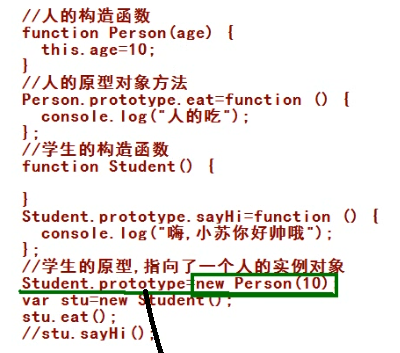
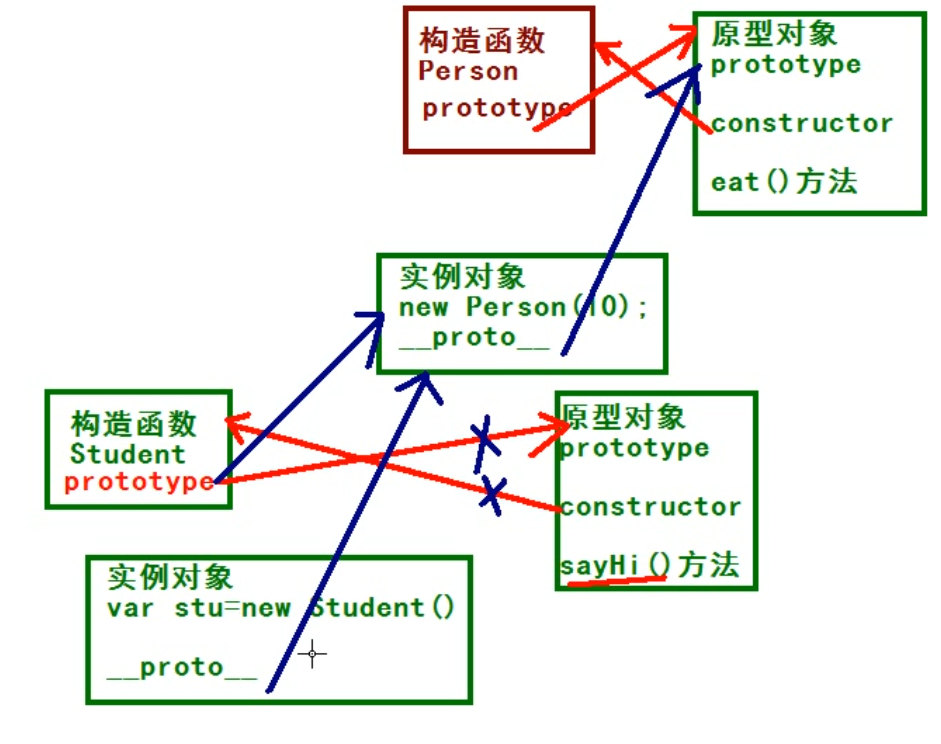
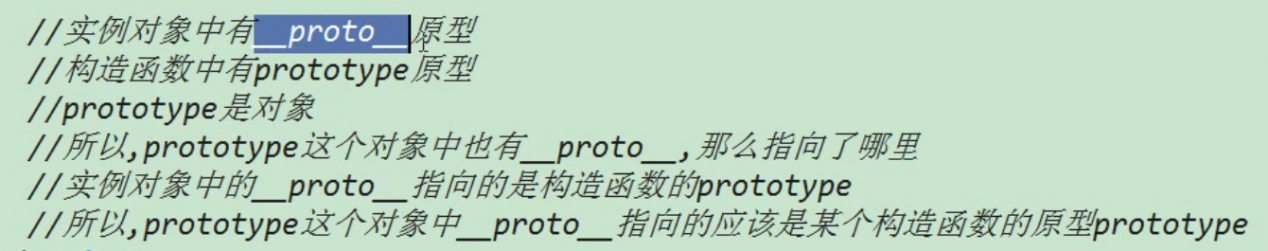
