

# JS CheatSheet

## Loops ↻

### For Loop

```
for (var i = 0; i < 10; i++) {
  document.write(i + ": " + i*3 + "<br />");
}
var sum = 0;
for (var i = 0; i < a.length; i++) {
  sum += a[i];
} // parsing an array
html = "";
for (var i of custOrder) {
  html += "<li>" + i + "</li>";
}
```

### While Loop

```
var i = 1; // initialize
while (i < 100) { // enters the cycle
  i *= 2; // increment to avo
  document.write(i + ", "); // output
}
```

### Do While Loop

```
var i = 1; // initialize
do { // enters cycle at
  i *= 2; // increment to avo
  document.write(i + ", "); // output
} while (i < 100) // repeats cycle if
```

### Break

```
for (var i = 0; i < 10; i++) {
  if (i == 5) { break; } // stops and ex
  document.write(i + ", "); // last output
}
```

### Continue

```
for (var i = 0; i < 10; i++) {
  if (i == 5) { continue; } // skips the re
  document.write(i + ", "); // skips 5
}
```

## Variables x

```
var a; // variable
var b = "init"; // string
var c = "Hi" + " " + "Joe"; // = "Hi Joe"
var d = 1 + 2 + "3"; // = "33"
var e = [2,3,5,8]; // array
var f = false; // boolean
var g = /()/; // RegEx
var h = function(){}; // function object
const PI = 3.14; // constant
var a = 1, b = 2, c = a + b; // one line
let z = 'zzz'; // block scope loca
```

### Strict mode

```
"use strict"; // Use strict mode to write secure
x = 1; // Throws an error because variable
```

## Basics ▶

### On page script

```
<script type="text/javascript"> ...
</script>
```

### Include external JS file

```
<script src="filename.js"></script>
```

### Delay - 1 second timeout

```
setTimeout(function () {
}, 1000);
```

### Functions

```
function addNumbers(a, b) {
  return a + b; ;
}
x = addNumbers(1, 2);
```

### Edit DOM element

```
document.getElementById("elementID").innerHTML = '
```

### Output

```
console.log(a); // write to the browse
document.write(a); // write to the HTML
alert(a); // output in an alert
confirm("Really?"); // yes/no dialog, retu
prompt("Your age?", "0"); // input dialog. Secor
```

### Comments

```
/* Multi line
   comment */
// One line
```

## If - Else ↓

```
if ((age >= 14) && (age < 19)) { // logical
  status = "Eligible."; // execute
} else { // else bl
  status = "Not eligible."; // execute
}
```

### Switch Statement

```
switch (new Date().getDay()) { // input is cu
  case 6: // if (day ==
    text = "Saturday";
    break;
  case 0: // if (day ==
    text = "Sunday";
    break;
  default: // else...
    text = "Whatever";
}
```

## Data Types ∞

```
var age = 18; // number
var name = "Jane"; // string
```

## Values

```
false, true // boolean
18, 3.14, 0b10011, 0xF6, NaN // number
"flower", 'John' // string
undefined, null, Infinity // special
```

## Operators

```
a = b + c - d; // addition, subtraction
a = b * (c / d); // multiplication, division
x = 100 % 48; // modulo. 100 / 48 remainder =
a++; b--; // postfix increment and decrem
```

## Bitwise operators

&	AND	5 & 1 (0101 & 0001)	1 (1)
	OR	5   1 (0101   0001)	5 (101)
~	NOT	~ 5 (~0101)	10 (1010)
^	XOR	5 ^ 1 (0101 ^ 0001)	4 (100)
<<	left shift	5 << 1 (0101 << 1)	10 (1010)
>>	right shift	5 >> 1 (0101 >> 1)	2 (10)
>>>	zero fill right shift	5 >>> 1 (0101 >>> 1)	2 (10)

## Arithmetic

```
a * (b + c) // grouping
person.age // member
person[age] // member
!(a == b) // logical not
a != b // not equal
typeof a // type (number, object, functi
x << 2 x >> 3 // minary shifting
a = b // assignment
a == b // equals
a != b // unequal
a === b // strict equal
a !== b // strict unequal
a < b a > b // less and greater than
a <= b a >= b // less or equal, greater or eq
a += b // a = a + b (works with - * %.
a && b // logical and
```

## Numbers and Math

```
var pi = 3.141;
pi.toFixed(0); // returns 3
pi.toFixed(2); // returns 3.14 - for worki
pi.toPrecision(2) // returns 3.1
pi.valueOf(); // returns number
Number(true); // converts to number
Number(new Date()) // number of milliseconds s
parseInt("3 months"); // returns the first number
parseFloat("3.5 days"); // returns 3.5
Number.MAX_VALUE // largest possible JS numb
Number.MIN_VALUE // smallest possible JS num
Number.NEGATIVE_INFINITY // -Infinity
Number.POSITIVE_INFINITY // Infinity
```

