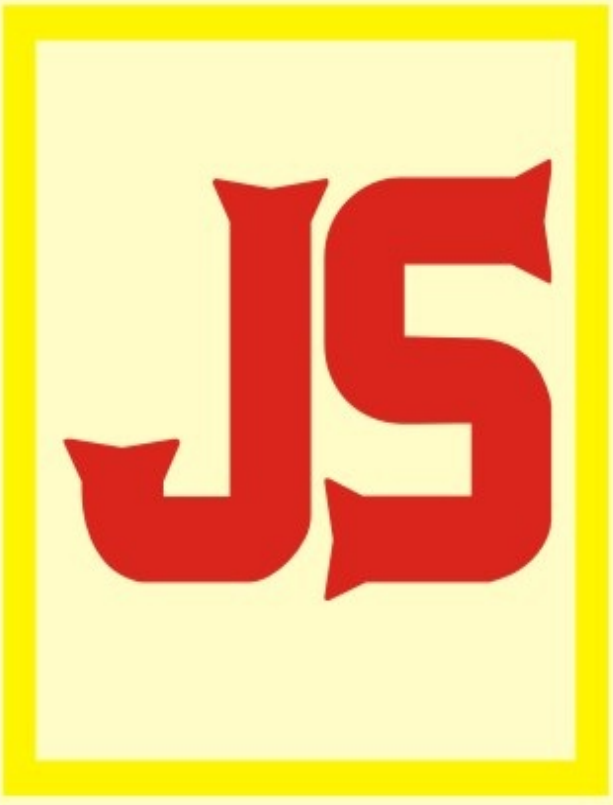
JavaScript Fundamental

A Step by Step Guide



Steven Bright

JavaScript Fundamental

A Step by Step Guide

By

Steven Bright

Also By Steven Bright:

CorelDraw How: The Fundamental of CorelDRAW

Tools and Function Lists: Engineering Tools Manual

Microsoft Word Fundamental: A Step by Step Guide

Web Design and Development Technology: Website How

The Best of Microsoft PowerPoint: Brand Exposure Techniques

Hacking Through Microsoft Excel Skills: A Step by Step Approach

The wave of Change: Adams Aliyu Oshiomhole: Why he is the man we need to Create the Atmosphere of Change we Desire to Build a Successful, Great and Powerful Nation

HOW TO USE THE BOOK

This is a documentation of the uses of JavaScript mostly in website development. Follow the Step by Step guide to be able to get the best out of this book. Work through it from chapter to chapter and you will be happy at the time you get to the last page. You will need a web authoring software like Dreamweaver or web editing software like Notepad++, and web browsers like Google chrome and Mozilla Firefox to preview your work.

TABLE OF CONTENT

INTRODUCTION

JavaScript is a scripting language. It is a language for the web, server, Personal computers, Laptops, Tablets, Phones, etc.

It is used to add interactivity to HTML pages that is giving the pages some level of dynamism.

A scripting language is a lightweight programming language. It can be inserted into HTML pages and it is an interpreted language (that is scripts execute without preliminary compilation)

Scripts can be place in the head or body section of HTML document. Nevertheless, it is a common practice to put scripts in the head section, or at the bottom of a web page such that they are all in one place and do not interfere with page content.

When a JavaScript interpreter is embedded in a web browser, the result is client side JavaScript. It combines the scripting ability of a JavaScript interpreter with the Document Object Model (DOM) defined by a web browser. Client-side JavaScript add some level of dynamic behavior to otherwise static web pages.

Although there are strong outward similarities between JavaScript and Java, including language name, syntax, and respective standard libraries, the two are distinct languages and differ greatly in their design.

Take note of the following

- JavaScript code is case sensitive
- White space between words and tabs are ignored
- JavaScript statements end with a semi- colon (;)
- Line breaks are ignored except within a statement
- Capitalize the first letter of every word except the first e.g. userLirstName
- To declare variables, use the keyword var and the variable name: var username. To assign values to variables, add an equal sign and the value: var userName = “Smith” var price = 100

JAVASCRIPT SYNTAX

This is a set of rules that determine how the language will be written by the programmer and interpreted by the browser.

