JavaScript Type Conversion

‹ Previous Next ›

JavaScript Type Conversion

JavaScript variables can be converted to a new variable and another data type:

- By the use of a JavaScript function
- **Automatically** by JavaScript itself

Converting Strings to Numbers

The global method `Number()` can convert strings to numbers.

Strings containing numbers (like `"3.14"`) convert to numbers (like `3.14`).

Empty strings convert to `0`.

Anything else converts to `NaN` (Not a Number).

```
Number("3.14")    // returns 3.14
Number(" ")       // returns 0
Number("")        // returns 0
Number("99 88")   // returns NaN
```

Number Methods

In the chapter Number Methods, you will find more methods that can be used to convert strings to numbers:

Method	Description
Number()	Returns a number, converted from its argument
parseFloat()	Parses a string and returns a floating point number
parseInt()	Parses a string and returns an integer

The Unary + Operator

The **unary + operator** can be used to convert a variable to a number:

Example

```
let y = "5";      // y is a string
let x = + y;      // x is a number
```

Try it Yourself »

If the variable cannot be converted, it will still become a number, but with the value `NaN` (Not a Number):

Example

```
let y = "John";   // y is a string
let x = + y;      // x is a number (NaN)
```

Try it Yourself »

Converting Numbers to Strings

The global method `String()` can convert numbers to strings.

It can be used on any type of numbers, literals, variables, or expressions:

Example

```
String(x)         // returns a string from a number variable x
String(123)       // returns a string from a number literal 123
String(100 + 23)  // returns a string from a number from an expression
```

Try it Yourself »

The Number method `toString()` does the same.

Example

```
x.toString()
(123).toString()
(100 + 23).toString()
```

Try it Yourself »

More Methods

In the chapter Number Methods, you will find more methods that can be used to convert numbers to strings:

Method	Description
toExponential()	Returns a string, with a number rounded and written using exponential notation.
toFixed()	Returns a string, with a number rounded and written with a specified number of decimals.
toPrecision()	Returns a string, with a number written with a specified length

Converting Dates to Numbers

The global method `Number()` can be used to convert dates to numbers.

```
d = new Date();
Number(d)          // returns 1404568027739
```

The date method `getTime()` does the same.

```
d = new Date();
d.getTime()        // returns 1404568027739
```

Converting Dates to Strings

The global method `String()` can convert dates to strings.

```
String(Date()) // returns "Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)"
```

The Date method `toString()` does the same.

Example

```
Date().toString() // returns "Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)"
```

In the chapter Date Methods, you will find more methods that can be used to convert dates to strings:

Method	Description
getDate()	Get the day as a number (1-31)
getDay()	Get the weekday as a number (0-6)
getFullYear()	Get the four digit year (yyyy)
getHours()	Get the hour (0-23)
getMilliseconds()	Get the milliseconds (0-999)
getMinutes()	Get the minutes (0-59)
getMonth()	Get the month (0-11)
getSeconds()	Get the seconds (0-59)
getTime()	Get the time (milliseconds since January 1, 1970)

Converting Booleans to Numbers

The global method `Number()` can also convert booleans to numbers.

```
Number(false)     // returns 0
Number(true)      // returns 1
```

Converting Booleans to Strings

The global method `String()` can convert booleans to strings.

```
String(false)     // returns "false"
String(true)      // returns "true"
```

The Boolean method `toString()` does the same.

```
false.toString()  // returns "false"
true.toString()   // returns "true"
```

Automatic Type Conversion

When JavaScript tries to operate on a "wrong" data type, it will try to convert the value to a "right" type.

The result is not always what you expect:

```
5 + null    // returns 5         because null is converted to 0
"5" + null  // returns "5null"   because null is converted to "null"
"5" + 2     // returns "52"      because 2 is converted to "2"
"5" - 2     // returns 3         because "5" is converted to 5
"5" * "2"   // returns 10        because "5" and "2" are converted to 5 and 2
```

Try it Yourself »

Automatic String Conversion

JavaScript automatically calls the variable's `toString()` function when you try to "output" an object or a variable:

```
document.getElementById("demo").innerHTML = myVar;

// if myVar = {name:"Fjohn"}  // toString converts to "[object Object]"
// if myVar = [1,2,3,4]       // toString converts to "1,2,3,4"
// if myVar = new Date()      // toString converts to "Fri Jul 18 2014 09:08:55 GMT+0200"
```

Numbers and booleans are also converted, but this is not very visible:

```
// if myVar = 123             // toString converts to "123"
// if myVar = true            // toString converts to "true"
// if myVar = false           // toString converts to "false"
```

JavaScript Type Conversion Table

This table shows the result of converting different JavaScript values to Number, String, and Boolean:

Original Value	Converted to Number	Converted to String	Converted to Boolean	Try It
false	0	"false"	false	Try it »
true	1	"true"	true	Try it »
0	0	"0"	false	Try it »
1	1	"1"	true	Try it »
"0"	0	"0"	true	Try it »
"000"	0	"000"	true	Try it »
"1"	1	"1"	true	Try it »
NaN	NaN	"NaN"	false	Try it »
Infinity	Infinity	"Infinity"	true	Try it »
-Infinity	-Infinity	"-Infinity"	true	Try it »
""	0	""	false	Try it »
"20"	20	"20"	true	Try it »
"twenty"	NaN	"twenty"	true	Try it »
[ ]	0	""	false	Try it »
[20]	20	"20"	true	Try it »
[10,20]	NaN	"10,20"	true	Try it »
["twenty"]	NaN	"twenty"	true	Try it »
["ten","twenty"]	NaN	"ten,twenty"	true	Try it »
function(){}	NaN	"function(){}"	true	Try it »
{ }	NaN	"[object Object]"	true	Try it »
null	0	"null"	false	Try it »
undefined	NaN	"undefined"	false	Try it »

Values in quotes indicate string values.

Red values indicate values (some) programmers might not expect.

‹ Previous Next ›