## Math.

```
var pi = Math.PI; // 3.141592653589793
Math.round(4.4); // = 4 - rounded
Math.round(4.5); // = 5
Math.pow(2,8); // = 256 - 2 to the power o
Math.sqrt(49); // = 7 - square root
Math.abs(-3.14); // = 3.14 - absolute, posit
Math.ceil(3.14); // = 4 - rounded up
Math.floor(3.99); // = 3 - rounded down
Math.sin(0); // = 0 - sine
```

```
var name = {first:"Jane", last:"Doe"}; // object
var truth = false; // boolean
var sheets = ["HTML","CSS","JS"]; // array
var a; typeof a; // undefin
var a = null; // value i
```

## Objects

```
var student = { // object name
  firstName:"Jane", // list of properti
  lastName:"Doe",
  age:18,
  height:170,
  fullName : function() { // object functio
    return this.firstName + " " + this.lastName
  }
};
student.age = 19; // setting value
student[age]++; // incrementing
name = student.fullName(); // call object functio
```

## Strings

```
var abc = "abcdefghijklmnopqrstuvwxy";
var esc = 'I don\'t \n know'; // \n new line
var len = abc.length; // string length
abc.indexOf("lmno"); // find substrings
abc.lastIndexOf("lmno"); // last occurrence
abc.slice(3, 6); // cuts out "def".
abc.replace("abc", "123"); // find and replac
abc.toUpperCase(); // convert to upper
abc.toLowerCase(); // convert to lower
abc.concat(" ", str2); // abc + " " + str
abc.charAt(2); // character at index
abc[2]; // unsafe, abc[2]
abc.charCodeAt(2); // character code
abc.split(","); // splitting a string
abc.split(""); // splitting on character
128.toString(16); // number to hex(:
```

## Events

```
<button onClick="myFunction();">
  Click here
</button>
```

### Mouse

onclick, oncontextmenu, ondblclick, onmousedown, onmouseenter, onmouseleave, onmousemove, onmouseover, onmouseout, onmouseup

### Keyboard

onkeydown, onkeypress, onkeyup

### Frame

onabort, onbeforeunload, onerror, onhashchange, onload, onpageshow, onpagehide, onresize, onscroll, onunload

### Form

onblur, onchange, onfocus, onfocusin, onfocusout, oninput, oninvalid, onreset, onsearch, onselect, onsubmit

### Drag

ondrag, ondragend, ondragenter, ondragleave, ondragover, ondragstart, ondrop

### Clipboard

oncopy, oncut, onpaste

```

Math.cos(Math.PI); // OTHERS: tan,atan,asin,ac
Math.min(0, 3, -2, 2); // = -2 - the lowest value
Math.max(0, 3, -2, 2); // = 3 - the highest value
Math.log(1); // = 0 natural logarithm
Math.exp(1); // = 2.7182pow(E,x)
Math.random(); // random number between 0
Math.floor(Math.random() * 5) + 1; // random integ

```

Constants like Math.PI:

E, PI, SQRT2, SQRT1\_2, LN2, LN10, LOG2E, Log10E

## Dates 31

Mon Feb 17 2020 13:42:03 GMT+0200 (Eastern European Standard Time)

```
var d = new Date();
```

1581939723047 milliseconds passed since 1970

Number(d)

```

Date("2017-06-23"); // date declara
Date("2017"); // is set to Ja
Date("2017-06-23T12:00:00-09:45"); // date - time
Date("June 23 2017"); // long date fo
Date("Jun 23 2017 07:45:00 GMT+0100 (Tokyo Time)");

```

Get Times

```
var d = new Date();
```

```
a = d.getDay(); // getting the weekday
```

```

getDate(); // day as a number (1-31)
getDay(); // weekday as a number (0-6)
getFullYear(); // four digit year (yyyy)
getHours(); // hour (0-23)
getMilliseconds(); // milliseconds (0-999)
getMinutes(); // minutes (0-59)
getMonth(); // month (0-11)
getSeconds(); // seconds (0-59)
getTime(); // milliseconds since 1970

```

Setting part of a date

```
var d = new Date();
```

```
d.setDate(d.getDate() + 7); // adds a week to a dat
```

```

setDate(); // day as a number (1-31)
setFullYear(); // year (optionally month and d
setHours(); // hour (0-23)
setMilliseconds(); // milliseconds (0-999)
setMinutes(); // minutes (0-59)
setMonth(); // month (0-11)
setSeconds(); // seconds (0-59)
setTime(); // milliseconds since 1970)

```

## Global Functions ()

```

eval(); // executes a string as
String(23); // return string from n
(23).toString(); // return string from n
Number("23"); // return number from s
decodeURI(enc); // decode URI. Result:
encodeURI(uri); // encode URI. Result:
decodeURIComponent(enc); // decode a URI compone
encodeURIComponent(uri); // encode a URI compone
isFinite(); // is variable a finite
isNaN(); // is variable an illeg
parseFloat(); // returns floating poi
parseInt(); // parses a string and

```

Media

onabort, oncanplay, oncanplaythrough, ondurationchange, onended, onerror, onloadeddata, onloadedmetadata, onloadstart, onpause, onplay, onplaying, onprogress, onratechange, onseeked, onseeking, onstalled, onsuspend, ontimeupdate, onvolumechange, onwaiting

