**第十六章 JS中的this**

**JavaScript语言中的this和其他语言中区别**

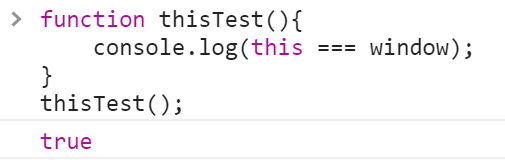
* **Java等面向对象语言中的this**在 Java 等面向对象的语言中，this 关键字的含义  
  是明确且具体的，即指代当前对象，一般在编译期  
  就已经确定下来，称为编译期绑定
* **JavaScript语言中的this**在 JavaScript 中，this 是动态绑定，或称为运行期绑定的。由于其运行期绑定的特性，JavaScript 中的 this 含义要丰富得多，它可以是全局对象、当前对象或者任意对象，这完全取决于函数的调用方式
* this不进行作用域传递（函数嵌套时的this缺陷）

**JavaScript语言中的this主要有以下4种应用场景**

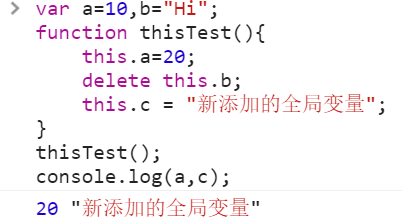
* 一般函数中的this
* 对象方法中的this
* 构造函数中的this
* 间接调用中的this

不管哪种场景下调用，this指向的都是调用此函数的主体

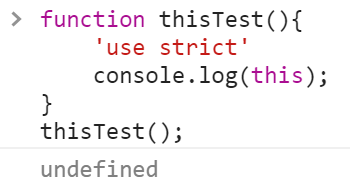
**一般函数中的this（非严格松散模式下）指代全局对象**



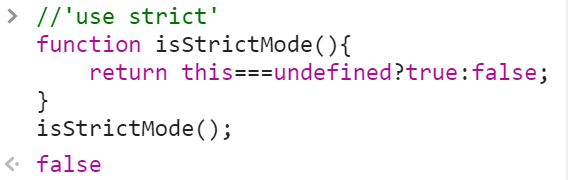
**可以通过this在函数内添加、删除、修改全局对象属性**



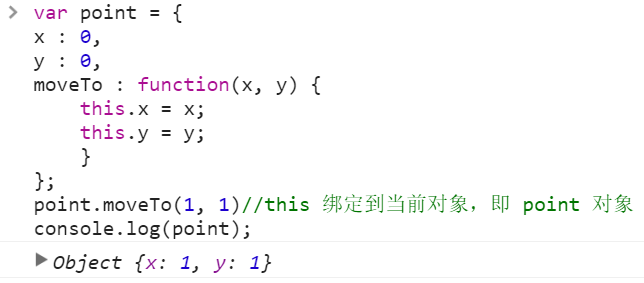
**一般函数中的this（严格模式）为undefined**



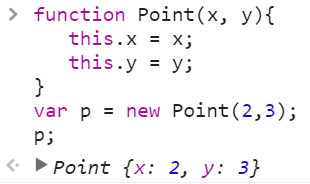
可以用此特性来判断当前是否为严格模式



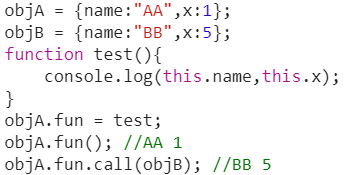
* 函数作为对象的一个属性时，称之为对象的方法
* 对象方法中的this指代调用此方法的对象（无嵌套的情况下）



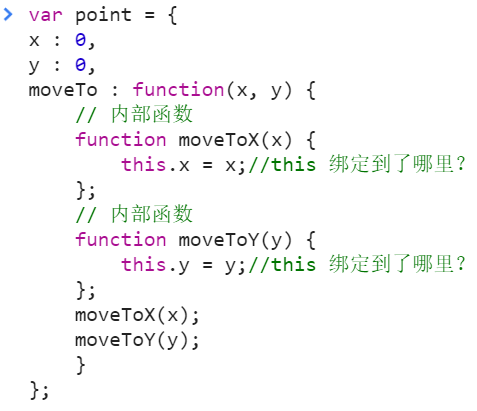
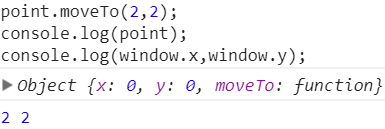
**构造函数中的this指代通过new新创建的对象**- JS（ES5） 并没有类（class）的概念，而是使用基于原型（prototype）的继承方式  
- JS中的构造函数充当了类的角色，如果不使用 new 调用，则和普通函数一样。  
- 如果作为构造函数正确调用时，构造函数中的this 绑定到新创建的对象上



