网页编码的常见问题以及解决方法

1. **判断input控件是否获取焦点**

获取密码框的焦点，如果用户是在密码框获取焦点的时候按下回车键，那么就进行某个操作，比如说登陆等等。

方法1：这个地方是根据input的id去判断

<ul style="list-style-type:*none*" >

<li>

<span >账号：</span>

<input type=*"text"* name=*"username"* id=*"username"* />

</li>

<li id=*"name-info"* class=*"login-info"* style="display: *none*;">

<span></span>

</li>

<li>

<span>密码：</span>

<input type=*"password"* name=*"password"* id=*"password"*/>

</li>

<li id=*"pwd-info"* class=*"login-info"* style="display: *none*;">

<span></span>

</li>

<li>

<input type=*"checkbox"* value=*"checkbox"* id=*"remember-login"* />

<span style="margin-left:*20px*;">记住登陆名</span>

</li>

</ul>

//监听回车事件

document.onkeydown=**function**(event){

//判断回车事件是否来自登陆输入密码文本框

**if**(document.activeElement.id=="password"){

**var** e = event || window.event || arguments.callee.caller.arguments[0];

**if**(e && e.keyCode==13){ // enter 键

checkLoginInfo();

}

}

};

1. **校验表单数据的正确性问题**

可以使用formValidator框架，具体看百度云盘web源码的formcheck.zip包

1. **在输入框中用灰色的信息提示用户，当输入框获得焦点时，信息消失。具体效果如下：**



**解决方案1：**1、首先将输入框的文本颜色设置为灰色

2、当输入框获得焦点，触发事件，将输入框内容设置为空，然后将文本颜色设置为黑色（表示是用户的输入），当输入框失去焦点时，如果输入框中无内容则将文本颜色设置为灰色，然后设置输入框提示内容。

具体函数：onfocus()获得焦点的响应函数

Onblur()失去焦点的响应函数

优点：支持IE，等通用浏览器

具体实现：

//响应对话框中确定添加按钮事件

$('#popur-heat-map ul li div').click(**function**(){

**var** str;

**var** str1;

name = $('#popur-heat-map ul li').find('input')[0].value;

url = $('#popur-heat-map ul li').find('input')[1].value;

//用正则表达式匹配url是否合法

str = url.match(/http:\/\/.+\..+/i);

str1 = url.match(/www.+\..+/i);

**if**(name==""||name==input\_text["name\_null"]||name==input\_text["name\_over"]||name==input\_text["name\_exist"]){//判断名称是否为null

//把input控件里面的文本颜色设置为灰色

$('#page\_setting\_name').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[0].value=input\_text["name\_null"];

}**else** **if** (name.length>50) {

$('#page\_setting\_name').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[0].value=input\_text["name\_over"];

}**else** **if**(str==**null**&&str1==**null**){

$('#page\_setting\_url').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[1].value=input\_text["url\_not\_right"];

}**else** **if** (url.length>100) {

$('#page\_setting\_url').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[1].value=input\_text["url\_over"];

}**else**{

**if**(isUrlExist(objArray, url)==1){//如果url已经存在则提示用户

$('#page\_setting\_url').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[1].value=input\_text["url\_exist"];

}**else** **if** (isNameExist(objArray,name)==1) {

$('#page\_setting\_name').attr("style","color:#999;");

$('#popur-heat-map ul li').find('input')[0].value=input\_text["name\_exist"];

}**else** **if** (isOverSize(objArray)==1) {//如果添加的数目超过10,则提示用户

//弹出提示框

$('#popur-add-data').css('display','block');

}**else**{//将用户输入插入到数据库

operateType = 1;//将操作类型置为插入数据

GetData(date\_time, name, url, operateType,0);//因为id是int类型，不能传null，因此在这里传0，不会影响数据的结果（后台没有调用该id），只是方便调通程序

//将input输入控件的值重新置为空

$('#popur-heat-map ul li').find('input')[0].value="";

$('#popur-heat-map ul li').find('input')[1].value="";

$('#setting\_add\_text\_url').text("");

// $('#popur-heat-map ul li').find('input')[0].placeholder="";

// $('#popur-heat-map ul li').find('input')[1].placeholder="";

$('.popur-layer').hide();

$('#popur-heat-map').hide();

}

}

});

<div id=*"popur-heat-map"* class=*"popur-box border-radius-3"*>

<div class=*"popur-header"*>

<span>新增转化目标</span>

<span class=*"popur-close-btn"*>X</span>

</div>

<ul>

<li>名称（最长50个字符）</li>

<li><input type=*"text"* class=*"placeholderCls"* id=*"page\_setting\_name"*

onfocus="if(this.value=='名称不能为空'||this.value=='你输入的名称超过规定长度50,请重新输入！'||this.value=='你输入的名称已经存在,请重新输入！'){this.value='';this.style.color='#000'}"

onblur="if(!value){value='名称不能为空';this.style.color='#999'}"/></li>

<li>转化目标URL</li>

<li><input type=*"text"* class=*"placeholderCls"* id=*"page\_setting\_url"*

onfocus="if(this.value=='你输入的url格式不对,请重新输入！'||this.value=='你输入的url超过规定长度100,请重新输入！'||this.value=='你输入的url已经存在,请重新输入！'){this.value='';this.style.color='#000'}"

onblur="if(!value){value='你输入的url格式不对,请重新输入！';this.style.color='#999'}"/></li>

<li><div class=*"button btn-blue-border border-radius-3"*>确定添加</div></li>

</ul>

</div>

**解决方案2：**使用input属性placeholder,具体可以查询html官网

优点：编写代码简单，只要浏览器支持HTML5就行。

缺点：不支持IE8及以下。

**解决方案3：**利用jquery先进行判断如果浏览器不支持placeholder则使用替代方法。

具体代码看百度云盘，web源码下的input**(推荐使用这种方法)**

<body>

<div class=*"content"*>

<input type=*"text"* name=*"name"* placeholder=*"输入你的姓名"* />

<input type=*"text"* name=*"address"* placeholder=*"输入你的地址"* />

</div>

<script type=*"text/javascript"* charset=*"utf-8"* src=*"js/LAB.js"*></script>

<script type=*"text/javascript"* charset=*"utf-8"*>

//下面代码的意思是等等jquery-1.6.1.min.js文件加载完后再去加载test.js文件，这样保证顺序性

$LAB.script("js/jquery-1.9.1.js").wait().script("js/modernizr.custom.js").wait().script("js/test.js")

.wait(**function**() {

// alert("初始化操作");

});

</script>

</body>

$(**function**(){

**if**(!Modernizr.input.placeholder){

$('input[type=text]:eq(0)').val('输入你的姓名');

$('input[type=text]:eq(1)').val('输入你的地址');

$('input[type=text]').focus(**function**(){

**if**($(**this**).val() === $(**this**).attr('placeholder')){

$(**this**).val('');

}

});

$('input[type=text]').blur(**function**(){

**if**($(**this**).val() === ''){

$(**this**).val($(**this**).attr('placeholder'));

}

});

}

});

1. **为弹出框添加阴影效果，box-shadow**

**情况1：非IE浏览器**

可以使用如下样式，在需要加入阴影的标签下设置如下样式：

box-shadow: 1px 1px 10px #666, -1px -1px 10px #666;

情况2：IE浏览器

由于box-shadow属性是CSS3.0才加入的，因此有些IE版本不支持CSS3.0，所以这个属性对于IE浏览器无效，不过我们可以采用下面的方法进行解决：

在需要加入阴影的标签下设置如下样式

在CSS文件中加入如下代码

*.box-shadow* {

filter: *progid*:*DXImageTransform.Microsoft.Shadow*(*color*=*'#969696',*

*Direction*=*135,* *Strength*=*5)*; /\*for ie6,7,8\*/

background-color: *#fff*;

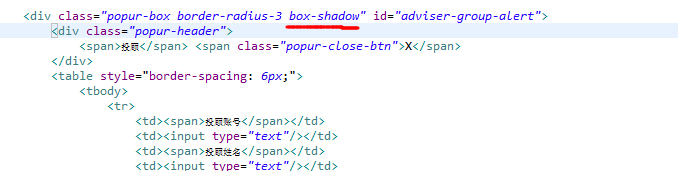
-moz-box-shadow: *2px* *2px* *5px* *#969696*; /\*firefox\*/

-webkit-box-shadow: *2px* *2px* *5px* *#969696*; /\*webkit\*/

box-shadow: *2px* *2px* *20px* *#969696*; /\*opera或ie9\*/

}

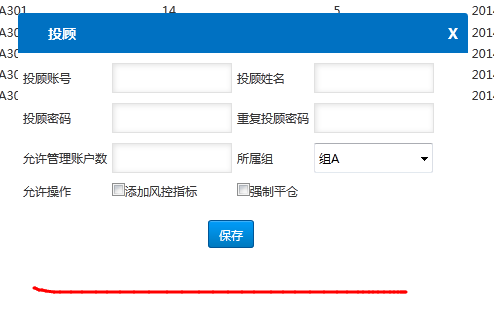
在需要阴影的标签下加入class= *box-shadow*



最终效果如下：



没有阴影的效果，如果背景颜色一样，就会出现这种效果，看着不是很好，如下：

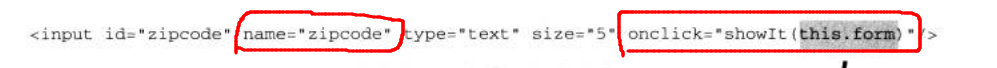


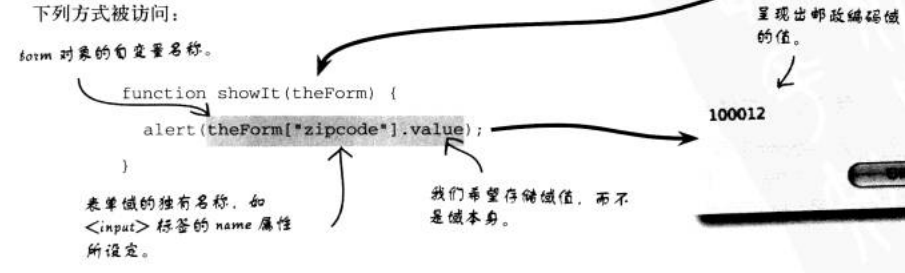
1. **如何得到表单中用户输入的数据**

方法1：

getElementById();

方法2：利用表单自身对象**form**





方法3：利用jQuery

$('#popur-heat-map ul li').find('input')[0].value

其中li是表单控件的父控件，[0]代表第一个表单输入控件

1. **获取html元素的点击事件、设置元素属性值、得到元素的值**

还是以上面的html的元素为例

解决方法（利用JQuery）：

获取div按钮（确定添加）的响应事件

$('#popur-heat-map ul li div').click(**function**(){函数体}

获取input控件的的内容值

$('#popur-heat-map ul li').find('input')[0].value

动态设置input控件的属性值

$('#heatmap\_add\_text\_name').attr("style","color:#999;");

动态设置元素<p>的内容



①$('#page\_setting\_pv').text("带来浏览量");

②document.[getElementById("page\_setting\_pv")](http://www.w3school.com.cn/jsref/met_doc_getelementbyid.asp).innerHTML = "带来浏览量";

根据id去获取

1. **获取键盘键的按下事件**

①回车按下事件

document.onkeydown=**function**(event){

**var** e = event || window.event || arguments.callee.caller.arguments[0];

**if**(e && e.keyCode==13){ // enter 键

}

};

②ESC按键

**if**(e && e.keyCode==27){ // 按 Esc

//要做的事情

}

③F2按键

**if**(e && e.keyCode==113){ // 按 F2

//要做的事情

}

1. **html元素的id与class的区别**

$('#page\_setting\_pv')这个是根据id值去获取一个元素，这样获得的元素只能而且是唯一的一个，不会获取多个。

$('.page\_setting\_pv')这个是根据元素class去获取，这样得到的元素可以是多个，只有某元素的class属性是page\_setting\_pv这条语句就可以获取到该元素

1. **对于jquery根据id或者class获取标签的问题**

如果id或者class里面含有动态的变量：

<table id=*"risk-table-day"* class=*"risk-table risk-table-width"*

cellpadding=*"1"*>

<tbody>

<tr>

<td><div class=*" ui-checkbox"*>

<input data-role=*"none"* value=*"checkbox"* style="float: *left*;"

class=*"risk-close-position-check"* +num type=*"checkbox"*>

</div>

<span>平仓线：</span>

</td>

</tr>

<tr class=*"risk-close-position1"* style="display: *table-row*;">

<td><span style="margin-left: *0px*;">平仓方式：</span></td>

<td><div class=*" ui-radio"*>

<input data-role=*"none"* value=*"checkbox"* type=*"radio"*>

</div>

<span>全平</span>

</td>

</tr>

</tbody>

</table>

这里就不能使用两级标签去获取，不能包含父标签，只能直接通过他自身去获取，如下就是错误的：

$('#*risk-table-day* .risk-close-position-check'+num).click(**function**(){

// alert("dianjil");

