## 每日一题 - JS 中 Number 类型的可以表示的范围是多少

### 信息卡片

* 时间：2019-08-01
* tag：Number 二进制 精度

### 题目描述

JS中Number是双精度浮点型， 意味着可以表示的范围是2^63次方么？如果不是的话，应该是多少呢？

### 思路

JavaScript 的 Number 类型为 IEEE 754 64 位浮点类型。 最近出了 stage3 BigInt 任意精度数字类型，已经进入 stage3 规范。

JavaScript 的 Number 类型使用 52 位表示小数位，11 位表示指数位，1 位表示符号位。 因此指数部分最大值为2^11 - 1 = 2047。当指数部分全部是1的时候，实际上表示的数字是NaN,Infinite或者-Infinite。

因此最大值为2^1023 \* (1 \* 2^0 + 1 \* 2^-1 + ... + 1 \* 2-52)，也就是2971 \* (2^53 - 1)，这个值就是javascript能表示的最大数字1.7976931348623157e+308

这个数字在计算器中是打印不出来的， 至于原因，大家自己想一下。

这个数字非常接近2^1024 ,其实我们可以稍微估算一下2 ^ 1024的值。

log(2^1024) = 1024\*log(2) = 1024 \* 0.30102999566398114 = 308.2547155599167

因此大概范围应该是 10^308

对应地，整数的表示范围就是-2^53 到 2^53 之间（不含两个端点）.

### 参考代码

答案是1.7976931348623157e+308， 其实也就是Number.MAX\_VALUE的值。

var biggestInt = Number.MAX\_VALUE;  
// 1.7976931348623157e+308

如果是整数的范围的话，JavaScript 能够准确表示的整数范围在-2^53 到 2^53 之间（不含两个端点）， 超过这个范围，无法精确表示这个整数。 (详情请参阅 [ECMAScript standard, chapter 6.1.6 The Number Type](https://www.ecma-international.org/ecma-262/#sec-ecmascript-language-types-number-type)):

var biggestInt = Number.MAX\_SAFE\_INTEGER;  
//9007199254740991  
var smallestInt = Number.MIN\_SAFE\_INTEGER;  
//-9007199254740991

在解析序列化的 JSON 时，如果 JSON 解析器将它们强制转换为 Number 类型，那么超出此范围的整数值可能会被破坏。在工作中使用 String 类型代替，是一个可行的解决方案。