# [jqGrid与Struts2的结合应用（十） —— 真正的自定义表单查询](http://blog.csdn.net/c622724/article/details/8372539)

分类： [JqGrid学习记录](http://blog.csdn.net/c622724/article/category/1309794) 2012-12-22 10:13 366人阅读 [评论](http://blog.csdn.net/c622724/article/details/8372539#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/c622724/article/details/8372539#report)

[jqGrid与Struts2的结合应用十](http://www.csdn.net/tag/jqGrid%e4%b8%8eStruts2%e7%9a%84%e7%bb%93%e5%90%88%e5%ba%94%e7%94%a8%e5%8d%81)[真正的自定义表单查询](http://www.csdn.net/tag/%e7%9c%9f%e6%ad%a3%e7%9a%84%e8%87%aa%e5%ae%9a%e4%b9%89%e8%a1%a8%e5%8d%95%e6%9f%a5%e8%af%a2)

在网上搜了一下，大多都是引用jqGrid的文档中的filterGrid方法。但这个方法不是真正意义的自定义查询。这篇文章主要讲述一下真正意义的自定义查询，顺便简单聊聊jqGrid源码中查询的实现方法。在这个系列的第五篇中（<http://blog.csdn.net/gengv/archive/2010/07/10/5725172.aspx> ）曾经讲到了如何进行数据查询。其中提到了一种“自定义查询”，这是jqGrid的文档中给出的一种所谓的自定义查询。其实就是把工具条查询，从Grid表格中移出来，然后可以控制输入框的横向或纵向排列，构成一种看似表单查询的查询方式。但是这种方式也有一定的局限性，比如不能以colModel中不存在的列作为查询字段进行查询，不能做到真正意义上的自定义表单布局等等。

昨天，CSDN的一位网友gaile给我的博客留言：

searchGrid的弹出的对话框的find按钮点击的时候是调用的triggerSearch()方法吗？那么重置按钮点击的时候都做了哪些操作啊？？我只知道它也发送了请求到server,但是如果我想点击重置的时候只是清除查询条件，而不让它发送请求到server端 该怎么做呢？？请问这些按钮的具体操作我能在哪个文件里找到？？谢谢了！  
  
还有，假如我想自定义serach弹出框的格式该怎么做呢?

他的问题实际上就是：如何实现自定义的表单查询。正好最近我也在考虑这个问题，昨晚特意查看了一下jqGrid的源码，顺便总结出这篇文章。

我们需要的是对查询的控制权，也就是最终发送给Server端的请求，而不只是通过jqGrid选项支配的那些自动生成的查询请求参数。

**1. 查询的实现思路：**

通过前面文章的介绍，可以知道jqGrid在查询中主要的任务是向Server发送有关于查询的参数，主要的参数包括search、searchField、searchOper、searchString、filters等（这些参数名可以通过选项进行更改，可以参考之前的文章）；Server得到参数后，进行后台数据查询，然后将结果发回客户端；jqGrid根据jsonReader的设置，解析返回的结果并显示到Grid表格中。

大体思路就是这样。既然如此，我们要想实现真正的自定义查询，要解决的无非就是如何把有关查询的参数“塞进”请求里。需要用到两个jqGrid选项：

**postData** ：可以参考本系列第九篇（http://blog.csdn.net/gengv/archive/2010/07/14/5735654.aspx）；当jqGrid向Server发送请求的时候，总是会将postData中的{name:value,...}对，作为参数一同传递到服务器端。因此我们只需要在发送查询请求的时候，将关于查询的参数放进postData就可以了。

**search** ：这个jqGrid选项，在jqGrid的官方文档中并没有列出来，但确实存在；它的值可以为true或false，用来控制是否为查询请求，如果为true，则在请求中加入search=true参数；可以使用jqGrid的setGridParam方法修改它。

**注1** ：search选项传入的这个search=true参数，其参数名称“search”是由jqGrid的prmNames选项中设置的，默认为“\_search”，此处是为了Server端java程序中成员变量命名方便，而设置为“search”了（具体细节可以参看本系列第一篇<http://blog.csdn.net/gengv/archive/2010/07/05/5714834.aspx> ）。

**注2** ：search选项传入请求的search参数值，优先于postData中的同名参数值。即如果serach选型值为flase，而postData中有个名值对为{search:true}，则请求为http://...action?search=false&...&search=true。此时第一个search参数（由search选项控制）将使第二个search参数（postData提供）被忽略。

**2. 自定义单条件查询的实现**

根据之前文章的介绍，当单条件查询发送请求时，参数应为如下形式：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

nd： 1278688214496

page： 1

rows： 15

search： **true**    // (1)

searchField： id    // (2)

searchOper： ge    // (3)

searchString： 9000    // (4)

sidx：

sord： asc

其中(1)(2)(3)(4)是我们最关注的。  
  
下面看看如何来实现单条件自定义查询：  
HTML部分：

**[xhtml]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

...

**<body>**

**< h3>**

        jqGrid测试 09

**< /h3>**

**< div** id="mysearch"**></div>**

**< br** **/>**

**< div>**

**< table** id="gridTable"**></table>**

**< div** id="gridPager"**></div>**

**< /div>**

**< br** **/>**

**< div>**

**< button** onclick="openSingleSearchDialog()"**>**自定义单条件查询**</button>**

**< button** onclick="clearSearch()"**>**重置**</button>**

**< /div>**

**< div** id="singleSearchDialog"**>**

**< table** class="formTable"**>**

**< thead>**

**< tr>**

**< th>**查询条件**</th>**

**< th>**查询方式**</th>**

**< th>**查询值**</th>**

**< /tr>**

**< /thead>**

**< tbody>**

**< tr>**

**< td>**

**< input** type="radio" name="searchField" value="id"**>**编码**</input><br** **/>**

**< input** type="radio" name="searchField" value="lastName"**>**姓**</input><br** **/>**

**< input** type="radio" name="searchField" value="firstName"**>**名**</input>**

**< /td>**

**< td>**

**< select** id="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" id="searchString"**></input>**

**< /td>**

**< /tr>**

**< /tbody>**

**< /table>**

**< /div>**

**</body>**

...

HTML中增加了一个自定义的<div>，singleSearchDialog；而且使用radio组来选择查询字段。

再看看javascript部分：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

$(**function**(){

    $("#gridTable").jqGrid({

        ...

        prmNames: {search: "search"},

        ...

