

---

Beginner's Essential

# Javascript Cheat Sheet

The language of the web.

# Table of Contents

Javascript Basics	2
Variables	2
Arrays	3
Operators	4
Functions	5
Loops	7
If - Else Statements	7
Strings	7
Regular Expressions	9
Numbers and Math	10
Dealing with Dates	12
DOM Node	14
Working with the Browser	18
Events	21
Errors	27

# Javascript Basics

## Including JavaScript in an HTML Page

```
<script type="text/javascript">  
  //JS code goes here  
</script>
```

## Call an External JavaScript File

```
<script src="myscript.js"></script><code></code>
```

## Including Comments

```
//  
Single line comments
```

```
/* comment here */  
Multi-line comments
```

# Variables

## var, const, let

**var**

The most common variable. Can be reassigned but only accessed within a function. Variables defined with var move to the top when code is executed.

**const**

Cannot be reassigned and not accessible before they appear within the code.

**let**

Similar to const, however, let variable can be reassigned but not re-declared.

## Data Types

```
var age = 23
```

Numbers

```
var x
```

Variables

```
var a = "init"
```

Text (strings)

```
var b = 1 + 2 + 3
```

Operations

```
var c = true
```

True or false statements

```
const PI = 3.14
```

Constant numbers

```
var name = {firstName:"John", lastName:"Doe"}
```

Objects

## Objects

```
var person = {  
  firstName:"John",  
  lastName:"Doe",  
  age:20,  
  nationality:"German"  
};
```

## Arrays

```
var fruit = ["Banana", "Apple", "Pear"];
```

### Array Methods

```
concat()
```

Join several arrays into one

```
indexOf()
```

Returns the first position at which a given element appears in an array

```
join()
```

Combine elements of an array into a single string and return the string

```
lastIndexOf()
```

Gives the last position at which a given element appears in an array

### **pop()**

Removes the last element of an array

### **push()**

Add a new element at the end

### **reverse()**

Reverse the order of the elements in an array

### **shift()**

Remove the first element of an array

### **slice()**

Pulls a copy of a portion of an array into a new array of 4 24

### **sort()**

Sorts elements alphabetically

### **splice()**

Adds elements in a specified way and position

### **toString()**

Converts elements to strings

### **unshift()**

Adds a new element to the beginning

### **valueOf()**

Returns the primitive value of the specified object

## **Operators**

### **Basic Operators**

- +** Addition
- Subtraction
- \*** Multiplication
- /** Division
- (..)** Grouping operator
- %** Modulus (remainder)
- ++** Increment numbers
- Decrement numbers

## Comparison Operators

```
== Equal to
=== Equal value and equal type
!= Not equal
!== Not equal value or not equal type
> Greater than
< Less than
>= Greater than or equal to
<= Less than or equal to
? Ternary operator
```

## Logical Operators

```
&& Logical and
|| Logical or
! Logical not
```

## Bitwise Operators

```
& AND statement
| OR statement
~ NOT
^ XOR
<< Left shift
>> Right shift
>>> Zero fill right shift
```

# Functions

```
function name(parameter1, parameter2, parameter3) {
  // what the function does
}
```

## Outputting Data

`alert()`

Output data in an alert box in the browser window

`confirm()`

Opens up a yes/no dialog and returns true/false depending on user click

`console.log()`

Writes information to the browser console, good for debugging purposes

**document.write ()**

Write directly to the HTML document

**prompt ()**

Creates an dialogue for user input

## **Global Functions**

**decodeURI ()**

Decodes a Uniform Resource Identifier (URI) created by encodeURI or similar

**decodeURIComponent ()**

Decodes a URI component

**encodeURI ()**

Encodes a URI into UTF-8

**encodeURIComponent ()**

Same but for URI components

**eval ()**

Evaluates JavaScript code represented as a string

**isFinite ()**

Determines whether a passed value is a finite number

**isNaN ()**

Determines whether a value is NaN or not

**Number ()**

Returns a number converted from its argument

**parseFloat ()**

Parses an argument and returns a floating point number

**parseInt ()**

Parses its argument and returns an integer

# Loops

```
for (before loop; condition for loop; execute after loop) {  
  // what to do during the loop  
}  
for
```

The most common way to create a loop in Javascript

## while

Sets up conditions under which a loop executes

## do while

Similar to the while loop, however, it executes at least once and performs a check at the end to see if the condition is met to execute again

## break

Used to stop and exit the cycle at certain conditions

## continue

Skip parts of the cycle if certain conditions are met of 7 24

# If - Else Statements

```
if (condition) {  
  // what to do if condition is met  
} else {  
  // what to do if condition is not met  
}
```

