JavaScript operators are used to assign values, compare values, perform arithmetic operations, and more.

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic between variables and/or values.

Given that **y = 5**, the table below explains the arithmetic operators:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator** | **Description** | **Example** | **Result in y** | **Result in x** | **Try it** |
| + | Addition | x = y + 2 | y = 5 | x = 7 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_add) |
| - | Subtraction | x = y - 2 | y = 5 | x = 3 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_sub) |
| \* | Multiplication | x = y \* 2 | y = 5 | x = 10 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_mult) |
| / | Division | x = y / 2 | y = 5 | x = 2.5 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_div) |
| % | Modulus (division remainder) | x = y % 2 | y = 5 | x = 1 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_mod) |
| ++ | Increment | x = ++y | y = 6 | x = 6 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_incr) |
| x = y++ | y = 6 | x = 5 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_incr2) |
| -- | Decrement | x = --y | y = 4 | x = 4 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_decr) |
| x = y-- | y = 4 | x = 5 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_decr2) |

For a tutorial about arithmetic operators, read our [JavaScript Arithmetic Tutorial](https://www.w3schools.com/js/js_arithmetic.asp).

JavaScript Assignment Operators

Assignment operators are used to assign values to JavaScript variables.

Given that **x = 10** and **y = 5**, the table below explains the assignment operators:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Example** | **Same As** | **Result in x** | **Try it** |
| = | x = y | x = y | x = 5 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_equal) |
| += | x += y | x = x + y | x = 15 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_plusequal) |
| -= | x -= y | x = x - y | x = 5 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_minequal) |
| \*= | x \*= y | x = x \* y | x = 50 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_multequal) |
| /= | x /= y | x = x / y | x = 2 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_divequal) |
| %= | x %= y | x = x % y | x = 0 | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_modequal) |

For a tutorial about assignment operators, read our [JavaScript Assignment Tutorial](https://www.w3schools.com/js/js_assignment.asp).

JavaScript String Operators

The + operator, and the += operator can also be used to concatenate (add) strings.

Given that **text1 = "Good "**, **text2 = "Morning"**, **and text3 = ""**, the table below explains the operators:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator** | **Example** | **text1** | **text2** | **text3** | **Try it** |
| + | text3 = text1 + text2 | "Good " | "Morning" | "Good Morning" | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_string1) |
| += | text1 += text2 | "Good Morning" | "Morning" | "" | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_string2) |

Comparison Operators

Comparison operators are used in logical statements to determine equality or difference between variables or values.

Given that **x = 5**, the table below explains the comparison operators:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Operator** | **Description** | **Comparing** | **Returns** | **Try it** |
| == | equal to | x == 8 | false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison1) |
| x == 5 | true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison2) |
| === | equal value and equal type | x === "5" | false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison3) |
| x === 5 | true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison4) |
| != | not equal | x != 8 | true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison5) |
| !== | not equal value or not equal type | x !== "5" | true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison6) |
| x !== 5 | false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison7) |
| > | greater than | x > 8 | false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison8) |
| < | less than | x < 8 | true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison9) |
| >= | greater than or equal to | x >= 8 | false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison10) |
| <= | less than or equal to | x <= 8 | *true* | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison11) |

For a tutorial about comparison operators, read our [JavaScript Comparisons Tutorial](https://www.w3schools.com/js/js_comparisons.asp).

Conditional (Ternary) Operator

The conditional operator assigns a value to a variable based on a condition.

|  |  |  |
| --- | --- | --- |
| **Syntax** | **Example** | **Try it** |
| *variablename* = (*condition*) ? *value1*:*value2* | voteable = (age < 18) ? "Too young":"Old enough"; | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_comparison) |

**Example explained:** If the variable "age" is a value below 18, the value of the variable "voteable" will be "Too young", otherwise the value of voteable will be "Old enough".

Logical Operators

Logical operators are used to determine the logic between variables or values.