    });

    $("#singleSearchDialog").dialog({   // (1)创建自定义查询对话框

        autoOpen: **false**,

        modal: **true**,

        resizable: **true**,

        width: 350,

        title: "自定义单条件查询",

        buttons: {

            "查询": singleSearch      // (2)在对话框中添加查询按钮

        }

    });

});

**var** openSingleSearchDialog = **function**() {

    $("#singleSearchDialog").dialog("open");

};

**var** resetSingleSearchDialog = **function**() {

    $("radio","#singleSearchDialog").attr("checked", **false**);

    $(":text","#singleSearchDialog").val("");

};

**var** singleSearch = **function**() {

**var** sdata = {   // (3)构建查询需要的参数

        searchField: $(":radio:checked", "#singleSearchDialog").val(),

        searchString: $("#searchString", "#singleSearchDialog").val(),

        searchOper: $("#searchOper", "#singleSearchDialog").val()

    };

    // (4)获得当前postData选项的值

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    // (5)将查询参数融入postData选项对象

    $.extend(postData, sdata);

    $("#gridTable").jqGrid("setGridParam", {

        search: **true**    // (6)将jqGrid的search选项设为true

    }).trigger("reloadGrid", [{page:1}]);   // (7)重新载入Grid表格，以使上述设置生效

    $("#singleSearchDialog").dialog("close");

};

**var** clearSearch = **function**() {

**var** sdata = {   // (8)构建一套空的查询参数

        searchField: "",

        searchString: "",

        searchOper: ""

    };

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    $.extend(postData, sdata);  //(9)将postData中的查询参数覆盖为空值

    $("#gridTable").jqGrid("setGridParam", {

        search: **false**   // (10)将jqGrid的search选项设为false

    }).trigger("reloadGrid", [{page:1}]);

    resetSingleSearchDialog();

};

说明：

(7)处设置完jqGrid参数后，重新载入表格时，jqGrid将会将postData中的参数加进请求参数中，因此将提交查询请求；在显式清除postData中的参数之前，每次查询都会带着这些查询参数。

(9)处将postData中的关于查询的参数全都覆盖为空值，重新载入表格后，请求中将不再传入“有效”的查询参数，即有查询参数名，但无查询参数值。例如：http://localhost:8085/JqGridTest/jqGridTest/jqGrid05.action?**search=false**&nd=1279865475529&rows=15&page=2&sidx=&sord=asc&**searchField=&searchString=&searchOper=**

**注意：**   
不论在查询或者清除查询之后条用trigger("reloadGrid")，都应该带上“[{page:1}]”，以设置page选项为1，即：

$("#gridTable").jqGrid(...).trigger("reloadGrid", [{page:1}]);

如果不加上这个参数的话，那么后续的查询中，page都始终保持之前操作时保存的值，对于后面再进行的查询很能产生副作用。举个例子来说，第一次查询使用查询条件A，如果满足条件的数据有110行，Grid中每页显示20行（即每次从Server最多返回20行），则总共有6页数据；这时翻页，一直翻到第4页；然后使用查询条件B再次进行查询，假设这次查询满足条件的数据总共只有35行，每页显示20行，也即只有2页数据，那么如果page的值还是4的话，则后台就找不到数据了。因此，应该使用trigger("reloadGrid", [{page:1}]);，这样总能保证，即使查询结果只有1页数据，也能被显示出来。



**3. 自定义多条件查询的实现**   
多条件查询与但条件查询的不同之处即在于使用filters参数，代替了searchField，searchOper，searchString参数。其参数形式为：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

filters: {

    "groupOp": "AND",

    "rules": [

        {"field":"id","op":"le","data":"1500"},

        {"field":"lastName","op":"bw","data":"LN-3"},

        {"field":"email","op":"cn","data":"sy"}

    ]

}

nd: 1278688110637

page:   1

rows:   15

search: **true**

sidx:

sord: asc

因此具体实现也与上述大同小异，只要加入正确的filters参数就行了。

首先看HTML部分，加入了multipleSearchDialog。

**[xhtml]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

...

**<div>**

    ...

**< button** onclick="openMultipleSearchDialog()"**>**自定义复杂条件查询**</button>**

**</div>**

...

**<div** id="multipleSearchDialog"**>**

**< table** class="formTable"**>**

**< thead>**

**< tr>**

**< th>**查询条件**</th>**

**< th>**查询方式**</th>**

**< th>**查询值**</th>**

**< /tr>**

**< /thead>**

**< tbody>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< /tbody>**

**< /table>**

**</div>**

...

再来看javascript部分：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

$(**function**(){

    $("#gridTable").jqGrid({

        ...

    });

    ...

    $("#multipleSearchDialog").dialog({

        autoOpen: **false**,

        modal: **true**,

        resizable: **true**,

        width: 350,

        title: "自定义多条件查询",

        buttons: {

            "查询": multipleSearch

        }

    });

});

...

**var** openMultipleSearchDialog = **function**() {

    $("#multipleSearchDialog").dialog("open");

};

**var** multipleSearch = **function**() {

**var** rules = "";

    $("tbody tr", "#multipleSearchDialog").each(**function**(i){    //(1)从multipleSearchDialog对话框中找到各个查询条件行

**var** searchField = $(".searchField", **this**).val();    //(2)获得查询字段

**var** searchOper = $(".searchOper", **this**).val();  //(3)获得查询方式

**var** searchString = $(".searchString", **this**).val();  //(4)获得查询值

**if**(searchField && searchOper && searchString) { //(5)如果三者皆有值且长度大于0，则将查询条件加入rules字符串

            rules += ',{"field":"' + searchField + '","op":"' + searchOper + '","data":"' + searchString + '"}';

        }

    });

**if**(rules) { //(6)如果rules不为空，且长度大于0，则去掉开头的逗号

        rules = rules.substring(1);

    }

    //(7)串联好filtersStr字符串

**var** filtersStr = '{"groupOp":"AND","rules":[' + rules + ']}';

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    //(8)将filters参数串加入postData选项

    $.extend(postData, {filters: filtersStr});

