

Variable types

	var	let	const
block scoped*	✗	✓	✓
tdz**	✗	✓	✓
creates global property*	✓	✗	✗
reassignable*	✓	✓	✗
redeclarable*	✓	✗	✗

https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Strict_mode
 * <https://medium.com/@allansendagi/block-scope-in-javascript-8fd2f909e848>
 ** <https://discuss.codecademy.com/t/what-is-the-difference-between-let-and-var-when-declaring-and-what-about-reassigning/492581/2>

<https://www.valentinog.com/blog/var/>
<https://www.freecodecamp.org/news/var-let-and-const-whats-the-difference/>
<https://fontawesome.com/v3/icons/>

Data types

var age = 18;	// number
var name = "Jane";	// string
var truth = false;	// boolean
var sheets = ["HTML", "CSS", "JS"];	// array
var a; typeof a;	// undefined
var a = null;	// value null

Objects:

```
var student = {
  firstName:"Jane",
  lastName:"Doe",
  age:18,
  height:170,
  fullName : function() {
    return this.firstName + " " +
    this.lastName;
  }
};
```

Data types (cont)

```
// list of properties and values
// object function
// object end
student.age = 19; // setting value
student[age]++; // incrementing
name = student.fullName(); // call object function
```

Operators

Comparison Operators

Comparison operators **compare** two values and return a boolean value, either `true` or `false`.

==	Equal to	5==5; //true
!=	Not equal to	5!=5; //false
>	Greater than	3>2; //true
>=	Greater than or equal to	3>=3; //true
<	Less than	3<2; //false
<=	Less than or equal to	2<=2; //true
===	Strict equal to	5=== '5'; //false
!==	Strict not equal to	5!== '5'; //true

Logical Operators

Logical operators perform logical operations and return a boolean value, either `true` or `false`.

&&	Logical AND	true && false; // false
	Logical OR	true false; // true
!	Logical NOT	!true; // false

Operators (cont)

? : Ternary operator

Returns value based on the condition

```
(5 > 3) ? 'success' : 'error'; //
condition ? expres sio nIfTrue : e
False
```

Assignment operators

Assignment operators are used to assign values

=	Assignment operator	a = 7;
+=	Addition assignment	a += 5;
*=	Multiplication Assignment	a = 3;
-=	Subtraction Assignment	a -= 2;
/=	Division Assignment	a /= 2;
%=	Remainder Assignment	a %= 2;
**=	Exponentiation Assignment	a = 2;

Arithmetic Operators

Arithmetic operators are used to perform arithmetic operations

+	Addition	x + y
-	Subtraction	x - y
*	Multiplication	x * y
/	Division	x / y
%	Remainder	x % y



Operators (cont)

++	Increment (increments by 1)	<code>++x</code> or <code>x++</code>
--	Decrement (decrements by 1)	<code>--x</code> or <code>x--</code>
**	Exponentiation (Power)	<code>x ** y</code>

Bitwise Operators

Bitwise operators perform operations on binary representations of numbers.

& Bitwise AND

| Bitwise OR

^ Bitwise XOR

~ Bitwise NOT

<< Left shift

>> Sign-propagating right shift

>>> Zero-fill right shift

String Operators

In JavaScript, you can also use the `+` operator to concatenate (join) two or more strings.

+ Concatenation operator

Other Operators

Operators (cont)

, evaluates each of its operands (from left to right) and returns the value of the last operand.

```
let x = 1;
x = (x++, x);
console.log(x); // 2
```

delete `delete x`

deletes an object's property, or an element of an array

typeof `typeof 3; // "number"`

returns a string indicating the data type

void `void(x)`

discards the expression's return value

in `prop in object`

returns true if the specified property is in the object

Operators (cont)

instanceof `object instanceof object`
returns `true` if the specified object is of the specified object type

Loopz

https://www.w3schools.com/js/js_loop_for.asp <https://www.programiz.com/javascript/for-loop>

https://www.w3schools.com/jsref/jsref_foreach.asp <https://www.programiz.com/javascript/forEach>

https://www.w3schools.com/js/js_loop_forof.asp <https://www.programiz.com/javascript/for-of>

https://www.w3schools.com/js/js_loop_forin.asp <https://www.programiz.com/javascript/for-in>

https://www.w3schools.com/js/js_loop_while.asp <https://www.programiz.com/javascript/while-loop>

https://www.w3schools.com/js/js_break.asp <https://www.programiz.com/javascript/break-statement>

<https://www.programiz.com/javascript/continue-statement>

https://www.w3schools.com/js/js_iterables.asp

https://www.w3schools.com/js/js_switch.asp <https://www.programiz.com/javascript/switch-statement>



By [krabat1 \(krabat1\)](#)
cheatography.com/krabat1/

Not published yet.
Last updated 5th January, 2024.
Page 2 of 4.

Sponsored by [Readable.com](#)
Measure your website readability!
<https://readable.com>

Global Methods

Math functions III. - Number Methods

`isFinite()` a value is a finite number?

`isInteger()` a value is an integer?

`isNaN()` a value is Number.NaN?

`isSafeInteger()` a value is a safe integer?

`toExponential(x)` Converts a num. into an exponential notation

`toFixed(x)` Formats a num. with x numbs of digits after the decimal point

`toLocaleString()` Converts a num. into a string, based on the locale settings

`toPrecision(x)` Formats a num. to x length

`toString()` Converts a num. to a string

`valueOf()` Returns the primitive value of a num.