# Strings

```
var person = "John Doe";
```

## Escape Characters

```
\'    - Single quote  
\"    - Double quote  
\\    - Backslash  
\b    - Backspace  
\f    - Form feed  
\n    - New line  
\r    - Carriage return  
\t    - Horizontal tabulator
```



`\v` - Vertical tabulator

## String Methods

`charAt()`

Returns a character at a specified position inside a string

`charCodeAt()`

Gives you the unicode of character at that position

`concat()`

Concatenates (joins) two or more strings into one

`fromCharCode()`

Returns a string created from the specified sequence of UTF-16 code units

`indexOf()`

Provides the position of the first occurrence of a specified text within a string

`lastIndexOf()`

Same as `indexOf()` but with the last occurrence, searching backwards

`match()`

Retrieves the matches of a string against a search pattern

`replace()`

Find and replace specific text in a string

`search()`

Executes a search for a matching text and returns its position

`slice()`

Extracts a section of a string and returns it as a new string

`split()`

Splits a string object into an array of strings at a specified position

`substr()`

Similar to `slice()` but extracts a substring depended on a specified number of characters

`substring()`

Also similar to `slice()` but can't accept negative indices

`toLowerCase()`

Convert strings to lowercase

`toUpperCase()`

Convert strings to uppercase

`valueOf()`

Returns the primitive value (that has no properties or methods) of a string object

## Regular Expressions

### Pattern Modifiers

`e` – Evaluate replacement  
`i` – Perform case-insensitive matching  
`g` – Perform global matching  
`m` – Perform multiple line matching  
`s` – Treat strings as single line  
`x` – Allow comments and whitespace in pattern  
`U` – Non Greedy pattern

### Brackets

`[abc]` Find any of the characters between the brackets  
`[^abc]` Find any character not in the brackets  
`[0-9]` Used to find any digit from 0 to 9  
`[A-z]` Find any character from uppercase A to lowercase z  
`(a|b|c)` Find any of the alternatives separated with |

### Metacharacters

`.` – Find a single character, except newline or line terminator  
`\w` – Word character  
`\W` – Non-word character  
`\d` – A digit  
`\D` – A non-digit character  
`\s` – Whitespace character  
`\S` – Non-whitespace character  
`\b` – Find a match at the beginning/end of a word  
`\B` – A match not at the beginning/end of a word  
`\0` – NUL character  
`\n` – A new line character  
`\f` – Form feed character  
`\r` – Carriage return character  
`\t` – Tab character  
`\v` – Vertical tab character

`\xxx` – The character specified by an octal number `xxx`  
`\xdd` – Character specified by a hexadecimal number `dd`  
`\uxxxx` – The Unicode character specified by a hexadecimal number `xxxx`

## Quantifiers

`n+` – Matches any string that contains at least one `n`  
`n*` – Any string that contains zero or more occurrences of `n`  
`n?` – A string that contains zero or one occurrences of `n`  
`n{X}` – String that contains a sequence of `X` `n`'s  
`n{X,Y}` – Strings that contains a sequence of `X` to `Y` `n`'s  
`n{X,}` – Matches any string that contains a sequence of at least `X` `n`'s  
`n$` – Any string with `n` at the end of it  
`^n` – String with `n` at the beginning of it  
`?=n` – Any string that is followed by a specific string `n`  
`?!n` – String that is not followed by a specific string `n`

# Numbers and Math

## Number Properties

### `MAX_VALUE`

The maximum numeric value representable in JavaScript

### `MIN_VALUE`

Smallest positive numeric value representable in JavaScript

### `NaN`

The “Not-a-Number” value

### `NEGATIVE_INFINITY`

The negative Infinity value

### `POSITIVE_INFINITY`

Positive Infinity value

## Number Methods

### `toExponential()`

Returns a string with a rounded number written as exponential notation

### `toFixed()`

Returns the string of a number with a specified number of decimals

**toFixed()**

String of a number written with a specified length

**toString()**

Returns a number as a string

**valueOf()**

Returns a number as a number

## Math Properties

<b>E</b>	Euler's number
<b>LN2</b>	The natural logarithm of 2
<b>LN10</b>	Natural logarithm of 10
<b>LOG2E</b>	Base 2 logarithm of E
<b>LOG10E</b>	Base 10 logarithm of E
<b>PI</b>	The number PI
<b>SQRT1_2</b>	Square root of 1/2
<b>SQRT2</b>	The square root of 2