    $("#gridTable").jqGrid("setGridParam", {

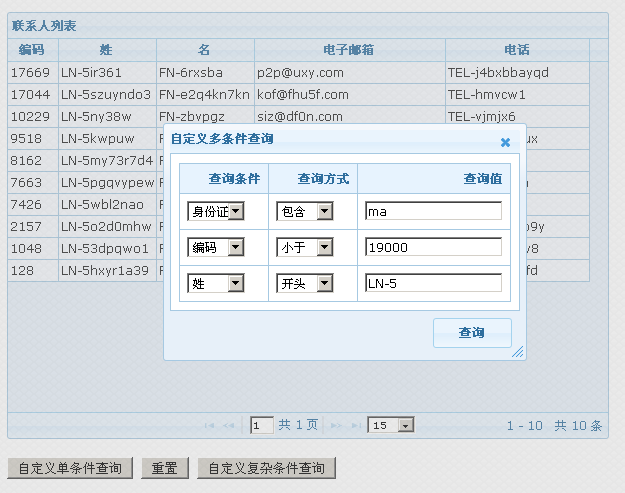
        search: **true**    //(9)将jqGrid的search选项设为true

    }).trigger("reloadGrid", [{page:1}]);   //(10)重新载入Grid表格

    $("#multipleSearchDialog").dialog("close");

};

OK，这样就实现了自定义多条件查询。同时可以看到，colModel中根本不存在身份证号（idCardNo）这一列，但是在查询中依然可以使用这个字段进行查询。  
  
另一方面，由于自己把握了查询参数的控制权，jqGrid的multipleSearch选项将变得无足轻重了。上面的两个例子，仅作说明之用，从外观上来看，还比较粗糙。但已经从根本上控制住了查询的实现。即使在复杂的页面设计，也不会影响查询的实现了。这与jqGrid官方文档中提供的“自定义查询”，是有本质区别的。  
  
对于页面布局的美化，以及动态添加查询条件行等等，只要基本的CSS+JavaScript知识就能实现，因此不再本文的讨论内容中了。



**4. jqGrid源码**   
jqGrid的源码可以从以下链接得到：  
<http://github.com/tonytomov/jqGrid>   
  
关于查询这部分的代码，可以查看：  
<http://github.com/tonytomov/jqGrid/blob/master/js/grid.formedit.js>

片段1：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

$t.SearchFilter = $("#"+fid).searchFilter(fields, { groupOps: p.groupOps, operators: oprtr, onClose:hideFilter, resetText: p.Reset, searchText: p.Find, windowTitle: p.caption,  rulesText:p.rulesText, matchText:p.matchText, onSearch: searchFilters, onReset: resetFilters,stringResult:p.stringResult, ajaxSelectOptions: $.extend({},$.jgrid.ajaxOptions,$t.p.ajaxSelectOptions ||{}), clone: p.cloneSearchRowOnAdd });

片段2：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

**function** searchFilters(filters) {

**var** hasFilters = (filters !== undefined),   //(1)

    grid = $("#"+$t.p.id), sdata={};

**if**(p.multipleSearch===**false**) {  //(2)

        sdata[p.sField] = filters.rules[0].field;

        sdata[p.sValue] = filters.rules[0].data;

        sdata[p.sOper] = filters.rules[0].op;

    } **else** {

        sdata[p.sFilter] = filters; //(3)

    }

    grid[0].p.search = hasFilters;  //(4)

    $.extend(grid[0].p.postData,sdata); //(5)

    grid.trigger("reloadGrid",[{page:1}]);  //(6)

**if**(p.closeAfterSearch) { hideFilter($("#"+fid)); }  //(7)

}

**function** resetFilters(op) {

**var** reload = op && op.hasOwnProperty("reload") ? op.reload : **true**;

    grid = $("#"+$t.p.id), sdata=[];

    grid[0].p.search = **false**;

**if**(p.multipleSearch===**false**) {

        sdata[p.sField] = sdata[p.sValue] = sdata[p.sOper] = "";

    } **else** {

        sdata[p.sFilter] = "";

    }

    $.extend(grid[0].p.postData,sdata);

**if**(reload) {

        grid.trigger("reloadGrid",[{page:1}]);

    }

**if**(p.closeAfterReset) { hideFilter($("#"+fid)); }

}

我没有从头到尾通读所有代码，没有时间，也没有那么高水平，呵呵~  
  
但从上面片段1中可以看出来：  
在查询对话框中点击“查询”按钮，则执行searchFilters方法；而点击“重置”按钮，则执行resetFilters方法。  
  
再看看searchFilters方法：

(1)首先判断是否存在filters；

(2)如果multipleSearch选项为false，则构建以sField、sOper、sValue所对应的参数名为内容的参数（默认的参数名为：searchField、searchOper、searchString）；

(3)如果multipleSearch选项为true，则构建sFilter对应的参数名的参数（默认为filters）；

(4)然后将查询参数置入postData选项；

(5)如果存在filters，则将jqGrid的search选项设为true；这与$("#gridTable").jqGrid("setGridParam", {     search: true });是一样的。

(6)重新载入表格数据（发送包含查询参数的请求）；

(7)如果closeAfterSearch为true，则查询完毕后，关闭查询对话框。

总体看来，与之前自定义的条件查询实现基本一样，所不同的是，这里把单条件和多条件查询和在一起了。  
  
而resetFilters方法，与之前自定义的实现，思路也基本一样，不再赘述。由于searchFilters和resetFilters是定义在searchGird方法中的函数，因此在外部调用时，还是有点困难的。

附上本文完整前端代码（后台代码可以参考之前的文章）：  
HTML部分：

**[xhtml]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd"**>**

**<html** xmlns="http://www.w3.org/1999/xhtml" xml:lang="zh-CN" lang="zh-CN"**>**

**< head>**

**< title>**jqGrid09**</title>**

**< link** rel="stylesheet" type="text/css" media="screen" href="../css/themes/aero/jquery-ui-1.8.1.custom.css" mce\_href="css/themes/aero/jquery-ui-1.8.1.custom.css" **/>**

**< link** rel="stylesheet" type="text/css" media="screen" href="../css/themes/aero/ui.jqgrid.custom.css" mce\_href="css/themes/aero/ui.jqgrid.custom.css" **/>**

**< link** href="../css/genericFormTable.css" mce\_href="css/genericFormTable.css" rel="stylesheet" type="text/css" **/>**

**< mce:script** type="text/javascript" src="../js/jquery-1.4.2.min.js" mce\_src="js/jquery-1.4.2.min.js"**></mce:script>**

**< mce:script** src="../js/jquery-ui-1.8.1.custom.min.js" mce\_src="js/jquery-ui-1.8.1.custom.min.js" type="text/javascript"**></mce:script>**

**< mce:script** src="../js/i18n/grid.locale-zh-CN.js" mce\_src="js/i18n/grid.locale-zh-CN.js" type="text/javascript"**></mce:script>**

**< mce:script** src="../js/jquery.jqGrid.min.js" mce\_src="js/jquery.jqGrid.min.js" type="text/javascript"**></mce:script>**

