Javascript Cheatsheet

Puvendran Pillay

June 11, 2016

1 Comments

```
// in-line comment
/* multi-line
comment */
```

2 Datatypes

1. undefined
2. null
3. boolean
4. string
5. number
6. object
7. array
8. float

3 Variables

1. Declare variable

```
var ourName;
```

2. Assign value to a variable

```
ourName = "something";
```

3. Can also be done once

```
var ourName = "something";
```

4. Variable should be initialized when declared.

5. Javascript variables names are *case-sensitive.*

4 Operators

1. Arithmetic Operators

   (a) Addition

```
var x = 5;     // assign the value 5 to x
var y = 2;     // assign the value 2 to y
var z = x + y  // assign the value 7 to z (x + y)
alert(z);      // outputs the value in an alert box
```

   (b) Subtraction

```
var x = 5;     // assign the value 5 to x
var y = 2;     // assign the value 2 to y
var z = x - y  // assign the value 3 to z (x - y)
alert(z);      // outputs the value in an alert box
```

   (c) Multiplication

```
var x = 5;     // assign the value 5 to x
var y = 2;     // assign the value 2 to y
var z = x * y  // assign the value 10 to z (x * y)
alert(z);      // outputs the value in an alert box
```

   (d) Division

```
var x = 5;     // assign the value 5 to x
var y = 2;     // assign the value 2 to y
var z = x / y  // assign the value 2.5 to z (x / y)
alert(z);      // outputs the value in an alert box
```

   (e) Modulus *(returns the remainder of division)*

```
var x = 5;     // assign the value 5 to x
var y = 2;     // assign the value 2 to y
var z = x % y  // assign the value 1 to z (x % y)
alert(z);      // outputs the value in an alert box
```

   (f) Increment

```
//same as var x = x + 1;
var x = 0;       // assigns the value of 0 to x
x++;             // increases the value by 1
alert(x);        // outputs the value in an alert box
```

(g) Decrement

```
//same as var x = x - 1;
var x = 0;        // assigns the value of 0 to x
x--;              // Decreases the value by 1
alert(x);         // outputs the value in an alert box
```

2. Assignment Operators

(a) assignment (=)

```
var x = 10;
alert(x);  // outputs the value in an alert box
```

(b) Addition assignment

```
var x = 10;
x += 5;    // same as x = x + 5;
alert(x);  // outputs the value in an alert box
```

(c) Subtraction assignment

```
var x = 10;
x -= 5;    // same as x = x - 5;
alert(x);  // outputs the value in an alert box
```

(d) Multiplication assignment

```
var x = 10;
x *= 5;    // same as x = x * 5;
alert(x);  // outputs the value in an alert box
```

(e) Division assignment

```
var x = 10;
x /= 5;    // same as x = x / 5;
alert(x);  // outputs the value in an alert box
```

(f) Modulus assignment

```
var x = 10;
x %= 5;    // same as x = x % 5;
alert(x);  // outputs the value in an alert box
```

5 Escape Characters

1. Single quote (\')

2. Double quote (\")

3. Backsplash (\\)

4. New line (\n)
5. Carriage return (\r)
6. Tab (\t)
7. Backspace (\b)
8. Form Feed (\f)
9. Examples

```
var x = 'It\'s alright';
var y = "We are the so-called \"Vikings\" from the north."
```

6 Concatenation

1. Concatenate String with another string

```
var x = "Hello";
var y = "World!";
alert(x + " " + y); // outputs x and y with space in the
    middle
```

2. Concatenate String with a variable

```
var x = "Hello";
x += "World!";  // attaches the word to x
alert(x);
```

7 ".length" Property

Used to find the total number of characters contained

```
var x ="Hello World!";
alert(x.length);
```

8 "[]"

Used to index character *(Javascript starts counting from 0 not 1)*

```
var x ="Hello World!";
alert(x[1]); //selects "e" from "Hello"
```

Strings are immutable that means you can't change individual characters in them.

```
var x = "Jello World";
x[1] ="H"; //it just doesn't work
x = "Hello World";  // you must change it completely
```

9 Arrays

1. One dimensional array

```
var x = ["Hello" , "World"];
```

2. Multi-dimensional array

```
var x =[["Hello" , "World"] , ["Good" , "Bye"]];
```

3. Arrays are mutable and data can be changed individually

4. “[]” are also used to index arrays

```
var x =[["Hello" , "World"] , ["Good" , "Bye"]];
alert(x[1][0]); // outputs "Good"
```

5. “.push()” property Used to append data *to “push in from the back”*

```
var x = ["Hello" , "World"];
x.push(["Nasi" , "Lemak"]);
alert(x);
```