## Math Methods

**abs(x)**

Returns the absolute (positive) value of x

**acos(x)**

The arccosine of x, in radians

**asin(x)**

Arcsine of x, in radians

**atan(x)**

The arctangent of x as a numeric value

**atan2(y, x)**

Arctangent of the quotient of its arguments

**ceil(x)**

Value of x rounded up to its nearest integer

**cos(x)**

The cosine of x (x is in radians)

**exp (x)**

Value of  $E^x$

**floor (x)**

The value of x rounded down to its nearest integer

**log (x)**

The natural logarithm (base E) of x

**max (x, y, z, . . . , n)**

Returns the number with the highest value

**min (x, y, z, . . . , n)**

Same for the number with the lowest value

**pow (x, y)**

X to the power of y

**random ()**

Returns a random number between 0 and 1

**round (x)**

The value of x rounded to its nearest integer

**sin (x)**

The sine of x (x is in radians)

**sqrt (x)**

Square root of x

**tan (x)**

The tangent of an angle

## Dealing with Dates

### Setting Dates

**Date ()**

Creates a new date object with the current date and time

**Date (2017, 5, 21, 3, 23, 10, 0)**

Create a custom date object. The numbers represent year, month, day, hour, minutes, seconds, milliseconds. You can omit anything you want except for year and month.

**Date ("2017-06-23")**

Date declaration as a string

## **Pulling Date and Time Values**

**getDate ()**

Get the day of the month as a number (1-31)

**getDay ()**

The weekday as a number (0-6)

**getFullYear ()**

Year as a four digit number (yyyy)

**getHours ()**

Get the hour (0-23)

**getMilliseconds ()**

The millisecond (0-999)

**getMinutes ()**

Get the minute (0-59)

**getMonth ()**

Month as a number (0-11)

**getSeconds ()**

Get the second (0-59)

**getTime ()**

Get the milliseconds since January 1, 1970

**getUTCDate ()**

The day (date) of the month in the specified date according to universal time (also available for day, month, fullyear, hours, minutes etc.)

**parse**

Parses a string representation of a date, and returns the number of milliseconds since January 1, 1970

## Set Part of a Date

### `setDate()`

Set the day as a number (1-31)

### `setFullYear()`

Sets the year (optionally month and day)

### `setHours()`

Set the hour (0-23)

### `setMilliseconds()`

Set milliseconds (0-999)

### `setMinutes()`

Sets the minutes (0-59)

### `setMonth()`

Set the month (0-11)

### `setSeconds()`

Sets the seconds (0-59)

### `setTime()`

Set the time (milliseconds since January 1, 1970)

### `setUTCDate()`

Sets the day of the month for a specified date according to universal time (also available for day, month, fullyear, hours, minutes etc.)

## DOM Node

### Node Properties

#### `attributes`

Returns a live collection of all attributes registered to and element

#### `baseURI`

Provides the absolute base URL of an HTML element

#### `childNodes`

Gives a collection of an element's child nodes

**firstChild**

Returns the first child node of an element

**lastChild**

The last child node of an element

**nextSibling**

Gives you the next node at the same node tree level

**nodeName**

Returns the name of a node

**nodeType**

Returns the type of a node

**nodeValue**

Sets or returns the value of a node

**ownerDocument**

The top-level document object for this node

**parentNode**

Returns the parent node of an element

**previousSibling**

Returns the node immediately preceding the current one

**textContent**

Sets or returns the textual content of a node and its descendants

**Node Methods****appendChild()**

Adds a new child node to an element as the last child node

**cloneNode()**

Clones an HTML element

**compareDocumentPosition()**

Compares the document position of two elements

**getFeature()**

Returns an object which implements the APIs of a specified feature



### **hasAttributes ()**

Returns true if an element has any attributes, otherwise false

### **hasChildNodes ()**

Returns true if an element has any child nodes, otherwise false

### **insertBefore ()**

Inserts a new child node before a specified, existing child node

### **isDefaultNamespace ()**

Returns true if a specified namespaceURI is the default, otherwise false

### **isEqualNode ()**

Checks if two elements are equal

### **isSameNode ()**

Checks if two elements are the same node

### **isSupported ()**

Returns true if a specified feature is supported on the element

### **lookupNamespaceURI ()**

Returns the namespaceURI associated with a given node

### **lookupPrefix ()**

Returns a DOMString containing the prefix for a given namespaceURI, if present

### **normalize ()**

Joins adjacent text nodes and removes empty text nodes in an element

### **removeChild ()**

Removes a child node from an element

### **replaceChild ()**

Replaces a child node in an element

## **Element Methods**

### **getAttribute ()**

Returns the specified attribute value of an element node

### **getAttributeNS ()**

Returns string value of the attribute with the specified namespace and name

**getAttributeNode ()**

Gets the specified attribute node

**getAttributeNodeNS ()**

Returns the attribute node for the attribute with the given namespace and name

**getElementsByTagName ()**

Provides a collection of all child elements with the specified tag name

**getElementsByTagNameNS ()**

Returns a live HTMLCollection of elements with a certain tag name belonging to the given namespace