**< mce:script** src="js/jqGrid09.js" mce\_src="js/jqGrid09.js" type="text/javascript"**></mce:script>**

**< /head>**

**< body>**

**< h3>**

            jqGrid测试 09

**< /h3>**

**< div** id="mysearch"**></div>**

**< br** **/>**

**< div>**

**< table** id="gridTable"**></table>**

**< div** id="gridPager"**></div>**

**< /div>**

**< br** **/>**

**< div>**

**< button** onclick="openSingleSearchDialog()"**>**自定义单条件查询**</button>**

**< button** onclick="clearSearch()"**>**重置**</button>**

**< button** onclick="openMultipleSearchDialog()"**>**自定义复杂条件查询**</button>**

**< /div>**

**< div** id="singleSearchDialog"**>**

**< table** class="formTable"**>**

**< thead>**

**< tr>**

**< th>**查询条件**</th>**

**< th>**查询方式**</th>**

**< th>**查询值**</th>**

**< /tr>**

**< /thead>**

**< tbody>**

**< tr>**

**< td>**

**< input** type="radio" name="searchField" value="id"**>**编码**</input><br/>**

**< input** type="radio" name="searchField" value="lastName"**>**姓**</input><br/>**

**< input** type="radio" name="searchField" value="firstName"**>**名**</input>**

**< /td>**

**< td>**

**< select** id="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" id="searchString"**></input>**

**< /td>**

**< /tr>**

**< /tbody>**

**< /table>**

**< /div>**

**< div** id="multipleSearchDialog"**>**

**< table** class="formTable"**>**

**< thead>**

**< tr>**

**< th>**查询条件**</th>**

**< th>**查询方式**</th>**

**< th>**查询值**</th>**

**< /tr>**

**< /thead>**

**< tbody>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< option** value="idCardNo"**>**身份证**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< option** value="cn"**>**包含**</option>**

**< option** value="bw"**>**开头**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< option** value="idCardNo"**>**身份证**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< option** value="cn"**>**包含**</option>**

**< option** value="bw"**>**开头**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< tr>**

**< td>**

**< select** class="searchField"**>**

**< option** value="id"**>**编码**</option>**

**< option** value="lastName"**>**姓**</option>**

**< option** value="firstName"**>**名**</option>**

**< option** value="idCardNo"**>**身份证**</option>**

**< /select>**

**< /td>**

**< td>**

**< select** class="searchOper"**>**

**< option** value="eq"**>**等于**</option>**

**< option** value="gt"**>**大于**</option>**

**< option** value="lt"**>**小于**</option>**

**< option** value="cn"**>**包含**</option>**

**< option** value="bw"**>**开头**</option>**

**< /select>**

**< /td>**

**< td>**

**< input** type="text" class="searchString"**></input>**

**< /td>**

**< /tr>**

**< /tbody>**

**< /table>**

**< /div>**

**< /body>**

**</html>**

JavaScript部分：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5758321)

$(**function**(){

    $("#gridTable").jqGrid({

        url: "listContacts.action",

        datatype: "json",

        mtype: "GET",

        height: 350,

        width: 600,

        colModel: [

              {name:"id",index:"id",label:"编码",width:40,sortable:**false**},

              {name:"lastName",index:"lastName",label:"姓",width:80,sortable:**false**},

              {name:"firstName",index:"firstName",label:"名",width:80,sortable:**false**},

              {name:"email",index:"email",label:"电子邮箱",width:160,sortable:**false**},

              {name:"telNo",index:"telNo",label:"电话",width:120,sortable:**false**}

        ],

        viewrecords: **true**,

        rowNum: 15,

        rowList: [15,50,100],

        prmNames: {search: "search"},

        jsonReader: {

            root:"gridModel",

            records: "record",

            repeatitems : **false**

        },

        pager: "#gridPager",

        caption: "联系人列表",

        hidegrid: **false**,

        shrikToFit: **true**

    });

    $("#singleSearchDialog").dialog({

        autoOpen: **false**,

        modal: **true**,

        resizable: **true**,

        width: 350,

        title: "自定义单条件查询",

        buttons: {

            "查询": singleSearch

        }

    });

    $("#multipleSearchDialog").dialog({

        autoOpen: **false**,

        modal: **true**,

        resizable: **true**,

        width: 350,

        title: "自定义多条件查询",

        buttons: {

            "查询": multipleSearch

        }

    });

});

**var** openSingleSearchDialog = **function**() {

    $("#singleSearchDialog").dialog("open");

};

**var** resetSingleSearchDialog = **function**() {

    $("radio","#singleSearchDialog").attr("checked", **false**);

    $(":text","#singleSearchDialog").val("");

};

**var** singleSearch = **function**() {

**var** sdata = {

        searchField: $(":radio:checked", "#singleSearchDialog").val(),

        searchString: $("#searchString", "#singleSearchDialog").val(),

        searchOper: $("#searchOper", "#singleSearchDialog").val()

    };

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    $.extend(postData, sdata);

    $("#gridTable").jqGrid("setGridParam", {

        search: **true**

    }).trigger("reloadGrid", [{page:1}]);

    $("#singleSearchDialog").dialog("close");

};

**var** clearSearch = **function**() {

**var** sdata = {

        searchField: "",

        searchString: "",

        searchOper: ""

    };

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    $.extend(postData, sdata);

    $("#gridTable").jqGrid("setGridParam", {

        search: **false**

    }).trigger("reloadGrid", [{page:1}]);

    resetSingleSearchDialog();

};

**var** openMultipleSearchDialog = **function**() {

    $("#multipleSearchDialog").dialog("open");

};

**var** multipleSearch = **function**() {

**var** rules = "";

    $("tbody tr", "#multipleSearchDialog").each(**function**(i){

**var** searchField = $(".searchField", **this**).val();

**var** searchOper = $(".searchOper", **this**).val();

**var** searchString = $(".searchString", **this**).val();

**if**(searchField && searchOper && searchString) {

            rules += ',{"field":"' + searchField + '","op":"' + searchOper + '","data":"' + searchString + '"}';

        }

    });

**if**(rules) {

        rules = rules.substring(1);

    }

**var** filtersStr = '{"groupOp":"AND","rules":[' + rules + ']}';

**var** postData = $("#gridTable").jqGrid("getGridParam", "postData");

    $.extend(postData, {filters: filtersStr});

    $("#gridTable").jqGrid("setGridParam", {

        search: **true**

    }).trigger("reloadGrid", [{page:1}]);