Math functions I.

`Math.round(x.y);` `Math.trunc(x.y)`
`// x.5 ^ x.4` `// x`

`Math.ceil(); Math.floor();` `Math.pow(x,y)`
`// ^` `// xy`

`Math.sign()` `Math.abs(-x);`
`// -x, 0, x → -1, 0, 1` `// x`

`Math.sqrt(x); Math.cbrt(y);` `Math.exp(x)`
`// √x √[3]y` `// value of Ex`

`Math.min(a,b,c,d); and Math.max(a,b,c,d);` `Math.random();`

`Math.log(); Math.log2(); Math.log10();`

Math functions II. - Angle functions

`sin(x)` // sine of x (x is in rad.)

`asin(x)` // arcsine of x, in rad.

`asinh(x)` // hyperb. arcsine of x

`sinh(x)` // hyperb. sine of x

`cos(x)` // cosine of x (x is in rad.)

`acos(x)` // arccosine of x, in rad.

`acosh(x)` // hyperb. arccosine of x

`cosh(x)` // hyperb. cosine of x

`tan(x)` // tang. of an angl

`atan(x)` // arctang. of x as a numeric value btw. -PI/2

`atan2(y, x)` and PI/2 rad.

`atanh(x)` // arctang. of the quotient of its arguments

`tanh(x)` // hyperb. arctang. of x // hyperb. tang. of a num.

String functions

`length` `str.length`

Returns the number of characters in a string

`substring()` `str.substring(indexStart, indexEnd)`

Returns a specified part of the string

`slice()` `str.slice(beginIndex, endIndex)`

Extracts and returns a section of the string

`substr()` `str.substr(beginIndex, length)`

String functions (cont)

similar to `slice()`, but the second parameter the extracted part.

`replace()` `str.replace(pattern, replacement)`

replace a substring/pattern in the string

`replaceAll()` `str.replaceAll(pattern, replacement)`

Returns string by replacing all matching pattern

`toUpperCase()` `str.toUpperCase()`

`toLowerCase()` `str.toLowerCase()`

Returns uppercase/lowercase representation of string

`concat()` `str.concat(str1, ...strN)`

Concatenates the arguments to the calling string

`repeat()` `str.repeat(count)`

Returns a string by repeating it given times

`trim()` `str.trim()`

Removes whitespace from both ends of a string

`padStart()` `str.padStart(targetLength, padString)`

`padEnd()` `str.padEnd(targetLength, padString)`

Pads a string at the start/end to a given length

`charAt()` `str.charAt(index)`

Returns character at a specified index in string

`charCodeAt()` `str.charCodeAt(index)`

Returns Unicode of the character at given index

`fromCharCode()` `String.fromCharCode(...codePoints)`

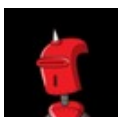
Returns a string from the given UTF-16 code units

`codePointAt()` `str.codePointAt(index)`

Returns the Unicode point value at given index

`fromCodePoint()` `String.fromCodePoint(...codePoints)`

Returns a string using the given code points



String functions (cont)	Booleans	Statements Reference
split() <code>str.split(separator, limit)</code>		
Returns the string divided into list of substring	The <code>Boolean()</code> function is used to convert various data types to boolean values.	
-- Search Methods --	You can use the <code>Boolean()</code> function to find out if an expression is true.	Date Reference
indexOf() <code>str.indexOf(searchValue, fromIndex)</code>	See Comparison and Logical Operators .	
lastIndexOf() <code>str.lastIndexOf(searchValue, fromIndex)</code>		Regular expressions
Returns the first index/last index of occurrence of a value		
search() <code>str.search(regex)</code>	Array Functions	Class Reference
Searches for specified value in the string	<code>// const array_name = [item1, item2, ...]; // var firstElement =</code>	
match() <code>str.match(regex)</code>	push() and unshift()	Error Reference
Returns result of matching string with a regex	adds an element at the end/beginning of the array.	
matchAll() <code>str.matchAll(regex)</code>	pop() and shift()	
Returns iterator of results matching with a regex	remove the last/first element from an array.	
includes() <code>str.includes(searchString, position)</code>	length	<code>// var last = array[array.length - 1]</code>
Checks if given string is found inside a string	Returns the number of elements in an array	
startsWith() <code>str.startsWith(searchString, position)</code>	Returns the number of elements in an array	
endsWith() <code>str.endsWith(searchString, position)</code>	Returns the number of elements in an array	<code>typeof (array) // object</code>
Checks if a string begins/ends with a specified string	Returns the number of elements in an array	<code>Array.isArray(array)</code>
localeCompare() <code>str.localeCompare(compareStr, locales, options)</code>	Objects	
Compares two strings in the current locale		
-- Template Literals -- <code>` \${...} `</code>	Events	
Template literals provide an easy way to interpolate variables and expressions into strings.		
	Event listener	
	Useful links	
	https://www.w3schools.com/jsref/default.asp	
	mdn web docs	
	educba.com - javascript-tutorial	
	https://www.javascripture.com/	
	Programiz - Getting Started	
	Codecademy - Cheatsheets / Learn JavaScript	
	htmlcheatsheet.com/js	
	Websitesetup - Javascript-Cheat-Sheet.pdf	
	javascript.info	