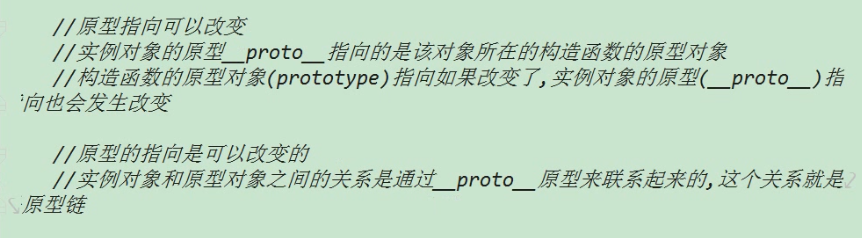
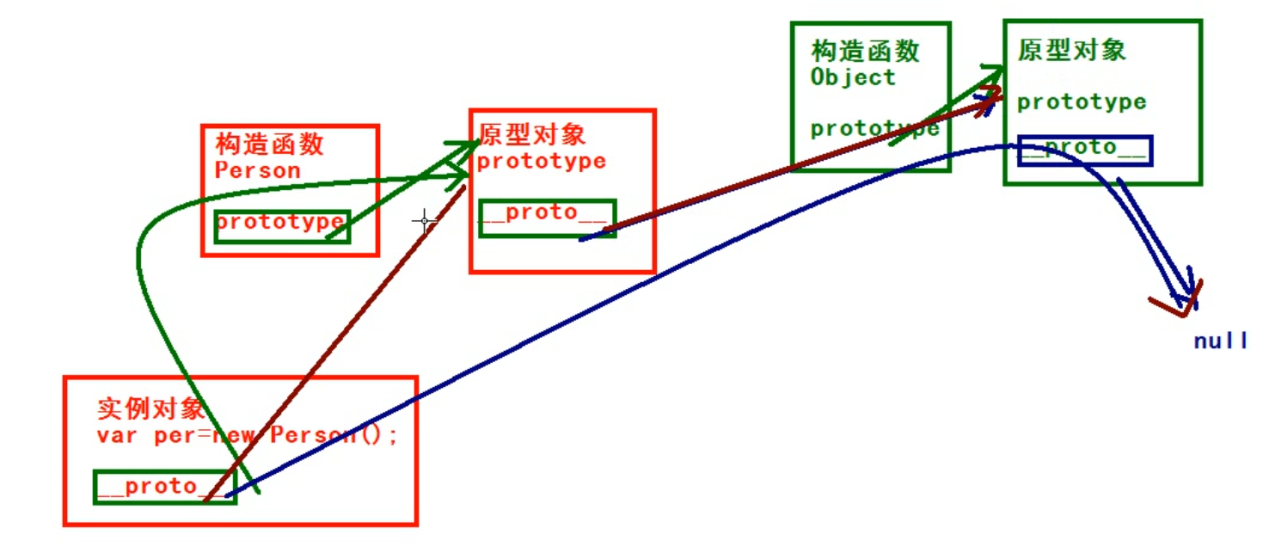
## 继承

