# 

**jQuery学习笔记**

**Java相关课程系列笔记之十二**

**笔记内容说明**

jQuery（程祖红老师主讲，占笔记内容100%）；

**目 录**

[一、 jQuery基础 1](#_Toc27464)

[1.1 jQuery的特点 1](#_Toc3077)

[1.2 jQuery编程的步骤 1](#_Toc6277)

[1.3 jQuery对象与DOM对象如何相互转换 1](#_Toc29837)

[1.4如何同时使用prototype和jQuery 1](#_Toc13003)

[1.5 EL表达式和jQuery函数的区别 2](#_Toc15425)

[二、 选择器 3](#_Toc25020)

[2.1什么是选择器 3](#_Toc16483)

[2.2基本选择器 3](#_Toc14616)

[2.3层次选择器 3](#_Toc19247)

[2.4基本过滤选择器 4](#_Toc12730)

[2.5内容过滤选择器 4](#_Toc13147)

[2.6可见性过滤选择器 5](#_Toc11168)

[2.7属性过滤选择器 5](#_Toc13447)

[2.8子元素过滤选择器 5](#_Toc18717)

[2.9表单对象属性过滤选择器 5](#_Toc12325)

[2.10表单选择器 6](#_Toc13474)

[三、 DOM操作 7](#_Toc17055)

[3.1查询 7](#_Toc11710)

[3.2创建 7](#_Toc14028)

[3.3插入节点 7](#_Toc24398)

[3.4删除节点 7](#_Toc29172)

[3.5如何将JavaScript代码与HTML分开 8](#_Toc20463)

[3.6复制节点 8](#_Toc1257)

[3.7属性 8](#_Toc1835)

[3.8样式操作 9](#_Toc13062)

[3.9遍历节点 9](#_Toc5377)

[3.10案例：员工列表（点击某行该行加亮，多选框被选中） 10](#_Toc1559)

[3.11案例：员工列表（点击部门隐藏或显示员工） 11](#_Toc22198)

[四、 事件 12](#_Toc32362)

[4.1事件绑定 12](#_Toc27402)

[4.2合成事件 12](#_Toc10406)

[4.3事件冒泡 可参考JavaScript笔记7.5 12](#_Toc6476)

[4.4 jQuery中事件处理 13](#_Toc28442)

[4.5动画 14](#_Toc18916)

[4.6类数组的操作 14](#_Toc27284)

[4.7案例：滚动广告条 15](#_Toc25673)

[五、 jQuery对Ajax编程的支持 17](#_Toc22287)

[5.1 load()方法 17](#_Toc17785)

[5.2案例：显示机票价格 17](#_Toc12984)

[5.3 $.get()方法 17](#_Toc23410)

[5.4 $.post()方法 18](#_Toc530)

[5.5案例：修改Ajax笔记中2.6案例：股票的实时行情 18](#_Toc5438)

[5.6 $.ajax()方法 18](#_Toc19009)

[5.7案例：搜索栏联想效果（服务器返回text） 18](#_Toc5264)

[5.8案例：下拉列表（服务器返回xml文本） 20](#_Toc6863)

[5.9案例：表单验证 21](#_Toc13566)

[5.10 jQuery的自定义方法 23](#_Toc2655)

[5.11 $.param()方法 24](#_Toc24158)

[5.12案例：自定义方法和$.param()方法使用（学了Struts2再看） 24](#_Toc26155)

# 一、jQuery基础

## 1.1 jQuery概述

jQuery是一种框架，对于浏览器的兼容问题，95％不用再去考虑了。

jQuery利用选择器查找要操作的节点（DOM对象），然后将这些节点封装成一个jQuery对象。封装的目的有两个：

1. 兼容不同的浏览器

2. 简化代码

通过调用jQuery对象的方法或者属性来实现对底层的DOM对象的操作。jQuery特点简单概括就是：选择器 + 调方法。

## 1.2 jQuery编程

step1：引入jQuery框架（www.jquery.org下载），min为去掉所有格式的压缩版

<script language="javascript" src="js/jquery-1.4.1.min.js"></script>

step2：使用选择器查找要操作的节点（该节点会被封装成一个jQuery对象，并返回）

var $obj=$('#di');//ID选择器，查找的节点ID为d1

step3：调用jQuery对象的方法或者属性

$obj.css('font-size','60px');//调用jQuery的css()方法

* 注意事项：jQuery是一个大的匿名函数，且内部有很多函数（类似Java中的内部类），它的大部分函数返回对象都是jQuery对象（它自己），所以可以继续“．”，例如：

function f1(){

var $obj=$('#d1');//为了强调返回的是jQuery对象，命名习惯用$开头来声明变量

$obj.css('font-size','60px').css('font-style','italic'); }

## 1.3 jQuery对象与DOM对象如何相互转换

1）dom对象如何转化为jQuery对象

使用函数：$(dom对象)即可，例如：

function f2(){

var obj=document.getElementById('d1');

var $obj=$(obj);//将dom节点封装成jQuery对象

$obj.html('hello java'); }

2）jQuery对象如何转化为dom对象

方式一：$obj.get(0); 方式二：$obj.get()[0];

function f3(){

var $obj=$('#d1'); //方式一：var obj=$obj.get(0);

var obj=$obj.get()[0];//方式二 obj.innerHTML='hello perl'; }

## 1.4如何同时使用prototype和jQuery

step1：先导入prototype.js，再导入jQuery.js

step2：将jQuery的$函数换一个名字：var $a=jQuery.noConflict();//注意大小写

* 注意事项：函数名就是一个变量，指向函数对象，例如：

<script language="javascript" src="js/prototype-1.6.0.3.js"></script>

<script language="javascript" src="js/jquery-1.4.3.js"></script>

function f1(){//无效 var obj=$('d1'); }

//无效是因为jQuery是后引入的，所以prototype被jQuery替换

function f1(){//为了避免冲突，可以将jQuery的$函数换一个名字$a

var $a=jQuery.noConflict(); var obj=$('d1');

obj.innerHTML='hello prototype'; $a('#d1').html('hello jQuery'); }

## 1.5 EL表达式和jQuery函数的区别

1）${}：EL表达式，在服务器端运行（JSTL标签库也在服务器端运行，EL和JSTL标签库本质是Java代码）。

2）$()：jQuery函数，在浏览器中运行（JavaScript也在浏览器中运行）。

# 选择器

## 2.1什么是选择器

jQuery模仿CSS选择器的语法提供了一种用来方便查找要操作的节点的语法规则。

## 2.2基本选择器

1）#id：ID选择器，如：$('#d1').css('color','red');//编号1变

2）.class：类选择器，如：$('.s1').css('font-size','60px');//编号2和3变

3）element：元素选择器，如：$('div').css('font-size','60px');//编号1和2变

4）selector1,selector2...selectorn：选择器合并，如：$('#d1,p').css('font-size','60px');//编号1和3变

