# JavaScript Cheat Sheet

Table of Contents

[JavaScript Cheat Sheet 1](#_Toc163491449)

[Array 1](#_Toc163491450)

[Method Calls 1](#_Toc163491451)

[Concatenate 1](#_Toc163491452)

[Copy Within 2](#_Toc163491453)

[Fill 2](#_Toc163491454)

[Include 2](#_Toc163491455)

[Index Of 3](#_Toc163491456)

[Is Array 3](#_Toc163491457)

[Join 3](#_Toc163491458)

[Last Index Of 3](#_Toc163491459)

[Reserse 3](#_Toc163491460)

[Slice 3](#_Toc163491461)

[Splice 3](#_Toc163491462)

[To String 3](#_Toc163491463)

[Value Of 4](#_Toc163491464)

[Data Structures 4](#_Toc163491465)

[Entries 4](#_Toc163491466)

[Keys 4](#_Toc163491467)

[Pop 5](#_Toc163491468)

[Push 5](#_Toc163491469)

[Shift 5](#_Toc163491470)

[Sort 5](#_Toc163491471)

[Unshift 5](#_Toc163491472)

[Functions 6](#_Toc163491473)

[Every 6](#_Toc163491474)

[Filter 6](#_Toc163491475)

[Find 6](#_Toc163491476)

[Find Index 6](#_Toc163491477)

[Map 6](#_Toc163491478)

[Protypes 6](#_Toc163491479)

[Reduce 7](#_Toc163491480)

[Reduce Right 7](#_Toc163491481)

[Some 7](#_Toc163491482)

[String 7](#_Toc163491483)

[Method Calls 7](#_Toc163491484)

[Character At 7](#_Toc163491485)

[Character Code At 8](#_Toc163491486)

[Concatenate 8](#_Toc163491487)

[From Character Code 8](#_Toc163491488)

[Index Of 8](#_Toc163491489)

[Last Index Of 8](#_Toc163491490)

[Length 8](#_Toc163491491)

[Repeat 8](#_Toc163491492)

[Slice 8](#_Toc163491493)

[Split 8](#_Toc163491494)

[Substr 8](#_Toc163491495)

[Sub String 9](#_Toc163491496)

[To Lower Case 9](#_Toc163491497)

[To String 9](#_Toc163491498)

[To Upper Case 9](#_Toc163491499)

[Trim 9](#_Toc163491500)

[Value Of 9](#_Toc163491501)

[Search Calls 9](#_Toc163491502)

[Ends With 9](#_Toc163491503)

[Includes 9](#_Toc163491504)

[Last Index Of 9](#_Toc163491505)

[Locale Compare 9](#_Toc163491506)

[Match 10](#_Toc163491507)

[Replace 10](#_Toc163491508)

[Search 10](#_Toc163491509)

[Starts With 10](#_Toc163491510)

[Regular Expression 10](#_Toc163491511)

[Object 10](#_Toc163491512)

[Object Notation 10](#_Toc163491513)

[Object Constructor 10](#_Toc163491514)

[RegExpr Classes 11](#_Toc163491515)

[RegExpr 11](#_Toc163491516)

[RegExpr Methods 11](#_Toc163491517)

[Exec 11](#_Toc163491518)

[Test 11](#_Toc163491519)

[String Methods 12](#_Toc163491520)

[Match 12](#_Toc163491521)

[Match All 12](#_Toc163491522)

[Replace 12](#_Toc163491523)

[Replace All 12](#_Toc163491524)

[Search 12](#_Toc163491525)

[Modifiers 12](#_Toc163491526)

[Start and End Indices Flag 12](#_Toc163491527)

[Global Flag 12](#_Toc163491528)

[Case-insensitive Search Flag 13](#_Toc163491529)

[Multi-line Search Flag 13](#_Toc163491530)

[Match Newline Characters Flag 13](#_Toc163491531)

[Unicode Flag 13](#_Toc163491532)

[Sticky Flag 13](#_Toc163491533)

[Groups and Ranges 13](#_Toc163491534)

[Matches Either 13](#_Toc163491535)

[Character Class 13](#_Toc163491536)

[Negated Character Class 14](#_Toc163491537)

[Capturing Group 14](#_Toc163491538)

[Capturing Group Number 14](#_Toc163491539)

[Meta Characters 14](#_Toc163491540)

[Single Character 14](#_Toc163491541)

[Word Character 14](#_Toc163491542)

[Non-word Character 14](#_Toc163491543)

[Digit Character 15](#_Toc163491544)