**if**(**this**.checked){

$('#*risk-table-day* .risk-close-position'+num).show();

}**else**{

$('#*risk-table-day* .risk-close-position'+num).hide();

}

});

正确的应该是：

$('.risk-close-position-check'+num).click(**function**(){

// alert("dianjil");

**if**(**this**.checked){

$('.risk-close-position'+num).show();

}**else**{

$('.risk-close-position'+num).hide();

}

});

1. **jquery得到多个控件，并且获取他们的某些属性值**

**问题描述：**在一个表格下，有多个checkbox，想获取这些checkbox哪些被用户点击选择了，并且获取这些被选择的checkbox的id值。

**解决方法1：**

/\*\*

\* 此方法用于展示网站的列表

\* **@param** obj

\*/

**function** setSiteTable(obj){

**var** html = "";

html = html + "<tr class=\"table-th-black\">";

html = html + "<th></th>";

html = html + "<th>" + siteTableTitle.name + "</th>";

html = html + "</tr>";

**for** (**var** v = 0; v < obj.length; v++) {

html = html + "<tr>";

html = html + "<td><input type=\"checkbox\" id=\""+obj[v].id+"\" class=\"site-list-permission\"/></td>";

html = html + "<td>"+obj[v].name+"</td>";

html = html + "</tr>";

}

$('#site-list-table').html(html);

}

/\*\*

\* 此方法用于获取用户点击权限，并将用户所选择的拼接成一个字符串

\* **@returns** {String} 返回的结果

\*/

**function** getPermissionStr(){

**var** permissionStr = "";

//获取class为site-list-permission的控件的个数 **关键代码：**

**var** size = $('.site-list-permission').size();

**for** (**var** v = 0; v < size; v++) {

**if**( $('.site-list-permission')[v].checked){

permissionStr = permissionStr + $($('.site-list-permission')[v]).attr("id");

permissionStr = permissionStr + ";";

}

}

//去掉末尾的‘;’字符

permissionStr = permissionStr.substring(0,permissionStr.length-1);

**return** permissionStr;

}

1. **Jquery对象与document对象间的转化**

描述：由于Jquery对象与document对象不一样，能使用的方法也不尽相同，因此有时我们需要用到document对象的特有方法时就需要将Jquery对象转化为document对象，比如：

如下得到的是一个input的text对象，要想设置该文本框的值，使用Jquery对象方法如下

$('#risk-positionmethod-text').val(data[5]);

如果我们想使用document对象的方法时则需要将Jquery对象转化为document对象，如何转化呢？这个很简单，应该Jquery对象中保存的是document对象的数组，我们在调用Jquery对象的方法时，实际上是对Jquery对象数组里的document对象进行方法调用，因此Jquery的对象不能调用document对象的方法，是因为document对象是针对单个标签的，Jquery对象已经封装好了document对象的方法。我们要想调用document对象的方法只需要取出Jquery对象中的document对象，具体实现如下：

$('#risk-positionmethod-text')[0].value = data[5];

1. **关于设置标签属性的方法attr和prop**

prop()方法是Jquery 1.6版本以后推出的一个新方法，在以前关于attributes和properties都是使用attr()方法，不过在Jquery 1.6以后为了修复attr()的一些bug，加入了prop()方法，因此在Jquery1.6以后，有些地方就不能再使用attr()方法，只能使用prop()方法，要不然就达不到我们想要的效果。

如下列举一些一般使用的情况：

$(window).prop();

$(document).prop();

window或document中使用.attr()方法在jQuery1.6中不能正常运行，因为window和document中不能有attributes。它们包含properties(比如：location或readyState)，必须使用.prop()方法操作或简单地使用javascript原生的方法。在jQuery1.6.1中，window和document中使用.attr()将被自动转成使用.prop，而不是抛出一个错误。

$(':checkbox').prop();

$('option').prop();

properties就是浏览器用来记录当前值的东西。当用户点击一个checkbox元素或选中一个select元素的一个option时，boolean properties保持最新。但相应的boolean attributes是不一样的，正如上面所述，它们仅被浏览器用来保存初始值。

$('#risk-radio-position-rate'+num).prop("checked","checked");或者

$('#risk-radio-position-rate'+num).prop("checked",true);

$('#risk-radio-position-rate'+num).prop("checked",false);

如何选择：

prop()方法应该被用来处理boolean attributes/properties以及在html(比如：window.location)中不存在的properties。其他所有的attributes(在html中你看到的那些)可以而且应该继续使用.attr()方法来进行操作

1. **如何动态显示和隐藏某些控件**

**其中一种方法** 涉及到两个方法**（建议使用这种方法，这样不会存在布局变化的问题）**

1. show(), $('.popur-layer').show();将class为popur-layer的元素显示出来
2. hide(), $('.popur-layer').hide();将class为popur-layer的元素隐藏

**另外一种方法：（该种方法可能存在标签显示和隐藏出现布局变化的问题）**

**var** box=$('#popur-delete-data');

box.css('display','none');

box.css('display','block');//把id值为popur-delete-data的元素显示出来

得到id值为popur-delete-data的元素，并用css将他的display属性设置为none，表示隐藏元素

**第三种方法：**设置元素的属性visibility



1. **Radio是否被选中以及改变图标的选中状态**

/\*\*

\* 此方法用于设置radio是否被选中

\*/

**function** riskChangeRadioStyle(t){

**if**($(t).val()==1){

$(t).attr("value","0");

$(t).removeAttr("checked");

} **else** {

$(t).val("1");

$(t).attr("checked","checked");

}

}

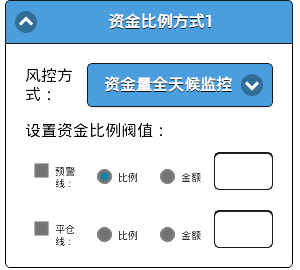
$("tr td .risk-radio-button").click(**function**(){

riskChangeRadioStyle(**this**);

});

html = html + "<td>" +

"<input type=\"radio\" value=\"checkbox\" class=\"risk-radio-button\"/>";



1. **Check box是否被选中**

根据元素的t.checked去获取，如果该值为true说明被选中，如果为false则说明未选中

$('.risk-close-position-check'+num).click(**function**(){

// alert("dianjil");

**if**(**this**.checked){

$('.risk-close-position'+num).show();

}**else**{

$('.risk-close-position'+num).hide();

}

});

html = html + "<td>" +

"<input type=\"checkbox\" value=\"checkbox\" style=\"float: left;\" class=\"risk-close-position-check"+num+"\" onclick=\"checkBox(this)\"/>";

1. **动态设置checkbox被选中或者取消选中**

**方法1：使用jquery方法**

**未选中：**

$("#add-account-table tr td input[type='checkbox']").attr("checked",**false**);

**选中：**

$("#add-account-table tr td input[type='checkbox']").attr("checked",**true**);

1. **判断浏览器是否支持placeholder属性，该属性作用请查询html官方文档**

/\*\*

\* 判断浏览器是否支持placeholder

\*/

**function** isPlaceholer(){

**var** input = document.createElement('input');

**return** "placeholder" **in** input;

}

1. **设置时间格式，返回2014/06/14的形式的字符串**

/\*\*

\* 格式化date类型，以特定方式显示出来

\*/

**function** formatDate(date) {

**var** y = date.getFullYear();

**var** m = date.getMonth()+1;//获取当前月份的日期

**var** d = date.getDate();

**var** M = m >= 10? m:("0"+m);

**var** D = d >= 10? d:("0"+d);

**return** y+"/"+M+"/"+D;

}

1. **格式化时间，显示效果2014-06-14 00:00:00**

/\*\*

\* 2014-05-09 00:00:00

\*/

**function** formatDate(date)

{

**var** dateStr = padStr(date.getFullYear()) +'-'+

padStr(1 + date.getMonth()) +'-'+

padStr(date.getDate()) +' '+

padStr(date.getHours()) +':'+

padStr(date.getMinutes()) +':'+

padStr(date.getSeconds());

**return** dateStr;

}

1. **格式化时间，显示效果是00:00:00，传进的参数是秒**

/\*\*

\* 此方法用于格式化平均访问时长

\* **@param** timeData秒

\* **@returns** {String}

\*/

**function** getTimeFormat(timeData)

{

**if**(timeData >= 60)

{

theTime1 = parseInt(timeData/60);

theTime = parseInt(timeData%60);

**if**(theTime1>=60)

{

theTime2 = parseInt(theTime1/60);

theTime1 = parseInt(theTime1%60);

**if**(theTime2 > 0)

{

theTime2 = (theTime2 >= 10? theTime2:("0"+theTime2));

theTime1 = (theTime1 >= 10? theTime1:("0"+theTime1));

theTime = (theTime >= 10? theTime:("0"+theTime));

**return** theTime2+":"+theTime1+":"+ theTime;

}

**else**

{

theTime1 = (theTime1 >= 10? theTime1:("0"+theTime1));

theTime = (theTime >= 10? theTime:("0"+theTime));

**return** "00:"+theTime1+":"+theTime;

}

}

**else**

{

theTime1 = (theTime1 >= 10? theTime1:("0"+theTime1));

theTime = (theTime >= 10? theTime:("0"+theTime));

**return** "00:"+theTime1+":"+theTime;

}

}

**else**

{

timeData = (timeData >= 10? timeData:("0"+timeData));

**return** "00:00:"+timeData;

}

}

1. **将毫秒转化为特定的日期格式，如：2014-07-31 07:00:00**

/\*\*

\* 此方法用于格式化时间

\*/

**var** formatTime = **function**(time, format){

**var** t = **new** Date(time);

**var** tf = **function**(i){**return** (i < 10 ? '0' : '') + i};

**return** format.replace(/yyyy|MM|dd|HH|mm|ss/g, **function**(a){

**switch**(a){

**case** 'yyyy':

**return** tf(t.getFullYear());

**break**;

**case** 'MM':

**return** tf(t.getMonth() + 1);

**break**;

**case** 'mm':

**return** tf(t.getMinutes());

**break**;

**case** 'dd':

**return** tf(t.getDate());

**break**;

**case** 'HH':

**return** tf(t.getHours());

**break**;

**case** 'ss':

**return** tf(t.getSeconds());

**break**;

}

});

}

使用：formatTime(data[v].time,'yyyy-MM-dd HH:mm:ss')

1. **校验用户输入的时间格式是否为07:00或者7:00**

/\*\*

\* 此方法用于检查用户输入时间段的合法性，合法则返回true，不合法返回false;

\* **@param** str需要检查合法性的时间字符

\* **@returns** {Boolean}

\*/

**function** riskControlCheckTime(str){

**var** flag = **false**;

**var** r = str.match(/^(\d{1,2}):(\d{1,2})$/);

**if**(r!=**null**){

//比如07:59,冒号前面的数字必须小于24，而冒号后面的数字必须小于60

**if**(**new** Number(str.split(":")[0])<24&&**new** Number(str.split(":")[1])<60){

flag = **true**;

}

}

**return** flag;

}

1. **Js中让程序延时的方法=Thread.sleep()**

/\*\*

\*此方法用于 让线程休息一会儿

\* **@param** numberMillis

\*/

**function** sleep(numberMillis) {

**var** now = **new** Date();

**var** exitTime = now.getTime() + numberMillis;

**while** (**true**) {

now = **new** Date();

**if** (now.getTime() > exitTime)

**return**;

}

}

1. **获取特定某天的日期**

/\*\*

\* 获取当天前的多少天

\*d代表特定时间

\*t代表天数

\*/

**function** getNextDay(d,t){

d = **new** Date(d);

d = d - 1000\*60\*60\*24\*t;

d = **new** Date(d);

**return** d.getFullYear()+"-"+(d.getMonth()+1)+"-"+d.getDate();

}

例子：想获取‘2014-07-14’的第前五天的日期

getNextDay("2014-07-14",5);

1. **Date类型间对比大小**

方法1：Date.parse(new Date())将Date类型转化可比较大小是数据类型

**if**((Date.parse(starttime)-Date.parse(getCurrentDay()))==0&&(Date.parse(endtime)-Date.parse(getCurrentDay()))==0){

starttime = dateAddday(getCurrentDay(),-1);

endtime = dateAddday(getCurrentDay(),-1);

}

**对比月份是否相同：**

if(dt1.getMonth()==dt2.getMonth())  
 {

方法2：

Starttime.getTime()>endtime.getTime();

