**关注”千锋互联”微信公众号( ID: qianfengjiaoyu ),更多学习资源(电子书/面试题/简历模板/视频/公开课等等)与你分享**



### 了解NaN与typeof

深入阐述NaN属性以及它为何被视为一种number类型

首先，*NaN*不是关键字（和*true*，*false*，*null*等不同），**它是全局对象上的一个属性**。*NaN*的值和*Number.NaN*的值相同：

NaN; // NaN

Number.NaN; // NaN

以下几种操作会得到*NaN*：

1. 0除以0
2. 无穷大除以无穷大
3. 无穷大乘以0
4. 任何把NaN当做运算对象的操作
5. 把非数值字符串或*undefined*转换为数字

为何typeof NaN操作返回“number”呢？

typeof NaN; // "number"

ECMAScript标准规定Number必须是IEEE-754浮点数。这包括了*Infinity*，*-Infinity*以及*NaN*。

按照定义，运行结果为一个**未定义数值**的操作会返回*NaN*。这就是为什么在JavaScript中除了作为全局对象的一个属性之外，*NaN*还是*Number*对象的一个部分：*Number.NaN*。**它依旧是一种数值类型**，只不过是作为一个未被定义的[实数](http://en.wikipedia.org/wiki/Real_number)而已。

*NaN*还被用来表示任何未被[ECMAScript规范包括](http://www.ecma-international.org/ecma-262/5.1/#sec-8.5)的数值。

**计算机的运算是受限的**

请看以下运算:

(3.2317006071311 \* 10e616) / (3.2317006071311 \* 10e616); // NaN

如[Wikipedia](http://en.wikipedia.org/wiki/Real_number)上说的那样：

计算机的运算并不能直接操作实数，而是只能操作一个有理数的有限子集，子集的大小取决于用于存储这些数值时使用的比特位的多少。

在普通的运算中，3.2317006071311 \* 10^616是一个**有限实数**，但在ECMAScript规范中，它实在**太大**（比*Number.MAX\_VALUE*还要大）了，因此只能使用**Infinity**表示。当用无穷大除以无穷大时也会得到*NaN*。当然，在普通的运算中，由于两个操作数都是无穷大，因此运算的结果显然应该等于1。

在这个例子中，NaN用来表示一个由于运算数太大而不能计算出的实数（这里是1）。因此如果typeof *NaN*的返回值不是“number”的话就有点反直觉了。毕竟在这个例子中，NaN只是用来表示一个无法由计算机运算得出的数值。

（注意: 3.2317006071311 \* 10^616这个值是*Number.MAX\_VALUE*的平方）

**NaN是无序的**

根据IEEE 754浮点数标准，与*NaN*的比较应该始终返回一个无序的结果。这意味着，*NaN*不等于，大于，或小于任何值，**包括它自己**：

NaN < 1; // false

NaN > 1; // false

NaN == NaN; // false

// 但我们还是可以检测NaN:

isNaN(NaN); // true

这就是为何你不能通过与*NaN*比较来判断某个值是否是*NaN*的原因，替代方案就是使用*isNaN()*方法。

当然，我们也可以把原生的*isNaN*函数替换为以下的代码：

// 原生实现

function isNaN(x) {

// 强制转换成数字

x = Number(x);

// 如果x是NaN，那么NaN != NaN会返回true，否则返回false

return x != x;

}

在*isNaN*的原生实现中，即使在传入的值是undefined或者不能被转换为数值类型的情况下，该函数也会返回true。

当然，我不建议你覆盖原生实现。不过，某些库中包含了它们自己的实现。比如，[Underscore](http://underscorejs.org/)的实现方式是这样的：

\_.isNaN = function(obj) {

// NaN是唯一和自身进行===比较返回false的值

return obj !== obj;

};

但是，它的行为和原生的*isNaN()*并不一致：

var x; // undefined

isNaN(x); // true

isNaN(undefined); // true

isNaN("a"); // true

Underscore的版本:

var x; // undefined

\_.isNaN(x); // false

\_.isNaN(undefined); // false

\_.isNaN("a"); // false

我不确定，但我猜Underscore的意图是要确定某个值的确是*NaN*，因为满足该函数检查条件的值只有1个，就是*NaN*。

**布尔值不是NaN**

请看以下代码：

isNaN(true); // false

isNaN(false); // false

这是因为布尔值实际是用一个比特位的数值类型来实现的，因此在比较时它们会被转化为对应的数值：

Number(true); // 1

Number(false); // 0

**关注”千锋互联”微信公众号( ID: qianfengjiaoyu ),更多学习资源(电子书/面试题/简历模板/视频/公开课等等)与你分享**

