这几个方法在js的高级[编程](http://www.2cto.com/kf" \t "_blank)中经常用到，对于新手来说可能还不知道他们有什么区别，我把我的体会总结下来，供大家参考：

首先，定义一个对象：

复制代码

 1  function Parent() {this.name = "wenbo";}

 2  Parent.prototype.alertP = function() {

 3      alert("Parent");

 4  }

 5

 6  function Child() {this.age = 23;}

 7  Child.prototype.alertC = function() {

 8      alert("Child");

 9  }

10

11  function F() {}

12  F.prototype = Parent.prototype;

13  Child.prototype = new F();

14  Child.prototype.constructor = Child;

15  var child = new Child();

复制代码

1,instanceof()   vs   isPrototypeOf():

instanceof:判断该对象是否为另一个对象的实例。

1  console.log(child instanceof  Parent); //true

2  console.log(child instanceof  Child); //true

isPrototypeOf:判断一个对象象是否为一个实例的原型。

1 Parent.prototype.isPrototypeOf(child);//true

2 Child.prototype.isPrototypeOf(child);//true

2， hasOwnProperty()   vs  propertyIsEnumerable()：

 hasOwnProperty：判断对象是否有某个特定的属性，(注意说的是对象的属性，而不是对象原型的属性)必须用字符串指定该属性。

1 Parent.hasOwnProperty("name");//true

2 Child.hasOwnProperty("age");//true

3 Parent.hasOwnProperty("alertP");//false

4 Child.hasOwnProperty("alertC");//false

 propertyIsEnumerable：判断给定的属性是否可以用 for...in 语句进行枚举。由于 for ... in 枚举是包含原型链上的属性的，但propertyIsEnumerable作用于原型方法上时，始终是返回false的，你可以这么认为，for...in可以枚举对象本身的属性和原型上的属性，而propertyIsEnumerable只能判断本身的属性是否可以枚举。此外，预定义的属性不是可列举的，而用户定义的属性总是可列举的。所以如果你只想遍历对象本身的属性，可以：

1 for (var key in obj) {

2   if (obj.hasOwnProperty(key) {

3      //do something

4   }

5 }