    $("#multipleSearchDialog").dialog("close");

};

# jqGrid选项补充说明

这篇中将补充说明一些之前没有讲到的细节内容，比如一些特殊功能的jqGrid选项、jqGrid的事件等等。

**1. jqGrid选项补充说明**   
  
之前已经介绍了许多经常用到的jqGrid选项，还有一部分虽然还没有介绍，但却能肩负特殊的功能。例如：

* **altRows** ：当次此选项为true时，表格的行将呈现交替样式显示（zebra-striped）。
* **altclass** ：此选项只在altRows为true时才有效，交替显示的样式；默认值为：ui-priority-secondary。
* **datastr** ：当datatype选项为xmlstring或jsonstring时，设置此选项可以充当Grid中的数据内容。
* **deepempty** ：如果有事件或者插件被绑定到表格的cell上，则应该经此选项设为true；此选项将使用jQuery的empty方法来处理行及其子元素的回收工作；虽然会有性能下降，但可以避免内存泄露。
* **forceFit** ：如果此选项为true，则当调整某一列的宽度的时候，其右侧相邻的列也会相应调整宽度，以保持整个表格的宽度不变；但此选项受制于shrinkToFit选项，当shrinkToFit为false时，此选项也将被忽略。
* **gridstate** ：当使用了hiddengrid、hidegrid、caption选项的时候，决定表格的状态；可能的值为visible和hidden，默认值为visible。
* **hidegrid** ：由于控制是否启用或禁用表格标题栏中的“显示/隐藏”表格按钮；只有在caption属性为非空字符串的情况下有效。
* **hiddengrid** ：如果设为true，则表格在初始化状态下是隐藏的，只有标题栏显示，且不会像Server发送请求，也就不会载入任何数据；只有第一次点击标题栏中的“显示/隐藏”表格按钮后，才会向Server发送请求，载入数据，显示表格；此选项只有在caption选项非空，且hidegrid为true的情况下才有效。
* **lastpage** ：只读选项，表示从请求返回的总页码数，默认为0。
* **lastsort** ：只读选项，表示最后一次排序使用的列的索引序数，从0开始，默认为0。
* **loadonce** ：如果此项为true，则只向Server发送一次请求。当第一次请求结束后，datatype选项自动变更为local，所有的后续操作都只是基于客户端的；而pager的各项功能也都将被禁用。
* **multiselect** ：如果此项为true，则允许多行同时选择；在表格的左端将出现新的一列，用于多行选择。
* **multikey** ：此项只在multiselect为true时有意义，定义按哪个键来进行多行选择，可能的值：shiftKey、altKey、ctrlKey。
* **multiboxonly** ：此项只在multiselect为true时有意义；如果此项为true，则只有点击多选列中的checkbox，才能进行多行选择，点击行的其他位置，都只是单行选择。
* **multiselectWidth** ：设置多选列的宽度。
* **selarrrow** ：只读选项，表示在multiselect为true时，多选行的id数组。
* **postData** ：{name:value,...}类型的数组，此数组中的元素将会作为参数，被直接传入url；
* **sortname** ：表格初始化时使用的排序列的index。
* **sortorder** ：表格初始化时使用的排序方式，asc或desc。
* **scroll** ：创建动态滚动的表格，可以使用垂直滚动条来载入数据。此选项可以接受boolean值或integer值；默认为false；如果为true，则表格将保持从开始到现在所有曾经显示行；如果为设为数值（例如1），则表格只保持当前显示的数据行，以防内存泄露。（具体参见官方文档<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:options> ）

**2. colModel选项补充说明**

* **align** ：文本对齐方式，left、center、right。
* **datefmt** ：日期格式，默认为ISO格式（Y-m-d），使用的是PHP风格的日期格式，参见官方文档。
* **formatter** ：使用预定义的格式，或者用于控制此列格式的自定义函数名。具体参见官方文档（<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:predefined_formatter> ）。
* **key** ：当从Server得到的数据中不包括id的时候，设定该列充当id，以代表唯一的行id；最多可以有一列设置此选项为true；如果多于一列设置了此项为true，则第一个设定的列将充当id列，其余的将被忽略。

注：

* name选项的值可以包含“.”,例如patentCaseType.caption，即对应从gridModel（根据我的jsonReader设置）中的元素的patentCaseType属性的caption属性；
* name选项主要用途是用来解析从response中传来的对应数据；
* index选项主要用途是用来在查询和排序时，向Server传递对应的参数；

**3. jqGrid事件**   
  
事件说明列表可以参考官方文档（<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:events> ），不再赘述。  
  
需要说明的是，自从3.6.3版本以后，可以在对应的事件方法函数中使用$(this)来应用jqGrid对象。

# j[qGrid与Struts2的结合应用（八） —— 奇妙的subGrid子表格](http://blog.csdn.net/c622724/article/details/8372553)

分类： [JqGrid学习记录](http://blog.csdn.net/c622724/article/category/1309794) 2012-12-22 10:15 89人阅读 [评论](http://blog.csdn.net/c622724/article/details/8372553#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/c622724/article/details/8372553#report)

[jqGrid与Struts2的结合应用八](http://www.csdn.net/tag/jqGrid%e4%b8%8eStruts2%e7%9a%84%e7%bb%93%e5%90%88%e5%ba%94%e7%94%a8%e5%85%ab)

jqGrid的一项高级功能就是嵌套子表格，使用起来也非常简单。使用的方式有两种：

        使用普通的subGrid子表格；

* 使用一个完整jqGrid作为子表格；

显然第二种方式更加通用，功能更加强大，因此我主要使用第二种方式。至于第一种方式，可以具体参考官方文档（<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:subgrid> ）。  
  
**使用完整jqGrid作为子表格**   
  
使用子表格，涉及到jqGrid的三个选项：

* **subGrid** ：首先必须将jqGrid的subGrid选项设置为true，默认为false；当此项设为true的时候，Grid表格的最左边将会添加一列，里面有一个“+”图标，用于展开子格；
* **subGridRowExpanded** ：当点击“+”展开子表格时，将触发此选项定义的事件方法；
* **subGridRowColapsed** ：当点击“-”收起子表格时，将触发此选项定义的事件方法；

**注1** ：subGridRowExpanded定义的事件方法函数将会得到两个参数：

* **subgrid\_id** ：子表格的id；当子表格展开的时候，在主表格中会创建一个div元素用来容纳子表格，subgrid\_id就是这个div的id。
* **row\_id** ：主表格中所要展开子表格的行的id。