1. **处理url显示格式，如果长度超过50，则50以后的用…代替**

/\*\*

\* 处理表格有url显示的方式

\* **@param** url

\* **@returns** {String}

\*/

**function** getLimitUrl(url)

{

**var** length = 50;

**var** tempUrl;

**if**(url.length > length)

tempUrl = url.substr(0, length) + "...";

**else**

tempUrl = url;

**return** tempUrl;

}

1. **对于<a>标签超链接属性href值的问题**
2. **如果值是http打头的**

值是http开头则直接赋值就行

**<a href=***"http://www.baidu.com"***></a>**

1. **如果值是www开头的**

值是www开头则需要在前面加双斜杠，要不然浏览器会加上默认的工程http前缀，这样的话显示的url就不对了

**<a href=***"//www.baidu.com"***></a>**

1. **Html路径问题**

**问题描述：**当我们在一个web项目下，对数据进行ajax请求的时候，涉及到一个路径的问题（也就是请求的url问题），还有就是当我们进行页面跳转的时候，如果进行跳转的两个页面不在同一个包下面，那么也会涉及到一个相对路径的问题，如果我们每次都自己去根据相对路径进行手动输入这个url。如果以后发布的ip地址改变，我们还得再次手动修改，那么会相当麻烦。

**思路：**我们可以将基本的web项目的url动态获取到，保存到一个变量中，然后在这个基本的url基础上再去加上具体的路径，这样即使ip改变也不会影响web程序，我们以后也无需再次修改，除非jsp页面的位置变动。

**方法1：**动态获取到web项目的url，包括端口号，保存到一个js变量中，这样我们在js文件中就可以随便使用了。

代码如下：head.jsp文件的所有代码，以后要使用可以直接将这个jsp文件引用过去就行了。

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"* />

<meta http-equiv=*"X-UA-Compatible"* content=*"IE=edge,chrome=1"* />

<meta name=*"viewport"* content=*"width=device-width, initial-scale=1.0"*>

<title>国金期货</title>

<!-- 下面的Java代码是将动态获取到请求的url，然后保存到basePath变量中 -->

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+ ((request.getServerPort()==80)?"":":"+request.getServerPort()) +path+"/";

%>

<script src=*"*<%=basePath %>*jquery-1.6.1.min.js"* charset=*"utf-8"*></script>

<!-- 下面的js代码是将调用服务器的java变量，并声明一个js变量来保存basePath变量中 的值，然后在js中就可以使用该变量了-->

<script type=*"text/javascript"* charset=*"utf-8"*>

**var** basePath = "<%=basePath %>";

$(**function**(){

getData();

});

/\*\*

\* 向服务器发送风控消息获取警告或者平仓数据的请求

\*/

**function** getData()

{

$.ajax({

url:basePath+"message/get",//在这个地方使用basePath变量，这样就不用再去考虑什么相对地址，这样方便后期修改

contentType:"application/x-www-form-urlencoded; charset=utf-8",

dataType: 'json',

type:"get",

data:{token:token,clientID:accountId,type:riskControlInfoType},

cache:**false**,

async:**false**,

success:**function**(response){

}

});

}

</script>

</head>

<body>

<div>

<input type=*"submit"* value=*"开始"* onclick="start()" />

</div>

<div id=*"messages"*><%=basePath %></div>

</body>

</html>

1. **http请求相对路径的问题**

**问题描述：**

假如我们有一个项目: MyApp 在该项目下, 有一个jsp文件夹 该文件夹下包括:  
login.jsp    // 登陆页面  
register.jps  // 注册页面  
我们在浏览器中输入地址 (注意: 地址的内容): http://localhost:8080/MyApp/jsp/login.jsp  
这时候, 浏览器会链接到 "登陆页面" (login.jsp)  
在login.jsp文件内包含了如下 "部分代码": <a href="jsp/register.jsp">注册用户 </a>  
那么,如果我们点击这个链接,就会在浏览器地址栏中, 出现如下错误链接: http://localhost:8080/MyApp/jsp/jsp/register.jsp  
为什么会出现"/jsp/jsp/register.jsp"呢?  
因为, 网页中的"相对链接", 是相对于你所 "请求的URL路径" 所决定的. 即:  
因为这里请求路径是:http://localhost:8080/MyApp/jsp/login.jsp  
那么, 浏览器 就会在这个路径下(即:http://localhost:8080/MyApp/jsp/)去找 jsp/register.jsp  
所以就会出现如下错误内容: <http://localhost:8080/MyApp/jsp/jsp/register.jsp>

**解决方法1：**

**使用绝对路径**

**缺点：每次使用都需要加上**${pageScope.basePath}，比较麻烦

**<**%@ page language="java" pageEncoding="GBK" contentType="text/html;charset=gbk" isELIgnored="false"%**>**

**<**%

String path = request.getContextPath();

// 获得本项目的地址(例如: http://localhost:8080/MyApp/)赋值给basePath变量

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

// 将 "项目路径basePath" 放入pageContext中，待以后用EL表达式读出。

pageContext.setAttribute("basePath",basePath);

%**>**

**<html>**

**<head>** **</head>**

**<body>**

**<a** href="${pageScope.basePath}jsp/register.jsp"**>**

**</body>**

**</html>**

**解决方法2：**

**优点：这是一个比较通用的方法，能完美解决上面的问题**

**代码如下：**

**<**%@ page language="java" pageEncoding="GBK" contentType="text/html;charset=gbk" isELIgnored="false"%**>**

**<**%

String path = request.getContextPath();

// 获得项目完全路径（假设你的项目叫MyApp，那么获得到的地址就是 http://localhost:8080/MyApp/）:

String basePath = request.getScheme()+"://"+request.getServerName()+":"+request.getServerPort()+path+"/";

%**>**

**<html>**

**<head>**

<!-- base需要放到head中 -->

**<base** href=" <%=basePath%>"**>**

**</head>**

// 这里我们就可以直接使用相对路径(即: 相对于base标签)

**<a** href="jsp/login.jsp"**>**Login **</a>**

**</html>**

还可以如下这样，到时在js代码中就可以直接用basePath进行操作

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

**var** basePath = "<%=basePath %>";

</script>

1. **判断字符串是否是URl格式的**

**解决方法：使用正则表达式**

//用正则表达式匹配url是否合法

//其中url是需要判断的字符串

str = url.match(/http:\/\/.+\..+/i);

str1 = url.match(/www.+\..+/i);

如果满足url格式，则str或者str1不为null

因此判断str,str1是否为null就知道url是否满足格式

1. **针对乱码问题，转码**

1.如果使用ajax发送请求的话，请求类型是GET，则controller层接收到的数据编码是ISO-8859-1，因此如果数据中有中文的话，最好将中文转为utf-8

$.ajax({

url:basePath+"/system/mail",

contentType:"application/x-www-form-urlencoded; charset=UTF-8",

type:"GET",

dataType:"json",

cache:**false**,

data: {

// name:name,url:url,operateType:operateType,id:id,

mailName:"转化页设置",mailAddr:addr},

success:**function**(response){

}

});

转码代码为：

//将ajax请求传来的参数mailName编码转为utf-8

mailName = new String(mailName.getBytes("ISO-8859-1"),"utf-8");

2.如果使用的是post请求，那么参数则不需要转码，mailName不需要在controller层进行转码

**var** data = {mailName:"转化页设置",mailAddr:addr};

$.post(basePath+"/system/mail",data,**function**(result){

//根据函数返回值的不同进行不同处理，返回值是由controller请求处理层的函数返回的

**if**(result=="1"){

$('.popur-box').hide();

$('.popur-layer').hide();

$('#popur-send-email-success').css('display','block');

}

**else**{

$('.popur-box').hide();

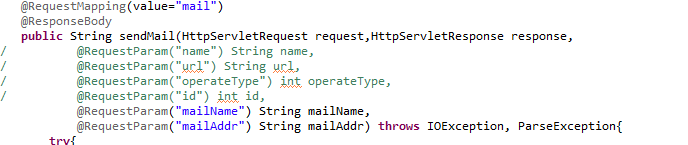
$('.popur-layer').hide();

$('#popur-send-email-false').css('display','block');

}

});

result返回值就是下面函数的返回值



1. **针对url参数传递中文乱码问题**

**问题描述：url中有中文，接收到该参数时会出现乱码**

[**http://localhost:8080/naomi-lzhd-web/page/page-heat-map-detail?name**](http://localhost:8080/naomi-lzhd-web/page/page-heat-map-detail?name)**=海尔商城&pv=43&click=0**

**解决方法1：**

**发送端：**

**//** page\_url传输的url

**var** page\_url = "page-heat-map-detail?name="+**encodeURI(obj[i]["url\_name"])**+"&pv="+obj[i].pv+"&click="+obj[i].click\_times;

//传输之前进行转码

//注意：这地方是先将可能存在乱码的中文先转码再整个转码

page\_url = encodeURI(page\_url);

html = html + "<td><a href="+page\_url+" target=\"\_blank\">查看热力图</a></td>";

**接收端：**

//进行解码

name = java.net.URLDecoder.*decode*(name ,"UTF-8");

1. **页面跳转**

**问题描述：在两个页面间实现跳转**

**方法1：直接使用html的跳转方法，在页面间进行跳转。**

**优点：比较方便**

**缺点：不能传递参数**

**实现：index.html是个相对路径**

window.location.href="index.html";

**方法2：通过后台进行跳转，此方法进行传参比较方便**

**此方法参考问题（利用js在两个jsp页面间传递参数）**

1. **利用js在两个jsp页面间传递参数**

**方法1：**

a.jsp发送一个带参数的url请求（请求b.jsp），b.jsp在controller层接收到参数，然后将参数加入到页面对象中，在页面中取得参数，这样就可以在b.jsp的js代码中使用a.jsp传来的参数了

具体实现如下：

此段代码是在<a>中设置href属性值，其中是带有参数url的请求

a.jsp中的js代码

$(*'#page\_down\_href'*).attr(*"href"*,basePath+*"page/page-down-detail?url="*+url);

下面这段代码是接收到<a>的超链接请求,并获得请求的参数，将参数加入到页面对象

b.jsp中的controller层代码

/\*\*

\* 此方法用于映射访问该页面的路径，比如一个页面visitors-loyalty.jsp，它放在client包文件夹下

\* 那么该类名上面的映射是文件的父路径，访问该文件路径为：localhost:8080/.../client/visitors-loyalty

\*/

@RequestMapping(value="page-down-detail")

**public** ModelAndView show(HttpServletRequest request,HttpServletResponse response,

@RequestParam("url") String url){

ModelAndView mov=**new** ModelAndView("/show/page-down-detail"

+ "");

//将参数加入到页面对象

mov.addObject("url", url);

// System.out.println("显示页面上下游详细页面");

**return** mov;

}

此代码用于获取页面对象中的属性值

B.jsp中的代码

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

**var** pageDownUrl;

$(**function**(){

pageDownUrl='${url}';

});

</script>

最后就可以在b.jsp中的js代码直接使用该参数的值

利用获得的参数去请求数据库的值

getData(starttime, endtime,type,pageDownUrl);

1. **格式化数字，保留t位小数**

/\*\*

\* 格式化数字

\* **@param** data要格式化的数字

\* **@param** t保留多少位小数

\* **@returns**

\*/

**function** getDataFormat(data,t)

{

**return** data.toFixed(t);

}

1. **从字符串中提取数字**

//text是需要提取数字的字符串

var value = text.replace(/[^0-9]/ig,"");

1. **Jsp中数字的处理方法**

Math.ceil(number) //返回大于等于其数值参数的最小整数。

比如Math.ceil( 2.5)得到的结果是3

Math.floor(number) //返回小于等于其数值参数的最大整数。

比如Math.floor(2.5)得到的结果是2

Math.round(number) //四舍五入

获取过滤数字

var str="\_\_20\_\_40\_\_43\_\_on\_\_";   
var strAry=str.split("\_\_");//得到一个20 40 43的数组   
for(var i=0;i<strAry.length;i++)   
alert(strAry[i]);

1. **验证变量是否是数字**

Js中验证a是否是数字，如果是则返回true，不是则返回false

isNaN(变量名a);