**hasAttribute ()**

Returns true if an element has any attributes, otherwise false

**hasAttributeNS ()**

Provides a true/false value indicating whether the current element in a given namespace has the specified attribute

**removeAttribute ()**

Removes a specified attribute from an element

**removeAttributeNS ()**

Removes the specified attribute from an element within a certain namespace

**removeAttributeNode ()**

Takes away a specified attribute node and returns the removed node

**setAttribute ()**

Sets or changes the specified attribute to a specified value

**setAttributeNS ()**

Adds a new attribute or changes the value of an attribute with the given namespace and name

**setAttributeNode ()**

Sets or changes the specified attribute node

**setAttributeNodeNS ()**

Adds a new namespaced attribute node to an element

# Working with the Browser

## Window Properties

### `closed`

Checks whether a window has been closed or not and returns true or false

### `defaultStatus`

Sets or returns the default text in the statusbar of a window

### `document`

Returns the document object for the window

### `frames`

Returns all <iframe> elements in the current window

### `history`

Provides the History object for the window

### `innerHeight`

The inner height of a window's content area

### `innerWidth`

The inner width of the content area

### `length`

Find out the number of <iframe> elements in the window

### `location`

Returns the location object for the window

### `name`

Sets or returns the name of a window

### `navigator`

Returns the Navigator object for the window

### `opener`

Returns a reference to the window that created the window

### `outerHeight`

The outer height of a window, including toolbars/ scrollbars

### **outerWidth**

The outer width of a window, including toolbars/ scrollbars

### **pageXOffset**

Number of pixels the current document has been scrolled horizontally

### **pageYOffset**

Number of pixels the document has been scrolled vertically

### **parent**

The parent window of the current window

### **screen**

Returns the Screen object for the window

### **screenLeft**

The horizontal coordinate of the window (relative to screen)

### **screenTop**

The vertical coordinate of the window

### **screenX**

Same as screenLeft but needed for some browsers

### **screenY**

Same as screenTop but needed for some browsers

### **self**

Returns the current window

### **status**

Sets or returns the text in the statusbar of a window

### **top**

Returns the topmost browser window

## **Window Methods**

### **alert()**

Displays an alert box with a message and an OK button

### **blur()**

Removes focus from the current window

**clearInterval ()**

Clears a timer set with setInterval()

**clearTimeout ()**

Clears a timer set with setTimeout()

**close ()**

Closes the current window

**confirm ()**

Displays a dialogue box with a message and an OK and Cancel button

**focus ()**

Sets focus to the current window

**moveBy ()**

Moves a window relative to its current position

**moveTo ()**

Moves a window to a specified position

**open ()**

Opens a new browser window

**print ()**

Prints the content of the current window

**prompt ()**

Displays a dialogue box that prompts the visitor for input

**resizeBy ()**

Resizes the window by the specified number of pixels

**resizeTo ()**

Resizes the window to a specified width and height

**scrollBy ()**

Scrolls the document by a specified number of pixels

**scrollTo ()**

Scrolls the document to specific coordinates

### **setInterval()**

Calls a function or evaluates an expression at specified intervals

### **setTimeout()**

Calls a function or evaluates an expression after a specified interval

### **stop()**

Stops the window from loading

## **Screen Properties**

### **availHeight**

Returns the height of the screen (excluding the Windows Taskbar)

### **availWidth**

Returns the width of the screen (excluding the Windows Taskbar)