5）\*：所有选择器，如：$('\*').css('font-size','60px');

6）案例：

<div id="d1">hello jquery</div><!-- 1 -->

<div class="s1">hello java</div><!-- 2 -->

<p class="s1">hello perl</p><!-- 3 -->

<input type="button" value="基本选择器的使用" onclick="f1();"/>

* 注意事项：当jQuery选择器查找到了多个DOM节点，则仍然是封装成“一个”jQuery对象，在调用jQuery对象的属性或者方法时，默认情况下，会作用于底层所有的DOM节点之上。如：$('.s1').css('font-size','60px');则编号2和3都变。

## 2.3层次选择器

1. select1 select2：所有后代（要符合select2的要求）。

例如：$('#d1 div').css('font-size','60px');//d2d3d4d5变

1. select1>select2：只考虑子节点（要符合select2的要求），孙子不管～

例如：$('#d1>div').css('font-size','60px');//d2d3d5变

1. select1+select2：下一个兄弟（要符合select2的要求），儿子不管～

例如：$('#d3+div').css('font-size','60px');//d5变，d2不管

1. select1~select2：下面所有的兄弟（要符合select2的要求），上面的兄弟不管～兄弟中的儿子也不管～

例如：$('#d2~div').css('background-color','yellow');//d3d5变

5）案例：

<div id="d1">

<div id="d2">hello 1</div>

<div id="d3" style="width:200px;height:200px;background-color:red;">

<div id="d4" style="width:150px;height:150px;background-color:silver;">hello 2</div>

</div>

<div id="d5">hello 3</div>

</div>

<input type="button" value="层次选择器的使用" onclick="f1();" />

## 2.4基本过滤选择器

1）:first：第一行。2）:last：最后一行。3）:not(selector)：把满足要求的选择器排除在外。

4）:even：偶数行，下标从0开始。 5）:odd：奇数行，下标从0开始。

6）:eq(index)：等于下标的元素，下标从0开始。

7）:gt(index)：大于下标的元素，下标从0开始。

8）:lt(index)：小于下标的元素，下标从0开始。

* 注意事项：过滤器前是没有空格的。是xx:first而不是xx :first。

9）案例：step1：页面表格

<table id="t1" border="1" width="60%" cellpadding="0" cellspacing="0">

<thead><tr><td>姓名</td><td>年龄</td></tr></thead>

<tbody><tr><td>岳飞</td><td>33</td></tr>

<tr id="tr2"><td>赵构</td><td>32</td></tr>

<tr><td>韩世忠</td><td>31</td></tr>

<tr><td>梁红玉</td><td>22</td></tr></tbody></table>

<input type="button" value="点这儿" onclick="f1();" />

step2：jQuery代码

<script language="javascript" src="../js/jquery-1.4.3.js"></script>

<!-- 相对路径，../表示向上跳一级 -->

<script language="javascript" type="text/javascript">

function f1(){

$('#t1 tr:first').css('background-color','#cccccc');

$('#t1 tr:last').css('background-color','#cccccc');

$('tbody tr:even').css('background-color','#fff8dc');

$('tbody tr:odd').css('background-color','yellow'); }

function f2(){ $('tbody tr:eq(1)').css('font-style','italic').css('font-size','50px'); }

function f3(){ $('tbody tr:not(#tr2)').css('font-size','50px'); }

function f4(){ $('tbody tr:eq(1) td:eq(1)').css('background-color','red');

//过滤器前（“：”前）是没有空格的 }

</script>

## 2.5内容过滤选择器

1）:contains(text)：匹配包含给定文本的元素。

2）:empty：匹配所有不包含子元素或者文本的空元素。

3）:has(selector)：匹配含有选择器所匹配的元素的元素。

4）:parent：匹配含有子元素或者文本的元素（与empty正好相反）。

5）案例：step1：页面

<div>吃饭了吗？</div><div></div>

<div><p>一会要下课了</p></div><input type="button" value="点这儿" onclick="f1();"/>

step2：jQuery代码

<script language="javascript" src="../js/jquery-1.4.3.js"></script>

<script language="javascript" type="text/javascript">

function f1(){ $('div:contains(吃饭)').css('font-size','50px'); }

function f2(){//当参数比较多，采用对象的方式传递

$('div:empty').css({'width':'400px', 'height':'80px','border':'2px solid red'}); }

function f3(){ $('div:has(p)').css('font-size','80px'); }

function f4(){ $('div:parent').css('border','2px solid blue'); }

</script>

## 2.6可见性过滤选择器

1）:hidden 匹配所有不可见元素，或者type为hidden的元素。

2）:visible 匹配所有的可见元素。

3）案例：step1：页面

<div>hell jQuery</div><div style="display:none;">hello java</div>

<input type="button" value="点这儿" onclick="f1();" />

step2：jQuery代码

<script language="javascript" src="../js/jquery-1.4.3.js"></script>

<script language="javascript" type="text/javascript">

function f1(){ $('div:hidden').css('display','block');//或$('div:hidden').show('slow');

$('div:hidden').show(500);//毫秒 }

function f2(){ $('div:visible').hide(800); }

</script>

## 2.7属性过滤选择器

1. [attribute]：有某个属性的元素。

例如：$('div[id]').css('font-size','60px');//div中有id属性的元素，编号1变

2）[attribute=value]：某个属性的值与指定的值相同的元素。

例如：$('div[id=d1]').css('font-size','60px');//编号1变

1. [attribute!=value]：某个属性的值与指定的值不相同的元素。

例如：$('div[id!=d1]').css('font-size','60px');//编号2变

4）案例：

<div id="d1">hell jQuery</div><!-- 1 --> <div>hello java</div><!-- 2 -->

<input type="button" value="点这儿" onclick="f1();" />

## 2.8子元素过滤选择器

1. :nth-child(index/even/odd)：对符合条件的每个节点的子节点作相同操作。

例如：$('ul li:nth-child(2)').css('font-size','60px');//item2和item22一起变

2）案例：

<ul> <li>item1</li> <li>item2</li> <li>item3</li></ul>

<ul> <li>item11</li><li>item22</li><li>item33</li></ul>

<input type="button" value="点这儿" onclick="f1();" />

* 注意事项：
* 子元素过滤器中index从1开始。
* 基本过滤器中eq的index从0开始，如上例中：$('ul li:eq(1)').css('font-size','60px');则为item2变化。