1. **处理数字，将其变为每三位用逗号隔开的格式，比如：33,444.666**

**方法1： 这种方法比较简单明了，不过在IE下会存在问题，永远都会出现小数点后两位，不管是否存在小数。比如234,，处理后的结果为234.00；另外0也会出现问题，0被处理后出现.00的结果。**

/\*\*

\* 此方法用于处理数字，弄成每三位逗号隔开，格式：100,444,333,455.55

\* **@returns**

\*/

**function** handleNumber(num){

**return** parseFloat(num).toLocaleString();

}

**方法2：该方法稍微复杂了点，不过在三大浏览器都不会出现上面问题，能完美解决兼容性。**

/\*\*

\* 此方法用于处理数字，弄成每三位逗号隔开，格式：100,444,333,455.55

\* **@returns**

\*/

**function** handleNumber(num){

**var** b = parseInt(num).toString();

**var** len = b.length;

**if**(len <= 3){

**return** b;

}

**var** r = len%3;

**return** r>0?b.slice(0,r)+","+b.slice(r,len).match(/\d{3}/g).join(","):b.slice(r,len).match(/\d{3}/g).join(",");

}

1. **验证邮箱格式是否正确**

/\*\*

\* 此方法用于校验邮箱格式是否正确

\* **@param** email 需要校验的邮箱字符串

\* **@returns** {Boolean} 返回true表示校验成功、符合格式要求，false表示校验不成功

\*/

**function** checkEmail(email){

**if** (email.match(/^\w+([-+.]\w+)\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$/)==**null**) {

**return** **false**;

}**else** {

**return** **true**;

}

}

**/^\w+([-+.]\w+)[\*@\w+([-.]\w+)\*\.\w+([-.]\w+)\*$/](mailto:*@\\w+(%5b-.%5d\\w+)*\\.\\w+(%5b-.%5d\\w+)*$/)**

**这段正则的意思是：邮箱地址 必须由 大小写字母 或 数字 或下划线开头，其后可以跟上任意的 \w字符 和 中划线 加号 英文句号 @ 跟上任意的 \w字符 和 中划线 加号 英文句号 . 跟上任意的 \w字符 和 中划线 加号 英文句号**

1. **数组排序**

**var** contentListIndex = **new** Array();

从小到大排序：

contentListIndex.sort(**function**(a,b){**return** a>b?1:-1});

从大到小排序：

contentListIndex.sort(**function** (a,b){ **return** a<b?1:-1});//从大到小排序

1. **动态画柱状图**

/\*

\* 设置年龄数据的横向柱状图

\*/

**function** setAgeBarChart(obj)

{

**var** index\_b = **new** Array();

**var** data = **new** Array();

**var** len = obj.length;

//设置条目

**var** totalCount = 0;

index\_b[0] = age\_type["ten\_to\_nineteen"];

index\_b[1] = age\_type["twenty\_to\_twentynine"];

index\_b[2] = age\_type["thirty\_to\_thirtynine"];

index\_b[3] = age\_type["fourty\_to\_fourtynine"];

index\_b[4] = age\_type["over\_fifty"];

**for**(**var** v = 0; v < len; v++)

{

data[v] = obj[v].visitorAttributeIndex[index["v\_uv"]];

data[v] = Number(data[v]);

totalCount = totalCount + data[v];

}

$('#table\_line\_age').highcharts({//关联到html页面，id为table\_line\_age的元素内画图

chart : {

height: 200,

defaultSeriesType: 'bar' //设置柱状图的显示方向，bar表示纵向显示，column表示横向显示

},

title: {

text: '',//不显示柱状图的标题

x: -20 //center

},

// subtitle: {

// text: '',

// x: -20

// },

xAxis: {

categories: index\_b,//["2014-03-27", "2014-03-28", "2014-03-29", "2014-03-30", "2014-03-31"]

labels: {

step:1,

style:{

fontSize:10

}

}

},

yAxis: {

title: {

text: ''

},

plotLines: [{

value: 0,

width: 1,

color: '#808080'

}],

labels: {

formatter: **function**() {//设置纵坐标显示的内容

**if**(totalCount!=0){

**return** getDataFormat((**this**.value/totalCount)\*100,1)+"%";

}

}

},

max: totalCount

},

tooltip: {

valueSuffix: '',

formatter:**function**(){

**var** s = '';

//计算占比

**var** y = **this**.y;

**var** rate = 0;

**if**(totalCount != 0)

{

rate = getDataFormat((y/totalCount)\*100,1);

}

**else**

rate = 0.0;

//console.log(this.series.yData[0]);

s += '<b>'+**this**.x+'</b>';

s += '<br/>'+sex["peoplenum"]+ **this**.y+ " | " + sex["rate"] + (rate+"%");

**return** s;

}

},

legend: {

// layout: 'vertical',

enabled:**false**,//设置柱状图下面的series元素为不显示

verticalAlign: 'bottom',

borderWidth: 0

},

colors: [

'#48BEF4'

],

series: [{

// name: "",

data: data

}]

});

// return;

}

<div class=*"row"*>

<div class=*"col-xs-12"* ></div>

</div>

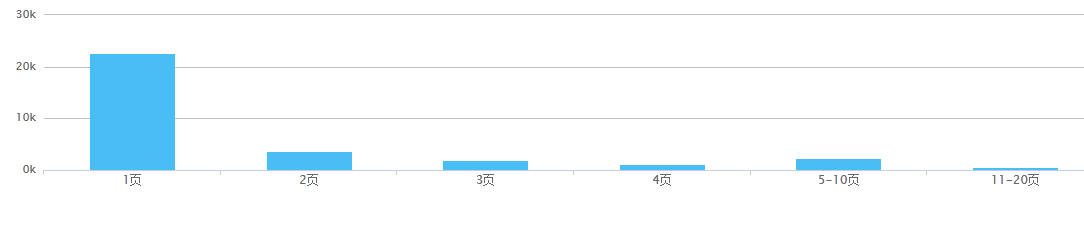
<div class=*"row"*>

<div class=*"col-xs-12"* id=*"table\_line\_age"* ></div>

</div>

</div>

效果如图所示：



1. **用js画矩形区域**

注：oCanvas.setCapture();

oCanvas.releaseCapture();

<style type=*"text/css"*>

*.box* {

background: *#2D679A*;

width: *0px*;

height: *0px*;

position: *absolute*;

opacity: *0.7*;

cursor: *move*;

width: *100%*;

border-style: *solid*;

border-width: *0px*;

}

</style>

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

**function** $Id(id)

{

**return** document.getElementById(id);

}

window.onload = **function**(e) {

e = e || window.event;

// startX, startY 为鼠标点击时初始坐标

// diffX, diffY 为鼠标初始坐标与 box 左上角坐标之差，用于拖动

**var** startX, startY, diffX, diffY;

// 是否拖动，初始为 false

**var** dragging = **false**;

**var** oCanvas=$Id('heatmap\_flash');

// 鼠标按下

$('#heatmap\_flash').mousedown(**function**(e) {

**if**(oCanvas.setCapture)

{

oCanvas.setCapture();

}

e = e || window.event;

startX = e.pageX;

startY = e.pageY;

**var** target = obj = e.target || e.srcElement;

// 如果鼠标在 box 上被按下

**if** (target.className.match(/box/)) {

// 允许拖动

dragging = **true**;

// 设置当前 box 的 id 为 moving\_box

**if** (document.getElementById("moving\_box") != **null**) {

document.getElementById("moving\_box").removeAttribute("id");

}

target.id = "moving\_box";

// 计算坐标差值

diffX = startX - target.offsetLeft;

diffY = startY - target.offsetTop;

} **else** {

areaNum++;

// 在页面创建 box

**var** active\_box = document.createElement("div");

active\_box.id = "active\_box";

active\_box.className = "box";

active\_box.style.top = startY + 'px';

active\_box.style.left = startX + 'px';

document.body.appendChild(active\_box);

}

// 鼠标移动

$('#heatmap\_flash').mouseup(**function**(e3) {

e3 = e3 || window.event;

**var** endX = e3.pageX;

**var** endY = e3.pageY;

**var** oLeft= e3.clientX;

**var** oTop= e3.clientY;

drawArea(e3);

setAlertData(endX,startY,startX);

// 禁止拖动

dragging = **false**;

**if** (document.getElementById("active\_box") !== **null**) {

**var** ab = document.getElementById("active\_box");

ab.removeAttribute("id");

// 如果长宽均小于 3px，移除 box

**if** (ab.offsetWidth < 3 || ab.offsetHeight < 3) {

document.body.removeChild(ab);

}

}

**if**(oCanvas.releaseCapture)

{

oCanvas.releaseCapture();

}

});

// 鼠标抬起

$('#heatmap\_flash').mousemove(**function**(e2) {

e2 = e2 || window.event;

**if** (document.getElementById("active\_box") != **null**) {

**var** ab = document.getElementById("active\_box");

ab.style.width = e2.pageX - startX + 'px';

ab.style.height = e2.pageY - startY + 'px';

}

// 移动，更新 box 坐标

**if** (document.getElementById("moving\_box") != **null** && dragging) {

**var** mb = document.getElementById("moving\_box");

mb.style.top = e2.pageY - diffY + 'px';

mb.style.left = e2.pageX - diffX + 'px';

}

});

};

});

</script>

</head>

<body style="height:*5000px*;">

<p>鼠标指针位于： <span></span></p>

<div class=*"clearfixed"* style="width:*2000px*;margin-left:*0px*;height:*10000px*;" id=*"heatmap\_flash"* ></div>

</body>

1. **动态生成表格**

/\*\*

\* 设置页面表格的内容

\*/

**function** setTable(obj)

{

// alert("设置表格数据");

**var** state;

// var ids = new Array();

**var** html = "<tr class=\"table-th-black\">"+

"<th></th>";

html= html + "<th>"+ table\_title["name"] +"</th>";

html= html + "<th>"+ table\_title["target"] +"</th>";

html= html + "<th>"+ table\_title["num"] +"</th>";

html= html + "<th>"+ table\_title["state"] +"</th>";

html= html + "<th>"+ table\_title["create\_time"] +"</th>";

html= html + "<th></th>";

html = html + "</tr>";

**for**(**var** i=0; i<obj.length;i++)//填充数据到表格

{

html = html + "<tr>";

html = html + "<td>"+(i+1)+"</td>";

html = html + "<td>" + obj[i]["name"] + "</td>";

//该条目是可点击的超链接

//alert(obj[i]["url"].substring(0,4));

**if**(obj[i]["url"].substring(0,4) == "http"){

html = html + "<td><a href=\""+obj[i]["url"]+"\" target=\"\_blank\" title=\""+obj[i]["url"]+"\">" + getLimitUrl(obj[i]["url"]) + "</a></td>";

}**else**{

html = html + "<td><a href=\"//"+obj[i]["url"]+"\" target=\"\_blank\" title=\""+obj[i]["url"]+"\">" + getLimitUrl(obj[i]["url"]) + "</a></td>";

}

html = html + "<td>" + obj[i]["trans"] + "</td>";

state = obj[i]["state"];

**if**(state==0){

html = html + "<td><div class=\"icon icon-playing\"></div><span>统计中</span></td>";

}**else**{

html = html + "<td><div class=\"icon icon-stop\"></div><span>已停止</span></td>";

}

html = html + "<td>" + obj[i]["createTime"] + "</td>";

//将每条数据的id值放进每个超链接的title属性里，方便以后删除操作获取该值

html = html + "<td><a href=\"#\" class=\"red-a pageseting\_delete\" title=\""+obj[i]["id"]+"\">删除</a></td>";

html = html + "</tr>";

}

$('#table-table').html(html); //关联到表格 <tbody id = "table-table" >

}

1. **表格分页的实现**

/\*\*

\* 分页实现

\*/

**var** pageNav = pageNav||{};

**var** pageNumber = 0;//表格条目的页数

pageNav.fn=**null**;

pageNav.pre="<<";

pageNav.next=">>";

/\*\*

\* 显示表格右下角的页数的数据

\*/

pageNav.nav=**function**(a,b)

{

**if**(1>=b)

**return** **this**.pn=**this**.p=1,**this**.pHtml2(1);

b<a&&(a=b);

**var** c="";

1>=a?a=1:(c+=**this**.pHtml(a-1,b,pageNav.pre),c+=**this**.pHtml(1,b,"1"));

**this**.p=a;

**this**.pn=b;

**var** d=2,e=6>b?b:6;4<=a&&(c+="<li><a>...</a></li>",d=a-2,e=a+2,e=b<e?b:e);

**for**(;d<a;d++)

c+=**this**.pHtml(d,b);

c+=**this**.pHtml2(a);

**for**(d=a+1;d<=e;d++)c+=**this**.pHtml(d,b);

e<b&&(c=c+"<li><a>...</a></li>"+**this**.pHtml(b,b));

a<b&&(c+=**this**.pHtml(a+1,b,pageNav.next));

**return** c;

};

pageNav.pHtml=**function**(a,b,c)

{

**return**" <li><a href='javascript:pageNav.go("+a+","+b+");' onclick ='onclick\_pagination("+a+")'>"+(c||a)+"</a></li> ";

};

pageNav.pHtml2=**function**(a)

{

//GetData(date\_start, date\_end, userGroupFlag, pageCount, Number(a));

**return**" <li class=\"active\"><a>"+a+"</a></li> ";

};

pageNav.go=**function**(a,b)

