

JavaScript cheat sheet

Holmer Hensen
NISLab
University of Southern Denmark

November 1, 2002

1 JavaScript Document Template

A JavaScript can be included both in the header and/or the body of the document, the choice depends on the task. There can be several JavaScripts in a document.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>JavaScript Template</title>
    <script language="javascript" type="text/javascript">
      <!-- Hide script from old browsers

      /*
       * Author: Firstname Lastname <email>
       * Date:
       * Version:
       * Last modified:
       *
       * Purpose: Description of the problem the program was written to solve.
       * Known Bugs:
       */

      // End hiding script from old browsers -->
    </script>

    <noscript>
      <b>You don't have JavaScript enabled or your browser does not support JavaScript</b>
    </noscript>
  </head>
  ...
```

2 Semicolons, whitespace and case sensitive

In JavaScript semicolons are optional. Use them anyway, it is more secure! Whitespaces are ignored by JavaScript, unless as part of a String (e.g. "I like JavaScript") and for separating keywords, variable names, numbers and functions. JavaScript is case sensitive, that means: `bgColor` is not equal to `bgcolor`.

3 Comments

3.1 Single-line comment

```
// e.g. to comment the meaning of a variable
```

3.2 Multiple-line comment

```
/* e.g. to comment purpose and parameters of a function
 * Function: saySomething
 * Purpose: displaying a message in an alert box
 * Parameters:
 *   message: the string to be displaying in the alert box
 * Return: The return value of this function is void
 *         (no return value)
 * Side Effects: This function changes no global variables.
 */
```

4 Variables

4.1 Variable assignment

Using `var` outside of function is optional. Assigning variables using `var` inside of functions is required if (a) we want to declare a local variable and a global variable with the same name already exists.(b)If recursive functions use variables with the same name.

```
x = 42;
var x = 42;
```

4.2 Variable scope

Variables in JavaScript have in general global scope. However, variables assigned inside of functions have function-local-scope. Take care, if you define a local variable in a function and a global variable with the same name exists, the global variable is not visible any more inside the function, no matter where you define the local variable.

5 Arrays

```
beans = ["Java beans","Coffee beans","Cacao beans"];
myCars = new Array("Jaguar", "Mercedes", "Rolls Royce");

noOfEntries = myCars.length;

myCD = new Array(42) \\declares array with 42 elements
myCD[0] = "Queen" \\first element
myCD[1] = "Carlos Santana"
myCD[2] = "Mozart"
myCD[3] = "Chopin"
...
myCD[41] = "Shubidua" \\last element

\\Associative Array
assoarr = {color1: "green", color2: "yellow", color3: "white"};
```

6 Types

```
Types in JavaScript:
  object
  function
  string
  number
  boolean
  undefined
```

6.1 typeof

The `typeof` operator is used to determine the type (see above) of a variable, string, keyword or object. `typeof` returns a string.

```
var aPowerFunc = new Function("x", "return x * x")
var aDate = new Date()
var aString = "I am a string"
var aNumber = 42

typeof aPowerFunc is object
typeof aDate is object
typeof aString is string
typeof aNumber is number
typeof nonExistVar is undefined
typeof Date is function \\ predefined object Date
```

7 Objects

```
function objDef(name) {
  this.objvar = name
}

\\ creation of an instance of the above defined object "class"
myObj = new objDef("Hans")
```

7.1 Methods, Properties

A *method* is a function associated with an object. *Properties* are defined by assigning a value to it.

```
\\Property
\\objectName.propertyName = value

\\Method
\\objectName.methodName = function

function calcArea(radius)
return(radius * radius * Math.PI);

function circle(color, radius)
this.fillColor = color;
this.area = calcArea(radius);
```

8 Operators

Arithmetic Operators	
+	Addition
-	Subtraction and unary negation
++	Increment
--	Decrement
*	Multiplication
/	Division
%	Modulus

String Operators	
+	Concatenation
+=	Adds two strings and assigns the result to the first

Logical Operators	
&&	AND
	OR
!	NOT

Assignment Operators	
=	variable assignment
+=	Adds two numbers and assigns the result to the first
-=	Subtracts two numbers and assigns the result to the first
*=	Multiplies two numbers and assigns the result to the first
/=	Divides two numbers and assigns the result to the first
%=	Calculates the modulus of two numbers and assigns the result to the first

Assignment Operators	
==	equal
!=	not equal
===	strict equal (equal and same type)
!==	not equal and/or not same type
>	greater than
>=	greater than or equal
<	less than
<=	less than or equal

Other Operators	
new	Create Object
delete	Delete Object
this	to refer to the current Object
typeof	returns string with type of operand
void	evaluates expression without returning a value

9 Conditional Statements

9.1 if statement

<pre>if (condition) { statements }</pre>
<pre>if (condition) { statements } else { statements }</pre>
<pre>condition ? expr1 : expr2</pre>

9.2 switch

```
switch (expression){
  case label:
    statement;
    break;
  case label2:
    statement;
    break;
  default : statement;
}
```

10 Loops

10.1 for statement

```
for(var i=0; i < array.length; i++){
  ...
}
```

10.2 do ...while statement

```
var i=0;
do {
  i+=1;
} while (i < 5)
```

10.3 while statement

```
n=8
while( n > 0) {
  n--
}
```

10.4 for ...in statement

The for ...in statement iterates over all the properties of an object.