The basic syntax is as follow:

<script>

< - -

var myDate=new

Date();

myDate.setDate

(myDate.getDate());

document.write(myDate)

- - >

</script></br>

Explanation

- The script tags i.e. < script ></ script > tell the browser to expect a script in between them.
- < — - - -> : These are the normal HTML comment tag but in Javascript it is used to tell the browsers that do not support JavaScript or with JavaScript disabled to ignore the code in between. This prevents code from being written out to your user's browser.
- Document.write: This is the part that writes the actual text onto users browsers.
- Where to put your script: you can put your script in the head section, body, or as an external file.

WHAT JAVASCRIPT IS USE FOR IN WEB DEVELPMENT

The basic things JavaScript is use for in web development are:

1. JavaScript gives HTML designers a programming tool
2. JavaScript can put dynamic text into an HTML page
3. JavaScript create pages dynamically such as date, or other external data, it can produce pages that are customized to the user.
4. JavaScript can react to events
5. JavaScript can read and write HTML elements
6. JavaScript can be used to validate input data
7. JavaScript can be used to detect the visitor’s browser
8. JavaScript can be used to create cookies

TYPES OF JAVASCRIPT

There are two types of JavaScript that is Internal and External Scripts.

Inline JavaScript:

Example:

<script>

alert (“WELCOME!”);

</script>

External JavaScript:

You use the SRC attribute of the <SCRIPT> tag to call JavaScript code from an external text file. This is useful if you have a lot of code or you want to run the external file from several pages, since any number of pages can call the same external JavaScript file.

Example:

The script below was saved as a miscript.js file:

alert(“WELCOME!”);

This was then called upon by the syntax below put in the head section just before the closing head tag:

<script src=“miscript.js”></script>

Standalone Javascript

This script loads as a page is loaded. A good example is given bellow:

<script>

alert(‘Welcome!’)

</script>

WHERE YOU CAN PLACE JAVACRIPT

JavaScript can be place in the following part of a document:

1. The head of a document
2. The body of a document
3. As an external .js file

JAVASCRIPT DISPLAY METHODS

JavaScript can “display” data in different ways as described below:

I’m going to use a the same example to show the possibilities of JavaScript display using these different methods.

Writing into an alert box using

window.alert().

script>

window.alert(24+ 44);

< /script >

Note that when the script above is tested, an alert window comes up and display the result of 24+44.

Writing into the HTML output using

document.write() .

< button

onclick =“document.write(24 + 44)”> </button >

Note that when the button is clicked, a document comes up that displays only the result.

Writing into the browser console, using

console.log() .

< script>

console.log(24 + 44);

< /script >

You will need to activate JavaScript console in your browser e.g Google Chrome using F12 and then type in the above script, you then press enter and it will give you the result.

Writing into an HTML element, using

innerHTML .

< p id=“mon”>< /p >

< script>

document.getElementById

(“demo”).innerHTML = 24 +

44 ;

< /script >

Note you set an id property, and script gets the result using that I’d.

POPUP BOXES

JavaScript has three types of Popup boxes that is:

- Alert box
- Confirm box
- Prompt box

Alert Box

It is used to ensure that information gets to a user. When an alert box pops up, a user must click OK to proceed.

Example:

<script>

alert(“WELCOME!”);

</script>

Confirm Box

It is used if one wants a user to verify or confirm information. When a confirm box pops up, a user must click OK or CANCEL to proceed.

Example 1:

<script>

confirm(“BUY”);

</script>

Example2:

<script>

var r=confirm

(“press a button”);

if (“r==true”)

{x=“You pressed OK!”;

}

else

{x=“You pressed cancel!”;

}

</script>

Prompt Box

It is used if one wants a user to input a value before entering a page. When a confirm box pops up, a user must click OK or CANCEL to proceed after entering a value.

Example:

<script>

prompt(“BUY”, “YES”);

</script>

Popup Box Line Break

This is used to display line break inside in popup box. It is done by inserting a back-slash followed by the character n between the values.