{

//将计算得到的页数显示到表格右下角的页面，根据id找到控件的显示位置

document.getElementById("pageNav").innerHTML=**this**.nav(a,b);

**null**!=**this**.fn&&**this**.fn(**this**.p,**this**.pn);

};

/\*

\* 分页实现 end

\*/

**var** pageCount = 20;

**var** pageTotal = 1; //一共多少页

**var** startPage = 1; // 第一页

**var** object;

/\*\*

\* 获取显示条目数，然后显示表格

\* **@param** obj

\*/

**function** setPageMaker(obj)

{

//获取每页显示数据条目数的点击事件

$('.pageCount').change(**function**(){

//得到用户选择的控件的数据

pageCount = $(**this**).find("option:selected").text();

//显示到表格

setFlowTable(obj,pageCount,1);

});

//计算一共多少页

pageTotal = (obj.length / Number(pageCount)).toFixed(0);

//计算最后页的显示数据条数

pageRemainder = (obj.length % Number(pageCount));

**if**( Number(pageRemainder) / Number(pageCount) < 0.5 && Number(pageRemainder) / Number(pageCount) >0)

{

pageTotal = Number(pageTotal) + 1;

}

//将页数显示到表格的右下角

pageNav.go(1,pageTotal);

}

/\*

\*响应用户点击页数的事件，并将表格刷新，重新显示数据

\*/

**function** onclick\_pagination(value)

{

// var nowPage;

**if** (value == '<<'){

startPage = Number(startPage) - 1;

}

**else** **if**(value == '>>'){

startPage = Number(startPage) + 1;

}

**else**{

startPage = value;

}

//startPage = value;

setFlowTable(object, pageCount, startPage);

}

/\*\*

\* 设置流量分析表格的内容

\*/

**function** setFlowTable(obj,pageCount,nowPage)

{

**var** start = (nowPage-1)\*pageCount;

**var** end = nowPage\*pageCount;

**if**(end>obj.length){//如果结束的位数超过了数据已有的长度，则end应该等于数据的长度

end = obj.length;

}

**var** html = "<tr class=\"table-th-black\">"+

"<th></th>";

html= html + "<th>"+ common\_title["url"] +"</th>";

html= html + "<th><span class=\"fl\">"+flow\_title["uv"]+"</span><!--div class=\"icon icon-arrow-down\"--></div></th>";

html= html + "<th>"+ flow\_title["ip"] +"</th>";

html= html + "<th>"+ flow\_title["stream"] +"</th>";

html= html + "<th>"+ common\_title["history"] +"</th>";

html = html + "</tr>";

**for**(**var** i=start; i<end;i++)//填充数据到表格

{

**此处省略**

}

}

//设置tab下面的数值

$('#entrance\_flow\_uv').text(obj[obj.length-1].v\_uv);

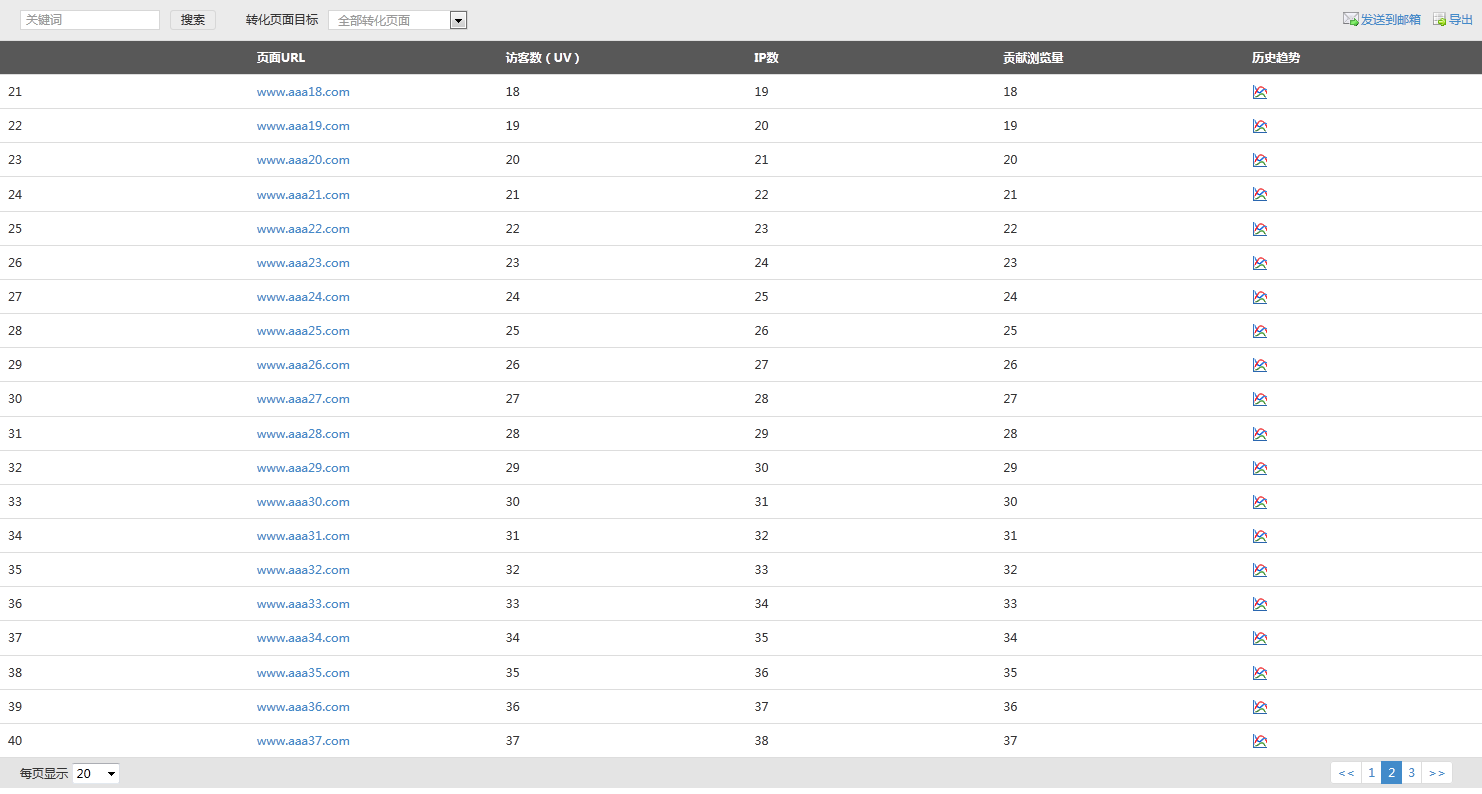
$('#entrance\_flow\_ip').text(obj[obj.length-1].v\_ip);

$('#entrance\_flow\_stream').text(obj[obj.length-1].v\_downstream\_pv);

$('#flow-table').html(html); //关联到表格 <tbody id = "table-table" >

}

效果如下图：



1. **Html中GET和POST请求的区别**

1. get是从服务器上获取数据，post是向服务器传送数据。

2. get是把参数数据队列加到提交表单的ACTION属性所指的URL中，值和表单内各个字段一一对应，在URL中可以看到。post是通过HTTP post机制，将表单内各个字段与其内容放置在HTML HEADER内一起传送到ACTION属性所指的URL地址。用户看不到这个过程。

3. 对于get方式，服务器端用Request.QueryString获取变量的值，对于post方式，服务器端用Request.Form获取提交的数据。

4. get传送的数据量较小，不能大于2KB。post传送的数据量较大，一般被默认为不受限制。但理论上，IIS4中最大量为80KB，IIS5中为100KB。

5. get安全性非常低，post安全性较高。但是执行效率却比Post方法好。

建议：

1、get方式的安全性较Post方式要差些，包含机密信息的话，建议用Post数据提交方式；  
 2、在做数据查询时，建议用Get方式；而在做数据添加、修改或删除时，建议用Post方式；

1. **针对获取html元素失败的问题**

1.超出了元素的作用范围去获取，肯定获取不到，情况如下：

如果元素不是动态生成的，则在$(function(){}函数里面获取是没有任何问题的。

如果元素是动态生成的，比如动态生成一个表格，如果要获取表格中的元素，则获取代码应该在表格生成代码之后，而且获取代码一定要在表格生成代码的函数中。具体结构如下所示：

/\*\*

\* 设置页面表格的内容

\*/

**function** setTable(obj)

函数体

{

生成表格的函数体

//响应用户的删除动作

$('#table-table').find('a').click(**function**(){

获取元素的函数体

}

注：该获取动作不能放进$(function(){}函数里面，因为超出了他的作用范围

}

1. **在jsp页面中导入CSS文件、JSP文件、Java文件、JS文件**
2. 导入CSS文件

其中basePath是相对路径，rel=*"stylesheet"*定义一个外部加载的样式表

<link rel=*"stylesheet"* href=*"*<%=basePath %>*css/common.css"*>

1. 导入JSP文件

当被导入的jsp文件与导入它的jsp文件在同一个包的时候，可以如下这样

<%@ include file=*"head.jsp"*%>

如果被导入的jsp文件与导入它的jsp文件不在同一目录下，则必须使用相对路径

示例：如a.jsp要导入b.jsp文件，b.jsp文件在a,jsp文件的父目录下，则代码如下：

<%@include file=*"../b.jsp"* %>

1. 导入Java文件

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<%@page import=*"com.naomi.haier.lzhd.web.admin.AdminManger"*%>

<%@page import=*"java.util.\*"*%>

1. 导入JS文件

type=*"text/javascript"表示文件类型*

<script type=*"text/javascript"* src=*"*<%=basePath %>*js/util/summary.js"*></script>

<script src=*"*<%=basePath %>*js/pageAnalysis/visited-pages.js"*></script>

1. **利用Spring MVC，从jsp向服务器发送请求**

**JSP端：**

/\*\*

\* 向数据库发送数据请求参数，再获取数据显示到页面

\*/

**function** GetData(date\_time,name,url,operateType,id)

{

// alert("请求数据");

**var** date\_time = **new** Date(date\_time);

$.ajax({

url:basePath+"/system/pageSettings",

contentType:"application/x-www-form-urlencoded; charset=utf-8",

type:"GET",

dataType:"json",

cache:**false**,

data: {date\_time:date\_time.getTime(),name:name,url:url,operateType:operateType,id:id},

success:**function**(response){

// console.log(response);

obj = response[0].pageSettingsList;

setTable(obj);//设置表格

}

});

}

**Controller端：**

/\*\*

\* 此方法用于热力图页面请求数据

\* **@param** request

\* **@param** response

\* **@param** start\_time开始时间

\* **@param** end\_time结束时间

\* **@param** name网站名称

\* **@param** url网站url

\* **@param** operateType操作类型

\* **@return**

\* **@throws** JsonParseException

\* **@throws** JsonMappingException

\* **@throws** IOException

\* **@throws** ParseException

\*/

@RequestMapping(value="/pageHeatMap")

@ResponseBody

**public** String getPageHeatMapData(HttpServletRequest request, HttpServletResponse response,

@RequestParam("id") **int** id,

@RequestParam("start\_time") String start\_time,

@RequestParam("end\_time") String end\_time,

@RequestParam("name") String name,

@RequestParam("url") String url,

@RequestParam("operateType") **int** operateType)

**throws** JsonParseException, JsonMappingException, IOException, ParseException{

siteId = AdminManger.*GetUserCurSite*(request.getSession());

classId = AdminManger.*GetUserCurClass*(request.getSession());

//将ISO-8859-1转码成utf-8

name = **new** String(name.getBytes("ISO-8859-1"),"utf-8");

phmRequest = getRequest(id,start\_time,end\_time,name,url,operateType,siteId,classId);

pageHeatMapResponse = pageHeatMapSvc.getPageHeatMap(phmRequest);

str = mapper.writeValueAsString(**new** Object[]{pageHeatMapResponse});

// System.out.println("请求返回的:"+str);

**return** str;

}

1. **页面加载完成后，调入js动态代码的入口函数**

1.//此函数是当页面加载完成后，动态执行js代码时是从该函数开始执行

优点：使用JQuery，编码方便

缺点：如果页面有图形，那么可能存在图形未加载完，页面就已经显示出来，但是显示的效果完全不对，因为这时的图片的宽高可能会不起作用，从而打乱了布局

$(document).ready(**function**() {

//函数体

//一般一些页面初始化的代码会放在该函数里面

});

上面的代码也可以简写成这样

$(**function**(){

}

2.下面方法可以解决$(document).ready(**function**() {}函数的上述缺点

$(window).load(**function**(){

//此函数里面一般放入动态生成图形的代码，比如动态生成柱状图，表格等等

});