**注2** ：在subGridRowExpanded定义的事件方法函数中，即可根据以上两个参数，创建新的jqGrid对象，其中的options选项可参考主表格的jqGrid选项来运用。可以理解为在主表格中在嵌套一个新的jqGrid对象。当然，在subGridRowExpanded定义的事件函数中，也不一定非得嵌套一个新的jqGrid到主表格，可以将其放置于任何自己希望的位置，甚至做些别的操作，根本不去里理会子表格，比如使用$.ajax()向Server发送ajax请求，并根据返回的json，做些自定义的操作。  
  
**注3** ：应当注意subGridRowColapsed的拼写并太符合规范，但却不得不遵循这个拼写。哈哈~  
  
**注4** ：相对于subGridRowExpanded必须定义事件函数来完成操作，subGridRowColapsed不用刻意定义对应的事件函数；因为当行收起的时候，上面提到的那个动态创建用来容纳子表格的div，其里面的内容将会被自动移除(removed)。  
  
现在来看看代码中到底多了哪些内容。

FetchPatentCasesAction代码：

**[java]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5733058)

1. **package** cn.gengv.struts2ex.jqGrid;
2. **import** java.util.List;
3. **import** com.byzl.hare.dao.impl.Criterion;
4. **import** com.byzl.hare.model.Contact;
5. **import** com.byzl.hare.model.PatentCase;
6. **import** com.byzl.hare.service.ContactService;
7. @SuppressWarnings("serial")
8. **public** **class** FetchPatentCasesAction **extends** JqGridBaseAction<PatentCase> {
10. **private** ContactService contactService;
12. **private** Contact contact;
14. @Override
15. **public** String execute() {
16. **return** **this**.refreshGridModel();
17. }
19. @Override
20. **public** **int** getResultSize() {
21. **return** **this**.contactService.queryPatentCasesCount(contact);
22. }
23. @Override
24. **public** **int** getResultSize(List<Criterion> criteria) {
25. **return** **this**.contactService.queryPatentCasesCount(contact);
26. }
27. @Override
28. **public** List<PatentCase> listResults(**int** from, **int** length) {
29. **return** **this**.contactService.queryPatentCases(contact, from, length);
31. }
32. @Override
33. **public** List<PatentCase> listResults(List<Criterion> criteria, **int** from,
34. **int** length) {
35. **return** **this**.contactService.queryPatentCases(contact, from, length);
36. }
37. @Override
38. **public** **void** sortResults(List<PatentCase> results, String field, String order) {
39. // TODO Auto-generated method stub
41. }
42. **public** **void** setContactService(ContactService contactService) {
43. **this**.contactService = contactService;
44. }
45. **public** Contact getContact() {
46. **return** contact;
47. }
48. **public** **void** setContact(Contact contact) {
49. **this**.contact = contact;
50. }
51. }

Action设置：

**[xhtml]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5733058)

1. **<action** name="fetchPatentCases" class="cn.gengv.struts2ex.jqGrid.FetchPatentCasesAction"**>**
2. **< result** name="success" type="json"**>**
3. **< param** name="includeProperties"**>**
4. gridModel/[/d+/]/./w+,
5. rows,page,total,record
6. **< /param>**
7. **< param** name="noCache"**>**true**</param>**
8. **< param** name="ignoreHierarchy"**>**false**</param>**
9. **< /result>**
10. **</action>**

javascript代码：

**[javascript]** [view plaincopy](http://blog.csdn.net/gengv/article/details/5733058)

1. $(**function**(){
2. // 配置jqGrid组件
3. $("#gridTable").jqGrid({
4. url: "jqGrid05.action",
5. datatype: "json",
6. mtype: "GET",
7. height: 350,
8. width: 600,
9. colModel: [
10. {name:"id",index:"id",label:"编码",width:40},
11. {name:"lastName",index:"lastName",label:"姓",width:80},
12. {name:"firstName",index:"firstName",label:"名",width:80},
13. {name:"email",index:"email",label:"电子邮箱",width:160,sortable:**false**},
14. {name:"telNo",index:"telNo",label:"电话",width:120,sortable:**false**}
15. ],
16. viewrecords: **true**,
17. rowNum: 15,
18. rowList: [15,50,100],
19. prmNames: {search: "search"},
20. jsonReader: {
21. root:"gridModel",
22. records: "record",
23. repeatitems : **false**,
24. },
25. pager: "#gridPager",
26. caption: "联系人列表",
27. hidegrid: **false**,
28. shrikToFit: **true**,
29. subGrid: **true**,  // (1)开启子表格支持
30. subGridRowExpanded: **function**(subgrid\_id, row\_id) {  // (2)子表格容器的id和需要展开子表格的行id，将传入此事件函数
31. **var** subgrid\_table\_id;
32. subgrid\_table\_id = subgrid\_id + "\_t";   // (3)根据subgrid\_id定义对应的子表格的table的id
34. **var** subgrid\_pager\_id;
35. subgrid\_pager\_id = subgrid\_id + "\_pgr"  // (4)根据subgrid\_id定义对应的子表格的pager的id
37. // (5)动态添加子报表的table和pager
38. $("#" + subgrid\_id).html("<table id='"+subgrid\_table\_id+"' class='scroll'></table><div id='"+subgrid\_pager\_id+"' class='scroll'></div>");
40. // (6)创建jqGrid对象
41. $("#" + subgrid\_table\_id).jqGrid({
42. url: "fetchPatentCases.action?contact.id="+row\_id,  // (7)子表格数据对应的url，注意传入的contact.id参数
43. datatype: "json",
44. colNames: ['编号','内部编码','名称','申请号'],
45. colModel: [
46. {name:"id",index:"id",width:80,key:**true**},
47. {name:"internalNo",index:"internalNo",width:130},
48. {name:"name",index:"name",width:80,align:"right"},
49. {name:"applicationNo",index:"applicationNo",width:80,align:"right"}
50. ],
51. jsonReader: {   // (8)针对子表格的jsonReader设置
52. root:"gridModel",
53. records: "record",
54. repeatitems : **false**
55. },
56. prmNames: {search: "search"},
57. pager: subgrid\_pager\_id,
58. viewrecords: **true**,
59. height: "100%",
60. rowNum: 5
61. });
62. }
63. });
65. });