6. “.pop()” property Used to unappend data *to “pop out the back”*

```
var x = ["Hello" , "World"];
x.push();
alert(x);
```

7. “.shift()” property Used to unprepend data *to “shift out the front”*

```
var x = ["Hello" , "World"];
x.push();
alert(x);
```

8. “.unshift()” property Used to prepend data *to “unshift the front”*

```
var x = ["Hello" , "World"];
x.unshift(["Nasi" , "Lemak"]);
alert(x);
```

10 Functions

1. Use the keyword “function” to create a function

```
function favFood(){
  alert("I like eating my favourite food");
}
```

2. Functions can have arguments

```
function favFood(name , food){
  var name = prompt("What is your name?");
  var food = prompt("What is your favourite food?");
  alert( name + " " + "like eating" + " " + food);
}
```

3. Scopes Scope is the visiblity of variables

   (a) Local scope *(only available within functions)*

   ```
   function favFood(name , food){
     var name = prompt("What is your name?");  // Local scope!
     var food = prompt("What is your favourite food?");  // Local scope!
     alert( name + " " + "like eating" + " " + food );
   }
   alert( name + " " + "likes" + " " + food );
   ```

   (b) Global scope *(available everywhere)*

   ```
   var name; // declare outside
   var food; // the function to make it global

   function favFood(name , food){
     name = prompt("What is your name?");
     food = prompt("What is your favourite food?");
     alert( name + " " + "like eating" + " " + food );
   }
   alert( name + " " + "likes"+ " " + food );  // can be used again
   ```

   (c) Local scope takes predence over Global scope

4. "return" statement The return statement stops the execution of a function and returns a value from that function.

```
function helloWorld(){
  return 1 + 1;
}
```

11 Boolean

Boolean are either True or False

12 Conditional Statements

Used to specify a block of code to execute, if conditions are true

1. If...elseif...else Statements

```
var age = 20;
// this condition is false, so the code moves to the next one
if (age < 17){
  alert("You can’t drive!");
} elseif (age > 17){
    alert("You can drive!");
}else{
  alert("You should drive!");
}
```

2. Comparison and Logical Operators

   (a) Equal (==)

   (b) Strict Equal (===)

   (c) Not Equal (!=)

   (d) Not Strict Equal (!==)

   (e) Greater than (>)

   (f) Greater than or Equal to (>=)

   (g) Less than (<)

   (h) Less than or Equal to (<=)

   (i) AND operator (&&)

   (j) OR operator (||)

3. Switch Statements The switch expression is evaluated once. The value of the expression is compared with the values of each case. If there is a match, the associated block of code is executed.

```
var age = 20;
switch (age){
  case 16:
    alert("You can’t drive!");
  break;
  case 17:
  case 18:
  case 19:
  case 20:
    alert("You can drive!");
  break;
default:
  alert("You should drive!");
}
```

13 Objects

Objects contains more than one value

1. Example

```
  var dog ={
    "name":"Clifford",
    "legs": 4,
    "goodBoy": TRUE,
  };
```

2. Methods Used to change data in a object

   (a) (.) or ([]) to access properties in an object

   ```
     var dog ={
       "name":"Clifford",
       "legs": 4,
       "goodBoy": TRUE,
   };
     alert(dog.name);
   ```

   (b) “.delete” method to remove the properties

   (c) “hasOwnProperty()” method to check whether property exists

14 ”for” Loop

Loops can execute a block of code a number of times

```
  function counter(){
    var count="";
    var i;
    for (i = 0; i < 5; i++){
    count += "Counting" + i;}
    console.log(count);
  }
  counter();
```

15 Math Function

1. “Math.random()” function to generate a random number between 0 and 1

2. “Math.floor()” function to round down a number to the nearest integer

16 Regular Expression

A regular expression is a sequence of characters that forms a search pattern. The search pattern can be used for text search and text replace operations.

1. Example

```
var text = "this is a text"
var i =/this/gi //search term in between two //
var thisCount = testString.match(i).length;
alert(thisCount);
```

2. “g” expression *(global)* to find all matches

3. “i” expression to ignore case-sensitivity

4. “\d” to retrieve digits

5. “\s” to find whitespace

6. “\S” to find non-whitespace

7. “ \d+” to get multiple results

17 Constructor functions

to add properties in an object

```
var Car = function() {
  //"this" refers to the new object being created by the
      constructor
  this.wheels = 4;
  this.engines = 1;
  this.seats = 1;
};
//"new" keyword to call constructor
// name of contructor is CAPITALIZED
var myCar =  new Car(); {
  this.wheels = 4;
  myCar.nickname = "Vroom";
}
```