继承

一、原型

当定义一个函数对象的时候，会包含一个预定义的属性，叫`prototype`，这就属性称之为原型对象。

```
function Person(name)
{
    this.name = name;
    this.showMe = function(){
        alert(this.name);
    }
};
var one = new Person("javascript");
one.showMe();//javascript
console.log(one.prototype);//undefined证明new出来的普通对象是没有原型的
console.log(Person.prototype);//就是Person函数本身
console.log(Person.prototype.constructor)//function Person(name){...}
```

通过function定义的Person就是一个函数对象，使用function定义的函数对象存在prototype属性，而使用new生成的对象就没有prototype的属性，一般被称作普通对象，但是有_proto_属性。

我们可以通过给原型添加属性和方法来给给对象添加属性或方法

```
functionHero(){
    this.name = "zhangwuji";
    this.sayMe = function(){
        alert("this is zhnagwuji.");
    }
}
//通过原型增加的属性和方法
Hero.prototype.name = "zhouzhiruo";
Hero.prototype.sayMe = function(){
    alert("this is zhouzhiruo.");
}
var hero = new Hero();
alert(hero.name);//zhangwuji
hero.sayMe();//this is zhnagwuji
//delete this.name
alert(hero.name);//zhouzhiruo
```

当函数对象本身的属性或方法与原型的属性或方法同名的时候：

1、默认调用的是函数对象本身的属性或方法.
2、通过原型增加的属性或方法的确是存在的.
3、函数对象本身的属性或方法的优先级要高于原型的属性或方法.