

### Misc. functions

```
typeof null;
num.toString(radix);
parseInt("3 blind mice");
parseInt("0xFF");
parseInt("ff", 16);
parseInt("eleven");
javascript:alert("Hi there");
eval("3 + 12");
escape(str);
unescape(str);
```

Returns string "object" (or whatever the thing is).  
Returns num var as a string var.  
Returns 3.  
Returns 255.  
Returns 255 also.  
Returns NaN.  
Execute JS as a URL.  
Evaluates a text string as code.  
Returns str in web-compliant form. E.g. "hi ho" = "hi%20ho".  
Returns str from web-compliant form. Undoes escape().

### Window functions

```
alert("Are you sure about that?");
confirm("Continue loading page?");
prompt("Enter your name, please.");
close();
find(), home(), print(), stop();
focus(), blur();
moveBy(), moveTo();
resizeBy(), resizeTo();
scrollBy(), scrollTo();
var intervalID = setInterval("bounce()", 10000);
clearInterval(intervalID);
setTimeout("display_time()", 10000);
clearTimeout();
var w = window.open("smallwin.html", "SmallWin",
    "width=400,height=300,status,resizeable,menubar");
```

Prompts with OK box.  
Prompts with OK / CANCEL message box.  
Prompts with text box.  
Close the window.  
Duplicates of buttons. Not in IE4.  
Cause us focus, or lose focus. Not in IE3.  
Move the window.  
Resize the window.  
Scroll the document displayed in the window.  
Set func to be repeatedly called w/ delay. Call OUT of func.  
Cancel function to be repeatedly invoked with delay.  
Call display\_time() in 1 sec. Put in display\_time() to loop.  
Cancel function invoked *once* after delay.  
Opens another window, loading smallwin.html, with name SmallWin, and given dimensions.

### Window properties

```
closed
defaultStatus, status
document
frames[]
history
history.back();
history.forward();
history.go();
innerHeight, innerWidth, outerHeight, outerWidth
locationbar, menubar, personalbar, scrollbars, statusbar, toolbar
name
opener
```

Returns true if window has been closed (useful for open()).  
Sets default status line, current status line (in status bar).  
Refers to the current document object (the HTML page).  
Refers to frames, if any.  
A reference to the history object, represents browsing hist.  
Go back a link.  
Go forward a link.  
Goes to a link, buggy in NS2&3, weird in IE3 – best avoid.  
Inner and outer dimensions of the window; not in IE4.  
References to visibility of parts. Not in IE4.  
Name of current window. Useful for <A TARGET>, for ex.  
Reference to Window opened this, or null if opened by user.

### Document functions

```
document.write("<h2> Table of Factorials</h2>");
document.writeln("Hi there.");
document.forms[i].elements[j++];
document.close();
```

Outputs to current doc, writes HTML.  
Outputs with a <CR> appended at end.  
Access forms and form elements via array scripting.  
Closes this window-if opened by JS, in later browsers.

### Document properties

```
document.location
document.forms[0], document.myform
document.alinkColor
document.anchors[]
document.applets[]
document.backgroundColor = "#040404";
document.cookie
document.embeds[]
document.fgColor = "blue";
document.images[]
document.lastModified
document.linkColor
document.links[]
document.referrer
document.title
document.URL
document.vlinkColor
document.domain
```

Represents URL document displayed. Set to load new doc.  
Refer to document forms.  
Color of hyperlink while clicked on (same as <BODY> tag).  
Hyperlink array.  
Applet array.  
Background color of document.  
Allows JS to read / write cookies. == "" if not set.  
Embedded array.  
Text color of document (same as <BODY> tag).  
Images array.  
Returns string of the date we were last modified.  
Color of unclicked links. Same as LINK attr. in <BODY>.  
Links array.  
URL of doc that ref'd us, if any.  
The title (<TITLE>) of this document.  
URL we were loaded from, same as location.href.  
Visited link color. Same as VLINK in <BODY>.  
Returns the name of the domain you're currently at.

## Navigator properties

navigator.appName  
navigator.appVersion  
navigator.userAgent  
navigator.appCodeName  
navigator.platform  
navigator.language  
navigator.userLanguage, navigator.systemLanguage  
navigator.javaEnabled()

The simple name of the web browser.  
The version number and/or other version info about browser.  
appName and appVersion combined, usually.  
The code name of the browser. E.g., "Mozilla."  
Platform they're running on as of JS1.2.  
Language of browser. "en" (English). NS4+, not IE.  
IE4+ version of navigator.language property.  
Returns true if Java supported and enabled on this browser.