说明：

* 由于FetchPatentCasesAction继承自JqGridBaseAction<PatentCase>，因此可以像之前的例子那样，使用这个Action类来应对jqGrid子表格的请求。
* FetchPatentCasesAction的Action设置也可以仿照其他针对jqGrid的Action来设置。
* 对于子表格Grid来说，和主表格Grid并没有什么分别，主表格中可以用的选项，在子表格中也可以使用。因此需要设置jsonReader等选项来配合FetchPatentCasesAction中的成员变量。

OK，就这么简单，子表格功能完成了。  
  


# [jqGrid与Struts2的结合应用（七） —— 浅谈排序](http://blog.csdn.net/c622724/article/details/8372560)

分类： [JqGrid学习记录](http://blog.csdn.net/c622724/article/category/1309794) 2012-12-22 10:16 104人阅读 [评论](http://blog.csdn.net/c622724/article/details/8372560#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/c622724/article/details/8372560#report)

[jqGrid与Struts2的结合应用七](http://www.csdn.net/tag/jqGrid%e4%b8%8eStruts2%e7%9a%84%e7%bb%93%e5%90%88%e5%ba%94%e7%94%a8%e4%b8%83)

终于讲到排序了，这一部分应该说还是比较好理解的。

jqGrid通过colModel选项中的sortable来控制是否可以以某列的值排序。sortable的默认值是true，当设为false时，即此列不能用于排序。

[javascript] [view plaincopy](http://blog.csdn.net/gengv/article/details/5732698)

$(function(){

    $("#gridTable").jqGrid({

        ...

        colModel: [

              {name:"id",index:"id",label:"编码",width:40},

              {name:"lastName",index:"lastName",label:"姓",width:80},

              {name:"firstName",index:"firstName",label:"名",width:80},

              {name:"email",index:"email",label:"电子邮箱",width:160,sortable:false},

              {name:"telNo",index:"telNo",label:"电话",width:120,sortable:false}

        ],

        ...

    });

});

当点击sortable为true的列首时，jqGrid会向Server发送排序请求，例如：  
http://localhost:8085/Hare/jqGridTest/jqGrid05.action?search=false&nd=1279006749246&rows=15&page=3&sidx=firstName&sord=asc  
  
注：其中sord和sidx参数名都是在jqGrid的prmNames选项中设定的（可参考本系列文章的第一篇）。而sidx参数的值即各列的colModel的index选项值。（在查询和排序时，发送的关于列的参数都是基于colModel的index属性的）  
  
后面的事情就交给服务器端的Action来处理了，还拿我们的Contact联系人列表为例。  
  
既然我们可能会分别使用不同的字段来排序，那么就必须为Contact提供不同的Comparator来简化比较操作。因此我写了一个针对Contact的Comparator的工厂类，用来根据不同的字段提供不同的Comparator。  
  
ContactComparatorFactory的代码：

[java] [view plaincopy](http://blog.csdn.net/gengv/article/details/5732698)

package cn.gengv.struts2ex.jqGrid;

import java.text.Collator;

import java.util.Comparator;

import com.byzl.hare.model.Contact;

public class ContactComparatorFactory {

    private static Collator collator\_Chinese = Collator.getInstance(java.util.Locale.CHINA);

    private final static Comparator<Contact> idComparator = new IdComparator();

    private final static Comparator<Contact> firstNameComparator = new FirstNameComparator();

    private final static Comparator<Contact> lastNameComparator = new LastNameComparator();

    private final static Comparator<Contact> fullNameComparator = new FullNameComparator();

    private final static Comparator<Contact> idCardNoNoComparator = new IdCardNoComparator();

    private final static Comparator<Contact> nationalityComparator = new NationalityComparator();

    public static Comparator<Contact> getComparator(String compareType) {

        if ("id".equalsIgnoreCase(compareType)) {

            return idComparator;

        } else if ("firstName".equalsIgnoreCase(compareType)) {

            return firstNameComparator;

        } else if ("lastName".equalsIgnoreCase(compareType)) {

            return lastNameComparator;

        } else if ("fullName".equalsIgnoreCase(compareType)) {

            return fullNameComparator;

        } else if ("idCardNoNo".equalsIgnoreCase(compareType)) {

            return idCardNoNoComparator;

        } else if ("nationality".equalsIgnoreCase(compareType)) {

            return nationalityComparator;

        } else {

            return null;

        }

    }

    public static class IdComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                int id1 = c1.getId();

                int id2 = c2.getId();

                return id1 == id2 ? 0 : (id1 < id2 ? -1 : 1);

            }

        }

    }

    public static class FirstNameComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                String s1 = c1.getFirstName();

                String s2 = c2.getFirstName();

                if (s1 == null && s2 == null) {

                    return 0;

                } else if (s1 == null && s2 != null) {

                    return -1;

                } else if (s1 != null && s2 == null) {

                    return 1;

                } else {

                    return collator\_Chinese.compare(s1, s2);

                }

            }

        }

    }

    public static class LastNameComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                String s1 = c1.getLastName();

                String s2 = c2.getLastName();

                if (s1 == null && s2 == null) {

                    return 0;

                } else if (s1 == null && s2 != null) {

                    return -1;

                } else if (s1 != null && s2 == null) {

                    return 1;

                } else {

                    return collator\_Chinese.compare(s1, s2);

                }

            }

        }

    }

    public static class FullNameComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                String s1 = c1.getFullName();

                String s2 = c2.getFullName();

                if (s1 == null && s2 == null) {

                    return 0;

                } else if (s1 == null && s2 != null) {

                    return -1;

                } else if (s1 != null && s2 == null) {

                    return 1;

                } else {

                    return collator\_Chinese.compare(s1, s2);

                }

            }

        }

    }

    public static class IdCardNoComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                String s1 = c1.getIdCardNo();

                String s2 = c2.getIdCardNo();

                if (s1 == null && s2 == null) {

                    return 0;

                } else if (s1 == null && s2 != null) {

                    return -1;

                } else if (s1 != null && s2 == null) {

                    return 1;

                } else {

                    return s1.compareToIgnoreCase(s2);

                }

            }

        }

    }

    public static class NationalityComparator implements Comparator<Contact> {

        public int compare(Contact c1, Contact c2) {

            if (c1 == null && c2 == null) {

                return 0;

            } else if (c1 == null && c2 != null) {

                return -1;

            } else if (c1 != null && c2 == null) {

                return 1;

            } else {

                String s1 = c1.getNationality();

                String s2 = c2.getNationality();

                if (s1 == null && s2 == null) {

                    return 0;

                } else if (s1 == null && s2 != null) {

                    return -1;

                } else if (s1 != null && s2 == null) {

                    return 1;

                } else {

                    return collator\_Chinese.compare(s1, s2);

                }

            }

        }

    }

}

然后再来看JqGridBaseAction，其中添加了一个抽象方法，用来将数据结果进行排序。

[java] [view plaincopy](http://blog.csdn.net/gengv/article/details/5732698)

package cn.gengv.struts2ex.jqGrid;

// import ...

