JS CheatSheet

If - Else ⇅

```
if ((age >= 14) && (age < 19)) {        // logical condition
status = "Eligible.";                 // executed if condition is t
} else {                                // else block is optional
status = "Not eligible.";             // executed if condition is t
}
```

Switch Statement

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

Ads

Dates 31

Thu Aug 31 2023 11:10:30 GMT+0700 (GMT+07:00)

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

1693455030616 miliseconds passed since 1970

```
Number(d)
```

```
Date("2017-06-23");                 // date declaration
Date("2017");                       // is set to Jan 01
Date("2017-06-23T12:00:00-09:45");  // date - time YYYY-MM-DDTHH:
Date("June 23 2017");               // long date format
Date("Jun 23 2017 07:45:00 GMT+0100 (Tokyo Time)"); // time zone
```

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 date

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

Ads

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 = "Hello World!";
```

Output

```
console.log(a);             // write to the browser console
document.write(a);          // write to the HTML
alert(a);                   // output in an alert box
confirm("Really?");         // yes/no dialog, returns true/false
prompt("Your age?","0");    // input dialog. Second argument is t
```

Comments

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

Data Types ℜ

```
var age = 18;                           // number
var name = "Jane";                      // string
var name = {first:"Jane", last:"Doe"};  // object
var truth = false;                      // boolean
var sheets = ["HTML","CSS","JS"];       // array
var a; typeof a;                        // undefined
var a = null;                           // value null
```

Objects

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

Numbers and Math ∑

```
var pi = 3.141;
pi.toFixed(0);          // returns 3
pi.toFixed(2);          // returns 3.14 - for working with money
pi.toPrecision(2)       // returns 3.1
pi.valueOf();           // returns number
Number(true);           // converts to number
Number(new Date())      // number of milliseconds since 1970
parseInt("3 months");   // returns the first number: 3
parseFloat("3.5 days"); // returns 3.5
Number.MAX_VALUE        // largest possible JS number
Number.MIN_VALUE        // smallest possible JS number
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 of 8
Math.sqrt(49);          // = 7 - square root
Math.abs(-3.14);        // = 3.14 - absolute, positive value
Math.ceil(3.14);        // = 4 - rounded up
Math.floor(3.99);       // = 3 - rounded down
Math.sin(0);            // = 0 - sine
Math.cos(Math.PI);      // OTHERS: tan,atan,asin,acos,
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 and 1
Math.floor(Math.random() * 5) + 1;  // random integer, from 1 to
```

Constants like Math.PI:

E, PI, SQRT2, SQRT1_2, LN2, LN10, LOG2E, Log10E

Global Functions ()

```
eval();                     // executes a string as if it was scr
String(23);                 // return string from number
(23).toString();            // return string from number
Number("23");               // return number from string
decodeURI(enc);             // decode URI. Result: "my page.asp"
encodeURI(uri);             // encode URI. Result: "my%page.asp"
decodeURIComponent(enc);    // decode a URI component
encodeURIComponent(uri);    // encode a URI component
isFinite();                 // is variable a finite, legal number
isNaN();                    // is variable an illegal number
parseFloat();               // returns floating point number of s
parseInt();                 // parses a string and returns an int
```

JSON j

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

Send

```
var myObj = { "name":"Jane", "age":18, "city":"Chicago" };  // cr
var myJSON = JSON.stringify(myObj);                         // st
window.location = "demo.php?x=" + myJSON;                   // se
```

Storing and retrieving

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

Useful Links ↲

JS cleaner | Obfuscator | Can I use?
Node.js | jQuery | RegEx tester

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 if statement
i *= 2;                     // increment to avoid infinite loop
document.write(i + ", ");   // output
}
```

Do While Loop

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

Break

```
for (var i = 0; i < 10; i++) {
if (i == 5) { break; }          // stops and exits the cycle
document.write(i + ", ");       // last output number is 4
}
```

Continue

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

Strings ⊗

```
var abc = "abcdefghijklmnopqrstuvwxyz";
var esc = 'I don\'t \n know';   // \n new line
var len = abc.length;           // string length
abc.indexOf("lmno");            // find substring, -1 if doesn't
abc.lastIndexOf("lmno");        // last occurance
abc.slice(3, 6);                // cuts out "def", negative value
abc.replace("abc","123");       // find and replace, takes regula
abc.toUpperCase();              // convert to upper case
abc.toLowerCase();              // convert to lower case
abc.concat(" ", str2);          // abc + " " + str2
abc.charAt(2);                  // character at index: "c"
abc[2];                         // unsafe, abc[2] = "C" doesn't w
abc.charCodeAt(2);              // character code at index: "c" -
abc.split(",");                 // splitting a string on commas g
abc.split("");                  // splitting on characters
128.toString(16);               // number to hex(16), octal (8) o
```

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

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

Regular Expressions \n

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

Modifiers

i	*perform case-insensitive matching*
g	*perform a global match*
m	*perform multiline matching*