```
\\see Object for creation of Student class
student1 = new Student("Hans Christian", "Andersen", 1805);
for (props in student1) {
  document.write(student1.props)
}
```

10.5 break and continue statement

A `break` terminates the innermost enclosing loop of: `while`, `do-while`, `for`, `switch`

A `continue` jumps back to the `while` condition.

```
while(i > 0){
  if(a[i]=="Orange")
    break;
  i--;
}
while(i > 0){
  if(a[i!="Orange")
    continue;
  oranges++;
}
```

11 Predefined Objects

Date
Array
Boolean
Function
Math
Number
RegExp
String

11.1 Date

Date	
<code>getDate()</code>	Day of month (1-31)
<code>getDay()</code>	Day of week (0-6)
<code>getFullYear()</code>	0-99 or four digits after 1999
<code>getFullYear()</code>	Year from 1900 up
<code>getHours()</code>	Hours (0-23)
<code>getMonth()</code>	Month (0-11 [sic!])
<code>getSeconds()</code>	Seconds (0-59)
<code>getTime()</code>	Number of milliseconds since 1 January 1970
<code>setDate()</code> , <code>setHours()</code> , <code>setYear()</code> , ...	Sets date, hours, year, etc.
<code>parse()</code>	Returns time in milliseconds since 1 January 1970 from given date/time string parameter
<code>toGMTString()</code>	
<code>toUTCString()</code>	
<code>toString()</code>	

11.2 Date examples

```
now = new Date \\current date nowDay = now.getDay()\\(0-6)
nowHour = now.getHours()\\(0-23)
```

11.3 Boolean

```
x = true; \\logical true value
y = false; \\logical false value
z = new Boolean(false); \\x is false
z2 = new Boolean("false"); \\x is true !
z3 = new Boolean(true); \\x is true
```

11.4 Math

Math methods	
abs	absolute value
sin, cos, tan	standard trigonometric functions
acos, asin, atan	
min, max	returns max/min value of to arguments
round	rounds argument
sqrt	square root
pow	exponential; first arg base, second arg exponent
random	generates a random number between 0 and 1

Math constants	
Math.E	Euler's constant
Math.PI	natural logarithm base 10
Math.SQRT1_2	square root of 1/2
Math.SQRT2	square root of 2

11.5 Math examples

```
a = Math.PI * r*r;
with(Math){
a = PI * r*r;
x = sin(y);
}
```

11.6 RegExp

Math methods	
[xyz]	set of characters; Matches any of the enclosed characters
[^xyz]	negated set of characters; Matches any of the non enclosed characters
*	matches preceding character 0 or more times
+	matches preceding character 1 or more times
?	matches preceding character 0 or 1 time
.	matches any character except newline
()	to mark a matched substring for future reference
x y	matches either x or y
[\b]	matches backslash
\	escape character
^	beginning of line
\$	end of line

11.7 RegExp examples

```
var message = "String";
var re = /r./;
message.match(re);
message.search(re); \\returns startindex or -1 if not found
message.replace(re, "i");
```

11.8 String

String methods	
bold	Same as <bold> tags
italics	Same as <i> tags
concat	Concatenates 'adds' to the first string the
charAt	returns character at the given position; param: index
fontcolor	Set the string in <font color= <i>color</i> > tags
fontsize	Set the string in <font size= <i>size</i> > tags
indexOf	Return startpos of given string in the search string param: string to search for; return: startpos or -1 if not found
lastIndexOf	Returns index of last occurrence of param value or returns -1 if no occurrences
length	Length of string; length of "bold" is 4
link	Creates HTML hyperlink
big	Same as <big> tags
small	Same as <small> tags
strike	Same as <strike> tags
sub	Same as <sub> tags
substring	Returns substring of calling string from startindex to endindex and lastindex; params: startindex, lastindex
toLowerCase	Converts string to lowercase
toUpperCase	Converts string to uppercase
replace	params: search value, replace value
split	splits a string at the given separator params: separator, string to split

11.9 String examples

```
var message = "Simple String";
message[0] \\returns "S"
document.write(message.bold()); \\writes the message in bold
message.charAt(0) \\same as message[0]
message.concat(" enhanced"); \\gives "Simple String enhanced"
message.length \\returns 13
message.indexOf("Simple"); \\returns 0
message.lastIndexOf("String"); \\returns 7
message.substring(2,5); \\returns "mple"
```

12 Event Handling

Notice, that it is `ondblclick` and not `onDbLcLick`

12.1 Form Event Handling

Form Events	
onSubmit	Event when the user presses the submit Button
onReset	Event when the user presses the reset Button
onChange	Event when user changes a form field
onSelect	Event when the user selects text in either an INPUT or the TEXTAREA field
onBlur	Event when field loses focus
onFocus	Event when field comes into focus, when field is entered

Key Events	
onKeyDown	Event key pressed
onKeyUp	Event key released
onKeyPress	Event key pressed and released

Mouse Events	
onMouseover	Event triggered when entering the area
onMouseout	Event triggered when leaving the area
onMouseMove	Event when mouse moved
onClick	Event triggered after pressing and releasing mouse button
onDblclick	Event triggered after double clicking mouse button

Window Events	
onLoad	Event triggered after loading the document
onUnload	Event triggered after leaving webpage
onResize	Event triggered when resizing the window
onError	Event triggered when JavaScript error occurs