## Math functions

Math.round(x/15);  
Math.pow(x,y);  
Math.sqrt(x\*x + y\*y);  
Math.random();  
Math.max(i, j);  
Math.min(i, j);  
Math.floor(j);  
Math.ceil(j);

Rounds to the nearest integer.  
Returns  $x^y$ .  
Returns square root of argument.  
Returns random between 0.0 – 0.1.  
Returns greater of two numbers.  
Returns lesser of two numbers.  
Rounds j down.  
Rounds j up.

## String functions

str.length;  
str.charAt(str.length - 1);  
str.substring(1, 4);  
str.indexOf("a"), str.lastIndexOf(" ");  
str.anchor(name), str.big(), str.blink(), str.bold(), str.fixed(), str.italics(), str.link(href);  
str.small(), str.strike(), str.sub(), str.sup()  
str.fontcolor("#090909"), str.fontSize(1-7 | "+2");  
str.match(), str.replace(), str.search();  
str.slice(2, 6);  
str = "1,2,3,4,5"; arr = str.split(",");  
str.substr(5,2);  
str.toUpperCase(), str.toLowerCase();

Returns a string's character length.  
Returns the last character of a string. SEE NOTES BELOW.  
Returns str[1] through str[3].  
Returns position of first "a" / last " " in string str, -1 if none.  
Return str with certain formatting imposed upon it.  
anchor=<A NAME=name>, link=<A HREF>, rest obvious.  
Set the string's font color / size.  
Regex / string match / replace functions.  
Returns str[2] through str[5], neg. args start from end.  
Returns array of substrings, split by delimiter ",".  
Returns str[5] through str[5+2].  
Convert a string's case.

## Vital notes

- Semi-colons are optional, but recommended.
- JavaScript is case-sensitive; HTML embedded names (such as **onClick**) are not.
- Always declare variables with **var**. Variables not declared with **var** are global automatically. Keep vars declared on top for clarity. Ex: **var ind = 0;**
- Fun with strings: "Hi there" and 'Hi there' are both legal string definitions.
- Octal number definitions begin with a 0. Hex begins with 0x (or 0X). Ex: 026, 0xAF, 0377, 0xff...
- JavaScript represents all numbers as floating point. Numbers can be extremely large, like: -999 tril <-> +999 tril.
- String indexing, like **str[str.length - 1]**, is supported by Nav4+, not IE4 though (IE5?).
- null is a special value in JS. It is not equivalent to 0. It represents the lack of an object, number, string, etc. Sometimes, converted to 0, though.
- Functions can be nested since JS1.2.
- undefined can be tested for by making an uninitialized variable: **var undefined; if (myform["checkbox" + ind] == undefined) ...**

## Useful code tidbits

```
<input type = "button" ... onClick = "alert('You clicked me!')">
var square = new Function("x", "return x*x;");
var square = function(x) { return x*x; }
image.width <-> image["width"]
var pattern = new RegExp("\bjava\b", "i");
var o = new Object();
var point = { x:2.3, y:-1.2 };
var sq = { upleft: { x:point.x, y:point.y }, lowright: { x:(point.x+side), y:(point.y+side) }};
document.images[i].width;
var a = new Array(); a[0] = 1.2; a[1] = "JavaScript"; a[2] = true; a[3] = { x:1, y:3 };
var a = [1.2, "JavaScript", true, { x:1, y:3 }];
var matrix = [[1,2,3], [4,5,6], [7,8,9]];
var sparseArray = [1,,,5];
for (var i in obj);
Circle.prototype.pi = 3.14159;
<body bgcolor = "&{favorite_color()};">
```

When user clicks button, execute "onClick" portion.  
Function literal – variable holds function definition.  
Function literal. square(144) would execute it.  
Two ways to access object properties.  
Creates RegExp object (regular-expression).  
Makes a general object... you can make up properties.  
Object literal – general object with init'd properties.  
Object literal, with sq.upleft.x == point.x, for example.  
Way to access images as array of document object.  
Creates an array. Once made, indexed elems. added easy.  
Alternate way since JS1.2.  
Nested array definition.  
Makes array with some undefined elements.  
The for/in loop loops through the properties of an object.  
Sets a pi val for all Circle objects.  
&{ JS-statements; }; used in NS3+, !IE4, embed JS in HTML.

## Notable constants

Number.MAX\_VALUE  
Number.MIN\_VALUE  
Number.NaN  
Number.POSITIVE\_INFINITY  
Number.NEGATIVE\_INFINITY

Largest representable number.  
Most negative representable number.  
Special Not-a-number value.  
Special value to represent infinity  
Special value to represent negative infinity.