## 2.9表单对象属性过滤选择器

1. :enabled：没有被禁用。

例如：$('#form1 input:enabled').attr('disabled',true);//设置属性，可见元素为不可见

1. :disabled：被禁用。

例如：$('#form1 input:disabled').css('border','1px dotted red');//设置不可见元素的样式

$('#form1 input:disabled').attr('disabled',false);//设置属性，不可见元素为可见

1. :checked：单选框、多选框中被选中的选项。

例如：alert($('#form2 input:checked').val());//把值输出，类似于value属性

1. :selected：下拉列表中被选中的选项。

例如：alert($('#form3 option:selected').val());

5）案例：

<form id="form1"> username:<input name="username" /><br />

name:<input name="name" disabled="disabled"/><br /></form>

<form id="form2">

爱好：做饭<input type="checkbox" name="interest" value="cooking"/>

钓鱼<input type="checkbox" name="interest" value="fishing" checked="checked"/>

足球<input type="checkbox" name="interest" value="football"/></form>

<form id="form3">

<option value="bj">北京</option>

<option value="cs" selected="selected">长沙</option>

<option value="wh">武汉</option></form>

<input type="button" value="点这儿" onclick="f1();" />

## 2.10表单选择器

1）:input：input元素。 2）:text：文本框。 3）:pasword：密码框。

4）:radio：单选。 5）:checkbox：多选。 6）:submit：提交按钮。

7）:image：图片。 8）:reset：重置按钮。 9）:button：普通按钮。

10）:file：文件。 11）:hidden：隐藏域。

# DOM操作

## 3.1查询

利用选择器找到要操作的节点之后，获得节点的值、属性值、文本以及html内容。

1）html()：html内容，如：alert($('#d1').html())，相当于innerHTML属性，下例中也会把span输出（输出标记中的所有内容），即<span>hello jQuery</span>。

2）text()：文本，如：alert($('#d1').text())，相当于innerText属性，由于有兼容性问题，所以没讲。只输出文本内容hello jQuery（标记中的文本内容）。

3）val()：节点的值，如：alert($('#username').val())，结果为文本框中输入的值。

4）attr()：属性值，如：alert($('#d1').attr('id'))，结果为d1。

* 注意事项：此外，这几个方法也可以用来修改节点的内容、值、文本内容、属性值。如：

$('#d1').html('hello java'); $('#username').val('chang');

$('#d1').attr('style','color:red;font-size:50px;');

5）案例：

<div id="d1"><span>hello jQuery</span></div>

username：<input name="username" id="username"/><br />

<input type="button" value="点这儿" onclick="f1();"/>

## 3.2创建

$(html);//直接写html语句即可，如：var $obj=$('<div>常</div>');

## 3.3插入节点

1）append()：向每个匹配的元素内部最后追加内容（添加的元素作为最后一个孩子）

例如：var $obj=$('<div>抗金英雄</div>'); $('body').append($obj);

2）prepend()：向每个匹配的元素内部最前添加内容（添加的元素作为第一个孩子）

例如：var $obj=$('<div>抗金英雄</div>'); $('body').prepend($obj);

3）after()：向每个匹配的元素之后插入内容（在该元素之后添加兄弟节点）

例如：$('ul').after('<p>hello</p>');

4）before()：向每个匹配的元素之前插入内容（在该元素之前添加兄弟节点）

例如：$('ul').before('<p>hello</p>');

* 注意事项：都可以简化为：$('body').append/prepend/after/before('<div>抗金英雄</div>');

5）案例：

<a href="javascript:;" onclick="f1();">岳飞是谁？</a>

<ul> <li>item1</li> <li>item2</li> <li>item3</li></ul>

## 3.4删除节点

1）remove()：删除节点，如：$('ul li:eq(1)').remove();

2）remove(selector)：删除满足selector的节点，如：$('ul li').remove('#l2');

3）empty()：清空节点，相当于innerHTML=""，如：$('ul li:eq(1)').empty();

4）案例：

<ul> <li>item1</li> <li id="l2">item2</li><li>item3</li></ul>

<input type="button" value="点这儿" onclick="f1();"/>

## 3.5如何将JavaScript代码与HTML分开

1）为何要分开？

为了使行为和数据分开。

2）问题分析

<script language="javascript">

var obj=document.getElementById("d1");

obj.onclick=function(){ this.innerHTML='hello java'; }

</script>

此处的var obj=document.getElementById("d1");没有写在一个函数里（即直接在script标签中写的），再由于浏览器是逐行解析，那么obj将是null，因为DOM树还没有生成。

3）当引入自己写的JS文件时，JS代码写

window.onload=function(){//这里的JS代码会在页面加载完成之后执行。

var obj=document.getElementById("d1");

obj.onclick=function(){ this.innerHTML='hello java'; } };

4）当引入jQuery框架时，JS代码：

①使用如下主结构

$(function(){ //这的JS代码会在页面加载完成之后执行。 });

②在主结构中添加点击事件使用如下结构（即不在html页面添加onclick）

$(function(){//主结构，当页面加载完毕后会执行这里的代码

$('#d1').click(function(){//现在是个jQuery对象，不是dom对象，

//所以不能用$('#d1').onclick，要用jQuery的函数

//this代表绑定了该事件的dom对象，this.innerHTML='hello java dom';//方式一

$(this).html('hello java jquery');//方式二 }); });

③上述做法的另一个好处是：可以不用修改代码，直接把代码放入.js文件中

## 3.6复制节点

1）clone()：复制节点（不复制行为）。

2）clone(true)：使复制的节点也具有行为（将事件处理代码一块复制）。

3）案例：step1：页面

<ul> <li>item1</li> <li>item2</li> <li>item3</li></ul>

<input type="button" value="点这儿" id="b1"/>

step2：jQuery代码

<script language="javascript" type="text/javascript">

$(function(){ $('ul li:eq(2)').click(function(){//给item2动态绑定一个行为

$(this).css('font-size','60px'); });

$('#b1').click(function(){//给button动态绑定一个行为

var $obj=$('ul li:eq(2)').clone(true);//连同行为一起复制

$('ul').append($obj);//追加为ul元素的最后一个节点 }); });

