1、闭包

//为避免冲突，将我们的方法用一个匿名方法包裹起来

(function($) {

//扩展这个方法到jquery

$.fn.extend({

//插件名字

pluginname: function() {

//遍历匹配元素的集合

return this.each(function() {

//在这里编写相应代码进行处理

});

}

});

//传递jQuery到方法中，这样我们可以使用任何javascript中的变量来代替"$"

})(jQuery);

2、接下来，我们给插件中加入一些可以改变的属性，这样用户可以根据自己的需要来做一些更改。同时，我们应提供相应的默认值。

(function($){

$.fn.extend({

//将可选择的变量传递给方法

pluginname: function(options) {

//设置默认值并用逗号隔开

var defaults = {

padding: 20,

mouseOverColor : '#000000',

mouseOutColor : '#ffffff'

}

var options = $.extend(defaults, options);

return this.each(function() {

var o = options;

//这里编写相应代码

//可以像下边这样获取变量值

alert(o.padding);

});

}

});

})(jQuery);

3、**实现jQuery插件**

(function ($) {

$.fn.extend({

//插件名称 - paddingList

paddingList: function (options) {

//参数和默认值

var defaults = {

animatePadding: 10,

hoverColor: "Black"

};

var options = $.extend(defaults, options);

return this.each(function () {

var o = options;

//将元素集合赋给变量 本例中是 ul对象

var obj = $(this);

//得到ul中的a对象

var items = $("li a", obj);

//添加hover()事件到a

items.hover(function () {

$(this).css("color", o.hoverColor);

//queue false表示不添加到动画队列中

$(this).animate({ paddingLeft: o.animatePadding }, { queue: false, duration: 300 });

}, function () {

$(this).css("color", "");

$(this).animate({ paddingLeft: "0" }, { queue: true, duration: 300 });

});

});

}

});

})(jQuery);

4、最后，使用插件的方法如下：

//使用插件

$(document).ready(function() {

$("#catagory").paddingList({ animatePadding: 30, hoverColor: "Red" });

});