# 每日一题 - 如何正确判断一个字符串是否是Number

### 信息卡片

* 时间：2019-08-12
* tag：数值判断 isNumeric

### 问题描述

实现函数isNumeric,使得其能够正确判断以下各种情况:

function isNumeric(str) {  
 // your code goes here  
}  
  
console.assert(isNumeric('1000'));  
console.assert(isNumeric('-100.'));  
console.assert(isNumeric('.1'));  
console.assert(isNumeric('-3.2'));  
console.assert(isNumeric('001'));  
console.assert(isNumeric('+4.5'));  
console.assert(isNumeric('1e3'));  
console.assert(isNumeric('1e-3'));  
console.assert(isNumeric('-100e-3'));  
  
console.assert(!isNumeric('++3'));  
console.assert(!isNumeric('-100..'));  
console.assert(!isNumeric('3abc'));  
console.assert(!isNumeric('abc'));  
console.assert(!isNumeric('-3e3.2'));  
console.assert(!isNumeric('Infinity'));  
console.assert(!isNumeric('-Infinity'));  
console.assert(!isNumeric(''));

### 参考实现

###### 方案一

最简单的是直接考虑所有需要过滤的情况，使用原生Number()方法转化判断

function isNumeric(str) {  
 // 剔除空的情况，因为会被转化为数值0  
 if (str === '' || str == null) return false;  
 // 直接将字符串转为数值类型  
 const num = Number(str);  
 /\*   
 最后还需要进行无穷值的过滤  
 如果传入的值是引用类型也没关系，因为Number()会把引用类型转化为NaN，  
 NaN与Infinity是可以比较的，效果可判断出NaN不是Number类型，只是这样写可能不是很直观  
 \*/  
 return num < Infinity && num > -Infinity;  
}

可以输入各种类型判断

###### 方案二

类似方案一的过滤实现

function isNumeric(str) {  
 // 直接过滤掉空和非字符串的类型，也就是同时将引用类型剔除  
 if (typeof str !== 'string' || str === '') {  
 return false;  
 }  
 // 将可转化的字符串转化为数值  
 const num = Number(str);  
 // 存在无穷值的情况，判断过滤  
 return num < Infinity && num > -Infinity;  
}

可以输入各种类型判断，可惜过滤掉了字符类型，当然也符合本题需求，只是方案一完善点

###### 方案三 - 穷举

function isNumeric(str) {  
 // 过滤掉空字符的情况  
 if (str === '') return false;  
  
 // 穷举所有需要过滤的情况  
 const specialStr = ['Infinity', 'undefined', 'null'];  
 if (specialStr.some(s => str.indexOf(s) !== -1)) return false;  
  
 // 将过滤后的转化为数值类型  
 const n = Number(str);  
 // 如果值等于自身，也就是过滤掉了NaN  
 return n === n;  
}

这种穷举是没问题的，可以解决本题需求，判断的是string类型的输入；但是没有兼容考虑其他类型的输入，也就是indexOf()会报错，以及穷举需要考虑所以情况，要堵住所有漏洞，关于优化，好像要改动的有点多。。。

###### 方案四 - parseFloat + isFinite

function isNumeric(obj) {  
 /\*   
 parseFloat 是全局函数，可以直接调用，它可以过滤出所有非可转化的数组类型，判断值的第一个字符，如果不是数值类型就返回NaN，是就继续(注：Infinity无穷大也是数值类型)  
 isFinite检测无穷大，如果是就返回false  
 \*/  
 return !isNaN(parseFloat(obj)) && isFinite(obj);  
}

这两个函数就覆盖了所有类型，可以判断出所有类型的输入是否是数值类型

### 总结

* 推荐使用方案一和方案四