对象的声明方式

1、字面式声明对象,如下:

```
var ohj={
        属性名称:属性值,
        方法名称:function(){
            //函数执行体
        }
    }
```

2、new 操作符+Object 声明对象，如下：

```
var obj=new Object();
obj.属性名称=属性值;
obj.方法名称=function(){
    //函数执行体
}
```

3、构造函数声明对象，如下：

```
function test([参数列表]){
    this.属性名称=属性值;
    this.方法名称=function(){
        //函数执行体
    }
}
var obj=new test(参数);
```

4、工厂方式声明对象，如下：

```
function createObject(nam,age){
    var obj=new Object();
    obj.name=name;
    obj.age=age;
    obj.run=function(){
        return this.name+this.age
    };
    return obj;
}
var obj1=createObject(‘zhangsan’,100);
var obj2=createObject(‘lisi’,200)
```

5、原型模式声明对象

```
function test(){
    test.prototype.属性名称=属性值;
    test.prototype.方法名称=function(){
        //函数执行体
    }
}
var obj=new test();
```

6、混合模式声明对象

```
function test(参1，参2){
    this.属性名称1=参1;
    this.属性名称2=参2;
}
test.prototype.方法名称=function(){
    //执行代码
}
var obj=new test(参1，参2);
```

原型与原型链

1、凡是通过 new function() 创建的对象都是函数对象，其他的都是普通对象

2、js 中所有的函数对象都有一个 prototype 属性，这个属性引用了一个对象，即原型对象，也简称原型。普通对象没有 prototype,但有__proto__属性

3、js 在创建对象（不论是普通对象还是函数对象）的时候，都有一个叫做__proto__的内置属性，用于指向创建它的函数对象的原型对象 prototype(即它的构造函数的原型对象)

继承

1、原型继承

```
var a=function(){};
a.prototype.say()=function(){
    alert(“haha”);
}
a.prototype.gongzi=500;
var b=function(){};
b.prototype=new a();
b.prototype.gongzi=1000;
var obj=new b();
```

注：子元素的成员属性 gongzi 会覆盖父元素。

2、构造继承

```
function parents(name){
    this.name=name;
    this.say=function(){
        alert(this.name);
    }
}
function child(name,age){
```

```
        this.pobj=parents;
        this.pobj(name);
        this.age=age;
    }
    var c=new child(‘lisi’,20);
```

3、call 和 apply

传参方式不同：call 是参数列表，apply 是数组