## Object-based browser detection

Document object	Browser that supports it
document.images	NS3+, IE4+
!document.images	NS2, IE3
document.layers	NS4+
document.all	IE4+
document.layers    document.all	NS4+, IE4+

```
if (document.images) document.images[0].src = "/images/myimg1.jpg";
```

Example usage of browser object detection.

## Useful events

Handler	Triggered when	Supported by
OnAbort	Loading interrupted.	Image
OnBlur	Element loses input focus.	Text elms., Window, all other elms.
OnChange	User changes an elm., moves on.	Select, text input elements
OnClick	User single-click. Ret. false = cancel.	Link, button elements
OnError	Error occurs while loading an image.	Image
OnFocus	Element given input focus.	Text elms., Window, all other elms.
OnLoad	Document or image finishes loading.	Window, Image
OnMouseOut	Mouse moves off element.	Link
OnMouseOver	Mouse moves over elm.	Link
OnReset	Form reset request, false = no reset.	Form
OnSubmit	Form submit, false = no submit.	Form
OnUnload	Document is unloaded.	Window

## Cookie stuff

```
document.cookie = "version=" + escape(document.lastModified) +  
    "; path=/; expires=" + mydate.toGMTString();  
javascript:alert(document.cookie)  
document.cookie = "Name=Joe Bob; path="/;
```

Sets persistent cookie. `escape()` converts to web form, `unescape()` undoes from web form.

Type this in your browser to see the cookie set for site!  
Minimal cookie setting.

## Variable arguments

```
function add_all_together() {  
    for (i = 0; i < add_all_together.arguments.length; i++)  
        total += add_all_together.arguments[i];  
}
```

The **arguments** method stores the arguments themselves, and the number of arguments passed to each function, as we demonstrate here.

## Pre-load and update images

```
myimg1 = new Image();  
myimg1.src = "/images/image01.jpg";
```

Make a new image object,  
set the image src to preload it.

```
function imgfilter(imgobj, newimg) {  
    if (document.images)  
        document.images[imgobj] = newimg.src;  
}
```

If the browser supports the `document.images` method,  
set a new src for the `imgobj` image object argument.

## Date stuff

```
var now = new Date();  
var xmas = new Date(97, 11, 25);  
now.toLocaleString();  
xmas.toGMTString();
```

Date obj representing current date and time.  
Date obj for 25-Dec-97, note months index from 0!  
Returns string of date and time.  
Returns string of date and time in GMT time.

## Creating a Plain-Text Document

```
var w = window.open("", "console", "width=600,height=300,resizeable");  
w.document.open("text/plain");  
w.document.writeln(msg);
```

We specify the optional `[window.]open` because there's a `document.open()` function too.

## To insure a child window you want to update is still open

```
if (!w.closed) w.close();
```

This will close the window declared above if it's still open,  
only if that window hasn't been closed by the user.

## To generate a random number between X and Y

```
function generate( lbound, ubound ) {  
    return Math.floor( ( ubound - lbound + 1 ) * Math.random() + lbound );  
}
```

// This one I got somewhere. It doesn't work. It returns only odd numbers, and goes 1 beyond the set range sometimes???

```
function generate( x, y ) {  
    var range = y-x+1;  
    var i = ( "" + range ).length;  
    var num = ( Math.floor( Math.random() ) * Math.pow( 10, i ) ) % range +  
        parseInt( x );  
    return num;  
}
```

## Detecting Shockwave

```
<SCRIPT LANGUAGE="JavaScript">
<!--hiding contents of script from old browsers, just in case

//If this browser understands the mimeType property and recognizes the MIME Type //application/futuresplash"...
if (navigator.mimeType && navigator.mimeType["application/x-shockwave-flash"]){

    //...write out the following <EMBED> tag into the document.
    document.write('<EMBED SRC="flash_movie.swf" WIDTH="220" HEIGHT="110" LOOP="true" QUALITY="high">');
}

//Otherwise,...
else {

    //...write out the following <IMG> tag into the document. The image need
    //not be the same size as the Flash movie, but it may help you lay out the
    /page if you can predict the size of the object reliably.
    document.write('<IMG SRC="welcome.gif" WIDTH="220" HEIGHT="110" ALT="Non-Shockwave Welcome">');
}

//Done hiding from old browsers. -->
</SCRIPT>
```

## Using JS to Write to Frames

Just give the frames names, then access them by name:

```
<frameset cols="*,*">
<frame name=left src="a.html">
<frame name=right src="b.html">
</frameset>
```

```
top . right . document . open ();
top . right . document . writeln ("Hello.");
top . right . document . close ();
```

```
top.document.location = "newpage.htm";
```

Breaks from frames, goes to new page.