[Non-digit Character 15](#_Toc163491545)

[Whitespace Character 15](#_Toc163491546)

[Non-whitespace Character 15](#_Toc163491547)

[Beginning/End of a Word 15](#_Toc163491548)

[Not Beginning/End of a Word 15](#_Toc163491549)

[Null Character 15](#_Toc163491550)

[New Line Character 15](#_Toc163491551)

[Form Feed Character 15](#_Toc163491552)

[Carriage Return Character 15](#_Toc163491553)

[Quantifiers 16](#_Toc163491554)

[At Least One 16](#_Toc163491555)

[Zero or More 16](#_Toc163491556)

[Zero or One 16](#_Toc163491557)

[Sequence of X n's 16](#_Toc163491558)

[Sequence of X to Y n's 16](#_Toc163491559)

[Sequence of at least X n's 16](#_Toc163491560)

[End 16](#_Toc163491561)

[Beginning 17](#_Toc163491562)

[Followed by a specific string n 17](#_Toc163491563)

[Not followed by a specific string n 17](#_Toc163491564)

# Array

## Method Calls

### Concatenate

This method is used to merge two or more arrays. This method does not change the existing arrays, but instead returns a new array.

Code Example:

A screenshot of a computer code

Description automatically generated

Results:



Code Example:

A screenshot of a computer code

Description automatically generated

Results:



### Copy Within

Copies array elements to another position in an array, overwriting the existing values.

Code Example:

A close-up of a website

Description automatically generated

Results:



### Fill

Fill all the array elements with a static value.

### Include

Check if an array includes certain value. Return boolean.

Code Example:



Results:



### Index Of

Search an array for the item and return the index of that value.

### Is Array

Check whether an object is an array.

Code Example:

A close up of words

Description automatically generated

Results:

### Join

Convert the elements of an array into a string.

### Last Index Of

Search an array for the item. If there are many times certain item is in the array will return last index.

### Reserse

Reverse the order of the elements in an array.

### Slice

The slice() method returns a shallow copy of a portion of an array into a new array object selected from start to end (end not included) where start and end represent the index of items in that array. The original array will not be modified.

### Splice

The splice() method changes the contents of an array by removing or replacing existing elements and/or adding new elements.

### To String

 Convert an array to a string.

### Value Of

The valueOf() method returns the primitive value of the specified object.

## Data Structures

### Entries

Array Iterator object that contains the key/value pairs for each index in the array.

Code Example:

A screenshot of a computer code

Description automatically generated

Results:

A close-up of a number

Description automatically generated

### Keys

Create an Array Iterator object, containing the keys of the array.

Code Example:

A screenshot of a computer code

Description automatically generated

Results:

Blue text on a white background

Description automatically generated

### Pop

Remove the last element of an array.

### Push

 Adds the specified elements to the end of an array and returns the new length of the array.

### Shift

 Remove the first item of an array.

### Sort

The sort() method sorts the elements of an array.

### Unshift

Adds one or more elements to the beginning of an array and returns the new length of the array.

## Functions

### Every

The every() method tests whether all elements in the array pass the test implemented by the provided function. It returns a Boolean value.

### Filter

The filter() method creates a new array with all elements that pass the test implemented by the provided function.

### Find

Get the value of the first element in the array that meets the criteria.

### Find Index

Get the index of the first element in the array that has a value of the criteria.

### Map

The Map object holds key-value pairs and remembers the original insertion order of the keys.

Code Example:

A screenshot of a computer code

Description automatically generated

Results:



### Protypes

Create a method on the array.

### Reduce

The reduce() method executes a user-supplied “reducer” callback function on each element of the array, passing in the return value from the calculation on the preceding element.

Code Example:

A screenshot of a computer code

Description automatically generated

Results:



### Reduce Right

The reduceRight() method applies a function against an accumulator and each value of the array (from right-to-left) to reduce it to a single value.

### Some

 The some() method tests whether at least one element in the array passes the test implemented by the provided function. It returns true if, in the array, it finds an element for which the provided function returns true; otherwise it returns false. It doesn't modify the array.

# String

## Method Calls

### Character At

The String object's charAt() method returns a new string consisting of the single UTF-16 code unit located at the specified offset into the string.

### Character Code At

The charCodeAt() method returns an integer between 0 and 65535 representing the UTF-16 code unit at the given index. Return the Unicode of the first character in a string (the Unicode value for "H").

### Concatenate

The concat() function takes one or more parameters, and returns the modified string. Strings in JavaScript are immutable, so concat() doesn't modify the string in place.