原型发生改变是不造成隔代影响的：就是说你把爸爸的原型指向改成姥爷，然后把用爸爸创造儿子，然后改变爷爷的原型方法是不会对儿子造成影响的。毕竟你改指向的方法是创建爷爷的实例对象，就是说要改就要改在前面



就是说刚才的将student的原型指向修改时，是将prototype的指向改变成实例对象的\_proto\_

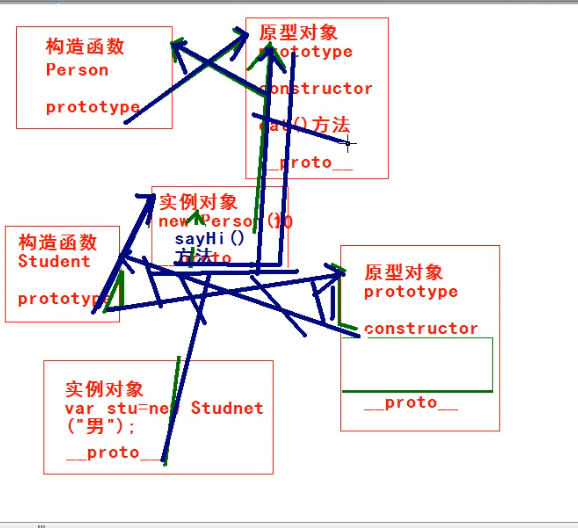


所以要在原型指向改变之后，添加方法。

## 同类型的问题。{}也是一个对象。



后添加的函数在实例对象（上级对象、上级原型），也就是说在实例对象的原型对向里是没有这个函数的。

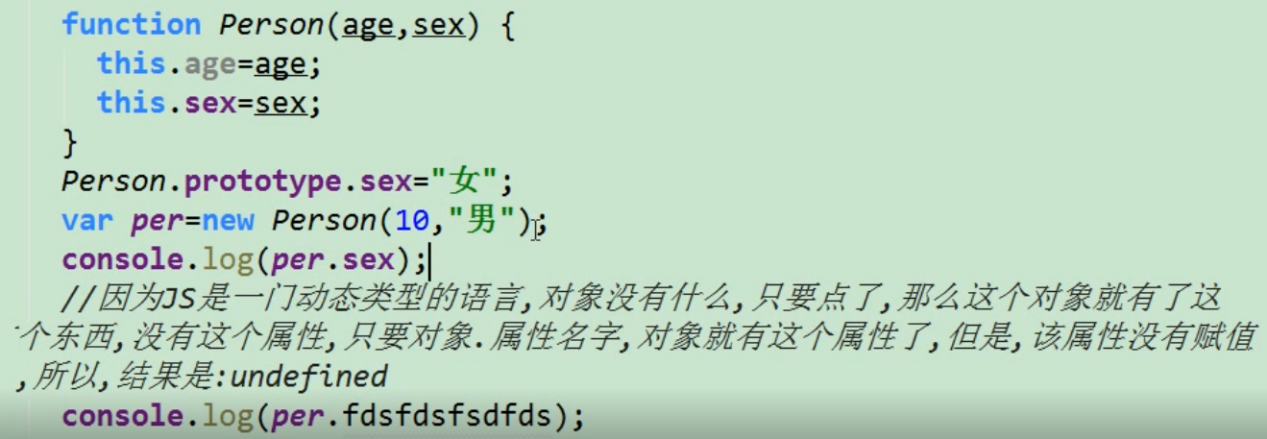
、

## 属性值重名问题

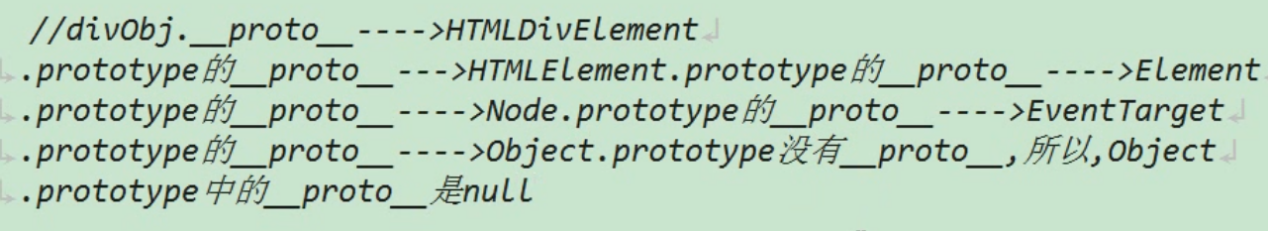


你改变实例对象的值理所当然的不会对原型里的属性值造成影响。

## Js是一门动态的语言的体现



原型链：



