1. 由于缺少私有作用域，开发者确定了一个规约，说明哪些属性和方法应该被看做私有的。这种规约规定在属性前后加下划线：

obj.\_color\_ = "blue";

这段代码中，属性 color 是私有的。注意，下划线并不改变属性是公用属性的事实，它只是告诉其他开发者，应该把该属性看作私有的。

1. 创建对象的方式

A、混合的构造函数/原型方式

function Car(sColor,iDoors,iMpg) {

this.color = sColor;

this.doors = iDoors;

this.mpg = iMpg;

this.drivers = new Array("Mike","John");

}

Car.prototype.showColor = function() {

alert(this.color);

};

var oCar1 = new Car("red",4,23);

var oCar2 = new Car("blue",3,25);

oCar1.drivers.push("Bill");

alert(oCar1.drivers); //输出 "Mike,John,Bill"

alert(oCar2.drivers); //输出 "Mike,John"

B、动态原型方法

function Car(sColor,iDoors,iMpg) {

this.color = sColor;

this.doors = iDoors;

this.mpg = iMpg;

this.drivers = new Array("Mike","John");

if (typeof Car.\_initialized == "undefined") {

Car.prototype.showColor = function() {

alert(this.color);

};

Car.\_initialized = true;

}

}

1. 继承机制实现
2. 对象冒充实现继承

function ClassA(sColor) {

this.color = sColor;

this.sayColor = function () {

alert(this.color);

};

}

function ClassB(sColor, sName) {

this.newMethod = ClassA; //赋值

this.newMethod(sColor); //调用

delete this.newMethod; //删除

this.name = sName;

this.sayName = function () {

alert(this.name);

};

}

注：所有新属性和新方法都必须在删除了新方法的代码行后定义。否则，可能会覆盖超类的相关属性和方法

1. Call

function ClassB(sColor, sName) {

//this.newMethod = ClassA;

//this.newMethod(color);

//delete this.newMethod;

ClassA.call(this, sColor);

this.name = sName;

this.sayName = function () {

alert(this.name);

};

}

1. Apply

function ClassB(sColor, sName) {

//this.newMethod = ClassA;

//this.newMethod(color);

//delete this.newMethod;

ClassA.apply(this, new Array(sColor));

this.name = sName;

this.sayName = function () {

alert(this.name);

};

}

可以把 ClassB 的整个 arguments 对象作为第二个参数传递给 apply() 方法：

function ClassB(sColor, sName) {

//this.newMethod = ClassA;

//this.newMethod(color);

//delete this.newMethod;

ClassA.apply(this, arguments);

this.name = sName;

this.sayName = function () {

alert(this.name);

};

}

1. 原型链

function ClassA() {

}

ClassA.prototype.color = "blue";

ClassA.prototype.sayColor = function () {

alert(this.color);

};

function ClassB() {

}

ClassB.prototype = new ClassA();

为 ClassB 类添加 name 属性和 sayName() 方法的代码如下：

function ClassB() {

}

ClassB.prototype = new ClassA();

ClassB.prototype.name = "";

ClassB.prototype.sayName = function () {

alert(this.name);

};

1. 原型链+构造函数

function ClassA(sColor) {

this.color = sColor;

}

ClassA.prototype.sayColor = function () {

alert(this.color);

};

function ClassB(sColor, sName) {

ClassA.call(this, sColor);

this.name = sName;

}

ClassB.prototype = new ClassA();

ClassB.prototype.sayName = function () {

alert(this.name);

};