</script>

## 3.7属性

1）attr('attrName')：读取属性。

2）attr('attrName','value')：设置一个属性。

3）attr({"attrName1":"value1","attrName2":"value2"})：设置多个属性。

例如：$("img").attr({ src: "test.jpg", alt: "Test Image" });

* 注意事项：
* 此处属性名可不用引号（单引或双引），但属性值必须用引号！
* 不要把样式当属性了。

4）removeAttr('attrName')：删除属性。

## 3.8样式操作

1. attr('class','')或者attr('style','')：读取和设置。

例如：读取样式：alert($('#d1').attr('class'));

设置样式：$('#d1').attr('class','s1'); 或 $('#d1').attr('style','color:red;font-style:italic;');

2）addClass('')：追加。

例如：$('#d1').addClass('s1 s2');//追加s1和s2两种样式

1. removeClass('') ：移除。

例如：$('#d1').removeClass('s1');//移除样式s1

1. removeClass('s1 s2 ...sn') ：移除多个样式

例如：$('#d1').removeClass('s1 s2');//移除样式s1和s2

5）removeClass()：删除所有样式。

6）toggleClass('')：样式来回切换，有该样式就删除，没有就添加。

例如：$('#d1').toggleClass('s3');//样式s3一会有一会没（来回切换）

1. hasClass('')：是否有某个样式。

例如：alert($('#d1').hasClass('s3'));//返回值true或false

8）css('')：只能读取style样式里某个属性的值。

* 注意事项：无法读取某个样式类

例如：<div id="d1" style="font-size:60px;" class="s3">hello jQuery</div>，则

alert($('#d1').css('font-size'));只能读出60px，若写alert($('#d1').css('s3'));则内容为空，读不出来。

1. css('','')：设置一个CSS样式。

例如：$('#d1').css('border','1px solid red');

1. css({'':'','':''})：设置多个样式。

例如：$('#d1').css({'border':'1px solid red','font-size':'50px'});

## 3.9遍历节点

1）children()：只考虑子元素（孩子），不考虑其它后代元素（孙子）

2）children(selector)：只考虑子元素（孩子），不考虑其它后代元素（孙子），然后还要满足selector的要求（再次过滤）

3）next()：下一个兄弟

4）next(selector)：下一个兄弟，然后还要满足selector的要求（再次过滤）

5）prev()：上一个兄弟

6）prev(selector)：上一个兄弟，然后还要满足selector的要求（再次过滤）

7）siblings()：兄弟们（上下都算）

8）siblings(selector)：兄弟们（上下都算），然后还要满足selector的要求（再次过滤）

9）find(selector)：从某一种节点开始查找所有符合selector要求的后代

10）parent()：父节点

11）案例：step1：页面

<div id="d1">

<div id="d2">hello 1</div><div id="d3">hello 2</div>

<div id="d4">hello 3</div><p>hello 4</p></div>

<input type="button" value="点这儿" id="b1"/>

step2：jQuery代码

<script language="javascript" type="text/javascript">

$(function(){ $('#b1').click(function(){

var $obj=$('#d1').children('div');//obj包含了4个节点

//length属性：获得jQuery对象包含的dom对象的个数

//alert($obj.length);//4

//$('#d4').next().css('font-size','40px');//p变

//$('#d3').siblings('div').css('font-size','40px');

//d2和d4变，若无过滤器，则d2、d4、p都变

$('#d1').find('p').css('font-size','40px');//p变

alert($('#d2').parent().attr('id'));//d1 }); });

</script>

## 3.10案例：员工列表（点击某行该行加亮，多选框被选中）

step1：页面表格

<table id="t1" border="1" width="60%" cellpadding="0" cellspacing="0">

<caption>员工信息</caption>

<thead style="background-color:silver">

<tr><td></td><td>姓名</td><td>薪水</td><td>年龄</td></tr></thead>

<tbody>

<tr><td><input type="checkbox"/></td><td>张三</td><td>4000</td><td>33</td></tr>

<tr><td><input type="checkbox" checked="checked"/></td><td>李四</td>

<td>5000</td><td>32</td></tr>

<tr><td><input type="checkbox"/></td><td>王五</td><td>3500</td><td>31</td></tr>

<tr><td><input type="checkbox"/></td><td>马六</td><td>5500</td><td>22</td></tr>

</tbody></table>

step2：样式

<style><!-- 注意样式顺序 -->

.row1{background-color:#fff8dc;}

.row2{background-color:#fff0f5;}

.selected{background-color:#ffff00;} </style>

step3：jQuery代码

<script language="javascript" type="text/javascript">

$(function(){ $('tbody tr:even').addClass('row1');

$('tbody tr:odd').addClass('row2');

$('tbody :checkbox:checked').parent().parent().addClass('selected');

$('tbody tr').click(function(){

if($(this).hasClass('selected')){

$(this).removeClass('selected');

$(this).find(':checkbox').attr('checked',false);

}else{ $(this).addClass('selected');

$(this).find(':checkbox').attr('checked',true); } }); });

</script>

## 3.11案例：员工列表（点击部门隐藏或显示员工）

step1：页面表格

<table width="50%" border="1" cellpadding="0" cellspacing="0">

<caption style="font-weight:800;">员工信息</caption>

<thead><tr><th>姓名</th><th>薪水</th><th>年龄</th></tr></thead>

<tbody>

<tr class="p" id="p1"><td colspan="3">部门一</td></tr>

<tr class="c\_p1"><td>张三</td><td>20000</td><td>23</td></tr>

<tr class="c\_p1"><td>李四</td><td>22000</td><td>22</td></tr>

<tr class="c\_p1"><td>王五</td><td>14000</td><td>26</td></tr>

<tr class="c\_p1"><td>马六</td><td>15000</td><td>21</td></tr>

<tr class="p" id="p2"><td colspan="3">部门二</td></tr>

<tr class="c\_p2"><td>张三</td><td>20000</td><td>23</td></tr>

<tr class="c\_p2"><td>李四</td><td>22000</td><td>22</td></tr>

<tr class="c\_p2"><td>王五</td><td>14000</td><td>26</td></tr>

<tr class="c\_p2"><td>马六</td><td>15000</td><td>21</td></tr>

<tr class="p" id="p3"><td colspan="3">部门三</td></tr>

<tr class="c\_p3"><td>张三</td><td>20000</td><td>23</td></tr>

<tr class="c\_p3"><td>李四</td><td>22000</td><td>22</td></tr>

<tr class="c\_p3"><td>王五</td><td>14000</td><td>26</td></tr>

<tr class="c\_p3"><td>马六</td><td>15000</td><td>21</td></tr>

</tbody>

</table>

step2：样式

<style>

thead{background-color:#cccccc;}

.p{ background-color:#FFE7BA;}

.selected{background-color:#FFFF00;} </style>

step3：jQuery代码

<script language="javascript" type="text/javascript">

$(function(){ $('.p').toggle(function(){

$(this).addClass('selected'); $(this).siblings('.c\_'+ this.id).hide(500);