Example:

<script>

alert(“Hello\nHow are You?”);

</script>

DATE AND TIME

Time

<script>

function time(){

time=new Date()

document.write(time.getHours() + “:” + time.getMinutes())

}

</script>

The above syntax is inserted into the head section and it is called up by the time function also inserted into the body function that is <script>time()</script> below:

Welcome Sadiku Monday. It is

<script>

time()

</script>

O’Clock!

Date and Time

The Date object is use to work with dates and times. Date objects are created with the Date() constructor.

Example:

<script>

var myDate=new

Date();

myDate.setDate

(myDate.getDate());

document.write(myDate)

</script></br>

The above syntax is inserted into the head section and it is called up by the time function also inserted into the body function that is <script>time()</script> below:

<script>

date()

</script>

FUNCTION

A function is a block of code that will be executed when someone calls it. The function is written inside a curly braces preceded by the “function” keyword.

Example:

<script>

function

myFunction(name, job)

{

alert(“Welcome ” + name +”, the ” + job);

}

</script></br>

The above is a function which is then called when a user clicks the onclick event handler below.

<button

onclick=“myFunction(‘Sadiku Moday’,

‘Engineer’)”>Try it

</button></br>

EVENT HANDLERS

Commands that are incorporated into the HTML source code and that carry out a predefined function or command given particular actions – are called event handlers. All event handlers begin with on.. e.g the onclick, onabort, onfocus, onchange, onload, onmouseover, are event handler.

onload

<script>

function hello(){

alert(“Hello World!”)

}

</script>

The above syntax is loaded once the page loads.

onclick

<script>

function message(element){

alert(“You clicked the ” + element + ” element!”)

}

</script>

The above syntax is inserted into the head section and it is called up by the hello function also inserted into the body function.

<form>

<input type=“radio” name=“Radio” onClick=“message(‘Radio Button

1’)”>Option 1

<input type=“radio” name=“Radio” onClick=“message(‘Radio Button

2’)”>Option 2

<input type=“checkbox” onClick=“message(‘Check Button’)”>Check

Button

<input type=“submit” value=“Send” onClick=“message(‘Send Button’)”>

<input type=“reset” value=“Reset” onClick=“message(‘Reset Button’)”>

<input type=“button” value=“Mine” onClick=“message(‘My very own Button’)”>

</form>

onMouseOverover

This tells the user when he onMouseOver the image in the event below:

<a href=“http://index.html”

onmouseover=“alert(‘you onMouseOverover the image’);return false”>

<img src=“images/cdw.jpg” width=“100” height=“30”>

Others are:

- onBlur
- onMouseOver
- onFocus
- onChange

COLOURS IN HTML DOCUMENT USING JAVASCRIPT

Using JavaScript, you can access the color settings of HTML documents. Elements you can

change include the text color, background color, and the colors of not yet visited, already visited, and active links.

```
<script>
function color(background, foreground, new_link, visited_link,
active_link){
document.bgColor=background
document.fgColor=foreground
document.linkColor=new_link
document.vlinkColor=visited_link
document.alinkColor=active_link
}
</script>
This is normal Text.<br>
<a href=“Page2.htm”>I have not visited Page 2 yet.</a><br>
<a href=“Page3.htm”>But I have already visited Page 3.</a><br>
<a href=“Page4.htm”>I am clicking on this link right now.</a><p>
<a href=“javascript:color(‘orange’,‘gray’,‘green’,‘yellow’,‘blue’)
“>Color change</a><br/>
```

REPEATED PERFORMANCE

In every programming language, there are times where you'll want to repeat the same command over and over. The commands for repeated performance are the same in nearly every language: they are called for and while. Here you'll see examples of each command.