### From Character Code

Convert a Unicode number into a character.

### Index Of

Search for the first occurrence of a certain String. Return the index of the position of the first character in the Character array.

### Last Index Of

Search a string for the last occurrence of a certain String. Return the index of the position of the first character in the Character array.

### Length

Return the number of characters in a string.

### Repeat

Make a new string by copying the original string number of times.

### Slice

The slice() method returns a shallow copy of a portion of an array into a new array object selected from start to end (end not included) where start and end represent the index of items in that array. The original array will not be modified.

### Split

The split() method divides a String into an ordered list of substrings, puts these substrings into an array, and returns the array. The division is done by searching for a pattern; where the pattern is provided as the first parameter in the method's call.

### Substr

The substr() method returns a portion of the string, starting at the specified index and extending for a given number of characters afterwards.

### Sub String

The substr() method returns a portion of the string, starting at the specified index and extending for a given number of characters afterwards.

### To Lower Case

Convert the string to lowercase letters.

### To String

Return the value of a string.

### To Upper Case

Convert the string to uppercase letters.

### Trim

Remove whitespace from both sides of a string.

### Value Of

Return the primitive value of a string object.

## Search Calls

### Ends With

Check if a string ends with.

### Includes

Check if a string includes a certain string.

### Last Index Of

Search a string for the last occurrence of a certain String. Return the index of the position of the first character in the Character array.

### Locale Compare

A Number indicates whether the reference string comes before, after, or is the same as the compare string in sort order. Returns one of three values: -1 if the reference string is sorted before the compare string, 0 if the two strings are equal, and 1 if the reference string is sorted after the compare string

### Match

Search for the same pattern in a string using Regular Expressions.

### Replace

The replace() method returns a new string with some or all matches of a pattern replaced by a replacement. The pattern can be a string or a RegExp, and the replacement can be a string or a function to be called for each match. If pattern is a string, only the first occurrence will be replaced.

### Search

The search() method executes a search for a match between a regular expression and this String object. Return the index position of the search string.

### Starts With

The startsWith() method determines whether a string begins with the characters of a specified string, returning true or false as appropriate.

# Regular Expression

## Object

### Object Notation

There are two ways to create a RegExp object: a literal notation and a constructor.

/cat/g

A screen shot of a computer

Description automatically generated

### Object Constructor

There are two ways to create a RegExp object: a literal notation and a constructor.

new RegExp("cat", "gi")

A screen shot of a computer

Description automatically generated

## RegExpr Classes

### RegExpr

The exec and test methods are similar to the match method. The only difference is that the match method returns all matches in an array, while with exec and test methods, we need a looping structure.

lastIndex is a read/write property of RegExp objects. For regular expressions with the "g" attribute set, it contains an integer that specifies the character position immediately following the last match found by the RegExp.exec() and RegExp.test() methods. These methods use this property as the starting point for their next search.

This property allows you to call those methods repeatedly, to loop through all matches in a string, and works only if the "g" modifier is set.

This property is read/write, so you can set it at any time to specify where the following search should begin in the target string. exec() and test() automatically reset the lastIndex to 0 when they fail to find a match (or another match).

## RegExpr Methods

### Exec

The exec() method executes a search for a match in a specified string. Returns a result array (match\_value,index,inpute\_string,group\_name), or null.

### Test

The test() method executes a search for a match between a regular expression and a specified string. Return boolean.

## String Methods

### Match

The match() method retrieves the result of matching a string of the match in an array.

### Match All

The matchAll() method returns an iterator of all results matching a string against a regular expression, including capturing groups.

### Replace

The replace() method returns a new string with first of a pattern replaced by a replacement. Replace first occurrence.

### Replace All

 The replaceAll() method returns a new string with all matches of a pattern replaced by a replacement.

### Search

The search() method executes a search for a match between a regular expression and this String object. Return index first occurrence.

