Print Page

JavaScript Operators Reference

« Previous (jsref_obj_number.a Next Chapter » (jsref_statements.as

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 »
-	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 »

For a tutorial about arithmetic operators, read our JavaScript Arithmetic Tutorial (/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 »
+=	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 (/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 »
+=	text1 += text2	"Good "	"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 (/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 »

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

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 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 » (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 » (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 » (tryit.asp?filename=tryjsref_oper_in)

« Previous (jsref_obj_number.a Next Chapter » (jsref_statements.as