

JavaScript Operators

[< Previous](#)[Next >](#)

Example

Assign values to variables and add them together:

```
var x = 5;           // assign the value 5 to x
var y = 2;           // assign the value 2 to y
var z = x + y;       // assign the value 7 to z (x + y)
```

[Try it Yourself »](#)

The **assignment** operator (=) assigns a value to a variable.

Assignment

```
var x = 10;
```

[Try it Yourself »](#)

The **addition** operator (+) adds numbers:

Adding

```
var x = 5;
var y = 2;
var z = x + y;
```

[Try it Yourself »](#)

The **multiplication** operator (*) multiplies numbers.

Multiplying

```
var x = 5;  
var y = 2;  
var z = x * y;
```

[Try it Yourself »](#)

JavaScript Arithmetic Operators

Arithmetic operators are used to perform arithmetic on numbers:

Operator	Description
+	Addition
-	Subtraction
*	Multiplication
/	Division
%	Modulus (Division Remainder)
++	Increment
--	Decrement

Arithmetic operators are fully described in the **JS Arithmetic** chapter.

JavaScript Assignment Operators

Assignment operators assign values to JavaScript variables.

Operator	Example	Same As
=	x = y	x = y
+=	x += y	x = x + y
-=	x -= y	x = x - y
*=	x *= y	x = x * y
/=	x /= y	x = x / y
%=	x %= y	x = x % y

The **addition assignment** operator (+=) adds a value to a variable.

Assignment

```
var x = 10;  
x += 5;
```

Try it Yourself »

Assignment operators are fully described in the **JS Assignment** chapter.

JavaScript String Operators

The + operator can also be used to add (concatenate) strings.

Example

```
var txt1 = "John";  
var txt2 = "Doe";  
var txt3 = txt1 + " " + txt2;
```

The result of txt3 will be:

John Doe

[Try it Yourself »](#)

The += assignment operator can also be used to add (concatenate) strings:

Example

```
var txt1 = "What a very ";  
txt1 += "nice day";
```

The result of txt1 will be:

What a very nice day

[Try it Yourself »](#)

When used on strings, the + operator is called the concatenation operator.

Adding Strings and Numbers

Adding two numbers, will return the sum, but adding a number and a string will return a string:


Example

```
var x = 5 + 5;  
var y = "5" + 5;  
var z = "Hello" + 5;
```

The result of x, y, and z will be:

10

55

 Hello5

Try it Yourself »

 If you add a number and a string, the result will be a string!

JavaScript Comparison Operators

Operator	Description
==	equal to
===	equal value and equal type
!=	not equal
!==	not equal value or not equal type
>	greater than
<	less than
>=	greater than or equal to
<=	less than or equal to
?	ternary operator

 Comparison operators are fully described in the **JS Comparisons** chapter.

JavaScript Logical Operators

Operator	Description
&&	logical and
	logical or
!	logical not

Logical operators are fully described in the **JS Comparisons** chapter.

JavaScript Type Operators

Operator	Description
typeof	Returns the type of a variable
instanceof	Returns true if an object is an instance of an object type

Type operators are fully described in the **JS Type Conversion** chapter.

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	5 & 1	0101 & 0001	0001	1
	OR	5 1	0101 0001	0101	5
~	NOT	~ 5	~0101	1010	10
^	XOR	5 ^ 1	0101 ^ 0001	0100	4
<<	Zero fill left shift	5 << 1	0101 << 1	1010	10
>>	Signed right shift	5 >> 1	0101 >> 1	0010	2
>>>	Zero fill right shift	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.

~00000000000000000000000000000000101 will return

1111111111111111111111111111111010

Bitwise operators are fully described in the **JS Bitwise** chapter.

Test Yourself with Exercises!

[Exercise 1 »](#)[Exercise 2 »](#)[Exercise 3 »](#)[Exercise 4 »](#)[Exercise 5 »](#)[< Previous](#)[Next >](#)

Copyright 1999-2018 by Refsnes Data. All Rights Reserved.