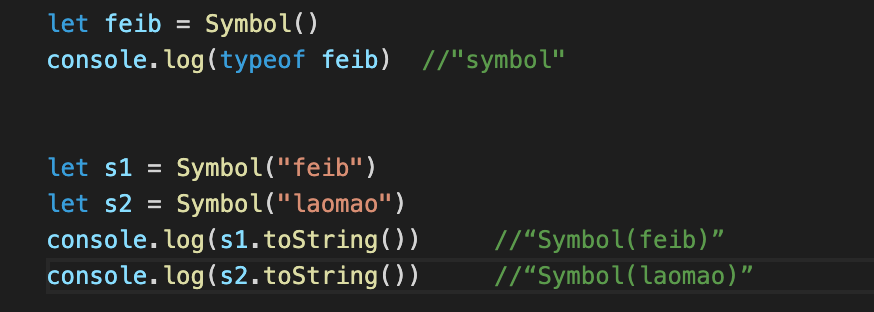
# ES6 Symbol

## 1、概述

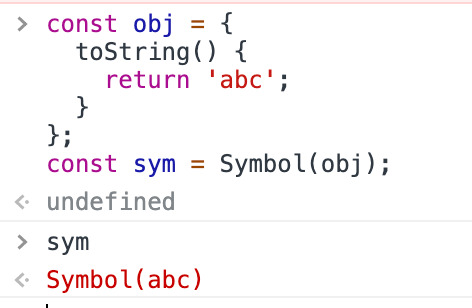
原始数据类型Symbol为了解决对象属性名冲突而提出。

Symbol函数前不能使用new命令，否则会报错。因为创建出来的Symbol是原始数据类型，不能是对象。

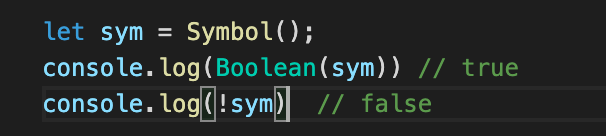
Symbol函数可以接受一个字符串作为参数，为了控制台显示或转为字符串时方便分别，否则控制台输出都只显示Symbol()



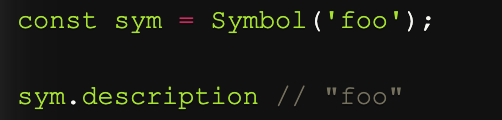
如果 Symbol 的参数是一个对象，就会调用该对象的toString方法，将其转为字符串，然后才生成一个 Symbol 值。



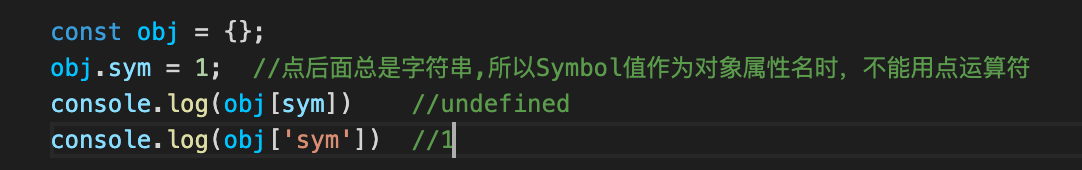
Symbol不仅可以转为字符串，还可以转为Boolean



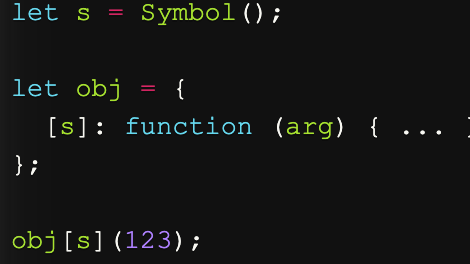
## 2、Symbol.prototype.description（ES2019才有)



## 3、作为属性名的 Symbol



在对象的内部，使用 Symbol 值定义属性时，Symbol 值必须放在方括号之中。如果不放在[]中，就变成字符串“s”了。



上面的增强写法



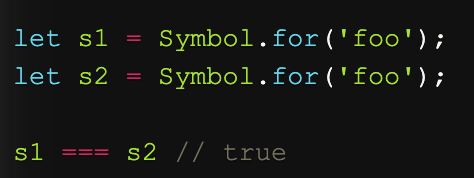
### 4、消除魔术字符串

### 5、Symbol作为对象属性名的遍历

symbol无法被for in、for of遍历，也不会被Object.keys()、Object.getOwnPropertyNames()和JSON.stringify()返回。

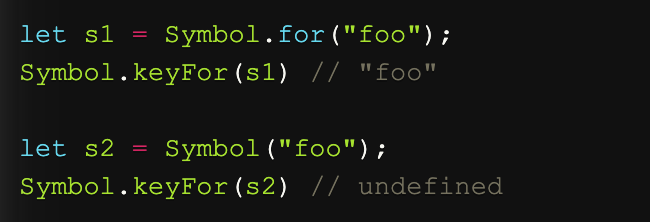
而Object.getOwnPropertySymbols(obj)返回一个数组，成员为对象的所有symbol属性名。而Reflect.ownKeys()返回的数组，既有symbol属性名，又有常规属性名。

### 6、Symbol.for()&Symbol.keyFor()



Symbol.for()和Symbol()不同的是，Symbol.for会先检查给定的key是否存在（Symbol是否已登记），如果存在则返回原来的Symbol，不存在则返回一个新的Symbol。

Symbol.keyFor()获取已登记的Symbol的key



Symbol.for()有全局登记特性，函数内部的Symbol.for也是登记在全局环境的

### 8、内置的Symbol值

### Symbol.hasInstance

A instanceof B时，实际调用的是B[Symbol.hasInstance](A)。

### Symbol.isConcatSpreadable

数组Concat时是否可以展开

### Symbol.species [§](https://es6.ruanyifeng.com/#docs/symbol#Symbol-species) [⇧](https://es6.ruanyifeng.com/#docs/symbol)

Symbol.species的作用在于，实例对象在运行过程中，需要再次调用自身的构造函数(constructor)时，会调用该属性指定的构造函数。它主要的用途是，有些类库是在基类的基础上修改的，那么子类使用继承的方法时，作者可能希望返回基类的实例，而不是子类的实例。