1. 什么是面向对象

可以这样的理解，面向对象就是把所有事物都进行了对象化。我们只需要对这个对象进行操作，就可以实现需要的功能。

1. 面向对象的三大特性
2. 封装：可以认为把某些属性与方法放到一起才会变成一个对象。
3. 继承：所有事物都是有祖辈的，只要把祖辈的东西继承下来再生成新的东西才事物才会有发展。
4. 多态：对象可以在不同的情况下实现不一样的功能。



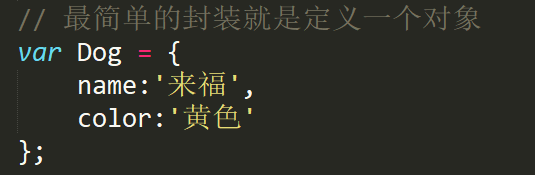
1. JS中的面向对象

JS是一个原生（基于）对象的程序语言。但因为在程序设计之初就没有打算把它做成非常大的语言，所以到SE6之前都没有真正意义上的面向对象编程方法。

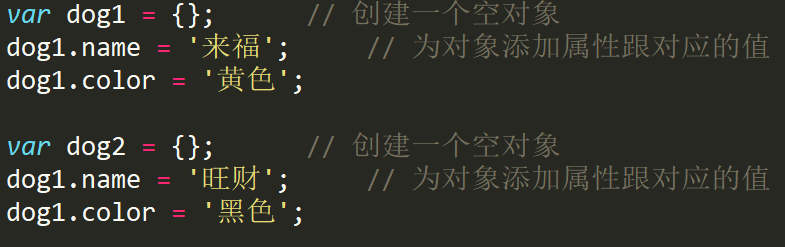
不过随着时代的发展，现在JS是可以使用面向对象来进行编程了，但严格来说只能是伪面向对象。

1. 封装JS对象

1、最简单的封装就是定义一个对象的数据类型的值。



也可以这样写。



上面的最简单定义（封装）对象的方法，但如果我们需要定义多个这样的对象就比较麻烦。所以我们可使用函数来完成这个事情。

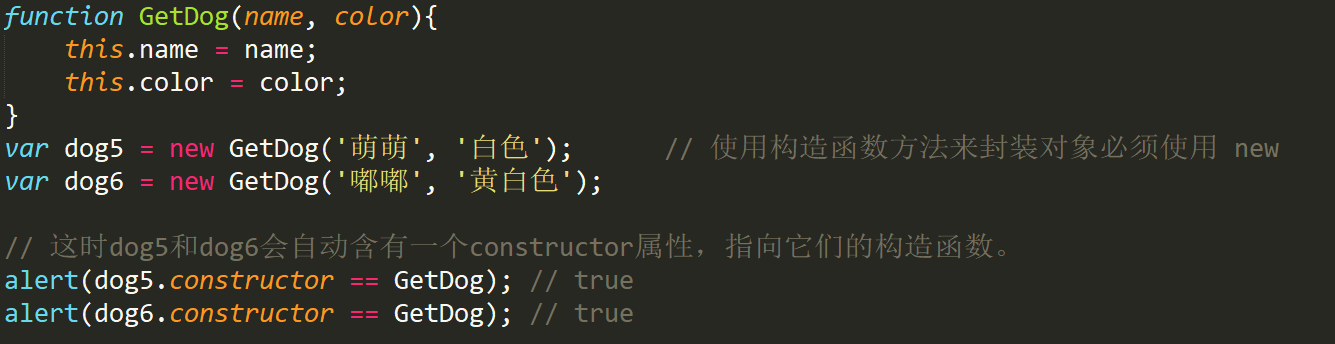


1. 使用构造函数来封装对象

dog3和dog4之间没有内在的联系，不能反映出它们是同一个原型对象的实例。

为了解决从原型对象生成实例的问题，Javascript提供了一个构造函数（Constructor）模式。

所谓"构造函数"，其实就是一个普通函数，但是内部使用了[this变量](http://www.ruanyifeng.com/blog/2010/04/using_this_keyword_in_javascript.html" \t "http://www.ruanyifeng.com/blog/2010/05/_blank)。对构造函数使用new运算符，就能生成实例，并且this变量会绑定在实例对象上。