## Js的三大属性

## 

## 1.封装

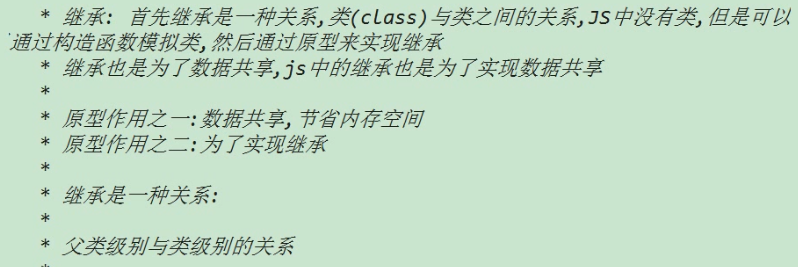
## 

## 2.多态



这里因为内存空间冗余问题所以js不会使用多态性。

## 3.继承



就是说当父类和子类成包含关系时，可以使用原型的方法实现继承，最终目的就是节省空间。

## 上面讲了那么多关于改变原型指向，以及属性重名的问题其实就是为了引出下面的继承。

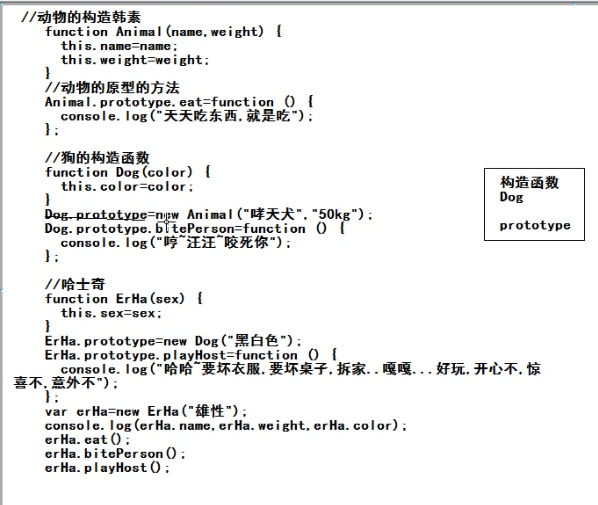


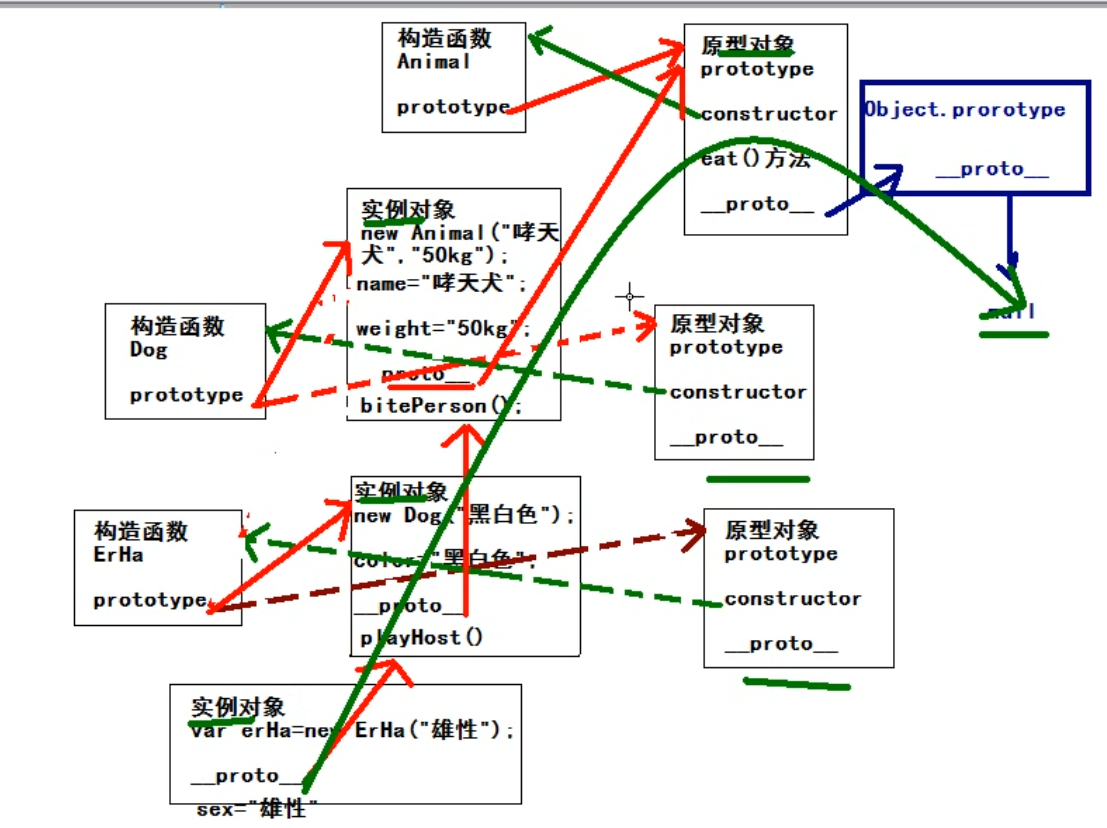
改变了student的原型指向之后，student的原型变成了person的实例对象，这个实例对象是有他的构造函数而来，所以有各种原型属性，而且还有原型方法，又因为上面的属性重名的道理，student原有的属性被很好的保留，而添加方法的方法就像上面的先改变指向再添加方法。