},function(){ $(this).removeClass('selected');

$(this).siblings('.c\_'+ this.id).show(500); }); });

</script>

# 事件

## 4.1事件绑定

1）方式一：bind( type,function(){} )

例如：$('#d1').bind('click',function(){//正式写法，常用于解决浏览器兼容性

$(this).html('hello java'); });

2）方式二（绑定的简写形式）：click( function(){} );

例如：$('#d1').click(function(){//简写形式

$(this).html('hello java'); });

## 4.2合成事件

1）hover(enter,leave)：模拟光标悬停事件。

例如：$(function(){ $('.s1').hover(function(){ $(this).addClass('s2');//光标进入

},function(){ $(this).removeClass('s2');//光标离开 }); });

2）不使用合成事件的方式。

例如：$('.s1').mouseenter(function(){//鼠标移入

$(this).addClass('s2');//绑定了mouseenter事件的div });

$('.s1').mouseleave(function(){//鼠标移出

$(this).removeClass('s2'); });

1. 案例：上面两点的样式与页面

<style><!-- 样式 -->

.s1{ width:100px; height:100px; background-color:silver; }

.s2{ background-color:yellow; } </style>

<div class="s1"></div><!-- 页面 -->

* 注意事项：细节！为何不用ID属性，即<div id="s1"></div>，样式选择器改名为#s1{...}。当执行函数时，相当于<div id="s1" class="s2"></div>，由于ID样式选择器比class样式选择器优先级高，所以不会有移入时背景变色效果！

4）toggle(function1(){},function2(){},...)：模拟光标连续单击事件。

例如：$(function(){ $('a').toggle(function(){ $('#d1').show('slow');

},function(){ $('#d1').hide('slow'); }); });

<a href="javascript:;">显示所有票价</a>

<div id="d1" style="display:none;">头等舱：￥2400<br />商务舱：￥2200<br />

经济舱：￥1200<br /></div>

## 4.3事件冒泡 可参考JavaScript笔记7.5

1）什么是事件冒泡

子节点产生的事件，会依次向上抛出给相应的父节点。

2）如何取消事件冒泡

使用event对象，e.cancelBubble=true;

3）事件对象有何作用

①取消冒泡：e.cancelBubble=true;

例如：<div id="d1" onclick="f2(event);">

<a href="javascript:;" onclick="f1(event);">Cilck Me</a></div>

function f1(e){ alert('点击了连接'); e.cancelBubble=true;//取消冒泡 }

function f2(e){ alert('点击了div'); }

②获得光标点击的坐标：两个属性：e.clientX e.clientY

例如：function f2(e){ alert(e.clientX+","+e.clientY); }

③找到事件源（详见JavaScript笔记7.6）：使用event对象：e.target || e.srcElement;要区分浏览器。

* 注意事项：两种获得事件源对象的方式最新的Chrome浏览器都支持。

## 4.4 jQuery中事件处理

1）获得事件对象

click(function(e){ //e：对底层的事件对象做了一个封装 });

2）事件对象的属性

①event.type：事件类型 ②event.target：返回事件源（是DOM对象）

③event.pageX/pageY：返回点击的坐标

例如：$(function(){

$('a').click(function(e){//对两个对象都绑定了click

//依据事件对象获得事件源，e是jQuery对象（封装了底层的事件对象）

var obj=e.target;//target属性返回的是一个dom对象

alert(obj.innerHTML);

alert(e.pageX+","+e.pageY);//通过事件对象获得光标点击的X、Y坐标

alert(e.type); }); });

<a href="javascript:;">Cilck 1</a><a href="javascript:;">Cilck 2</a>

3）停止冒泡：event.stopPropagation()

例如：$(function(){//e是Query封装后的对象，不能再用底层的cancelBubble属性

$('a').click(function(e){ alert('点击的是连接');

e.stopPropagation();//停止冒泡 });

$('div').click(function(e){ alert('点击的是div'); }); });

<div id="d1"><a href="javascript:;">Cilck Me</a></div>

4）停止默认行为：event.preventDefault()

* 注意事项：原来的写法为<a href="del.do" onclick="return false"></a>

如：$(function(){

$('a').click(function(e){ var flag=confirm('确定删除吗？');

if(!flag){ //阻止浏览器的默认行为，即不再向连接地址发请求

e.preventDefault(); } }); });

<a href="del.do">删除</a>

5）模拟操作：trigger('click')

$(function(){

$('#b1').click(function(){

//模拟用户点击了username文本框，即让username文本框产生焦点获得事件

$('#username').trigger('focus');

//$('#username').focus();//简写形式，模拟获得焦点

//$('#username').click();//模拟点击 }); });

## 4.5动画

1）show()/hide()

①作用：通过同时改变元素的“宽度”和“高度”来实现显示或隐藏。

②用法：show(速度,[回调函数]); （hide同理）

A.回调函数：整个动画执行完毕之后，会执行该函数。

B.速度：'slow'，'fast'，'normal' 或者使用毫秒数。

2）slideUp()/slideDown()

①作用：通过同时改变元素的“高度”来实现显示或隐藏。

②用法：同上。

3）案例：将4.2案例4）中显示机票价格修改

$(function(){ $('a').toggle(function(){ $('#d1').slideDown(800);//滑下

},function(){ $('#d1').slideUp(800);//收起 }); });

4）fadeIn()/fadeOut()

①作用：通过改变元素的不透明度来实现显示或隐藏。0透明0.5半透明1不透明

②用法：同上。

5）案例：将4.2案例4）中显示机票价格修改

$(function(){ $('#b1').toggle(function(){ $('#d1').fadeIn('slow');//淡入

},function(){ $('#d1').fadeOut('slow');//淡出 }); });

6）自定义动画animate(params,speed,[callback])