1. **$(window).load(function(){}和$(document).ready(function() {}的区别**

①执行时间

**$(window).load(function(){}**必须等到页面内包括图片的所有元素加载完毕后才能执行。  
        $(document).ready()是DOM结构绘制完毕后就执行，可能 DOM 元素关联的内容并没有加载完 。这样的话就可能出现页面显示的效果不对情况 。**$(window).load(function(){}**则不会

**②** 编写个数不同

window.onload不能同时编写多个，如果有多个window.onload方法，只会执行一个

以下代码无法正确执行：   
window.onload = function(){   
   alert(“text1”);   
};   
window.onload = function(){   
   alert(“text2”);   
};

结果只输出第二个

 $(document).ready()可以同时编写多个，并且都可以得到执行

以下代码正确执行：   
$(document).ready(function(){   
   alert(“Hello World”);   
});   
$(document).ready(function(){   
   alert(“Hello again”);   
});   
结果两次都输出

③简写手法

window.onload没有简化写法

$(document).ready(function(){})可以简写成$(function(){});

1. **获取屏幕宽高**

屏幕分辨率的高

window.screen.height

屏幕分辨率的宽

window.screen.width

屏幕可用工作区高度 不包括任务栏

window.screen.availHeight

屏幕可用工作区宽度 不包括任务栏

window.screen.availWidth

网页正文全文宽

document.body.scrollWidth

网页正文全文高

document.body.scrollHeight

网页可见区域宽(包括边线的宽)

document.body.offsetWidth

网页可见区域高(包括边线的宽)

document.body.offsetHeight

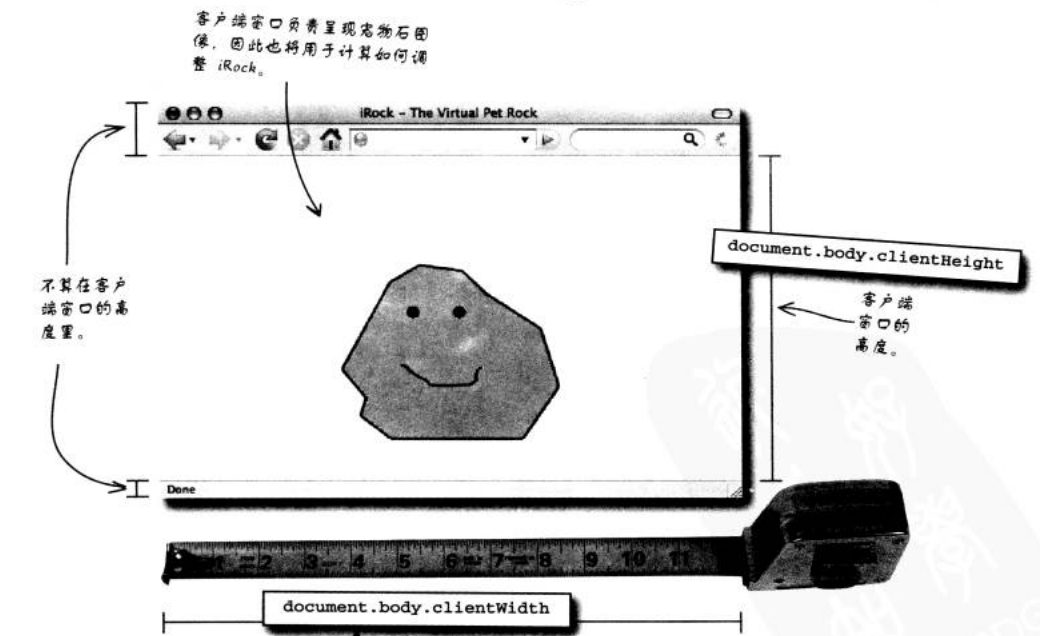
客户端窗口只是浏览器呈现网页的那部分窗口

网页可见区域宽

document.body.clientWidth

网页可见区域高

document.body.clientHeight



1. **Js中将变量转为可进行计算的数字(Number类型)**

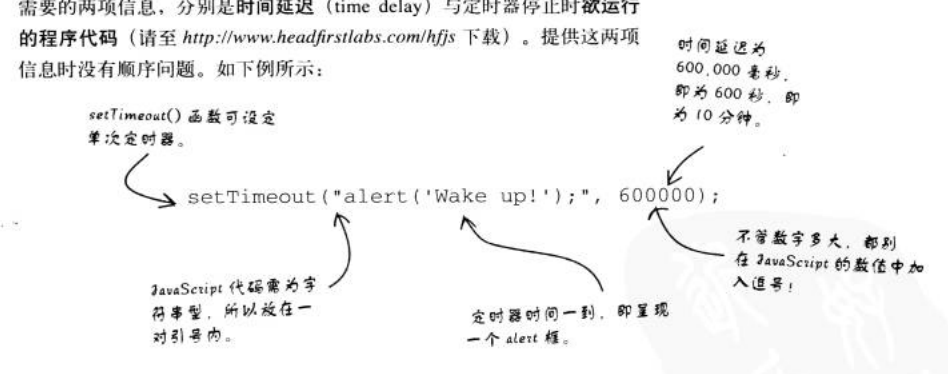
Js中也存在数字变量，如果想对某个变量进行数学计算（加减乘除），则必须先将变量转为数字变量

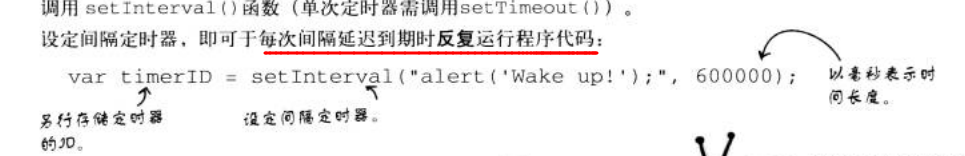
方法1：x = **new** Number(x);

方法2：x = parseInt(x);

1. **网页中设置定时器**

如果想让在一段时间后执行某个动作，则可以使用定时器





1. **网页中嵌入另一个网页，Iframe标签**
2. **浏览器网页的一些动作**

Onload() :每个标签都会有这个动作，当网页加载是触发该事件

Onresize():当浏览器窗口大小改变时触发该动作

OnMouseOver():鼠标在控件上面的时候触发

OnMouseOut():鼠标离开的时候触发

Onfocus()：元素获取焦点的响应事件

Onblur():元素失去焦点的响应事件

1. **HTML常用的一些属性**

标签自身属性

Z-index :设置标签的层次优先级，Z-index值越高则优先显示在表层，值越低有可能会被覆盖

Color : 字体颜色

Width : 标签宽度

Height : 标签高度

Css样式属性

display : 设置标签是否显示，none不显示，block是显示

opacity: 设置div背景透明度，大小范围是0.0-1.0

**注：IE不支持此属性，因此在设置时需要加上下面几句代码**

opacity: *0.5*;   /\* Firefox, Safari(WebKit), Opera)

filter:*alpha(opacity*=*50)*; /\* IE 4-7 \*/

filter: " *alpha(opacity*=*50)*"; /\* IE 8 \*/

position：对元素进行定位，absolute是绝对定位

overflow : 属性规定当内容溢出元素框时发生的事情

1. **检查浏览器是否支持cookie**

navigator.cookieEnabled;返回的是Boolean型，true代表支持cookie

1. **将数据存入cookie，并且读取cookie中的数据**

**方法1：使用js**

/\*\*

\* 此方法用于设在cookie

\* **@param** c\_name 名称

\* **@param** value 值

\* **@param** expiredays 过期天数

\*/

**function** setCookie(c\_name,value,expiredays)

{

**var** exdate=**new** Date();

exdate.setDate(exdate.getDate()+expiredays);

//escape()函数可对字符串进行编码，这样就可以在所有的计算机上读取该字符串。

document.cookie=c\_name+ "=" +escape(value)+

((expiredays==**null**) ? "" : ";expires="+exdate.toGMTString());

}

/\*\*

\* 此方法用于得到cookie,有cookie则返回cookie里面的值，没有则返回空串

\*/

**function** getCookie(c\_name)

{

**if** (document.cookie.length>0)

{

c\_start=document.cookie.indexOf(c\_name + "=");

**if** (c\_start!=-1)

{

c\_start=c\_start + c\_name.length+1 ;

c\_end=document.cookie.indexOf(";",c\_start);

**if** (c\_end==-1) c\_end=document.cookie.length;

//unescape()函数可对字符串进行解码

**return** unescape(document.cookie.substring(c\_start,c\_end));

}

}

**return** "";

}

方法2：

**var** getCookie = **function**() {

// 本地检查登录状态，如果已登录返回true

**var** arr,reg=**new** RegExp("(^| )loginStatus=([^;]\*)(;|$)");

**if**(arr=document.cookie.match(reg))

**return** unescape(arr[2]);

**else**

**return** **false**;

}

/\*\*

\* 如果 cookie 已设置，则显示欢迎词，否则显示提示框来要求用户输入名字。

\*/

**function** checkCookie()

{

username=getCookie('username');

**if** (username!=**null** && username!="")

{alert('Welcome again '+username+'!')}

**else**

{

username=prompt('Please enter your name:',"");

**if** (username!=**null** && username!="")

{

setCookie('username',username,365);

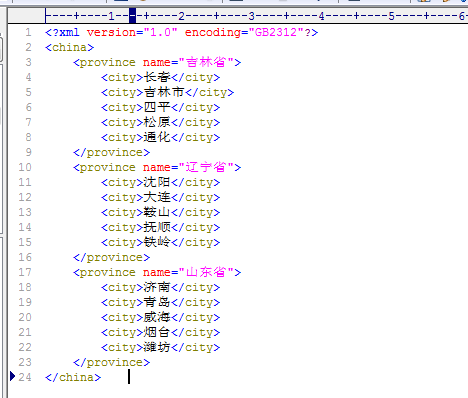
}

}

}

1. **解析XML文件的方法，IE兼容**

Xml文件结构



/\*\*

\* 跨浏览器的解析xml文件的函数

\* @param {Object} fileXmlName

\*/

function parseXML(fileXmlName){

var xmlDoc;

try {//Internet Explorer 创建一个空的xml文档

xmlDoc = new ActiveXObject("Microsoft.XMLDOM");

}

catch (e) {

try {//Firefox, Mozilla, Opera, 创建一个空的xml文档

xmlDoc = document.implementation.createDocument("", "", null);

}

catch (e) {

}

}

//关闭异步加载

xmlDoc.async = false;

//加载xml文件

xmlDoc.load(fileXmlName);

return xmlDoc;

}

<body>

<select id="province" name="province">

<option value="">请选择....</option>

<option value="吉林省">吉林省</option>

<option value="辽宁省">辽宁省</option>

<option value="山东省">山东省</option>

</select>

<select id="city" name="city">

<option value="">请选择.....</option>

</select>

</body>

<script language="JavaScript">

//页面中第一个下拉选的onchange事件

document.getElementById("province").onchange = function(){

//1 获取页面中选中的省份名称

var provinceValue = document.getElementById("province").value;

//2 解析xml文件

var docXml = parseXML("cities.xml");

//3 遍历xml文件中的省份

var provinceXmlElements = docXml.getElementsByTagName("province");

//清空

/\*

\* <select id="city" name="city">

<option value="">请选择.....</option>

<option value="长春">长春</option>

</select>

\*/

var cityElement = document.getElementById("city");

var optionsOld = cityElement.getElementsByTagName("option");

// for(var z=1;z<optionsOld.length;){

// cityElement.removeChild(optionsOld[z]);

// }

for(var z=optionsOld.length-1;z>0;z--){

cityElement.removeChild(optionsOld[z]);

}

//4 遍历xml文件中的所有省份

for(var i=0;i<provinceXmlElements.length;i++){

var provinceXmlElement = provinceXmlElements[i];

var provinceXmlValue = provinceXmlElement.getAttribute("name");

//5 判断页面中的省份名称与xml文件中的省份是否一致

if(provinceValue==provinceXmlValue){

//6 如果一致，读取对应省份的所有城市信息

var cityXmlElements = provinceXmlElement.getElementsByTagName("city");

for(var j=0;j<cityXmlElements.length;j++){

var cityXmlElement = cityXmlElements[j];

var cityXmlValue = cityXmlElement.firstChild.nodeValue;

//7 插入到页面中的第二个下拉选中

/\*

\* <select id="city" name="city">

<option value="">请选择.....</option>

<option value="长春">长春</option>

</select>

\*/

var optionElement = document.createElement("option");

optionElement.setAttribute("value",cityXmlValue);

var cityText = document.createTextNode(cityXmlValue);

optionElement.appendChild(cityText);

cityElement.appendChild(optionElement);