实例



这里的金丝猴就继承了猴子，动物的属性和方法。

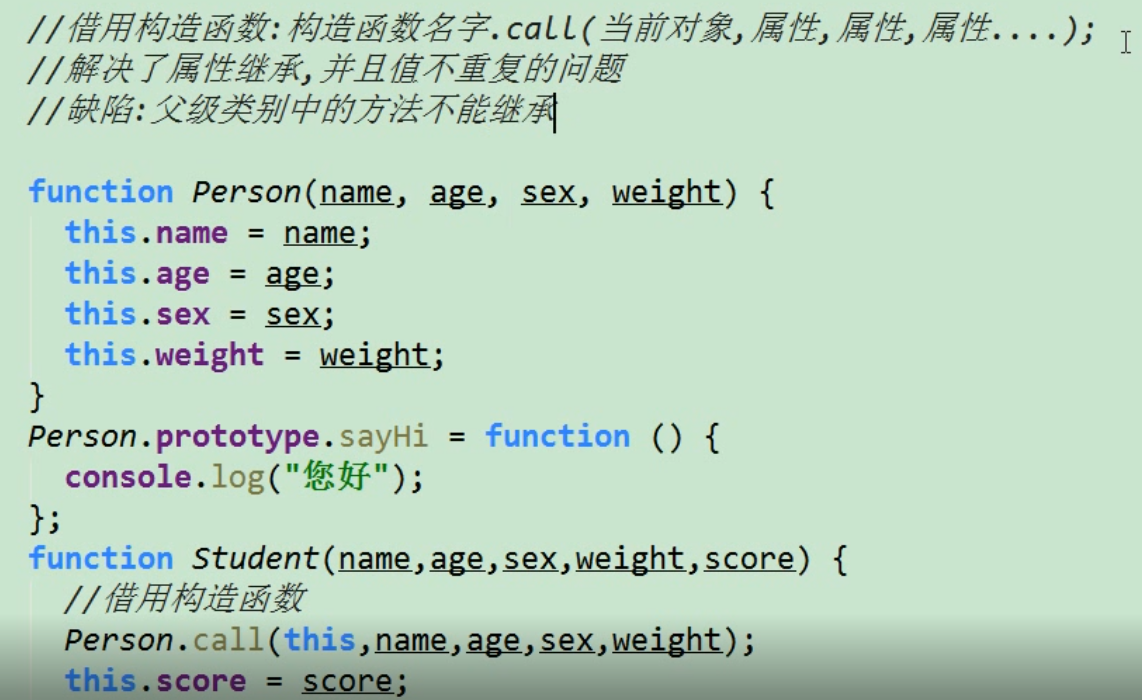




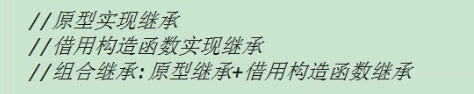
属性继承（借用继承）

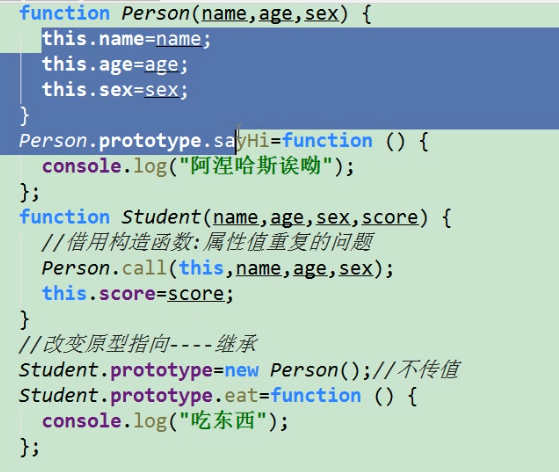
于是使用借用构造函数的方式继承值不同属性。

但是不能继承方法。

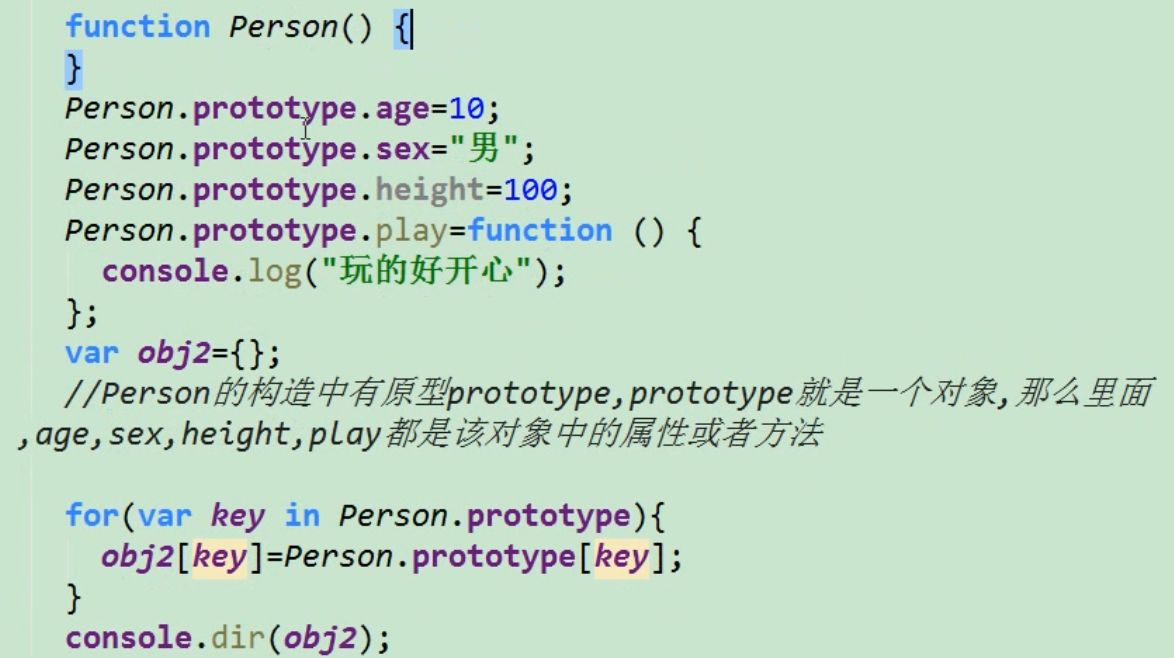


