

# JavaScript 类型转换

Number() 转换为数字，String() 转换为字符串，Boolean() 转化为布尔值。

## JavaScript 数据类型

在 JavaScript 中有 5 种不同的数据类型：

- string
- number
- boolean
- object
- function

3 种对象类型：

- Object
- Date
- Array

2 个不包含任何值的数据类型：

- null
- undefined

## typeof 操作符

你可以使用 **typeof** 操作符来查看 JavaScript 变量的数据类型。

### 实例

```
typeof "John"           // 返回 string
typeof 3.14              // 返回 number
typeof NaN              // 返回 number
typeof false            // 返回 boolean
typeof [1,2,3,4]         // 返回 object
typeof {name:'John', age:34} // 返回 object
typeof new Date()        // 返回 object
typeof function () {}    // 返回 function
typeof myCar             // 返回 undefined (如果 myCar 没有声明)
typeof null             // 返回 object
```

[尝试一下 »](#)

请注意：

- NaN 的数据类型是 number
- 数组(Array)的数据类型是 object
- 日期(Date)的数据类型为 object
- null 的数据类型是 object
- 未定义变量的数据类型为 undefined

如果对象是 JavaScript Array 或 JavaScript Date ，我们就无法通过 **typeof** 来判断他们的类型，因为都是 返回 object。

## constructor 属性

**constructor** 属性返回所有 JavaScript 变量的构造函数。

### 实例

```
"John".constructor      // 返回函数 String() { [native code] }
(3.14).constructor      // 返回函数 Number() { [native code] }
false.constructor       // 返回函数 Boolean() { [native code] }
[1,2,3,4].constructor   // 返回函数 Array() { [native code] }
{name:'John', age:34}.constructor // 返回函数 Object() { [native code] }
new Date().constructor   // 返回函数 Date() { [native code] }
function () {}.constructor // 返回函数 Function(){ [native code] }
```

尝试一下 »

你可以使用 constructor 属性来查看对象是否为数组 (包含字符串 "Array"):

### 实例

```
function isArray(myArray) {
    return myArray.constructor.toString().indexOf("Array") > -1;
}
```

尝试一下 »

你可以使用 constructor 属性来查看对象是否为日期 (包含字符串 "Date"):

### 实例

```
function isDate(myDate) {
    return myDate.constructor.toString().indexOf("Date") > -1;
}
```

尝试一下 »

## JavaScript 类型转换

JavaScript 变量可以转换为新变量或其他数据类型：

- 通过使用 JavaScript 函数
- 通过 JavaScript 自身自动转换

## 将数字转换为字符串

全局方法 **String()** 可以将数字转换为字符串。

该方法可用于任何类型的数字，字母，变量，表达式：

### 实例

```
String(x)           // 将变量 x 转换为字符串并返回
String(123)          // 将数字 123 转换为字符串并返回
String(100 + 23)     // 将数字表达式转换为字符串并返回
```

尝试一下 »

Number 方法 **toString()** 也是有同样的效果。

### 实例

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

尝试一下 »

在 [Number 方法](#) 章节中，你可以找到更多数字转换为字符串的方法：

方法	描述
toExponential()	把对象的值转换为指数计数法。
toFixed()	把数字转换为字符串，结果的小数点后有指定位数的数字。
toPrecision()	把数字格式化为指定的长度。

## 将布尔值转换为字符串

全局方法 **String()** 可以将布尔值转换为字符串。

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

Boolean 方法 **toString()** 也有相同的效果。

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

# 将日期转换为字符串

Date() 返回字符串。

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

全局方法 String() 可以将日期对象转换为字符串。

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

Date 方法 toString() 也有相同的效果。

实例

```
obj = new Date()  
obj.toString() // 返回 Thu Jul 17 2014 15:38:19 GMT+0200 (W. Europe Daylight Time)
```