## Modifiers

### Start and End Indices Flag

**d Flag** - The "d" flag indicates that the result of a regular expression match should contain the **start and end indices** of the substrings of each capture group. **Corresponding property** RegExp.prototype.hasIndices

### Global Flag

**g Flag** - The "g" flag indicates that the regular expression should be tested **against all possible matches** in a string. A regular expression defined as both global ("g") and sticky ("y") will ignore the global flag and perform sticky matches. **Corresponding property** RegExp.prototype.hasIndices

### Case-insensitive Search Flag

**i Flag** - Do a **case-insensitive** search. **Corresponding property** RegExp.prototype.ignoreCase

### Multi-line Search Flag

**m Flag** - Multi-line search. **Corresponding property** RegExp.prototype.multiline

### Match Newline Characters Flag

**s Flag** - Allows to match newline characters. **Corresponding property** RegExp.prototype.dotAll

### Unicode Flag

**u Flag** - "unicode"; treat a pattern as a sequence of unicode code points. **Corresponding property** RegExp.prototype.unicode

### Sticky Flag

**y Flag** - Perform a "sticky" search that matches starting at the current position in the target string. **Corresponding property** RegExp.prototype.sticky

## Groups and Ranges

### Matches Either

**(x|y)** - Matches either "x" or "y". For example, /green|red/ matches "green" in "green apple" and "red" in "red apple".

### Character Class

**[xyz] or [a-c]** - A character class. Matches any one of the enclosed characters. You can specify a range of characters by using a hyphen, but if the hyphen appears as the first or last character enclosed in the square brackets it is taken as a literal hyphen to be included in the character class as a normal character. For example, [abcd] is the same as [a-d]. They match the "b" in "brisket", and the "c" in "chop". For example, [abcd-] and [-abcd] match the "b" in "brisket", the "c" in "chop", and the "-" (hyphen) in "non-profit".

### Negated Character Class

**[^xyz] or [^a-c]** - A negated or complemented character class. That is, it matches anything that is not enclosed in the brackets. You can specify a range of characters by using a hyphen, but if the hyphen appears as the first or last character enclosed in the square brackets it is taken as a literal hyphen to be included in the character class as a normal character.  
For example, [^abc] is the same as [^a-c]. They initially match "o" in "bacon" and "h" in "chop".

### Capturing Group

**(x)** - Capturing group: Matches x and remembers the match. For example, /(foo)/ matches and remembers "foo" in "foo bar".

### Capturing Group Number

**\n** -Where "n" is a positive integer. \1 refers to the first capturing group in the regular expression. \2 will refer to the second capturing group and \n will refer to an nth capturing group.  
Where "n" is a positive integer. A back reference to the last substring matching the n parenthetical in the regular expression (counting left parentheses). For example, /apple(,)\sorange\1/ matches "apple, orange," in "apple, orange, cherry, peach".

## Meta Characters

### Single Character

**.** - Find a single character, except newline or line terminator.

### Word Character

**\w** - Find a word character. A word character is a character a-z, A-Z, 0-9, including \_ (underscore).

### Non-word Character

**\W** - Find a non-word character.

### Digit Character

**\d** - Find a digit character.

### Non-digit Character

**\D** - Find a non-digit character.

### Whitespace Character

**\s** - Find a whitespace character.

### Non-whitespace Character

**\S** - Find a non-whitespace character.

### Beginning/End of a Word

**\b** - Find a match at the beginning/end of a word, beginning like this: \bHI, end like this: HI\b.

### Not Beginning/End of a Word

**\B** - Find a match, but not at the beginning/end of a word.

### Null Character

**\0** - Find a null character.

### New Line Character

**\n** - Find a new line character.

### Form Feed Character

**\f** - Find a form feed character.

### Carriage Return Character

**\r** - Find a carriage return character.

Tab Character

**\t** - Find a tab character.

Vertical Tab Character

**\v** - Find a vertical tab character.

Octal Number

**\xxx** - Find the character specified by an octal number xxx.

Hexadecimal Number

**\xdd** - Find the character specified by a hexadecimal number dd.

Unicode Character

**\udddd** - Find the Unicode character specified by a hexadecimal number dddd.

## Quantifiers

### At Least One

**n+** - Matches any string that contains **at least one** n.

### Zero or More

**n**\* - Matches any string that contains **zero or more** occurrences of n.

### Zero or One

**n?** - Matches any string that contains **zero or one** occurrences of n.

### Sequence of X n's

**n{X}** - Matches any string that contains **a sequence of X n's**.

### Sequence of X to Y n's

**n{X,Y}** - Matches any string that contains **a sequence of X to Y n's**.

### Sequence of at least X n's

**n{X,}** - Matches any string that contains **a sequence of at least X n's**.

### End

**n$** - Matches any string with n at the **end** of it.

### Beginning

**^n** - Matches any string with n at the **beginning** of it.

### Followed by a specific string n

**?=n** - Matches any string that is **followed by a specific** **string n**.

### Not followed by a specific string n

**?!n** - Matches any string that is **not** **followed by a specific** **string n**.