组合继承

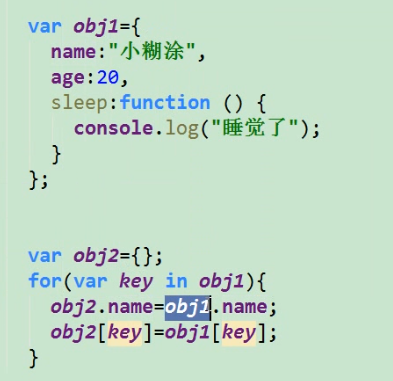


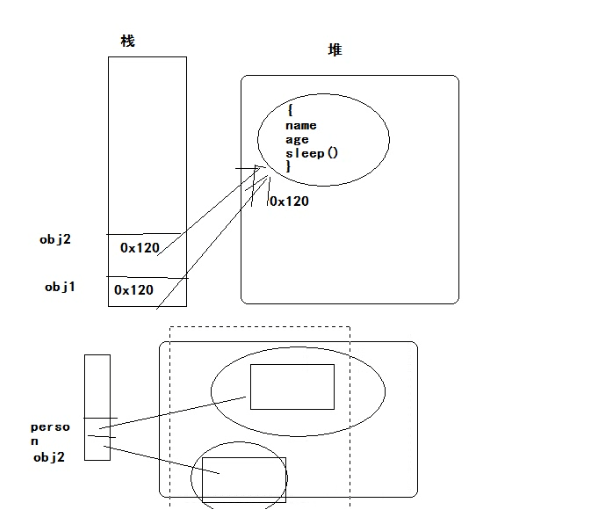


## 拷贝继承

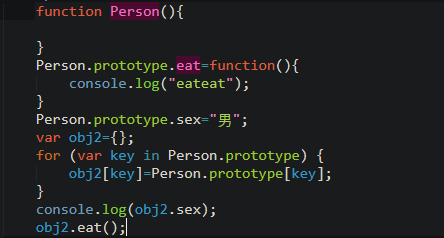


Js是一门动态的语言obj2没有那些属性，点.了就有了，再附上person的值就是拷贝继承。

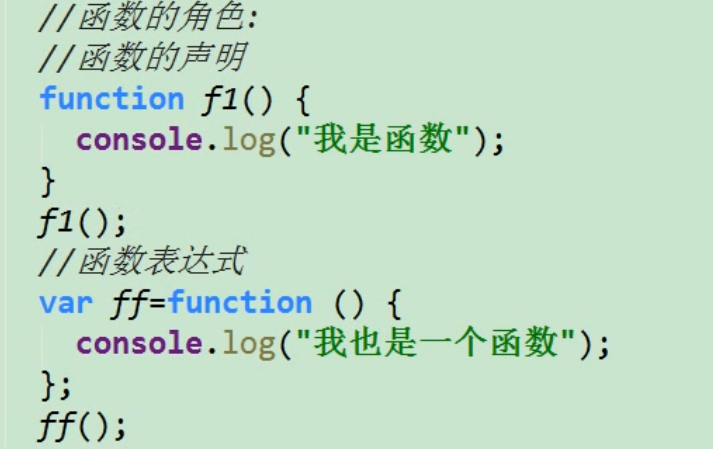