}

}

}

}

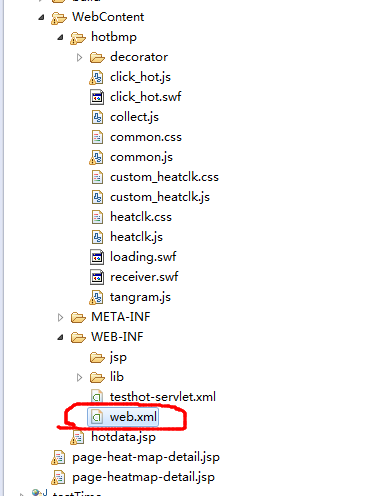
1. **Web无法加载某文件，但是路径是对的**

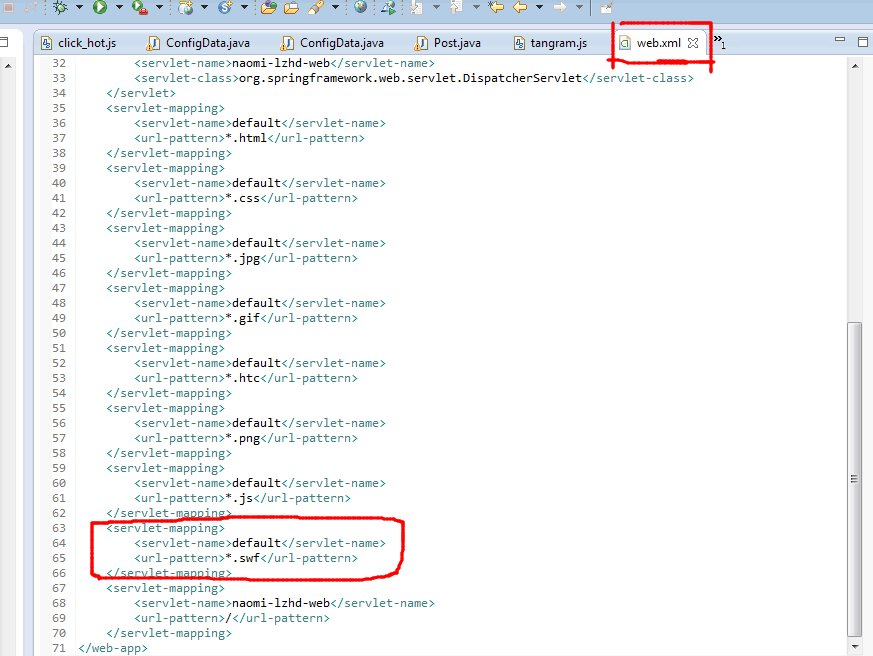
解决方法1：

检查web工程的web.xml文件夹是否将文件配置好

比如遇到flash文件，后缀名为.swf

查看web.xml是否已经配置默认加载该文件，如果没有配置那么即便路径是对的也无法加载





1. **在js、jsp中调用Java中的变量**

①在js中调用java变量

var 变量名 = <%=JAVA变量名 %>

**例子：**

**使用普通方法：**

<%

    String name = "bbb";

%>

<img alt="搜索" src="http://img.baidu.com/img/iknow/qb/select-search.png" id="selectsearch-icon">

**调用：**

var name = "<%=name %>";

**使用EL方法：**

<%   
 String name="zhouhaigang";  
 request.setAttribute("name", name);  
%>

调用  
${ name }

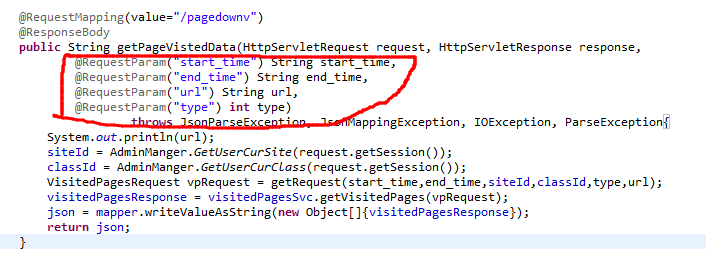
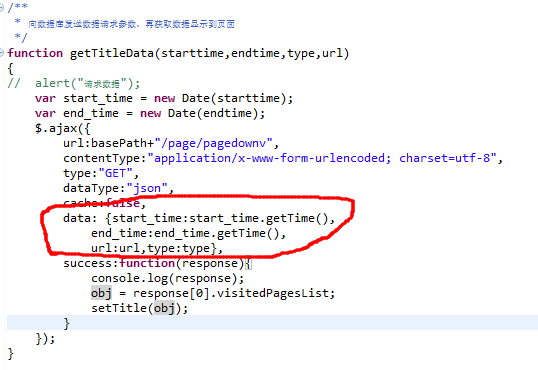
②在jsp中调用Java变量

<%=JAVA变量名 %>

1. **遇到网页请求报400状态**

解决方法1：

检查如下红圈部分的参数名称是否对应，如果不对应的话就会报以上的错误



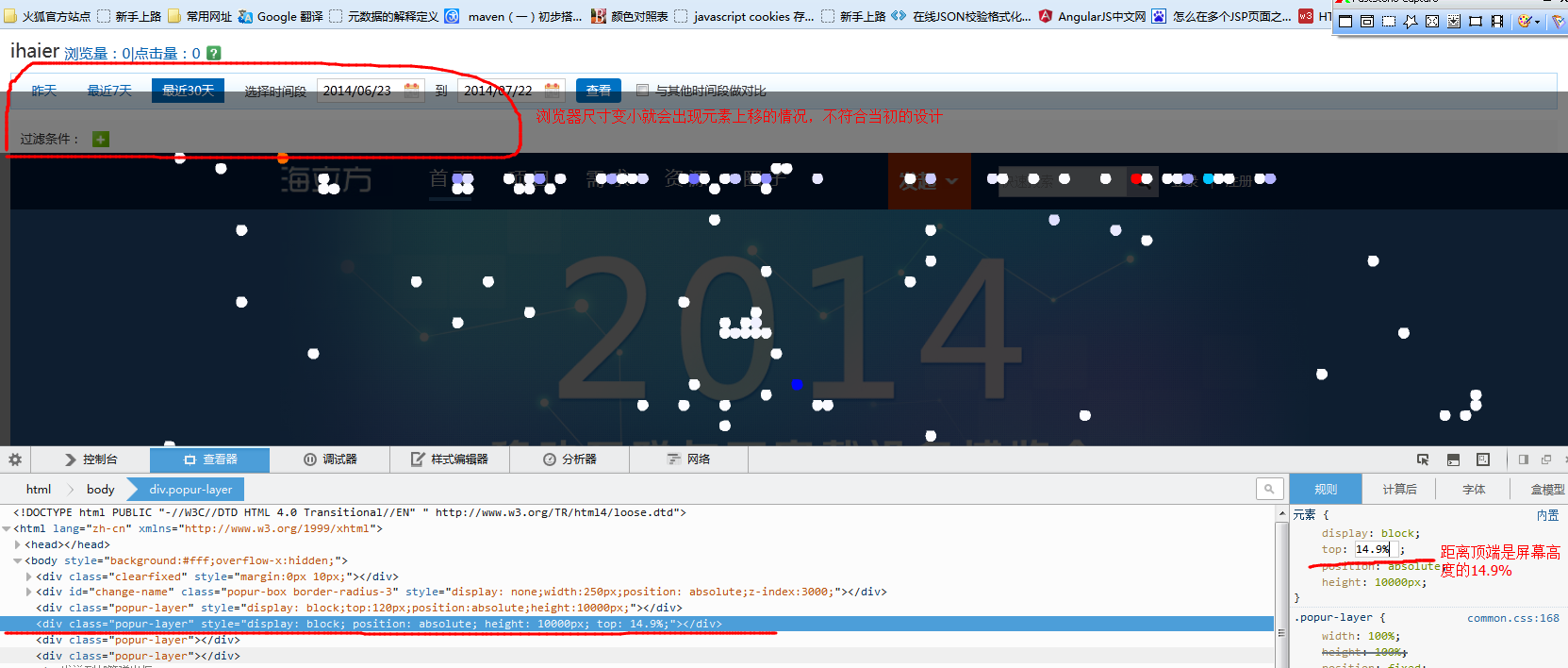
1. **Html元素定位问题position**

**利用position属性：**

①静止定位（static）：这种方法使得所有的元素最终位置都是一个静止位置。它没有特别的设定，遵循基本的定位规定，不能通过z-index进行层次分级。

②绝对定位（absolute）：这种方法允许用户指定元素的左上角、右下角或者其他的参考点和最近的父元素之间的关系将该元素从正常的文档流中拖出来。当页面滚动时，元素将保持它们各自的位置不变，随着页面一起滚动，就像粘在一起一样。

**注：**当使用绝对定位的时候如果设置百分比的话，元素布局会跟着浏览器大小的改变而改变，布局可能会出现不是当初想要的效果，因此一般设成固定的距离，比如：top:120px；这样就不会出现下面的情况



例子：如果采用绝对定位，那么在设置元素位置的时候会出现一个问题，比如用户网页下移的时候，就会出现定位不准的情况，因为获取的坐标是一浏览器窗口为基准的（详情请看44）

$('#change-name').css("margin-left", x +"px");

$('#change-name').css("margin-top", y +"px");



如何解决上面问题：

方法：在原来的数值加上body的偏移量document.body.scrollLeft

$('#change-name').css("margin-left",eval(document.body.scrollLeft + x)+"px");

$('#change-name').css("margin-top",eval(document.body.scrollTop+ y)+"px");

③固定定位（fixed）：这种方法允许元素相对于实际的浏览器窗口放置。

注：固定定位是相对浏览器窗口固定的，不会跟着网页上下的移动而移动，广告的位置适合使用这种定位。

④相对定位（relative）：这方法定义相对于在默认情况下浏览器把它放置的位置。这个很难解释，可喜的是这个方法使用的很少。

1. **html中position定位问题**

对于标签定位问题，如果标签的position属性声明为了absolute那么就可以使用left、top、right、bottom等定位属性，如果标签并未声明position属性或者是position属性的值不是absolute那么只能使用margin-left、margin-right等等属性，而不能使用left、right等

1. **获取用户鼠标点击位置**

方法1：

此方法是获取用户点击的像素位置，不会跟着网页上下移动而改变，意思就是该坐标的获取的值是固定的，不会因为网页上下移动的变化

注：此方法可以用于js框架文件中，如下是获取整个屏幕的鼠标点击位置

document.onclick=**function**(ev){

**//此处是为了IE兼容问题，如果是IE浏览器则oEvent = window.event，如果是其他浏览器则oEvent = ev.**

**var** oEvent = ev || window.event;

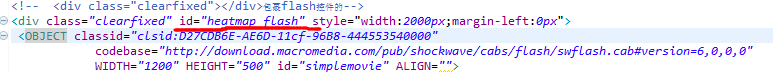
**var** oLeft=oEvent.clientX;

**var** oTop=oEvent.clientY;

};

如下是获取某个元素的鼠标点击位置

**注：作用范围不一样**



//获取用户点击位置

$('#heatmap\_flash').click(**function**(ev){

**var** oEvent = ev || window.event;

oLeft=oEvent.clientX;

oTop=oEvent.clientY;

**if**(oTop>120){

setAlertData(oLeft,oTop);

}

});

注：oEvent.clientX是获取相对于浏览器窗体的坐标，不会随着网页的上下移动而改变（一般用于那种网页高度小于屏幕高度的网页获取坐标）

oEvent.pageX是获取相对于整个网页的坐标，会随着网页上下移动而改变。（一般用于网页比较长，超出了屏幕的高度的网页的获取坐标）

1. **Option元素在IE下添加Onclick()事件不响应的问题**

**解决方法1：**

<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN">

<html>

<head>

<title>select-option onclick </title>

<script type="text/javascript" >

function simOptionClick4IE(){

var evt=window.event ;

var selectObj=evt?evt.srcElement:null;

// IE Only

if (evt && selectObj && evt.offsetY && evt.button!=2

&& (evt.offsetY > selectObj.offsetHeight || evt.offsetY<0 ) ) {

// 记录原先的选中项

var oldIdx = selectObj.selectedIndex;

setTimeout(function(){

var option=selectObj.options[selectObj.selectedIndex];

// 此时可以通过判断 oldIdx 是否等于 selectObj.selectedIndex

// 来判断用户是不是点击了同一个选项,进而做不同的处理.

showOptionValue(option)

}, 60);

}

}

function showOptionValue(opt,msg){

var now=new Date();

var dt= (1900+now.getYear())+'-'+(now.getMonth()+1)+'-'+now.getDate()+

' '+now.getHours()+':'+now.getHours()+':'+now.getSeconds()+'.'+now.getMilliseconds();

var resultZone=document.getElementById('reslut');

resultZone.style.margin="10px";

resultZone.innerHTML=dt +" 时,点击了: " + (opt.text + ' = '+opt.value);

