JavaScript

Operadores

Operadores

Operadores de Atribuição

Name	Shorthand operator	Meaning
Assignment	`x = f()`	`x = f()`
Addition assignment	`x += f()`	`x = x + f()`
Subtraction assignment	`x -= f()`	`x = x - f()`
Multiplication assignment	`x *= f()`	`x = x * f()`
Division assignment	`x /= f()`	`x = x / f()`
Remainder assignment	`x %= f()`	`x = x % f()`
Exponentiation assignment	`x **= f()`	`x = x ** f()`

Operadores

Operadores de Atribuição

Operator	Description	Examples returning true

Equal (==)	Returns `true` if the operands are equal.	`3 == var1`
		`"3" == var1`
		`3 == '3'`

Not equal (!=)	Returns `true` if the operands are not equal.	`var1 != 4` `var2 != "3"`
Strict equal (===)	Returns `true` if the operands are equal and of the same type. See also `Object.is` and sameness in JS.	`3 === var1`
Strict not equal (!==)	Returns `true` if the operands are of the same type but not equal, or are of different type.	`var1 !== "3"` `3 !== '3'`
Greater than (>)	Returns `true` if the left operand is greater than the right operand.	`var2 > var1` `"12" > 2`
Greater than or equal (>=)	Returns `true` if the left operand is greater than or equal to the right operand.	`var2 >= var1` `var1 >= 3`
Less than (<)	Returns `true` if the left operand is less than the right operand.	`var1 < var2` `"2" < 12`
Less than or equal (<=)	Returns `true` if the left operand is less than or equal to the right operand.	`var1 <= var2` `var2 <= 5`

Operadores

Operadores Aritméticos

Operator	Description	Example
Remainder (`%`)	Binary operator. Returns the integer remainder of dividing the two operands.	12 % 5 returns 2.
Increment (`++`)	Unary operator. Adds one to its operand. If used as a prefix operator (`++x`), returns the value of its operand after adding one; if used as a postfix operator (`x++`), returns the value of its operand before adding one.	If `x` is 3, then `++x` sets `x` to 4 and returns 4, whereas `x++` returns 3 and, only then, sets `x` to 4.
Decrement (`--`)	Unary operator. Subtracts one from its operand. The return value is analogous to that for the increment operator.	If `x` is 3, then `--x` sets `x` to 2 and returns 2, whereas `x--` returns 3 and, only then, sets `x` to 2.
Unary negation (`-`)	Unary operator. Returns the negation of its operand.	If `x` is 3, then `-x` returns -3.
Unary plus (`+`)	Unary operator. Attempts to convert the operand to a number, if it is not already.	`+"3"` returns `3`. `+true` returns `1`.
Exponentiation operator (`**`)	Calculates the `base` to the `exponent` power, that is, `base^exponent`	`2 ** 3` returns `8`. `10 ** -1` returns `0.1`.

Operadores

Operadores Lógicos

Operator	Usage	Description
Logical AND (`&&`)	`expr1 && expr2`	Returns `expr1` if it can be converted to `false`; otherwise, returns `expr2`. Thus, when used with Boolean values, `&&` returns `true` if both operands are true; otherwise, returns `false`.
Logical OR (`||`)	`expr1 || expr2`	Returns `expr1` if it can be converted to `true`; otherwise, returns `expr2`. Thus, when used with Boolean values, `||` returns `true` if either operand is true; if both are false, returns `false`.
Logical NOT (`!`)	`!expr`	Returns `false` if its single operand that can be converted to `true`; otherwise, returns `true`.

Operadores

Operador Ternários

```
condition ? val1 : val2
```

```
const status = age >= 18 ? "adult" : "minor";
```

Operadores

Operador in

```
// Arrays
const trees = ["redwood", "bay", "cedar", "oak", "maple"];
0 in trees; // returns true
3 in trees; // returns true
6 in trees; // returns false
"bay" in trees; // returns false
// (you must specify the index number, not the value at that index)
"length" in trees; // returns true (length is an Array property)

// built-in objects
"PI" in Math; // returns true
const myString = new String("coral");
"length" in myString; // returns true

// Custom objects
const mycar = { make: "Honda", model: "Accord", year: 1998 };
"make" in mycar; // returns true
"model" in mycar; // returns true
```

Operadores

Operador instanceOf

```
objectName instanceof objectType
```

```
const theDay = new Date(1995, 12, 17);
if (theDay instanceof Date) {
  // statements to execute
}
```