Given that **x = 6 and y = 3**, the table below explains the logical operators:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Description** | **Example** | **Try it** |
| && | and | (x < 10 && y > 1) is true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_and) |
| || | or | (x === 5 || y === 5) is false | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_or) |
| ! | not | !(x === y) is true | [Try it »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_not) |

JavaScript Bitwise Operators

Bit operators work on 32 bits numbers. Any numeric operand in the operation is converted into a 32 bit number. The result is converted back to a JavaScript number.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Operator** | **Description** | **Example** | **Same as** | **Result** | **Decimal** |
| & | AND | x = 5 & 1 | 0101 & 0001 | 0001 | 1 |
| | | OR | x = 5 | 1 | 0101 | 0001 | 0101 | 5 |
| ~ | NOT | x = ~ 5 | ~0101 | 1010 | 10 |
| ^ | XOR | x = 5 ^ 1 | 0101 ^ 0001 | 0100 | 4 |
| << | Left shift | x = 5 << 1 | 0101 << 1 | 1010 | 10 |
| >> | Right shift | x = 5 >> 1 | 0101 >> 1 | 0010 | 2 |

The examples above uses 4 bits unsigned examples. But JavaScript uses 32-bit signed numbers.  
  
Because of this, in JavaScript, ~ 5 will not return 10. It will return -6.  
  
~00000000000000000000000000000101 will return 11111111111111111111111111111010

The typeof Operator

The **typeof** operator returns the type of a variable, object, function or expression:

Example

typeof "John"                 // Returns string   
typeof 3.14                   // Returns number  
typeof NaN                    // Returns number  
typeof false                  // Returns boolean  
typeof [1, 2, 3, 4]           // Returns object  
typeof {name:'John', age:34}  // Returns object  
typeof new Date()             // Returns object  
typeof function () {}         // Returns function  
typeof myCar                  // Returns undefined (if myCar is not declared)  
typeof null                   // Returns object

[Try it Yourself »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_typeof)

Please observe:

* The data type of NaN is number
* The data type of an array is object
* The data type of a date is object
* The data type of null is object
* The data type of an undefined variable is undefined

You cannot use **typeof** to define if a JavaScript object is an array (or a date).

The delete Operator

The **delete** operator deletes a property from an object:

Example

var person = {firstName:"John", lastName:"Doe", age:50, eyeColor:"blue"};  
delete person.age;   // or delete person["age"];

[Try it Yourself »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_delete)

The delete operator deletes both the value of the property and the property itself.

After deletion, the property cannot be used before it is added back again.

The delete operator is designed to be used on object properties. It has no effect on variables or functions.

**Note:** The delete operator should not be used on predefined JavaScript object properties. It can crash your application.

The in Operator

The **in** operator returns true if the specified property is in the specified object, otherwise false:

Example

// Arrays  
var cars = ["Saab", "Volvo", "BMW"];  
"Saab" in cars          // Returns false (specify the index number instead of value)  
0 in cars               // Returns true  
1 in cars               // Returns true  
4 in cars               // Returns false (does not exist)  
"length" in cars        // Returns true  (length is an Array property)  
  
// Objects  
var person = {firstName:"John", lastName:"Doe", age:50};  
"firstName" in person   // Returns true  
"age" in person         // Returns true  
  
// Predefined objects  
"PI" in Math            // Returns true  
"NaN" in Number         // Returns true  
"length" in String      // Returns true

[Try it Yourself »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_in)

The instanceof Operator

The **instanceof** operator returns true if the specified object is an instance of the specified object:

Example

var cars = ["Saab", "Volvo", "BMW"];  
  
cars instanceof Array;          // Returns true  
cars instanceof Object;         // Returns true  
cars instanceof String;         // Returns false  
cars instanceof Number;         // Returns false

[Try it Yourself »](https://www.w3schools.com/jsref/tryit.asp?filename=tryjsref_oper_instanceof)

The void Operator

The **void** operator evaluates an expression and returns **undefined**. This operator is often used to obtain the undefined primitive value, using "void(0)" (useful when evaluating an expression without using the return value).

Example

<a href="javascript:void(0);">  
  Useless link  
</a>  
  
<a href="javascript:void(document.body.style.backgroundColor='red');">  
  Click me to change the background color of body to red  
</a>