①params：是一个JavaScript对象，描述动画执行结束之后元素的样式，比如：{'height':'200px'}

②speed：速度，只能用毫秒数。 ③callback：回调函数。

7）案例：step1：页面

<div id="d1"></div>

step2：样式

#d1{ width:100px; height:100px; background-color:#fff8cd; border:1px solid red;

position:absolute;/\*必须定义position属性，否则动不起来\*/ }

step3：jQuery代码

$(function(){ $('#d1').click(function(){

//$(this).animate({'left':'500px','top':'300px'},5000);//走最短距离，往右下走

$(this).animate({'left':'500px'},5000);//动画队列先执行完第一个，再执行第二个

$(this).animate({'top':'300px'},3000).fadeOut('slow'); }); });

## 4.6类数组的操作

jQuery对象里面可能包含多个DOM对象，所谓类数组，指的就是这些DOM对象。

1）each(function(i))：循环遍历每一个元素，this代表被遍历的DOM对象，$(this)代表被迭代的jQuery对象，i代表正在被遍历的那个对象的下标。下标从0开始。

例如：$('#b1').click(function(){ var $obj=$('ul li');

$obj.each(function(i){//i：表示正在被遍历的那个节点的下标，下标从0开始

if(i==0){//this：表示正在被遍历的那个DOM对象

$(this).css('font-size','50px');

}else if(i==1){ $(this).css('font-style','italic');

}else{ $(this).css('color','red'); } }); });

2）eq(index)：返回index位置处的jQuery对象。

例如：$('#b1').click(function(){ var $obj=$('ul li');

//eq：取下标对应处的DOM对象，然后将这个DOM对象封装成一个jQuery对象

var $o=$obj.eq(1); $o.css('font-size','50px');//item2变 });

3）index(obj)：返回下标，其中obj可以是DOM对象或者jQuery对象。

例如：$('#b1').click(function(){ var $obj=$('ul li');

var $o=$obj.eq(1);//item2

var index=$obj.index($o);//index：返回节点在类数组中的下标

alert(index);//1 });

4）length属性：获得jQuery对象包含的DOM对象的个数。

例如：$('#b1').click(function(){ var $obj=$('ul li');

alert($obj.length);//3 });

5）get()：返回DOM对象组成的数组。

例如：$('#b1').click(function(){ var $obj=$('ul li');

var arr=$obj.get(); arr[2].innerHTML='hello';//item3变 });

6）get(index)：返回第index个DOM对象。

例如：$('#b1').click(function(){ var $obj=$('ul li');

var obj=$obj.get(1);//get(index)：取index对应处的DOM对象

obj.innerHTML='hello jQuery';//item2变 });

7）案例：页面

<ul> <li>item1</li><li>item2</li><li>item3</li></ul>

<input type="button" value="点这儿" id="b1""/>

## 4.7案例：滚动广告条

step1：页面无序列表

<div id="d1">

<ul id="adv"><li><img src="images/1.jpg"/></li><li><img src="images/2.jpg"/></li>

<li><img src="images/3.jpg"/></li><li><img src="images/4.jpg"/></li>

<li><img src="images/5.jpg"/></li></ul>

<ul id="num"><li>1</li><li>2</li><li>3</li><li>4</li><li>5</li></ul></div>

step2：样式

\*{ margin:0px; padding:0px; }

#d1{ border : 1px solid #aaaaaa; margin-left:200px; margin-top:40px; width:548px; height:177px; overflow:hidden; position:relative; }

#adv,#num{ position:absolute; }

ul li{ list-style:none; display:inline; }

ul img{ width:548px; height:177px; display:block; }

#num{ right:5px; bottom:5px; }

#num li{ float: left; color: #FF7300; text-align: center; line-height: 16px;

width: 16px; height: 16px; font-family: Arial; font-size: 12px; cursor: pointer;

overflow: hidden; margin: 3px 1px; border: 1px solid #FF7300; }

.on{ line-height: 21px; width: 21px; height: 21px; font-size: 16px; margin: 0 1px;

border: 0; background-color:red; font-weight: bold; }

step3：jQuery代码

$(function(){

$('#num li').mouseenter(function(){

var index = $('#num li').index(this);//需要知道光标指向的是哪一个li

showImage(index);//依据index(下标)，滚动图片

}).eq(0).mouseenter();

var i = 0; var taskId;//光标进入，停止滚动图片，光标离开，开始滚动图片

$('#d1').hover(function(){ clearInterval(taskId);//光标进入，清除定时任务

},function(){ taskId = setInterval(function(){//光标离开，开始定时任务

showImage(i); i++;

if(i == 5){ i = 0; } },2000);

}).mouseleave(); });

//滚动图片函数，stop(true)：在执行当前动画之前，先清空之前累积的动画

function showImage(index){

$('#adv').stop(true).animate({'top':-index \* 177},1000);

$('#num li').eq(index).addClass('on').siblings().removeClass('on');//加亮光标指定的li }

# jQuery对Ajax编程的支持

## 5.1 load()方法

1. 作用：将服务器返回的数据直接插入到符合要求的节点之上，相当于：

obj.innerHTML=服务器返回的数据。

2）用法：$obj.load(url,[data]);

①url：请求地址，服务器上的某个组件的地址。

②data：请求参数，有两种形式：

A.请求字符串形式："name=chang&&age=23"

B.对象形式：{'name':'chang','age':23}

* 注意事项：load方法：当没有请求参数时，会使用get方式向服务器发请求，如果有请求参数，会使用post方式向服务器发请求。也不支持异步！

## 5.2案例：显示机票价格

step1：ActionServlet中service方法中if判断

if(action.equals("/priceInfo")){

String airline=request.getParameter("airline");

if(airline.equals("CA1000")){//简单操作，不读取数据库了

out.println("头等舱：￥2400<br />经济舱：￥1200");

}else{ out.println("头等舱：￥2200<br />经济舱：￥1000"); } }

step2：页面表格

<table cellpadding="0" cellspacing="0" width="60%" border="1">

<tr> <td>航班号</td><td>机型</td><td>起飞时间</td><td>到达时间</td>

<td>&nbsp;</td><td>经济舱价格</td><td>&nbsp;</td></tr>

<tr> <td>CA1000</td><td><a href="#">波音777</a></td><td>8:00am</td>

<td>10:00am</td><td><a href="javascript:;" class="s1">显示所有票价</a>

<div></div></td><td>￥1200</td>

<td><input type="button" value="订票" /></td></tr>

<tr> <td>MU1949</td><td><a href="#">空客230</a></td><td>18:00am</td>

<td>20:00am</td><td><a href="javascript:;" class="s1">显示所有票价</a>

<div></div></td><td>￥800</td>

<td><input type="button" value="订票" /></td></tr></table>

step3：jQuery代码

$(function(){

$('a.s1').toggle(function(){ var airline=$(this).parent().siblings().eq(0).text();

