JavaScript Number Methods

‹ Previous Next ›

JavaScript Number Methods

These **number methods** can be used on all JavaScript numbers:

Method	Description
toString()	Returns a number as a string
toExponential()	Returns a number written in exponential notation
toFixed()	Returns a number written with a number of decimals
toPrecision()	Returns a number written with a specified length
valueOf()	Returns a number as a number

The toString() Method

The `toString()` method returns a number as a string.

All number methods can be used on any type of numbers (literals, variables, or expressions):

Example

```
let x = 123;
x.toString();
(123).toString();
(100 + 23).toString();
```

Try it Yourself »

The toExponential() Method

`toExponential()` returns a string, with a number rounded and written using exponential notation.

A parameter defines the number of characters behind the decimal point:

Example

```
let x = 9.656;
x.toExponential(2);
x.toExponential(4);
x.toExponential(6);
```

Try it Yourself »

The parameter is optional. If you don't specify it, JavaScript will not round the number.

The toFixed() Method

`toFixed()` returns a string, with the number written with a specified number of decimals:

Example

```
let x = 9.656;
x.toFixed(0);
x.toFixed(2);
x.toFixed(4);
x.toFixed(6);
```

Try it Yourself »

`toFixed(2)` is perfect for working with money.

The toPrecision() Method

`toPrecision()` returns a string, with a number written with a specified length:

Example

```
let x = 9.656;
x.toPrecision();
x.toPrecision(2);
x.toPrecision(4);
x.toPrecision(6);
```

Try it Yourself »

The valueOf() Method

`valueOf()` returns a number as a number.

Example

```
let x = 123;
x.valueOf();
(123).valueOf();
(100 + 23).valueOf();
```

Try it Yourself »

In JavaScript, a number can be a primitive value (typeof = number) or an object (typeof = object).

The `valueOf()` method is used internally in JavaScript to convert Number objects to primitive values.

There is no reason to use it in your code.

All JavaScript data types have a `valueOf()` and a `toString()` method.

Converting Variables to Numbers

There are 3 JavaScript methods that can be used to convert a variable to a number:

Method	Description
Number()	Returns a number converted from its argument.
parseFloat()	Parses its argument and returns a floating point number
parseInt()	Parses its argument and returns a whole number

The methods above are not **number** methods. They are **global** JavaScript methods.

The Number() Method

The `Number()` method can be used to convert JavaScript variables to numbers:

Example

```
Number(true);
Number(false);
Number("10");
Number("  10");
Number("10  ");
Number(" 10  ");
Number("10.33");
Number("10,33");
Number("10 33");
Number("John");
```

Try it Yourself »

If the number cannot be converted, `NaN` (Not a Number) is returned.

The Number() Method Used on Dates

`Number()` can also convert a date to a number.

Example

```
Number(new Date("1970-01-01"))
```

Try it Yourself »

Note

The `Date()` method returns the number of milliseconds since 1.1.1970.

The number of milliseconds between 1970-01-02 and 1970-01-01 is 86400000:

Example

```
Number(new Date("1970-01-02"))
```

Try it Yourself »

Example

```
Number(new Date("2017-09-30"))
```

Try it Yourself »

The parseInt() Method

`parseInt()` parses a string and returns a whole number. Spaces are allowed. Only the first number is returned:

Example

```
parseInt("-10");
parseInt("-10.33");
parseInt("10");
parseInt("10.33");
parseInt("10 20 30");
parseInt("10 years");
parseInt("years 10");
```

Try it Yourself »

If the number cannot be converted, `NaN` (Not a Number) is returned.

The parseFloat() Method

`parseFloat()` parses a string and returns a number. Spaces are allowed. Only the first number is returned:

Example

```
parseFloat("10");
parseFloat("10.33");
parseFloat("10 20 30");
parseFloat("10 years");
parseFloat("years 10");
```

Try it Yourself »

If the number cannot be converted, `NaN` (Not a Number) is returned.

Number Object Methods

These **object methods** belong to the **Number** object:

Method	Description

Number.isInteger()	Returns true if the argument is an integer
Number.isSafeInteger()	Returns true if the argument is a safe integer
Number.parseFloat()	Converts a string to a number
Number.parseInt()	Converts a string to a whole number

Number Methods Cannot be Used on Variables

The number methods above belong to the JavaScript **Number Object**.

These methods can only be accessed like `Number.isInteger()`.

Using X.isInteger() where X is a variable, will result in an error:

`TypeError X.isInteger is not a function`.

The Number.isInteger() Method

The `Number.isInteger()` method returns `true` if the argument is an integer.

Example

```
Number.isInteger(10);
Number.isInteger(10.5);
```

Try it Yourself »

The Number.isSafeInteger() Method

A safe integer is an integer that can be exactly represented as a double precision number.

The `Number.isSafeInteger()` method returns `true` if the argument is a safe integer.

Example

```
Number.isSafeInteger(10);
Number.isSafeInteger(12345678901234567890);
```

Try it Yourself »

Safe integers are all integers from $-(2^{53} - 1)$ to $+(2^{53} - 1)$.
This is safe: 9007199254740991. This is not safe: 9007199254740992.

The Number.parseFloat() Method

`Number.parseFloat()` parses a string and returns a number.

Spaces are allowed. Only the first number is returned:

Example

```
Number.parseFloat("10");
Number.parseFloat("10.33");
Number.parseFloat("10 20 30");
Number.parseFloat("10 years");
Number.parseFloat("years 10");
```

Try it Yourself »

If the number cannot be converted, `NaN` (Not a Number) is returned.

Note

The **Number** methods `Number.parseInt()` and `Number.parseFloat()`

are the same as the

Global methods `parseInt()` and `parseFloat()`.

The purpose is modularization of globals (to make it easier to use the same JavaScript code outside the browser).

The Number.parseInt() Method

`Number.parseInt()` parses a string and returns a whole number.

Spaces are allowed. Only the first number is returned:

Example

```
Number.parseInt("-10");
Number.parseInt("-10.33");
Number.parseInt("10");
Number.parseInt("10.33");
Number.parseInt("10 20 30");
Number.parseInt("10 years");
Number.parseInt("years 10");
```

Try it Yourself »

If the number cannot be converted, `NaN` (Not a Number) is returned.

Complete JavaScript Number Reference

For a complete Number reference, visit our:

Complete JavaScript Number Reference.

The reference contains descriptions and examples of all Number properties and methods.

Exercise?

What is a legal JavaScript method for returning a number as a string?

- `intToString()`
- `toString()`
- `stringify()`

Submit Answer »

‹ Previous Next ›