

Object.seal()

概述

Object.seal() 方法可以让一个对象密封，并返回被密封后的对象。密封对象将会阻止向对象添加新的属性，并且会将所有已有属性的可配置性（**configurable**）置为不可配置（**false**），即不可修改属性的描述或删除属性。但是可写性描述（**writable**）为可写（**true**）的属性的值仍然被修改。

语法

```
Object.seal(obj)
```

参数

obj

将要被密封的对象

描述

通常情况下，一个对象是**可扩展的**（可以添加新的属性）。密封一个对象会让这个对象变的不能添加新属性，且所有已有属性会变的不可配置。属性不可配置的效果就是属性变的不可删除，以及一个数据属性不能被重新定义成为访问器属性，或者反之。但属性的值仍然可以修改。尝试删除一个密封对象的属性或者将某个密封对象的属性从数据属性转换成访问器属性，结果会静默失败或抛出

TypeError 异常（**严格模式**）。



不会影响从原型链上继承的属性。但 **__proto__** () 属性的值也会不能修改。

例子

```
1  var obj = {  
2      prop: function () {},  
3      foo: "bar"  
4  };  
5  
6  // 可以添加新的属性, 已有属性的值可以修改, 可以删除  
7  obj.foo = "baz";
```

```
8  obj.lumpy = "woof";
9  delete obj.prop;
10
11  var o = Object.seal(obj);
12
13  assert(o === obj);
14  assert(Object.isSealed(obj) === true);
15
16  // 仍然可以修改密封对象上的属性的值。
17  obj.foo = "quux";
18
19  // 但你不能把一个数据属性重定义成访问器属性。
20  Object.defineProperty(obj, "foo", { get: function() { return "g"; } });
21
22  // 现在,任何属性值以外的修改操作都会失败。
23  obj.quaxxor = "the friendly duck"; // 静默失败,新属性没有成功添加
24  delete obj.foo; // 静默失败,属性没有删除成功
25
26  // ...在严格模式中,会抛出TypeError异常
27  function fail() {
28    "use strict";
29    delete obj.foo; // 抛出TypeError异常
30    obj.sparky = "arf"; // 抛出TypeError异常
31  }
32  fail();
33
34  // 使用Object.defineProperty方法同样会抛出异常
35  Object.defineProperty(obj, "ohai", { value: 17 }); // 抛出TypeError异常
36  Object.defineProperty(obj, "foo", { value: "eit" }); // 成功将原有值改变
```

规范

Specification	Status	Comment
ECMAScript 5.1 (ECMA-262) Object.seal	 ST Standard	Initial definition. Implemented in JavaScript 1.8.5
ECMAScript 2015 (6th Edition, ECMA-262) Object.seal	 ST Standard	

浏览器兼容性

	Desktop		Mobile		
Feature	Firefox (Gecko)	Chrome	Internet Explorer	Opera	Safari
Basic support	4 (2.0)	6	9	未实现	5.1

相关链接

- [Object.isSealed](#)
- [Object.isFrozen](#)
- [Object.isExtensible](#)
- [Object.preventExtensions](#)
- [Object.freeze](#)