$(this).next().load('priceInfo.do','airline='+airline);

//或用对象形式$(this).next().load('priceInfo.do',{'airline=':airline});

$(this).html('显示经济舱价格');

},function(){ $(this).next().empty(); $(this).html('显示所有票价'); }); });

## 5.3 $.get()方法

1）作用：使用get方式向服务器发请求。

2）用法：$.get(url,[data],[callback],[type]);

①url：请求地址。 ②data：请求参数，有两种形式：同上。

③callback：是一个回调函数，格式：function(data,statusText)，其中，data是服务器返回的数据，statusText是服务器处理的状态。

④type：服务器返回的数据的类型，有五种：

A.html：返回的是一个html文档。 B.text：返回的是纯文本。

C.json：返回的是json字符串。 D.xml：返回的是一个Xml文档。

E.script：返回的是一个javascript脚本。

## 5.4 $.post()方法

1）作用：使用post方式向服务器发请求。 2）用法：与$.get()方法相同

## 5.5案例：修改Ajax笔记中2.6案例：股票的实时行情

step1：导入jQuery框架，删除step4中Ajax（JS）代码

step2：写jQuery代码，如下：

$(function(){ setInterval(quoto,3000); });

function quoto(){

$.get('quoto.do',function(data){//注意IE浏览器的缓存问题

//$.get()/$.post()/$.ajax()会自动将服务器返回的JSON字符串转换成JavaScript对象

$('#tb1').empty();//先清空之前的内容

for(i=0;i<data.length;i++){ $('#tb1').append( '<tr><td>'+data[i].name

+'</td><td>'+data[i].code+'</td><td>'+data[i].price+'</td></tr>'); }

},'json');//ActionServlet中发送的是JSON字符串，所以此处写json，记得导6个包，详见Ajax笔记2.5 }

## 5.6 $.ajax()方法

1）用法：$.ajax( { ... } )，参数顺序无所谓。

2）参数如下：①url(string)：请求地址。 ②type(string)：GET/POST。

③data(object/string)：请求参数。④dataType(string)：预期服务器返回的数据类型。

⑤success(function)：请求成功后调用的回调函数，有两个参数function(data,textStatus)

，其中data是服务器返回的数据，textStatus描述状态的字符串。

⑥error(function)：请求失败时调用的函数，有三个参数

function(xhr,textStatus,errorThrown)：

A.xhr：底层的XMLHttpRequest对象。 B.textStatus：错误的描述。

C.errorThrown：一般为null。

⑦async：true（缺省，异步）/false（同步）。

## 5.7案例：搜索栏联想效果（服务器返回text）

step1：ActionServlet中service方法中if判断

if(action.equals("/find")){

//if(1==1){ throw new ServletException("some error"); }//模拟错误

String key=request.getParameter("key");

if(key.equals("小")){ out.println("小学生作文,小学生,小米,小米2,小学生守则");

}else if(key.equals("小学")){ out.println("小学生作文,小学生");

}else if(key.equals("岳")){ out.println("岳飞,岳不群"); } }

step2：样式

<style> table{ margin-left:400px; margin-top:100px; font-size:20px; }

.selected{ background-color:#fff8cd; } </style>

step3：页面

<table cellpadding="0" cellspacing="0">

<tr><td><input name="key" id="key"/></td>

<td><input type="button" value="搜索"/></td></tr>

<tr><td colspan="2"><div id="tips"/></div></td></tr></table>

step4：jQuery代码：火狐某些低版本切换中文输入法时keyup事件失效，是浏览器的BUG

<script language="javascript" type="text/javascript">

$(function(){

$('#key').keyup(function(){

$.ajax( {

'url':'find.do', 'type':'post', 'data':'key='+$('#key').val(), 'dataType':'text',

'success':function(data){//data表示服务器返回的数据，如：岳飞,岳不群

var arr=data.split(',');//分解服务器返回的数据

$('#tips').empty();

for(i=0;i<arr.length;i++){ $('#tips').append(

"<div class='item'>"+arr[i]+"</div>"); }

//当光标经过提示项时加亮

$('.item').mouseenter(function(){

$(this).addClass('selected').siblings().removeClass('selected'); });

//当光标点击某个选项时，将该选项的值复制到key

$('.item').click(function(){

$('#key').val($(this).text()); $('#tips').empty(); });

}//success回调函数的括号

});//这的 } 是$.ajax中的括号

});//这的 } 是keyup中的括号

});//这的 } 是$函数中的括号

</script>

step5：为了解决keyup失效问题，使用input或propertychange事件都可，但IE浏览器只认识propertychange事件，其他浏览器只认识input事件，所以再次为了解决兼容性问题，修改如下：

<script language="javascript" type="text/javascript">

$(function(){

var event\_type='input';//要判断浏览器类型

if(navigator.userAgent.indexOf('MSIE')!= -1){//IE浏览器

event\_type='propertychange'; }

$('#key').bind(event\_type,fn); });

function fn(){

$.ajax( {

'url':'find.do', 'type':'post', 'data':'key='+$('#key').val(), 'dataType':'text',

'success':function(data){//data表示服务器返回的数据，如：岳飞,岳不群

var arr=data.split(',');//分解服务器返回的数据

$('#tips').empty();

for(i=0;i<arr.length;i++){ $('#tips').append(

"<div class='item' style='border:1px solid silver'>"+arr[i]+"</div>");

}

//当光标经过提示项时加亮

$('.item').mouseenter(function(){

$(this).addClass('selected').siblings().removeClass('selected'); });

//当光标点击某个选项时，将该选项的值复制到key

$('.item').click(function(){

$('#key').val($(this).text()); $('#tips').empty(); });

},

'error':function(){ alert('出错'); }

});//这的 } 是$.ajax中的括号

}

</script>

## 5.8案例：下拉列表（服务器返回xml文本）

step1：页面

<select style="width:120px;" id="s1">

<option value="bmw520">宝马520</option><option value="qq">QQ</option>

<option value="maiten">迈腾</option></select>

step2：样式

<style>

#d1{ width:300px; height:80px; background-color:#fff8dc; } </style>

step3：jQuery代码

<script type="text/javascript">

$(function(){

$('select').change(function(){

$('#d1').remove();

$.ajax( { 'url':'carinfo.do', 'type':'post', 'data':{'name':$('select').val()},

'dataType':'xml',

'success':function(data,statusText){

//data是一个dom节点，指向xml文档对应的那棵dom树。

var $obj = $(data);