Looping With For

Assuming that you want to publish a table of squared numbers from 1 to 100, now you could, of course, calculate all the values with your pocket calculater and then enter them into a table, or you could call on JavaScript to help you. For the JavaScript variation – which requires significantly less source code – you'll need the following function:

Example:

```
<script >
function square(){
for (var i=1; i<=100; i++){
document.write("<tr><td>")
document.write(i)
document.write("</td><td>")
document.write(i*i)
document.write("</td></tr>")
}
}
</script>
```

The one above is entered into the head section while the one below is enter into the body.

```
<table border=2>
<script>
square()
</script>
</table>
```

Looping with While

The while loop closely resembles the “for loop”, but the while loop doesn’t require any counter variables and there is no operation that changes the value of these counter variables. The for loop will simply run until the given condition is false. Its beginning is nearly identical to that of the for loop – it is used when you already have a counter variable with a value since no fixed value is given to the counter variable at the beginning of the loop.

Example:

```
<script >
function square(){
i=23
while (i<=100){
document.write(“<tr><td>”)
document.write(i)
document.write(“</td><td>”)
document.write(i*i)
document.write(“</td></tr>”)
i++
}
}
</script>
```

The one above is entered into the head section while the one below is enter into the body.

```
<table border=2>
<script>
square()
</script>
</table>
```

Array

The array object is use is use to store multiple values in a single variable. They are simply an ordered stack of data items with the same data type. JavaScript arrays start at zero. You can access and display an array element by referring to the name of the array and the element's index number e.g. document.write(country[7]) in the example below: that is it is the document.write that calls the country with the number it contains. change 7 to any other number in the array to see the effect.

Example:

```
<script>
var country=new Array();
country[0]="Nigeria";
country[1]="China";
country[2]="U.S";
country[3]="Iran";
country[4]="U.K";
country[5]="Russia";
country[6]="Egypt";
country[7]="Cuba";
country[8]="Japan";
document.write(country[7])
</script></br>
```

Note:

The letter 'A' in the array constructor Array() must be capital letter.

Factorial Table

This is use to get factorial values to get the factorial of numbers from 1 to 18(<19), we have the code below:

```
<script>
var fact = 1;
for(i = 1; i <19; i++) {
```

```
fact = fact*i;
document.write(i + “! = ” + fact + “<br>”);
}
</script>
```

DOCUMENT FORMATTING

Create a New Formatted Page

<SCRIPT language=“JavaScript”>

function newPage() {

var userName = prompt(“m:”, ””)

document.write(“<H1>Welcome ” + userName + “</H1>
”)

document.write(“<H2>to your new home page.</H2>”)

}

</SCRIPT>

To allow the user to create a new page, the link below call on the newpage function in the head section:

<A HREF=“JavaScript:newPage()”>Create-a-Page!</A>

Create a New Formatted Window

<SCRIPT language = “JavaScript”>

function openWin() {

window.open(“index.html”, “newWindow”,

“height=100,width=100,”)

}

</SCRIPT>

To allow the user to create a new page, the link below call on the newpage function in the head section:

<A HREF=“JavaScript:openWin()”>New Window!</A>

EVENT OBJECT

Document Object

This Uses either getElementById() or getElementsByName().

Using getElementById():

<script type=“text/javascript”>

function getElement() {

var x=document.getElementById(“myHeader”)

alert(“I am a ” + x.tagName + ” element”)

}

</script>

To allow the user to get element By Id, the link below call on the getElementById function in the head section:

<h2 id=“myHeader” onclick=“getElement()”>Click to see what element I am!</h2>

Using getElementsByName():

<script type=“text/javascript”>

function getElements() {

var x=document.getElementsByName(“myInput”)

alert(x.length + ” elements!”)

}

</script>

To allow the user to get element By Id, the link below call on the getElementById function in the head section:

<input name=“myInput” type=“text” size=“10”>

<input name=“myInput” type=“text” size=“10”>

<input name=“myInput” type=“text” size=“10”>

<input name=“myInput” type=“text” size=“10”>

<input name=“myInput” type=“text” size=“10”>

<input name=“myInput” type=“text” size=“10”>

Event Object: What are the coordinates of the cursor?

<script type=“text/javascript”>

function show_coords(event) {

x=event.clientX

y=event.clientY

alert(“X coords: ” + x + “, Y coords: ” + y)

}

</script>

To allow the user to get the coordinate of his cursor anywhere on the page, the link below call on the function show_coords function in the head section:

<body onmousedown=“show_coords(event)”>

Click in the document. An alert box will alert the x and y coordinates of the

cursor.

