1. javascript中，“函数”（方法）也是对象。
2. 一切对象都有一个根源。它是Object.prototype。
3. 根源之上再没有其他根源。Object.getPrototypeOf(Object.prototype)是null。js中除字面量以外的一切引用对象都来自这个“根源”对象。
4. 表达式Object.getPrototypeOf(Function) === Function.prototype的结果是真。这是Function特有的。实际上Function的prototype是一个内置函数，一切函数都派生自这个内置函数，这个内置函数是一个函数工厂。这个内置函数对象的prototype指向“根源”对象。
5. 表达式Object.prototype === Object.getPrototypeOf(Function.prototype)的结果是真。说明了Object跟Function二者之间的联系，是通过“根源”对象联系起来的。

Function和Object，既是函数，因为都可以Function()或者Object()这样的方式执行，又是对象，因为可以Function.a = 'a',Object.a = 'a'这样赋值。

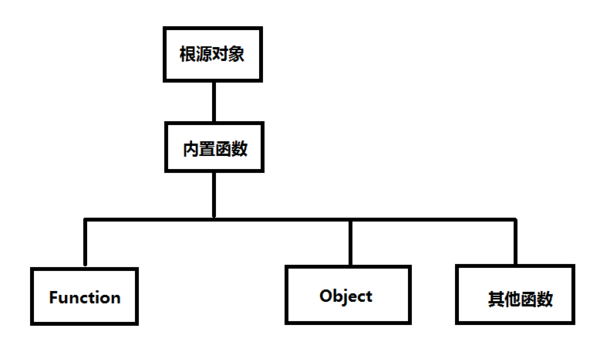
说它们是函数，是因为他们都是通过上面第4条中说的”内置函数工厂“，派生出来的，因而具备函数的特性。

说他们是对象。是因为他们都是通过上面第1条中的”根源“对象，派生出来的，因此具备对象的特征。

继续说一下，Function.prototype指向”内置函数“。而Object.prototype指向”根源对象“。

因而new Function会产生一个匿名函数，而new Object产生一个plain object。

他们之间的关系简单描述就像下面这样。

[](http://h.hiphotos.baidu.com/zhidao/pic/item/6f061d950a7b0208b4995a8b61d9f2d3572cc80c.jpg)