Animation

animationend, animationiteration, animationstart

Miscellaneous

transitionend, onmessage, onmousewheel, ononline, onoffline, onpopstate, onshow, onstorage, ontoggle, onwheel, ontouchcancel, ontouchend, ontouchmove, ontouchstart

## Arrays ≡

```

var dogs = ["Bulldog", "Beagle", "Labrador"];
var dogs = new Array("Bulldog", "Beagle", "Labrad

```

```

alert(dogs[1]); // access value at ind
dogs[0] = "Bull Terrier"; // change the first it

```

```

for (var i = 0; i < dogs.length; i++) { // pai
    console.log(dogs[i]);
}

```

Methods

```

dogs.toString(); // converti
dogs.join(" * "); // join: '
dogs.pop(); // remove
dogs.push("Chihuahua"); // add ne
dogs[dogs.length] = "Chihuahua"; // the sar
dogs.shift(); // remove
dogs.unshift("Chihuahua"); // add ne
delete dogs[0]; // change
dogs.splice(2, 0, "Pug", "Boxer"); // add ele
var animals = dogs.concat(cats,birds); // join tu
dogs.slice(1,4); // element
dogs.sort(); // sort si
dogs.reverse(); // sort si
x.sort(function(a, b){return a - b}); // numeric
x.sort(function(a, b){return b - a}); // numeric
highest = x[0]; // first :
x.sort(function(a, b){return 0.5 - Math.random()});

```

concat, copyWithin, every, fill, filter, find, findIndex, forEach, indexOf, isArray, join, lastIndexOf, map, pop, push, reduce, reduceRight, reverse, shift, slice, some, sort, splice, toString, unshift, valueOf

## Regular Expressions \n

```
var a = str.search(/CheatSheet/i);
```

Modifiers

<b>i</b>	perform case-insensitive matching
<b>g</b>	perform a global match
<b>m</b>	perform multiline matching

Patterns

<b>\</b>	Escape character
<b>\d</b>	find a digit
<b>\s</b>	find a whitespace character
<b>\b</b>	find match at beginning or end of a word

## Errors

```
try { // block of code to
  undefinedFunction();
}
catch(err) { // block to handle
  console.log(err.message);
}
```

### Throw error

```
throw "My error message"; // throw a text
```

### Input validation

```
var x = document.getElementById("mynum").value; //
try {
  if(x == "") throw "empty"; //
  if(isNaN(x)) throw "not a number";
  x = Number(x);
  if(x > 10) throw "too high";
}
catch(err) { //
  document.write("Input is " + err); //
  console.error(err); //
}
finally { //
  document.write("</br />Done"); //
}
```

### Error name values

<b>RangeError</b>	<i>A number is "out of range"</i>
<b>ReferenceError</b>	<i>An illegal reference has occurred</i>
<b>SyntaxError</b>	<i>A syntax error has occurred</i>
<b>TypeError</b>	<i>A type error has occurred</i>
<b>URIError</b>	<i>An encodeURI() error has occurred</i>

## Useful Links

<a href="#">JS cleaner</a>	<a href="#">Obfuscator</a>
<a href="#">Can I use?</a>	<a href="#">Node.js</a>
<a href="#">jQuery</a>	<a href="#">RegEx tester</a>

n+ *contains at least one n*  
n\* *contains zero or more occurrences of n*  
n? *contains zero or one occurrences of n*  
^ *Start of string*

## JSON j

```
var str = '{"names":[" + // cr
'{"first":"Hakuna","lastN":"Matata" },' +
'{"first":"Jane","lastN":"Doe" },' +
'{"first":"Air","last":"Jordan" }]}';
obj = JSON.parse(str); // pa
document.write(obj.names[1].first); // ac
```

### Send

```
var myObj = { "name":"Jane", "age":18, "city":"Ch
var myJSON = JSON.stringify(myObj);
window.location = "demo.php?x=" + myJSON;
```

### Storing and retrieving

```
myObj = { "name":"Jane", "age":18, "city":"Chicag
myJSON = JSON.stringify(myObj); //
localStorage.setItem("testJSON", myJSON);
text = localStorage.getItem("testJSON"); //
obj = JSON.parse(text);
document.write(obj.name);
```

## Promises p

```
function sum (a, b) {
  return Promise(function (resolve, reject) {
    setTimeout(function () {
      if (typeof a !== "number" || typeof b !== '
        return reject(new TypeError("Inputs must
      }
      resolve(a + b);
    }, 1000);
  });
}
var myPromise = sum(10, 5);
myPromise.then(function (result) {
  document.write(" 10 + 5: ", result);
  return sum(null, "foo"); // Invalid
}).then(function () { // Won't l
}).catch(function (err) { // The ca
  console.error(err); // => Ple:
});
```

### States

pending, fulfilled, rejected

### Properties

Promise.length, Promise.prototype

### Methods

Promise.all(iterable), Promise.race(iterable),  
Promise.reject(reason), Promise.resolve(value)