</body>

Event Object: What is the Unicode of the key pressed?

<script type=“text/javascript”>

function whichButton(event) {

alert(event.keyCode)

}

</script>

With the code above, Press a key on your keyboard. An alert box will alert the unicode of the key Pressed.

Press a key on your keyboard. An alert box will alert the unicode of the key pressed.

Event Object: Which element was clicked?

<script type=“text/javascript”>

```
function whichElement(e) {
var targ
if (!e) var e = window.event
if (e.target) targ = e.target
else if (e.srcElement) targ = e.srcElement
if (targ.nodeType == 3) // defeat Safari bug
targ = targ.parentNode
var tname
tname=targ.tagName
alert(“You clicked on a ” + tname + ” element.”)
}
</script>
```

The clicked element will be gotten by the code bellow calling onto the whichElement function in the head section:

```
<body onmousedown=“whichElement(event)”>
<p>Click somewhere in the document. An alert box will alert the tag name of the element you clicked on.</p>
<h3>This is a header</h3>
<p>This is a paragraph</p>
<img border=“0” src=“ball16.gif” width=“29” height=“28” alt=“Ball”>
</body>
```

IMAGE ROLLOVER

This changes the image once the user over the original one.

<A HREF=“URL”

onMouseOver=“document.hot.src=‘images/BUTTON 4.jpg’”

onMouseOut=“document.hot.src=‘images/BUTTON 5.jpg’”>

<IMG name=“hot” src=“hot2.gir”> </A>

JAVASCRIPT VARIABLE

JavaScript variables are containers for storing data values. These variables must be assigned as explained below:

In this example, m, n, and p, are variables and m and n will be given some values: To assign a variable, you use var- variable, = for assigning, set the value e.g 7, and end it with semi colon.

Examples

1. Declaring ariables

   var s = 9;

   var w = 39;

   var person = “Steven Bright”;

   var price = $2.99;

   var pi = 3.14;

2. var m = 9;

   var n = 23 ;

   var p = m + n;

   I.e p = 9+23 = 32

3. var m = 12;

   var n = 40 ;

   var p = m - n;

   I.e p = 12-40 = -28

Note: m stores the value 9, n stores the value 23, p stores the value 32.

Rules for Constructing Variable Names:

The general rules for constructing names for variables (unique identifiers) are given below and make sure you pay attention to them while naming your JavaScript variables:

- Names can contain letters, digits, underscores, and dollar signs.
- Names must begin with a letter.

- Names can also begin with $
- Names are case sensitive (y and Y are different variables).
- Reserved words (like JavaScript keywords) cannot be used as names.

JAVASCRIPT OPERATORS

JavaScript Arithmetic Operators

These operators are used to perform arithmetic on numbers be it literals or variables. Below are the various JavaScript arithmetic operators and their functions:

Operator and Description

+ Addition

- Subtraction

* Multiplication

/ Division

% Modulus

++ Increment

— Decrement

JavaScript Assignment Operators

These operators assign values to JavaScript variables.

Operators and their uses Explained

Operator =

x = y same as x = y

Operator +=

x += y same as x = x + y

Operator -=

x -= y same as x = x - y

*Operator *=*

x *= y same as x = x * y

Operator /=

x /= y same as x = x / y

Operator%=

x %= y same as x = x % y

Example

var x = 10;

x += 5 ;

x = x + =y same as x = x+y

where y = 5 in this case

Therefore,

x = 10+5 = 15.

JavaScript String Operators

This involves the use of + operator to add (concatenate) strings i.e texts.

Example

var txt1 = “The room” ;

var txt1 += “is hot” ;

The result of txt1 will be: The room is hot

ADDING STRINGS AND NUMBERS

Remember that adding two numbers will return the sum, but adding a number and a string will return a string:

Example

w = “steven” + 2006 ;

The result of w will be: steven2006

JAVASCRIPT BOOLEANS

JavaScript Boolean represents one of two values: that is true or false. This implies that JavaScript Boolean Values is True or False.