1. 使用Prototype模式来封装对象

先看看我们改一下的构造函数的例子，如图：

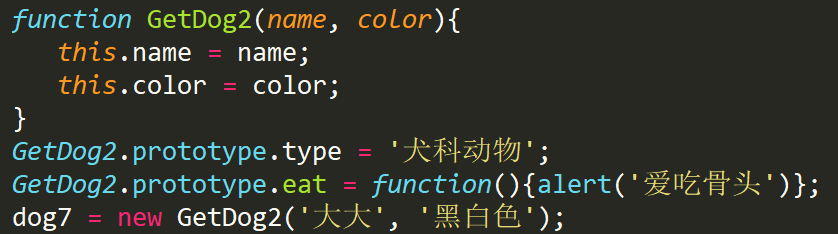


我们在构造函数中加入了一个固定的属性与固定的方法。

看起来没有问题，但实际上dog5跟dog6的type跟eat()是一样的，在new的时候会同 样的生成个一样的内容。

所以是一种内存的浪费与效率的低下。

所以我们可以使用prototype来做样的事情。



Javascript规定，每一个构造函数都有一个prototype属性，指向另一个对象。这个对象 的所有属性和方法，都会被构造函数的实例继承。

这意味着，我们可以把那些不变的属性和方法，直接定义在prototype对象上。

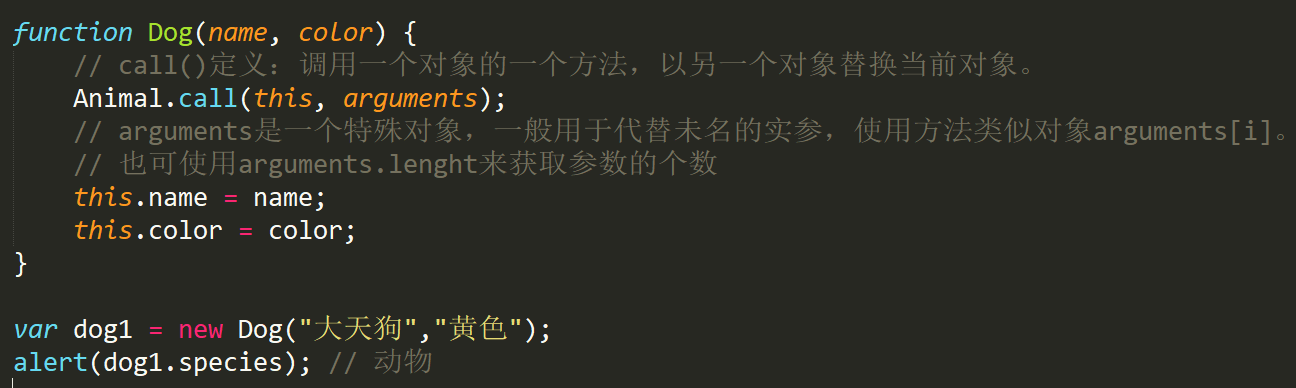
辅助用的内置方法：

isPrototypeOf()：这个方法用来判断，某个proptotype对象和某个实例之间的关系

hasOwnProperty()：用来判断某一个属性到底是本地属性，还是继承自prototype对象的属性。

in:运算符可以用来判断，某个实例是否含有某个属性，不管是不是本地属性。也可以用来遍历对象属性。

1. 对象的继承
2. 构造函数的继承
   1. 使用call或者apply来的进行继承



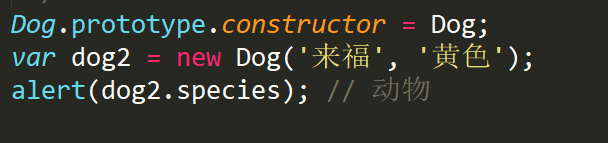
call()定义：调用一个对象的一个方法，以另一个对象替换当前对象。

arguments是一个特殊对象，一般用于代替未名的实参，使用方法类似对象arguments[i]。

也可使用arguments.lenght来获取参数的个数

b) 使用prototype属性来完成





任何一个prototype对象都有一个constructor属性，指向它的构造函数。

因此，在运行"Dog.prototype = new Animal();"这一行之后，dog2.constructor也指向Animal！

因此我们必须手动纠正，将Dog.prototype对象的constructor值改为Dog

c) 直接继承prototype

1. 非构造函数的继承