}

</script>

</head>

<body>

<select onclick="simOptionClick4IE()" >

<!-- 下面的 onclick="showOptionValue( this )" 是为 ff 和 opera而准备 -->

<option value="1" onclick="showOptionValue( this )" >aaaaa</option>

<option value="2" onclick="showOptionValue( this )" >bbbbb</option>

<option value="3" onclick="showOptionValue( this )" >ccccc</option>

</select>

<div id="reslut" ></div>

</body>

</html>

**解决方法2：**

$('#choose\_index').change(**function**(){

**var** index\_temp = $(**this**).find("option:selected").text();

$('a').filter('#index\_show')[0].innerHTML = index\_temp;

condition = $('a').filter('#index\_show')[0].innerHTML;

lineText = condition;

selectIndex(lineText);

GetData(starttime, endtime,historyUrl,condition);

});

<select id =*"choose\_index"* onchange=*"selectLabelShow()"*>

<option>浏览量(pv)</option>

<option>访客数(uv)</option>

<option style="display: *none*;">新访客数</option>

<option>访问次数(us)</option>

<option style="display: *none*;">IP数</option>

<option>平均访问时长</option>

<option style="display: *none*;">转化次数</option>

<option style="display: *none*;">转化率</option>

<option>跳出率</option>

</select>

**解决方法3：jquery方法，推荐使用第三种方法**

<select id=*"select-click"*>

<option>abc</option>

<option>ee</option>

<option>ccc</option>

<option>fff</option>

<option>huj</option>

</select>

$('#select-click').bind("change",**function**(){

**var** index = $(**this**).find("option:selected").index();

alert(index);

});

**或者**

$('#select-click').on("change",**function**(){

**var** index = $(**this**).find("option:selected").index();

alert(index);

});

1. **设置select的option元素的默认值**

设置select的第二个选项是默认值

$('#heatmap-vector')[0].selectedIndex = 1;

$('#heatmap-condition')[0].selectedIndex = 1;



效果：



1. **判断HTML元素是否已经存在**

此方法用于判断id为active\_box01的元素是否不存在

**if**(!document.getElementById("active\_box01")){

}

1. **Js获取中英文字符的长度**

/\*\*

\* JS 判断中英文字符长度

\* **@param** str

\* **@returns** {Number}

\*/

**function** strlen(str) {

**var** len = 0;

**for** (**var** i = 0; i < str.length; i++) {

// charCodeAt()返回指定字符的字符编码号

**var** c = str.charCodeAt(i);

//单字节加1

**if** ((c >= 0x0001 && c <= 0x007e) || (0xff60 <= c && c <= 0xff9f)) {

len++;

}

**else** {

len += 2;

}

}

**return** len;

}

1. **判断字符串中是否有中文**

**方法1：使用正则表达式**

/\*\*

\* 此方法用于判断字符串str中是否存在中文

\* **@param** str

\*/

**function** checkChinese(str){

**var** reg = **new** RegExp("[\\u4E00-\\u9FFF]+","g");

alert(reg.test(str));

}

例子：abc你好 返回true

Abc 返回false

Abc你好abc 返回true

**方法2：在 Unicode 字符中，因为汉字的编码都是大于255，所以只要编码大于255的都是汉字**

**function** checkChinese(str){

**var** flag = **false**;

**for** (**var** v = 0; v < str.length; v++) {

**if**(str.charCodeAt(v)>255){

flag = **true**;

}

}

**return** flag;

}

1. **Js对数据进行排序算法总结**

见网址：http://blog.csdn.net/hguisu/article/details/7776068

1. **Js中替换特定字符串中的符合规则的字符**

替换某个字符串中符合规则的所有字符串 ：将所有’&’替换成’$’

/\*\*

\* 将字符串中的特定字符&转换成/

\* **@param** str

\* **@returns**

\*/

**function** replaceChar(str){

str = str.replace(/\&/g,"$");

**return** str;

}

替换某个字符串中符合规则的第一个字符串：将第一个’&’替换成’$’

/\*\*

\* 将字符串中的特定字符&转换成/

\* **@param** str

\* **@returns**

\*/

**function** replaceChar(str){

str = str.replace(/&/,"$");or str = str.replace("&","$");

**return** str;

}

1. **url作为参数在页面间传参，获取url不准确的问题**

问题描述：两页面间进行传参，url作为其中的一个参数，但是如果url本身就带了参数，可能会在另外一个页面获取url时，出现url获取的不准确的现象。

原因：在获取url时，另外一个页面以为url中的参数是另外的单独参数，因此url获取不全。

解决方法：将url中的参数符号转化为其他符号，然后在接收参数的页面再将url进行还原。**具体实现可参考上一个问题（Js中替换特定字符串中的符合规则的字符）**

url=http://www.rrs.com/search?fcid=342$utm\_source=baidu&utm\_campaign=%2525E7%252583%2525AD%2525E6%2525B0%2525B4%2525E5%252599%2525A8%25252D%2525E9%252580%25259A%2525E7%252594%2525A8%2525E8%2525AF%25258D$utm\_adgroup=%2525E9%252580%25259A%2525E7%252594%2525A8%2525E8%2525AF%25258D%25252D%2525E6%25258E%252592%2525E5%252590%25258D&channel=web&s=baidu\_cpc&active=jiadian

1. **ajax异步请求的问题**

问题描述：当用户点击一个按钮进行数据请求的时候，我们是通过ajax进行请求的，如果用户双击按钮，就可能出现先弹出没数据的提示，然后显示出两条相同的数据。

原因：应该是用户双击按钮，进行两次ajax请求，当第一次请求的数据还没回来的时候第二次请求已经发出，因此第一次请求的数据可能延迟到第二次请求。

解决方法：在ajax请求中加入async:**false 表示进行同步请求，false为同步，这个 getKeyData ()方法中的Ajax请求将整个浏览器锁死，只有这个jsp执行结束后，才可以执行其它操作。**

/\*\*

\* 向数据库发送数据请求参数url，再获取数据显示到页面

\*/

**function** getKeyData(startTime,endTime,url,operateType,pageNum,pageCount,index)

{

**var** start\_time = **new** Date(startTime);

**var** end\_time = **new** Date(endTime);

$.ajax({

url:basePath+"/page/entrancePageKey",

contentType:"application/x-www-form-urlencoded; charset=utf-8",

type:"GET",

dataType:"json",

cache:**false**,

async:**false**,

data: {start\_time:start\_time.getTime(),end\_time:end\_time.getTime(),url:url,

pageNum:pageNum,pageCount:pageCount,index:index},

success:**function**(response){

}

});

}

1. **关于同源策略的问题**

**何为同源策略：**

同源策略(Same Origin Policy)，它是由Netscape提出的一个著名的安全策略。 现在所有支持JavaScript 的浏览器都会使用这个策略。同源策略阻止从一个域上加载的脚本去获取或操作另一个域上的文档属性。也就是说，受到请求的 URL 的域必须与当前 Web 页面的域相同。这说明浏览器隔离来自不同源的内容，以防止它们之间的操作。

简单来说就是一个URL由四部分组成，拿 [www.2cto.com](http://www.2cto.com) 来说（http的默认端口是80，https的默认端口是443。如果是默认端口，可以省略，所以这个URL等价[www.2cto.com:80](http://www.2cto.com:80)  
 协议：http

主机：[www.2cto.com](http://www.2cto.com)

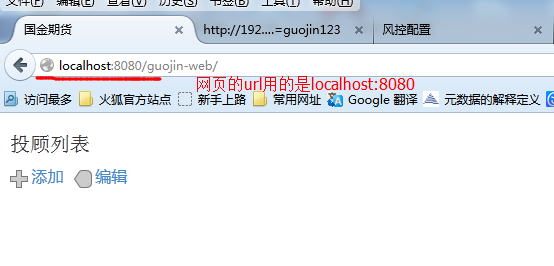
端口：80

路径：/

所谓的同源就是要求这个URL的协议，主机，端口三部分都相同。一般我们说的域或者domain也是这里的源的概念。

**什么情况下会产生同源策略问题：**

**情况1：**网页的url与该网页的ajax请求的url不一致，例如：





以上情况就会产生同源策略问题。



**关于这种情况的解决方法：将两个地方的url统一成相同的就可以。**

1. **让背景图片自适应控件的大小 background-image**

**问题描述：想让背景图片自动拉伸到窗口的大小**

**方法1：这个对于IE不兼容，在ie下只能显示图片自身的大小，其余都是空白，css代码如下**

background: *url(../images/login/header\_background.png)* *no-repeat*;

background-size: *100%* *100%*;

-moz-background-size: *100%* *100%*; /\* 老版本的 Firefox \*/

**方法2：对于ie使用滤镜，不过这个效果也不是很好，他是将图片重复的显示来填充一张图片不能覆盖的地方，CSS代码如下**

background-size: *100%* *100%*;

-moz-background-size: *100%* *100%*;/\* 老版本的 Firefox \*/

background:*url(../../images/login/background.png)*;

filter:*"progid:DXImageTransform.Microsoft.AlphaImageLoader(sizingMethod='scale')"*; /\* *sizingMethod='scale'就是将图片进行拉伸* \*/

效果如下：



**方法3：与方法2大同小异，只是多加了一个属性background-attachment:fixed，这种方法会根据不同浏览器显示不同效果，在火狐或者谷歌则显示一张图片的拉伸效果，比较理想，在ie浏览器下则显示效果跟方法2是一样的**

background-image:*url('../../images/login/background.png')*;

background-attachment:*fixed*;

background-size:*100%* *100%*; filter:*progid*:*DXImageTransform.Microsoft.AlphaImageLoader*(*src*=*'../../images/login/background.png',sizingMethod*=*'scale')*;

火狐浏览器：



IE浏览器：



1. **Jsp和html页面，编码问题**

要想让显示的页面能正常显示中文。

**html文件**：只需要在html的head标签里面加入如下代码就行

<head>

<meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"* />

</head>

完整文件如下：

<!DOCTYPE html HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<title>Insert title here</title>

<meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"* />

</head>

<body>

<div id=*"click"*>你好</div>

</body>

</html>

**Jsp文件：**jsp与html有区别，必须在文件的开始加上一段代码，要不然会出现乱码

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<!DOCTYPE html HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">

<html>

<head>

<title>Insert title here</title>

<meta http-equiv=*"content-type"* content=*"text/html; charset=UTF-8"* />

</head>

<body>

<div id=*"click"*>你好</div>

</body>

</html>

1. **判断变量是否为undefined**

**typeof**(timeId)=='undefined')

1. **为元素/标签附加某些重要的数据**

**方法1：**直接加到标签的属性上，作为属性值加上去。比如在标签的title属性上加入值1999.

<div id=*"content"*>abc</div>

$('#content').attr("title","1999");

**方法2:**使用jquery的data方法（**推荐使用**）

$('#content').data("name","gh");

alert( $('#content').data("name"));

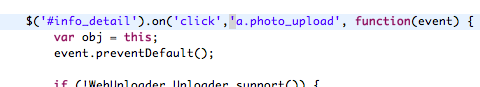
1. **将json数据格式的字符串用转成js对象**

{"msg":"天猫一勋旗舰店店铺半年内动态评分详情如下:","tab":{"trs":[["4.8分","低0.0%"],["4.7分","低0.0%"],["4.7分","低0.0%"]]}}这种格式的字符串转化为json对象

var msg = JSON.parse(obj[v].message);

1. **对动态添加的标签绑定事件**

使用jQuery的方法

****

其中#info\_detail是父id，a.photo\_upload是id 为info\_detail标签里面的a标签（并且class是photo\_upload），为这样的a标签绑定事件

1. **获取网站的根路径**

**问题描述：获取网站根路径**

**解决方法：**

/\*\*

\* 获取当前URL根目录

\*/

**var** getBasePath = **function**(){

**var** location = window.document.location;

**var** basePath = **null**;

**with**(location){

//获取当前网址，如： http://localhost:8083/uimcardprj/share/meun.jsp

**var** curWwwPath = href;

//获取主机地址之后的目录，如： uimcardprj/share/meun.jsp

**var** pathName = pathname;

**var** pos = curWwwPath.indexOf(pathName);

//获取主机地址，如： http://localhost:8083

basePath = curWwwPath.substring(0, pos);

}

**return** basePath;

}

**方法2：**利用jsp嵌入Java代码获取

<%@ page language=*"java"* contentType=*"text/html; charset=UTF-8"*

pageEncoding=*"UTF-8"*%>

<%

String path = request.getContextPath();

String basePath = request.getScheme()+"://"+request.getServerName()+ ((request.getServerPort()==80)?"":":"+request.getServerPort()) +path+"/";

%>

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

**var** basePath = "<%=basePath %>";

</script>

1. **双方各得**