### **colorDepth**

Returns the bit depth of the color palette for displaying images

### **height**

The total height of the screen

### **pixelDepth**

The color resolution of the screen in bits per pixel

### **width**

The total width of the screen

## **Events**

### **Mouse**

#### **onclick**

The event occurs when the user clicks on an element

#### **oncontextmenu**

User right-clicks on an element to open a context menu

#### **ondblclick**

The user double-clicks on an element

### **onmousedown**

User presses a mouse button over an element

### **onmouseenter**

The pointer moves onto an element

### **onmouseleave**

Pointer moves out of an element

### **onmousemove**

The pointer is moving while it is over an element

### **onmouseover**

When the pointer is moved onto an element or one of its children

### **onmouseout**

User moves the mouse pointer out of an element or one of its children

### **onmouseup**

The user releases a mouse button while over an element

## **Keyboard**

### **onkeydown**

When the user is pressing a key down

### **onkeypress**

The moment the user starts pressing a key

### **onkeyup**

The user releases a key

## **Frame**

### **onabort**

The loading of a media is aborted

### **onbeforeunload**

Event occurs before the document is about to be unloaded

### **onerror**

An error occurs while loading an external file

### **onhashchange**

There have been changes to the anchor part of a URL

### **onload**

When an object has loaded

### **onpagehide**

The user navigates away from a webpage

### **onpageshow**

When the user navigates to a webpage

### **onresize**

The document view is resized

### **onscroll**

An element's scrollbar is being scrolled

### **onunload**

Event occurs when a page has unloaded

## **Form**

### **onblur**

When an element loses focus

### **onchange**

The content of a form element changes (for <input>, <select>and <textarea>)

### **onfocus**

An element gets focus

### **onfocusin**

When an element is about to get focus

### **onfocusout**

The element is about to lose focus

### **oninput**

User input on an element

### **oninvalid**

An element is invalid



### **onreset**

A form is reset

### **onsearch**

The user writes something in a search field (for <input="search">)

### **onselect**

The user selects some text (for <input> and <textarea>)

### **onsubmit**

A form is submitted

## **Drag**

### **ondrag**

An element is dragged

### **ondragend**

The user has finished dragging the element

### **ondragenter**

The dragged element enters a drop target

### **ondragleave**

A dragged element leaves the drop target

### **ondragover**

The dragged element is on top of the drop target

### **ondragstart**

User starts to drag an element

### **ondrop**

Dragged element is dropped on the drop target

## **Clipboard**

### **oncopy**

User copies the content of an element

### **oncut**

The user cuts an element's content

### **onpaste**

A user pastes content in an element

## **Media**

### **onabort**

Media loading is aborted

### **oncanplay**

The browser can start playing media (e.g. a file has buffered enough)

### **oncanplaythrough**

When browser can play through media without stopping

### **ondurationchange**

The duration of the media changes

### **onended**

The media has reached its end

### **onerror**

Happens when an error occurs while loading an external file

### **onloadeddata**

Media data is loaded

### **onloadedmetadata**

Meta Metadata (like dimensions and duration) are loaded

### **onloadstart**

Browser starts looking for specified media

### **onpause**

Media is paused either by the user or automatically

### **onplay**

The media has been started or is no longer paused

### **onplaying**

Media is playing after having been paused or stopped for buffering

### **onprogress**

Browser is in the process of downloading the media

### **onratechange**

The playing speed of the media changes

### **onseeked**

User is finished moving/skipping to a new position in the media

### **onseeking**

The user starts moving/skipping

### **onstalled**

The browser is trying to load the media but it is not available

### **onsuspend**

Browser is intentionally not loading media

### **ontimeupdate**

The playing position has changed (e.g. because of fast forward)

### **onvolumechange**

Media volume has changed (including mute)

### **onwaiting**

Media paused but expected to resume (for example, buffering)

## **Animation**

### **animationend**

A CSS animation is complete

### **animationiteration**

CSS animation is repeated

### **animationstart**

CSS animation has started

## **Other**

### **transitionend**

Fired when a CSS transition has completed

### **onmessage**

A message is received through the event source

**onoffline**

Browser starts to work offline

**ononline**

The browser starts to work online

**onpopstate**

When the window's history changes

**onshow**

A <menu> element is shown as a context menu

**onstorage**

A Web Storage area is updated

**ontoggle**

The user opens or closes the <details> element

**onwheel**

Mouse wheel rolls up or down over an element

**ontouchcancel**

Screen touch is interrupted

**ontouchend**

User finger is removed from a touch screen

**ontouchmove**

A finger is dragged across the screen

**ontouchstart**

Finger is placed on touch screen

## Errors

**try**

Lets you define a block of code to test for errors

**catch**

Set up a block of code to execute in case of an error

### **throw**

Create custom error messages instead of the standard JavaScript errors

### **finally**

Lets you execute code, after try and catch, regardless of the result

## **Error Name Values**

### **name**

Sets or returns the error name

### **message**

Sets or returns an error message in string form

### **EvalError**

An error has occurred in the eval() function

### **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