$('#s1').after("<div id='d1'></div>");

$('#d1').html('报价:' + $obj.find('price').text()

+ '<br/> 描述:' + $obj.find('desc').text());

},

'error':function(xhr,e1,e2){

//xhr表示XmlHttpRequest对象，e1,e2 表示具体的错误信息

alert('系统出错');

}

});//这的 } 是$.ajax中的括号

});//这的 } 是change中的括号

});//这的 } 是$函数中的括号

</script>

step4：ActionServlet中service方法中if判断

if(path.equals("/carinfo")){ System.out.println("carInfo..");

//生成一份xml文档，返回给客户端

response.setContentType("text/xml;charset=utf-8");

PrintWriter out = response.getWriter(); StringBuffer sb = new StringBuffer();

sb.append("<msg>"); String name =request.getParameter("name");

if(name.equals("bmw520")){ sb.append("<price>50</price>");

sb.append("<desc>还不错</desc>");

}else if(name.equals("qq")){ sb.append("<price>5</price>");

sb.append("<desc>非常不错</desc>");

}else{ sb.append("<price>18</price>");

sb.append("<desc>没开过不知道</desc>"); }

sb.append("</msg>"); out.println(sb.toString()); out.close(); }

step5：访问http://localhost:8080/应用名/carinfo.do?name=bmw520 ，则会返回一颗Xml树

访问http://localhost:8080/应用名/carinfo.jsp正常显示。

## 5.9案例：表单验证

1）用户输入时，动态提示信息

step1：样式

<style> #d1{ width:500px; height:250px; border:1px solid red;

background-color:#fff8dc; margin:20 auto; }</style>

step2：页面

<div id="d1">

<form class="cmxform" id="signupForm">

<p><label for="username">用户名</label><input id="username" name="username" /></p>

<p><label for="password">密码</label><input id="password" name="password"

type="password" /></p>

<p><label for="confirm\_password">确认密码</label>

<input id="confirm\_password" name="confirm\_password" type="password" /></p>

<p><label for="email">邮箱</label><input id="email" name="email" /></p>

<p><label for="agree">是否同意</label><input type="checkbox" id="agree"

name="agree" /></p>

<p><input class="submit" type="submit" value="Submit" /></p>

</form>

</div>

step3：jQuery代码

<script type="text/javascript" src="../js/jquery-1.4.3.js"></script>

<script type="text/javascript" src="../js/jquery.validate.js"></script><!-- 导包 -->

<script type="text/javascript">

$(function(){

$("#signupForm").validate(

{ rules:

{ username:

{ required: true, minlength: 2 },

password:

{ required: true, minlength: 5 },

confirm\_password:

{ required: true, minlength: 5, equalTo: "#password" },

email:

{ required: true, email: true }

},

}

);

});

</script>

2）使用自定义错误提示信息，如动态提示信息为中文

step1：修改jQuery代码

<script type="text/javascript">

$(function(){

$("#signupForm").validate(

{ rules:

{ username:

{ required: true, minlength: 2 },

password:

{ required: true, minlength: 5 },

confirm\_password:

{ required: true, minlength: 5, equalTo: "#password" },

email:

{ required: true, email: true }

},

messages:

{ username:

{ required: "请输入用户名", minlength: "用户名至少2个字符" },

password:

{ required: "请输入密码", minlength: "密码不能少于5个字符" },

confirm\_password:

{ required: "请输入确认密码", minlength: "确认密码不能少于5

个字符",equalTo: "密码输入不一致" },

email: "请输入有效邮箱"

}

}

);

);

</script>

3）给验证信息添加图片，如点击提交，提示信息包括相应提示图片

step1：添加样式

span { font-weight: bold; padding-right: 1em; vertical-align: top; }

span.error { background:url("../image/unchecked.gif") no-repeat 0px 0px;

padding-left: 16px; }

span.success { background:url("../image/checked.gif") no-repeat 0px 0px;

padding-left: 16px; }

step2：修改jQuery代码

<script type="text/javascript">

$(function(){

$("#signupForm").validate(

{ rules:

{ username:

{ required: true, minlength: 2 },

password:

{ required: true, minlength: 5 },

confirm\_password:

{ required: true, minlength: 5, equalTo: "#password" },

email:

{ required: true, email: true }

},

messages:

{ username:

{ required: "请输入用户名", minlength: "用户名至少2个字符" },

password:

{ required: "请输入密码", minlength: "密码不能少于5个字符" },

confirm\_password:

{ required: "请输入确认密码", minlength: "确认密码不能少于5

个字符",equalTo: "密码输入不一致" },

email: "请输入有效邮箱"

},

errorElement: "span", //用来创建错误提示信息标签

success: function(label) {//验证成功后执行的回调函数

//label指向上面那个错误提示信息标签

//清空错误提示信息

label.text(" ").addClass("success");//加上自定义的success类

}

});

});

</script>

## 5.10 jQuery的自定义方法

1）语法：$.fn.funName=function(){...}

1. 案例：

$.fn.red=function(){ this.css("backgroundColor","red"); }

$("#msg\_error").red();//调用自定义的方法

## 5.11 $.param()方法

1）语法：

var obj={表单的名字:"表单的值"};//组成一个对象

$.param(obj);

## 5.12案例：自定义方法和$.param()方法使用（学了Struts2再看）

step1：创建自定义方法

$.fn.remote=function(url,errorMsg,errorCtn){

var b=false; var value=this.val(); var name=this.attr("name");

var obj={};//做了一个对象

obj[name]=value;//添加一个键值对

var params=$.param(obj);//考虑提交是中文的情况，先编码再传

$.ajax( { url:url,

date:params,

type:"post",

dataType:"json",

async:false,

success:function(data){//data为接收到的boolean值变量

if(data){ $(errorCtn).text(""); b=true;//验证成功

}else{ $(errorCtn).text(errorMsg);//验证失败 }

}

});

return b; }

step2：JSP页面调用自定义函数

<script type="text/javascript">

$(function(){ $('#save').click(function(){

var b=validateCostForm();

if(b){ $('#costForm').submit(); } }); })

function validateCostForm(){

var b1=false;

b1=$('#costName').remote("validateName.action","资费名称已被占用",

$('#msg\_costName'));

return b1 ; }

</script>

step3：创建验证用的Action，并返回一个boolean值变量

step4：在struts-cost.xml配置文件中，配置ValueStack栈顶返回的boolean值变量名字

* 注意事项：此处，需要学习Struts2后，并结合NetCTOSS项目，代码略……