这种拷贝的方式叫做浅拷贝，而下面直接改地址指向的，不常用也不知道叫啥拷贝。



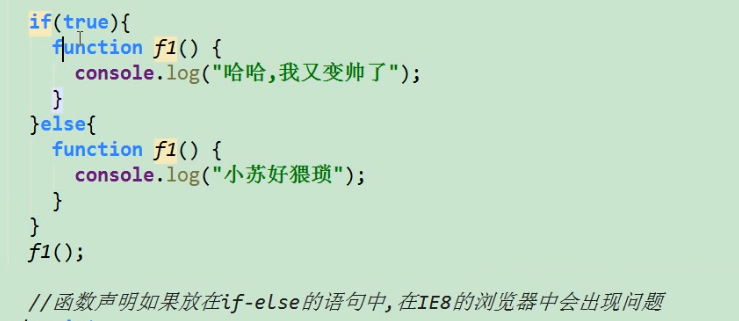
还有一种深拷贝，没学。



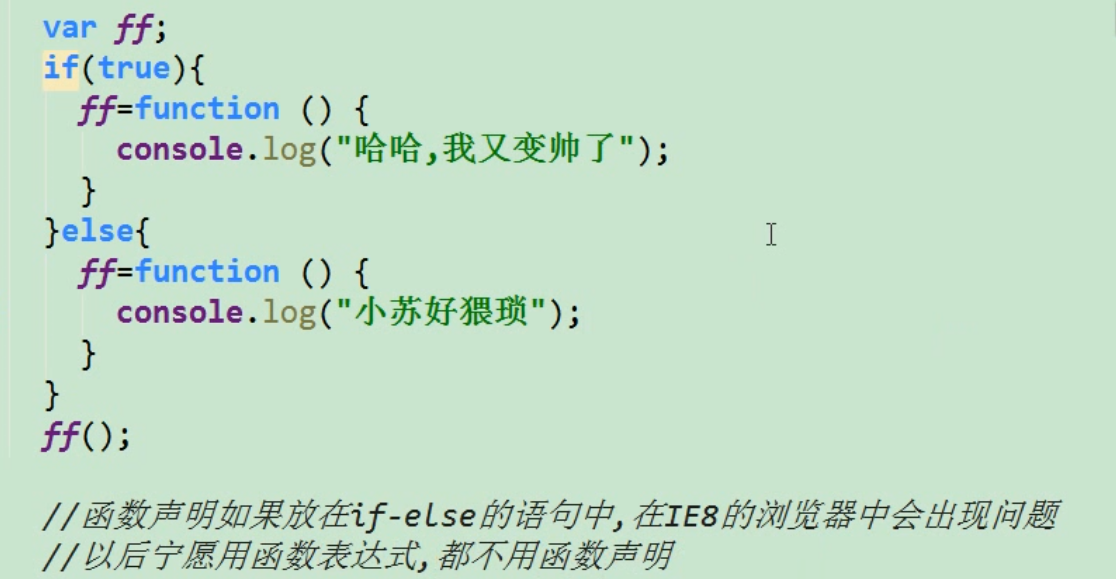
## 函数声明和表达式的区别



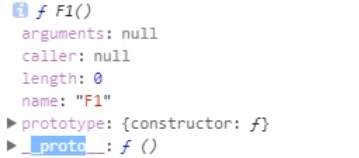
就是说函数的声明会先预解析，预解析解析的时候是不会执行if语句的判断的。

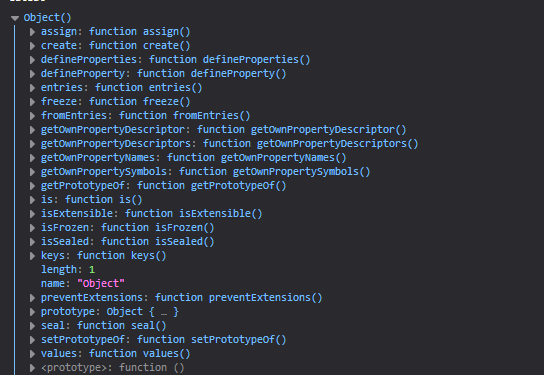


