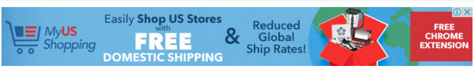
JS by Category
JS by Alphabet

JavaScript
JS Array
JS Boolean
JS Classes
JS Date
JS Error
JS Global
JS JSON
JS Math
JS Number
JS Operators
JS RegExp
JS Statements
JS String

HTML DOM


MyUS Shopping Easily Shop US Stores with FREE DOMESTIC SHIPPING & Reduced Global Ship Rates! FREE CHROME EXTENSION

Referência de operadores de JavaScript

‹ Anterior Próximo ›

Os operadores JavaScript são usados para atribuir valores, comparar valores, realizar operações aritméticas e muito mais.

Operadores aritméticos JavaScript

Operadores aritméticos são usados para realizar operações aritméticas entre variáveis e / ou valores.

Dado que **y = 5** , a tabela abaixo explica os operadores aritméticos:

Operator	Description	Example	Result in y	Result in x	Try it
+	Addition	x = y + 2	y = 5	x = 7	Try it »
-	Subtraction	x = y - 2	y = 5	x = 3	Try it »
*	Multiplication	x = y * 2	y = 5	x = 10	Try it »
/	Division	x = y / 2	y = 5	x = 2.5	Try it »
%	Modulus (division remainder)	x = y % 2	y = 5	x = 1	Try it »
++	Increment	x = ++y	y = 6	x = 6	Try it »
		x = y++	y = 6	x = 5	Try it »
--	Decrement	x = --y	y = 4	x = 4	Try it »
		x = y--	y = 4	x = 5	Try it »

Para obter um tutorial sobre operadores aritméticos, leia nosso Tutorial de Aritmética de JavaScript .

Operadores de atribuição de JavaScript

Operadores de atribuição são usados para atribuir valores a variáveis JavaScript.

Dado que **x = 10** e **y = 5** , a tabela abaixo explica os operadores de atribuição:

Operator	Example	Same As	Result in x	Try it
=	x = y	x = y	x = 5	Try it »
+=	x += y	x = x + y	x = 15	Try it »
-=	x -= y	x = x - y	x = 5	Try it »
*=	x *= y	x = x * y	x = 50	Try it »
/=	x /= y	x = x / y	x = 2	Try it »
%=	x %= y	x = x % y	x = 0	Try it »

For a tutorial about assignment operators, read our JavaScript Assignment Tutorial.

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 »
+=	text1 += text2	"Good Morning"	"Morning"	""	Try it »

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 »
		x == 5	true	Try it »
===	equal value and equal type	x === "5"	false	Try it »
		x === 5	true	Try it »
!=	not equal	x != 8	true	Try it »
!==	not equal value or not equal type	x !== "5"	true	Try it »
		x !== 5	false	Try it »
>	greater than	x > 8	false	Try it »
<	less than	x < 8	true	Try it »
>=	greater than or equal to	x >= 8	false	Try it »
<=	less than or equal to	x <= 8	*true*	Try it »

For a tutorial about comparison operators, read our JavaScript Comparisons Tutorial.

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 »

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 »
||	or	(x === 5 || y === 5) is false	Try it »
!	not	!(x === y) is true	Try it »

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 »

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

PROPAGANDA

MyUS Shopping Easily Shop US Stores with FREE DOMESTIC SHIPPING & Reduced Global Ship Rates! FREE CHROME EXTENSION

COLOR PICKER

COMO NÓS

Obtenha a certificação completando um curso hoje!
iniciar

JOGO DE CÓDIGOS

Jogar um jogo

Certificados
HTML
CSS
JavaScript
Front-end
Pitão
SQL
E mais

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 »

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 »

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 »

The void Operator

O operador **void** avalia uma expressão e retorna **indefinido** . Este operador é freqüentemente usado para obter o valor primitivo indefinido, usando "void (0)" (útil ao avaliar uma expressão sem usar o valor de retorno).

Exemplo

```
<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>
```

Tente você mesmo "

❮ Anterior

Próximo ❯

REPORTAR ERRO
FÓRUM
CERCA DE
FAZER COMPRAS

Principais tutoriais
HTML Tutorial
CSS Tutorial
JavaScript Tutorial
How To Tutorial
SQL Tutorial
Python Tutorial
W3.CSS Tutorial
Bootstrap Tutorial
PHP Tutorial
Java Tutorial
C++ Tutorial
jQuery Tutorial

Top References
HTML Reference
CSS Reference
JavaScript Reference
SQL Reference
Python Reference
W3.CSS Reference
Bootstrap Reference
PHP Reference
HTML Colors
Java Reference
Angular Reference
jQuery Reference

Top Examples
HTML Examples
CSS Examples
JavaScript Examples
How To Examples
SQL Examples
Python Examples
W3.CSS Examples
Bootstrap Examples
PHP Examples
Java Examples
XML Examples
jQuery Examples

Web Courses
HTML Course
CSS Course
JavaScript Course
Front End Course
SQL Course
Python Course
PHP Course
jQuery Course
Java Course
C++ Course
C# Course
XML Course

Get Certified »

W3Schools é otimizado para aprendizagem e treinamento. Os exemplos podem ser simplificados para melhorar a leitura e o aprendizado. Tutoriais, referências e exemplos são constantemente revisados para evitar erros, mas não podemos garantir a correção total de todo o conteúdo. Ao usar o W3Schools, você concorda em ter lido e aceito nossos termos de uso , cookies e política de privacidade .

Copyright 1999-2021 de Refsnes Data. Todos os direitos reservados.
W3Schools é desenvolvido por W3.CSS .