在这篇文章中，我将分享12个非常有用的JavaScript技巧。这些技巧可以帮助你减少并优化代码。

**1) 使用!!将变量转换成布尔类型**

有时，我们需要检查一些变量是否存在，或者它是否具有有效值，从而将它们的值视为true。对于做这样的检查，你可以使用||（双重否定运算符），它能自动将任何类型的数据转换为布尔值，只有这些变量才会返回false：0，null，""，undefined或NaN，其他的都返回true。我们来看看这个简单的例子：

function Account(cash) {

this.cash = cash;

this.hasMoney = !!cash;

}

var account = new Account(100.50);

console.log(account.cash); // 100.50

console.log(account.hasMoney); // true

var emptyAccount = new Account(0);

console.log(emptyAccount.cash); // 0

console.log(emptyAccount.hasMoney); // false

在这个例子中，如果account.cash的值大于零，则account.hasMoney的值就是true。

**2) 使用+将变量转换成数字**

这个转换超级简单，但它只适用于数字字符串，不然就会返回NaN（**不是数字**）。看看这个例子：

function toNumber(strNumber) {

return +strNumber;

}

console.log(toNumber("1234")); // 1234

console.log(toNumber("ACB")); // NaN

这个转换操作也可以作用于Date，在这种情况下，它将返回时间戳：

console.log(+new Date()) // 1461288164385

**3) 短路条件**

如果你看到过这种类似的代码:

if (conected) {

login();

}

那么你可以在这两个变量之间使用&&（AND运算符）来缩短代码。例如，前面的代码可以缩减到一行：

conected && login();

你也可以用这种方法来检查对象中是否存在某些属性或函数。类似于以下代码：

user && user.login();

**4) 使用||设置默认值**

在ES6中有默认参数这个功能。为了在旧版浏览器中模拟此功能，你可以使用||（OR运算符），并把默认值作为它的第二个参数。如果第一个参数返回false，那么第二个参数将会被作为默认值返回。看下这个例子：

function User(name, age) {

this.name = name || "Oliver Queen";

this.age = age || 27;

}

var user1 = new User();

console.log(user1.name); // Oliver Queen

console.log(user1.age); // 27

var user2 = new User("Barry Allen", 25);

console.log(user2.name); // Barry Allen

console.log(user2.age); // 25

**5) 在循环中缓存array.length**

这个技巧非常简单，并且在循环处理大数组时能够避免对性能造成巨大的影响。基本上几乎每个人都是这样使用for来循环遍历一个数组的：

for (var i = 0; i < array.length; i++) {

console.log(array[i]);

}

如果你使用较小的数组，那还好，但是如果处理大数组，则此代码将在每个循环里重复计算数组的大小，这会产生一定的延迟。为了避免这种情况，你可以在变量中缓存array.length，以便在循环中每次都使用缓存来代替array.length：

var length = array.length;

for (var i = 0; i < length; i++) {

console.log(array[i]);

}

为了更简洁，可以这么写：

for (var i = 0, length = array.length; i < length; i++) {

console.log(array[i]);

}

**6) 检测对象中的属性**

当你需要检查某些属性是否存在，避免运行未定义的函数或属性时，这个技巧非常有用。如果你打算编写跨浏览器代码，你也可能会用到这个技术。例如，我们假设你需要编写与旧版Internet Explorer 6兼容的代码，并且想要使用document.querySelector()来通过ID获取某些元素。 但是，在现代浏览器中，这个函数不存在。所以，要检查这个函数是否存在，你可以使用in运算符。看下这个例子：

if ('querySelector' in document) {

document.querySelector("#id");

} else {

document.getElementById("id");

}

在这种情况下，如果在document中没有querySelector函数，它就会使用document.getElementById()作为代替。

**7) 获取数组的最后一个元素**

Array.prototype.slice（begin，end）可以用来裁剪数组。但是如果没有设置结束参数end的值的话，该函数会自动将end设置为数组长度值。我认为很少有人知道这个函数可以接受负值，如果你将begin设置一个负数的话，你就能从数组中获取到倒数的元素：

var array = [1, 2, 3, 4, 5, 6];

console.log(array.slice(-1)); // [6]

console.log(array.slice(-2)); // [5,6]

console.log(array.slice(-3)); // [4,5,6]

**8) 数组截断**

这个技术可以锁定数组的大小，这对于要删除数组中固定数量的元素是非常有用的。例如，如果你有一个包含10个元素的数组，但是你只想获得前五个元素，则可以通过设置array.length = 5来阶段数组。看下这个例子：

var array = [1, 2, 3, 4, 5, 6];

console.log(array.length); // 6

array.length = 3;

console.log(array.length); // 3

console.log(array); // [1,2,3]

**9) 全部替换**

String.replace()函数允许使用String和Regex来替换字符串，这个函数本身只能替换第一个匹配的串。但是你可以在正则表达式末尾添加/g来模拟replaceAll()函数：

var string = "john john";

console.log(string.replace(/hn/, "ana")); // "joana john"

console.log(string.replace(/hn/g, "ana")); // "joana joana"

**10) 合并数组**

如果你需要合并两个数组，你可以使用Array.concat()函数：

var array1 = [1, 2, 3];

var array2 = [4, 5, 6];

console.log(array1.concat(array2)); // [1,2,3,4,5,6];

但是，这个函数对于大数组来说不并合适，因为它将会创建一个新的数组并消耗大量的内存。在这种情况下，你可以使用Array.push.apply（arr1，arr2），它不会创建一个新数组，而是将第二个数组合并到第一个数组中，以减少内存使用：

var array1 = [1, 2, 3];

var array2 = [4, 5, 6];

console.log(array1.push.apply(array1, array2)); // [1,2,3,4,5,6];

**11) 把NodeList转换成数组**

如果运行document.querySelectorAll("p")函数，它会返回一个DOM元素数组，即NodeList对象。但是这个对象并没有一些属于数组的函数，例如：sort()，reduce()，map()，filter()。为了启用这些函数，以及数组的其他的原生函数，你需要将NodeList转换为数组。要进行转换，只需使用这个函数：[] .slice.call（elements）：

var elements = document.querySelectorAll("p"); // NodeList

var arrayElements = [].slice.call(elements); // 现在已经转换成数组了

var arrayElements = Array.from(elements); // 把NodeList转换成数组的另外一个方法

**12) 对数组元素进行洗牌**

如果要像外部库Lodash那样对数据元素重新洗牌，只需使用这个技巧：

var list = [1, 2, 3];

console.log(list.sort(function() {

return Math.random() - 0.5

})); // [2,1,3]

**结论**

现在，你已经学到了一些有用的JS技巧，它们主要用于缩减JavaScript代码量，其中一些技巧在许多流行的JS框架都使用到，如Lodash，Underscore.js，Strings.js等。如果你知道其他JS技巧，欢迎分享！