所以我们采用函数的表达式的方式来声明函数。

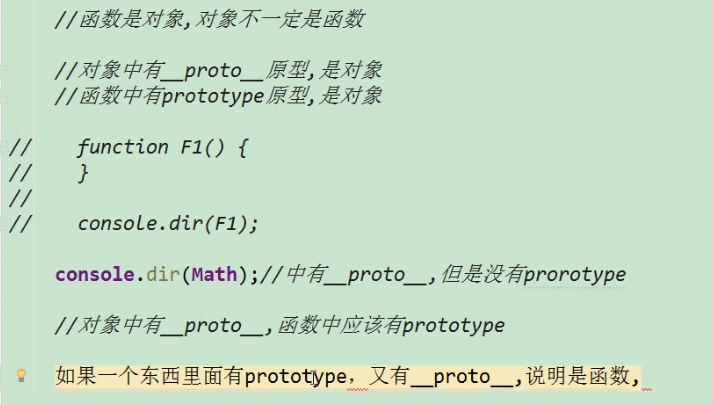


## 函数一定是对象，对象不一定是函数

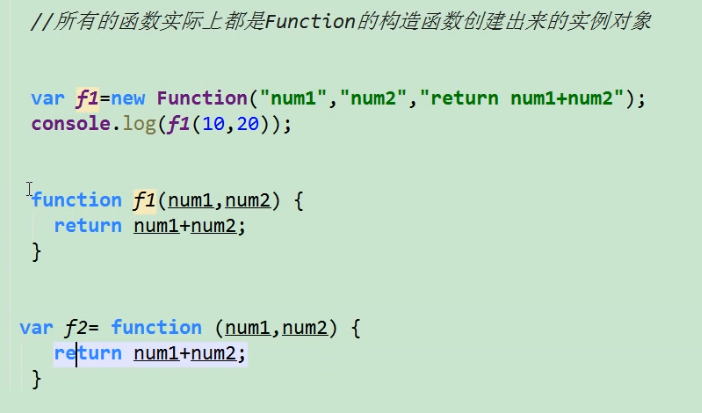
构造函数是对象。



Object好像不是对象，他没有原形\_\_proto\_\_属性，我不确定。



## 函数也是对象



Function也是有object的原型对象构造出来的。