遍历和检测属性的相关操作受如下情况影响：继承、枚举。

Object.getOwnPropertyNames(obj):返回obj的所有自有的属性键。Object.keys(obj)：返回obj的所有可枚举的属性键。

遍历所有属性的函数创建：

function getAllPropertyNames(obj){

var result=[];

while(obj){

Array.protorype.push.apply(result,Object.getOwnPropertyNames(obj));

obj=Object.getPrototypeOf(obj);

}

return result;

}

propKey in obj:如果obj拥有一个键为propKey的属性，则返回true。这个测试也包含继承的属性。

Object.prototype.hasOwnProperty(prokey):判断是否为自有属性。

for-in循环：返回所有可枚举的属性的键。Object.keys():返回所有（非继承）可枚举属性的键。Object.getOwnPropertyNames():返回所有自有属性的键。

计算对象自有属性的个数：Object。keys(obj).length

属性特性是属性的原子构建块。属性描述符是一种数据结构，用于编程处理特性。

所有的属性状态，包括它的数据和元数据，都存储在特性中。它们是拥有的字段，就像对象拥有的属性。

[[Get]]:持有getter，读取属性时调用的函数；[[Set]]：持有setter,为属性设置值时调用的函数。

[[Configurable]]:t特例：JS允许把非配置属性从可写状态变成只读状态。数组的属性length总是可写的，但不可配置。如果没有这个例外，就无法冻结数组。

为了创建一个对象完全相同的拷贝，必须确保两件事：拷贝必须具有与原对象相同的原型；拷贝必须具有与原对象相同的属性和特性。

继承的只读属性不能被赋值。

保护对象三个层次由弱到强：防止扩展、封闭、冻结。

防止扩展：Object.preventExtensions(obj) 检查对象是否可扩展：Object.isExtensible(obj)

封闭：Object.seal(obj) 检查一个对象是否封闭：Object.isSealed(obj)

冻结：Object.freeze(obj):所有的属性不可写，且封闭obj 检查一个对象是否冻结：Object.isFrozen(obj)

instanceof：可以用来检查一个对象是否为给定构造函数的实例。

constructor属性的用例：1.切换对象的构造函数2.确定对象的构造函数名3.创建相似对象4.指向父构造函数

如果相同的属性，在几个原型中同时存在，称为多态。

公有属性：存储在属性中的值（实例或它的原型）是可以公共访问的。

私有值：存储在环境中的数据和函数是私有的————只有构造函数和构造函数所创建的函数可以访问。

特权方法：原型中的公有方法不能访问私有数据，因此，我们需要特权方法————实例中的共有方法。

特权方法是公有的，且可以被任何实例调用，而这种方法也可以访问私有值，因为她们都是在构造函数中创建的。

Crockford私有模式的利弊:不优雅、完全安全、可能比较慢、会消耗更多的内存。

一个方法的主对象是一个对象，它具有一个属性且该属性的值指向这个方法。

泛型方法：借用原型方法。

chablis.incAge(1)：chablis是该方法调用的接收者，通过this传入incAge

wine.prototype.incAge.call(chablis,1)使用该模式，可以把不是wine实例的对象作为接收者 。

Object作为Map使用的缺陷：继承影响读取属性、覆写会影响调用方法、特殊属性\_\_proto\_\_