基本上javascript每个对象都都从原型继承属性（null除外）

Object.create（）它创建一个新对象，其中第一个参数是这个对象的原型。Object.create（）提供第二个可选参数，用以对对象的属性进行描述。

写程序时不知道属性名称，无法通过点运算符（.）来访问对象的属性，但是可以用[ ]运算符，因为它使用字符串值（字符串值是动态的，可以再运行时更改）而不是标识符（标识符是静态的，必须写死在程序中）作为索引对属性进行访问。

在javascript中，只有查询属性时才会体会到继承的存在，而设置则和继承无关，这是JavaScript中的一个重要特性，该特性可以让程序员可以有选择的覆盖继承的属性。

属性访问错误：属性访问并不总是返回或一个设置值。

查询一个并不存在的属性并不会报错，但是如果对象不存在，那么试图查询这个不存在对象的属性就会报错。设置属性失败时并不会报错。

在下列场景中给对象o设置属性p会失败

* o中的属性p是只读的；不能给只读属性重新赋值（defineProperty（）方法中有一个例外，可以对配置的只读属性重新赋值）
* o中的属性p是继承的，且它是只读的；不能通过同名自有属性覆盖只读的继承属性。
* o中不存在自有属性p；o没有使用setter方法继承属性p，并且o的扩展性（extensible attribute）是false。如果o中不存在p，而且没有setter方法可供调用，则p一定会添加至o中。但是如果o是不可扩展的，那么在o中不能定义新属性。

删除属性：delete运算符可以删除对象的属性。它的操作数应当是一个属性的访问表达式。delete只是断开属性和宿主的联系，而不会去操作属性中的的属性当delete表达式删除成功或者没有任何副作用（比如删除不存在的属性）时，后者delete 后不是一个属性访问表达式，它返回ture。

delete不能删除那么可配置性位flase的属性（尽管可以删除不可扩展对象的可配置属性）某些内置对象的对象是不可配置的，比如通过变量声明和函数声明创建的全局对象的属性，这些情况下delete操作会返回false。在严格模式下，删除一个不可配置的属性会报一个类型错误。

删除全局对象的可配置属性时 可以直接用delete this.p；

检测属性

判断某个属性是否存在于某个对象中。可以通过运in 运算符、hasOwnPreperty（）和propertyIsEnumerable（）方法来完成这个工作，甚至仅通过属性查询也可以做到这一点。

in运算符的左侧是属性名（字符串），右侧是对象，如果对象自有属性或继承属性中包含这个属性则返回ture。

对象的hasOwnPreperty（）方法用来检测给定的名字是否是对象的自有属性。对于继承属性他将返回false。

propertyIsEnumerable（）只检车到是自有属性且这个自有属性可枚举位ture时它才返回ture；

还可以使用”！==”来判断一个属性是undefined：

o.x !==undefined; ture:o中有属性X

o.y!==undefined; fales:o中没有属性Y

o.toString!==undefined; true:o继承了toString属性；

但是 in运算符可以区分不存在的属性和存在但是值为undefined的属性。“！==”则不行。

for/in循环可以在循环体中遍历对象中所有可枚举的属性（包括自有属性和继承属性），把属性名赋值给循环变量。

for(prop in o){

if(!o.hasOwnProperty(p)) continue; //跳过继承的属性

}

for(prop in o){

if(typeof o[prop]===”function”) continue ; //跳过方法