

elem.focus();

return false;

}else{

return true;

}

}

</script>

<form>

Selection: <select id='selection'>

<option>Please Choose</option>

<option>CA</option>

<option>WI</option>

<option>XX</option>

</select>

<input type='button'

onclick="madeSelection(document.getElementById('selection'), 'Please Choose Something')"

value='Check Field' />

</form>

**Full Form Validation:**

<script type='text/javascript'>

functionformValidator(){

// Make quick references to our fields

varfirstname = document.getElementById('firstname');

varaddr = document.getElementById('addr');

var zip = document.getElementById('zip');

var state = document.getElementById('state');

var username = document.getElementById('username');

var email = document.getElementById('email');

// Check each input in the order that it appears in the form!

if(isAlphabet(firstname, "Please enter only letters for your name")){

if(isAlphanumeric(addr, "Numbers and Letters Only for Address")){

if(isNumeric(zip, "Please enter a valid zip code")){

if(madeSelection(state, "Please Choose a State")){

if(lengthRestriction(username, 6, 8)){

if(emailValidator(email, "Please enter a valid email address")){

return true;

}

}

}

}

}

}

return false;

}

functionnotEmpty(elem, helperMsg){

if(elem.value.length == 0){

alert(helperMsg);

elem.focus(); // set the focus to this input

return false;

}

return true;

}

functionisNumeric(elem, helperMsg){

varnumericExpression = /^[0-9]+$/;

if(elem.value.match(numericExpression)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

functionisAlphabet(elem, helperMsg){

varalphaExp = /^[a-zA-Z]+$/;

if(elem.value.match(alphaExp)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

functionisAlphanumeric(elem, helperMsg){

varalphaExp = /^[0-9a-zA-Z]+$/;

if(elem.value.match(alphaExp)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

functionlengthRestriction(elem, min, max){

varuInput = elem.value;

if(uInput.length>= min &&uInput.length<= max){

return true;

}else{

alert("Please enter between " +min+ " and " +max+ " characters");

elem.focus();

return false;

}

}

functionmadeSelection(elem, helperMsg){

if(elem.value == "Please Choose"){

alert(helperMsg);

elem.focus();

return false;

}else{

return true;

}

}

functionemailValidator(elem, helperMsg){

varemailExp = /^[\w\-\.\+]+\@[a-zA-Z0-9\.\-]+\.[a-zA-z0-9]{2,4}$/;

if(elem.value.match(emailExp)){

return true;

}else{

alert(helperMsg);

elem.focus();

return false;

}

}

</script>

<form onsubmit='return formValidator()' >

First Name: <input type='text' id='firstname' /><br />

Address: <input type='text' id='addr' /><br />

Zip Code: <input type='text' id='zip' /><br />

State: <select id='state'>

<option>Please Choose</option>

<option>AL</option>

<option>CA</option>

<option>TX</option>

<option>WI</option>

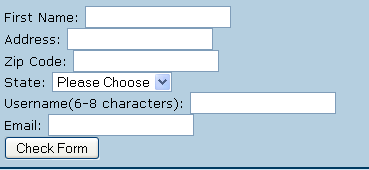
</select><br />

Username(6-8 characters): <input type='text' id='username' /><br />

Email: <input type='text' id='email' /><br />

<input type='submit' value='Check Form' />

</form>



# JavaScript Timing Events

JavaScript can be executed in time-intervals.

This is called timing events.

**JavaScript Timing Events**

With JavaScript, it is possible to execute some code after a specified time-interval. This is called timing events.

It's very easy to time events in JavaScript. The two key methods that are used are:

* setTimeout() - executes a code some time in the future
* clearTimeout() - cancels the setTimeout()

**Note:** The setTimeout() and clearTimeout() are both methods of the HTML DOM Window object

The setTimeout() method returns a value - In the statement above, the value is stored in a variable called t. If you want to cancel this setTimeout(), you can refer to it using the variable name.

The first parameter of setTimeout() is a string that contains a JavaScript statement. This statement could be a statement like "alert('5 seconds!')" or a call to a function, like "alertMsg()".

The second parameter indicates how many milliseconds from now you want to execute the first parameter.

There are 1000 milliseconds in one second.

**SetTimeOut Example:**

<html>  
<head>  
<script type="text/javascript">  
function timedMsg()  
{  
var t=setTimeout("alert('5 seconds!')",5000);  
}  
</script>  
</head>  
  
<body>  
<form>  
<input type="button" value="Display timed alertbox!"  
onClick="timedMsg()" />  
</form>  
</body>  
</html>

**ClearTimeOut Example:**

<html>  
<head>  
<script type="text/javascript">  
var c=0;  
var t;  
vartimer\_is\_on=0;  
  
function timedCount()  
{  
document.getElementById('txt').value=c;  
c=c+1;  
t=setTimeout("timedCount()",1000);  
}  
  
function doTimer()  
{  
if (!timer\_is\_on)  
  {  
  timer\_is\_on=1;  
  timedCount();  
  }  
}  
  
function stopCount()  
{  
clearTimeout(t);  
timer\_is\_on=0;  
}  
</script>  
</head>  
  
<body>  
<form>  
<input type="button" value="Start count!" onClick="doTimer()">  
<input type="text" id="txt">  
<input type="button" value="Stop count!" onClick="stopCount()">  
</form>  
</body>  
</html>