**通过call/applay间接调用的函数中的this（指代第一个参数）**- JS中函数也是对象（函数对象），也有属性和方法（length、call、apply等）  
- JS中函数可以通过call和apply进行间接调用，动态的指定由谁来调用此函数



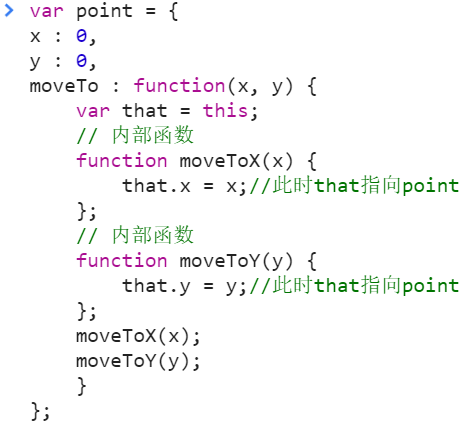
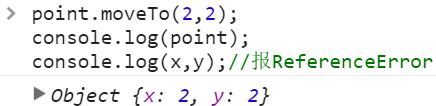
this不进行作用域传递、函数嵌套中的this存在缺陷

moveTo方法中嵌套的moveToX函数作为一般函数看待，此函数中的this此时指向的是全局对象window而不是point

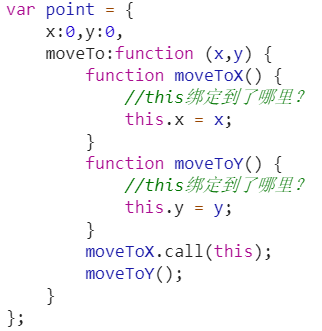
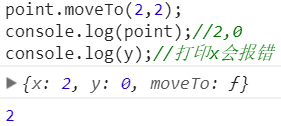
**如何解决对象方法中嵌套函数的this指向问题**

**方法一：使用变量（that或self）软绑定，使this指向正确**

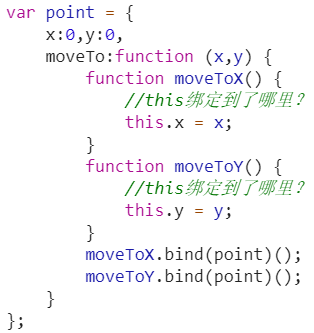
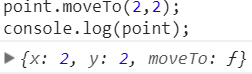
  moveTo方法中嵌套的函数中的that此时指向的是point对象。

这是软绑定形式，除此之外，还可以通过函数对象的bind方法进行硬绑定

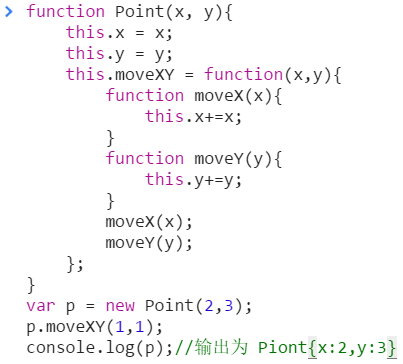
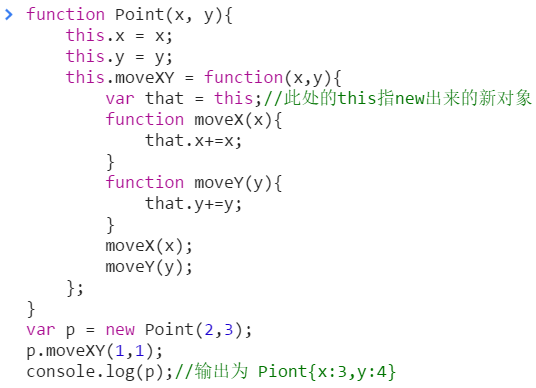
**方法二：使用call/apply间接调用，使this指向正确**

方法三：使用Function.prototype.bind，使this指向正确

**构造函数中的this同样存在函数嵌套缺陷，解决办法同上**

* **函数对象的内部方法 [[call]]  
  - 函数对象有一个叫[[Call]]内部方法，函数的执行其实是通过[[Call]]方法来执行的  
  - [[Call]]方法接收两个参数thisArg和argumentList  
  - thisArg和this的指向有直接关系，argumentList为函数的实参列表**
* **thisArg与4种情况的对应关系  
  - 普通方法调用thisArg为undefined。  
  - 通过call或apply调用，thisArg既为第一个参数。  
  - 通过对象调用，thisArg指向该对象。  
  - 在构造方法中，thisArg为新构造的对象**

**总原则：this指的是调用函数的那个主体**