Patterns

\	*Escape character*
\d	*find a digit*
\s	*find a whitespace character*
\b	*find match at beginning or end of a word*
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*
$	*End of string*
\uxxxx	*find the Unicode character*
	Any single character

Promises Þ

```
function sum (a, b) {
return Promise(function (resolve, reject) {
 setTimeout(function () {                                          /
   if (typeof a !== "number" || typeof b !== "number") {           /
       return reject(new TypeError("Inputs must be numbers"));
   }
   resolve(a + b);
 }, 1000);
});
}
var myPromise = sum(10, 5);
myPromsise.then(function (result) {
document.write(" 10 + 5: ", result);
return sum(null, "foo");              // Invalid data and return
}).then(function () {                  // Won't be called becaus
}).catch(function (err) {              // The catch handler is c
console.error(err);                   // => Please provide two nu
});
```

States

pending, fulfilled, rejected

Properties

Promise.length, Promise.prototype

Methods

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

Ads

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 local variable
```

Strict mode

```
"use strict";   // Use strict mode to write secure code
x = 1;          // Throws an error because variable is not declar
```

Values

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

Operators

```
a = b + c - d;      // addition, substraction
a = b * (c / d);    // multiplication, division
x = 100 % 48;       // modulo. 100 / 48 remainder = 4
a++; b--;           // postfix increment and decrement
```

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, function...)
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
a || b              // logical or
```

Arrays ≡

```
var dogs = ["Bulldog", "Beagle", "Labrador"];
var dogs = new Array("Bulldog", "Beagle", "Labrador");  // declar

alert(dogs[1]);             // access value at index, first item
dogs[0] = "Bull Terier";    // change the first item

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

Methods

```
dogs.toString();                        // convert to string: res
dogs.join(" * ");                       // join: "Bulldog * Beagl
dogs.pop();                             // remove last element
dogs.push("Chihuahua");                 // add new element to the
dogs[dogs.length] = "Chihuahua";        // the same as push
dogs.shift();                           // remove first element
dogs.unshift("Chihuahua");              // add new element to the
delete dogs[0];                         // change element to unde
dogs.splice(2, 0, "Pug", "Boxer");      // add elements (where, h
var animals = dogs.concat(cats,birds);  // join two arrays (dogs
dogs.slice(1,4);                        // elements from [1] to [
dogs.sort();                            // sort string alphabetic
dogs.reverse();                         // sort string in descend
x.sort(function(a, b){return a - b});   // numeric sort
x.sort(function(a, b){return b - a});   // numeric descending sor
highest = x[0];                         // first item in sorted a
x.sort(function(a, b){return 0.5 - Math.random()});     // 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

Errors ⚠

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

Throw error

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

Input validation

```
var x = document.getElementById("mynum").value; // get input valu
try {
if(x == "")  throw "empty";                 // error cases
if(isNaN(x)) throw "not a number";
x = Number(x);
if(x > 10)   throw "too high";
}
catch(err) {                                    // if there's an
document.write("Input is " + err);          // output error
console.error(err);                         // write the error in
}
finally {
document.write("</br />Done");              // executed regardles
}
```

Error name values

RangeError	*A number is "out of range"*
ReferenceError	*An illegal reference has occurred*
SyntaxError	*A syntax error has occurred*
TypeError	*A type error has occurred*
URIError	*An encodeURI() error has occurred*

Online Interactive JavaScript (JS) Cheat Sheet

JavaScript Cheat Seet contains useful code examples on a single page. This is not just a PDF page becase it's interactive! Find code for JS loops, variables, objects, data types, strings, events and many other categories. Copy-paste the code you need or just quickly check the JS syntax for your projects.

Choose to display or hide the comments, clicking the command in the top right corner.

- **Basics** – Introduction to JavaScript syntax. Learn how to include the scripts on a HTML page, how to declare a function, target a DOM element by it ID, how to output the data and how to write comments.
- **Loops** – Most programming languages allow to work with loops, which help in executing one or more statements up to a desired number of times. Find the "for" and "while" loop syntax in this section.
- **If - Else statements** – Conditional statements are used to perform different actions based on different conditions.
- **Variables** – Use variables (numbers, strings, arrays etc.) and learn the operators.
- **Data types** – You can declare many types of variables and declare your own objects in JavaScript.
- **Strings** – Learn how to work with JS strings and find the most common functions to work with this data type.
- **Events** – Use JavaScript event listeners to trigger functions.
- **Numbers and math** – Work with JS numbers, predefined constants and perform math functions.
- **Dates** – Get or modify current time and date.
- **Arrays** – Learn how to organize your variables in vectors and how to use them.
- **Global functions** – Predefined functions that are built in every browser that supports JS.
- **Regular expressions** – Use RegEx to define a search pattern.
- **Errors** – JS error handling.
- **JSON** – JavaScript Object Notation is syntax used for storing and exchanging data.
- **Promises** – The Promise object is used for asynchronous computation. See our example on how to declare one.

Bookmark this JavaScript cheat sheet with Ctrl + D!