Examples

1. Boolean(57> 34) //

   returns true

2. Boolean(45 > 78) //

   returns false

Comparisons and Conditions

In this case, everything with a “Real” value is True but everything Without a “Real” is False.

Examples

1. var y = 0 ;

   Boolean(y); //

   returns false

2. var y = null;

   Boolean(y); //

   returns false

3. var y= false ;

   Boolean(y); //

   returns false

JAVASCRIPT MATH OBJECT

The JavaScript Math object allows you to perform various mathematical tasks on numbers.

Math.round()

This rounds a number to the nearest integer:

Example

Math.round

(467.9); //

Returns 468

Math.min()

Math.min() is use to find the lowest value in a list of arguments:

Example

Math.min (7, 180 , 60, 32,

0 , -250); //

Returns -250

Math.max()

Math.max() is use to find the highest value in a list of arguments:

Example

Math.max(7 , 180 , 60, 32,

0 , -250); //

Returns 180

Math.random()

This returns a random number between 0 (inclusive), and 1 (exclusive):

Example

Math.random()

// returns a random

number

Math.ceil()

This rounds a number up to the nearest integer:

Example

Math.ceil

(37.3); //

Returns 37

Math.floor()

This rounds a number down to the nearest integer:

Example

Math.floor

(35.9); //

Returns 35

JAVASCRIPT USES OUTSIDE WEB PAGES

Apart from JavaScript use Web browsers and servers, JavaScript interpreters are embedded in a number of other tools and each of these applications provides its own object model that provides access to the host environment. Some of these tools are:

Embedded scripting language

The following are implemented using JavaScript:

Opera’s extensions, Google’s Chrome extensions, , Apple’s Dashboard Widgets, Microsoft’s Gadgets, Yahoo! Widgets. OpenOffice.org, an office application suite, as well as its popular fork LibreOffice.

Scripting engine

The following Scripting languages are also base on JavaScript:

Microsoft’s Active Scripting technology supports JScript as a scripting language.

The Qt C++ toolkit includes a QtScript module to interpret JavaScript, analogous to Java’s javax.script package.

The Java programming language introduced the javax.script package in version 6 that includes a JavaScript implementation based on Mozilla Rhino.

Application platform

The following Application platforms are also base on JavaScript:

Adobe Integrated Runtime is a JavaScript runtime that allows developers to create desktop applications.

The Mozilla platform, which underlies Firefox, Thunderbird, and some other Web browsers, uses JavaScript to implement the graphical user interface (GUI) of its various products.

GNOME Shell, the shell for the GNOME 3 desktop environment, also made JavaScript its default programming language.

CONCLUSION

JavaScript as a scripting language is mostly use for the web, web base applications, and development of mathematical base projects, but it also found great use in Embeded Scripting language, Sripting Engine, and Application platforms such as Mozilla platform and ActionScript.

Thank you for reading my book. If you enjoyed it, won’t you please take a moment to leave me a review at your favorite retailer? And also inform your friends to get their copy at their favorite retailer too.

Thanks!

Steven Bright

Author

ABOUT THE AUTHOR

Steven Bright is a Tech expert, graphic designer, web developer, and blogger. He is also the Author of:

CorelDraw How: The Fundamental of CorelDRAW

Tools and Function Lists: Engineering Tools Manual

Microsoft Word Fundamental: A Step by Step Guide

Web Design and Development Technology: Website How

The Best of Microsoft PowerPoint: Brand Exposure Techniques

Hacking Through Microsoft Excel Skills: A Step by Step Approach

The wave of Change: Adams Aliyu Oshiomhole: Why he is the man we need to Create the Atmosphere of Change we Desire to Build a Successful, Great and Powerful Nation

Connect with Steven Bright

Amazon author page: http://amazon.com/author/stevenbright

Facebook: http://www.facebook.com/steven.bright.9022

Blogs:

http://ebookstrati.blogspot.com

http://computerskillsworld.blogspot.com

http://hackingthroughsuccess.blogspot.com

###