在 [Date 方法](#) 章节中，你可以查看更多关于日期转换为字符串的函数：

方法	描述
getDate()	从 Date 对象返回一个月中的某一天 (1 ~ 31)。
getDay()	从 Date 对象返回一周中的某一天 (0 ~ 6)。
getFullYear()	从 Date 对象以四位数字返回年份。
getHours()	返回 Date 对象的小时 (0 ~ 23)。
getMilliseconds()	返回 Date 对象的毫秒(0 ~ 999)。
getMinutes()	返回 Date 对象的分钟 (0 ~ 59)。
getMonth()	从 Date 对象返回月份 (0 ~ 11)。
getSeconds()	返回 Date 对象的秒数 (0 ~ 59)。
getTime()	返回 1970 年 1 月 1 日至今的毫秒数。

# 将字符串转换为数字

全局方法 Number() 可以将字符串转换为数字。

字符串包含数字(如 "3.14") 转换为数字 (如 3.14)。

空字符串转换为 0。

其他的字符串会转换为 NaN (不是个数字)。

```
Number("3.14") // 返回 3.14  
Number(" ") // 返回 0
```

```
Number("")           // 返回 0
Number("99 88")      // 返回 NaN
```

在 [Number 方法](#) 章节中，你可以查看到更多关于字符串转为数字的方法：

方法	描述
parseFloat()	解析一个字符串，并返回一个浮点数。
parseInt()	解析一个字符串，并返回一个整数。

## 一元运算符 +

**Operator +** 可用于将变量转换为数字：

### 实例

```
var y = "5";           // y 是一个字符串
var x = + y;           // x 是一个数字
```

[尝试一下 »](#)

如果变量不能转换，它仍然会是一个数字，但值为 NaN (不是一个数字):

### 实例

```
var y = "John";        // y 是一个字符串
var x = + y;           // x 是一个数字 (NaN)
```

[尝试一下 »](#)

## 将布尔值转换为数字

全局方法 **Number()** 可将布尔值转换为数字。

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

## 将日期转换为数字

全局方法 **Number()** 可将日期转换为数字。

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

日期方法 **getTime()** 也有相同的效果。

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

# 自动转换类型

当 JavaScript 尝试操作一个 "错误" 的数据类型时，会自动转换为 "正确" 的数据类型。  
以下输出结果不是你所期望的：

```
5 + null    // 返回 5      null 转换为 0
"5" + null  // 返回"5null"  null 转换为 "null"
"5" + 1     // 返回 "51"   1 转换为 "1"
"5" - 1     // 返回 4      "5" 转换为 5
```

# 自动转换为字符串

当你尝试输出一个对象或一个变量时 JavaScript 会自动调用变量的 toString() 方法：

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

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


数字和布尔值也经常相互转换：


```
myVar = 123           // toString 转换为 "123"
myVar = true          // toString 转换为 "true"
myVar = false         // toString 转换为 "false"
```

下表展示了使用不同的数值转换为数字(Number), 字符串(String), 布尔值(Boolean):

原始值	转换为数字	转换为字符串	转换为布尔值	实例
false	0	"false"	false	<a href="#">尝试一下 »</a>
true	1	"true"	true	<a href="#">尝试一下 »</a>
0	0	"0"	false	<a href="#">尝试一下 »</a>
1	1	"1"	true	<a href="#">尝试一下 »</a>
"0"	0	"0"	true	<a href="#">尝试一下 »</a>
"000"	0	"000"	true	<a href="#">尝试一下 »</a>
"1"	1	"1"	true	

				尝试一下 »
NaN	NaN	"NaN"	false	尝试一下 »
Infinity	Infinity	"Infinity"	true	尝试一下 »
-Infinity	-Infinity	"-Infinity"	true	尝试一下 »
""	0	""	false	尝试一下 »
"20"	20	"20"	true	尝试一下 »
"Runoob"	NaN	"Runoob"	true	尝试一下 »
[]	0	""	true	尝试一下 »
[20]	20	"20"	true	尝试一下 »
[10,20]	NaN	"10,20"	true	尝试一下 »
["Runoob"]	NaN	"Runoob"	true	尝试一下 »
["Runoob","Google"]	NaN	"Runoob,Google"	true	尝试一下 »
function(){}	NaN	"function(){}"	true	尝试一下 »
{ }	NaN	"[object Object]"	true	尝试一下 »
null	0	"null"	false	尝试一下 »
undefined	NaN	"undefined"	false	尝试一下 »

 3 篇笔记

 写笔记