@SuppressWarnings("serial")

public abstract class JqGridBaseAction<T> extends ActionSupport {

    ...

    // (1)添加排序方法

    public abstract void sortResults(List<T> results, String field, String order);

    public String refreshGridModel() {

        try {

            List<Criterion> criteria = Collections.emptyList();

            if(search == true) {

                criteria = new ArrayList<Criterion>();

                if(filters != null && filters.length()>0) {

                    criteria.addAll(this.generateSearchCriteriaFromFilters(filters));

                }

                Criterion criterion = this.generateSearchCriterion(searchField, searchString, searchOper);

                if(criterion != null) {

                    criteria.add(criterion);

                }

            }

            List<T> results = Collections.emptyList();

            int from = rows \* (page - 1);

            int length = rows;

            if(loadonce) {

                from = 0;

                length = 100;

            }

            if(!criteria.isEmpty()) {

                record = this.getResultSize(criteria);

                results = this.listResults(criteria, from, length);

            } else {

                record = this.getResultSize();

                results = this.listResults(from, length);

            }

            // (2)将结果排序

            if(sidx != null && sord != null) {

                sortResults(results, sidx, sord);

            }

            this.setGridModel(results);

            total = (int) Math.ceil((double) record / (double) rows);

            return SUCCESS;

        } catch (Exception e) {

            e.printStackTrace();

            this.addActionError(e.getMessage());

            return ERROR;

        }

    }

    ...

}

而在ListContactsAction中提供了方法实现：

[java] [view plaincopy](http://blog.csdn.net/gengv/article/details/5732698)

package cn.gengv.struts2ex.jqGrid;

import java.util.Collections;

import java.util.Comparator;

import java.util.List;

import com.byzl.hare.dao.impl.Criterion;

import com.byzl.hare.model.Contact;

import com.byzl.hare.service.ContactService;

@SuppressWarnings("serial")

public class ListContactsAction extends JqGridBaseAction<Contact> {

    ...

    @Override

    public void sortResults(List<Contact> results, String field, String order) {

        // (1)根据field获得对应的Comparator

        Comparator<Contact> comparator = ContactComparatorFactory.getComparator(field);

        if(comparator != null) {

            // (2)使用Comparator排序

            Collections.sort(results, comparator);

            // (3)如果需要的排序顺序为desc，则颠倒顺序

            if("desc".equals(order)) {

                Collections.reverse(results);

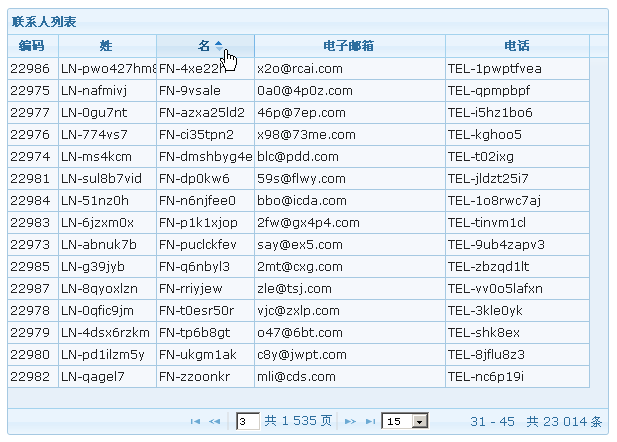
            }

        }

    }

    ...

}



不过这个例子存在一定的局限性，即只能将当前页中的数据根据某列进行排序；而不能跨页间进行数据排序。之所以存在这种局限，也是源于实际应用中的客观限制。还以这个例子来说，数据库里总共模拟了两万多条数据记录。如果每次要将这些记录进行排里的话，除非有数据库索引支持，否则所要消耗的时间也是相当客观的，对于用户体验来说，几乎就是灾难。如果数据量更多的话，结果可想而知。  
  
因此，我们应该换一个角度来看这个问题，用户之所以使用排序，更多的目的还是在于查找数据方便，既然我们可以提供条件查询（尤其是复杂条件查询），那么用户对于排序的需求也就不会那么迫切了。同时也可以体会到，排序更多地应用在少量数据的场合下。

# [jqGrid与Struts2的结合应用（四） —— 丰富多彩的Pager Bar](http://blog.csdn.net/johnstrive/article/details/42014425)

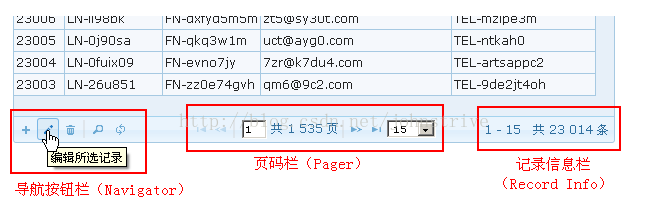
分类： [web前台](http://blog.csdn.net/johnstrive/article/category/677938) 2014-12-18 22:42 8人阅读 [评论](http://blog.csdn.net/johnstrive/article/details/42014425#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/johnstrive/article/details/42014425#report)

原文地址：http://blog.csdn.net/gengv/article/details/5720707

如不能转载，请联系博主，我马上删除。

在介绍排序与查询之前，先介绍一下Pager、Navigator和自定义按钮。  
  
默认情况下，Pager Bar分为3个部分：

* 页码栏区域（Pager）
* 导航按钮区域（Navigator Buttons）
* 记录信息区域（Record Information）



**1. Pager和Record Information的常用属性**

在jqGrid的options选项中，有许多是专门针对Pager的，之前的例子中已经使用了一个——“pager”选项，用来指明Pager Bar在页面中对应的区域。  
  
主要的属性有：

* **pager** ：必须是一个有效的html元素。这个元素可以根据页面的需要，被放置于页面的任何位置。在指定pager选项的值的时候，可以使用三种形式：'pager'、'#pager'、$('#pager')。但推荐使用第二种，即'#pager'。
* **pgbuttons** ：决定在Pager Bar里是否显示用于翻页的按键，默认为true。
* **pginput** ：决定在Pager Bar里是否显示用于输入目标页码的输入框，默认为true。
* **pgtext** ：显示当前页码状态的字符串，这个与所选用的language文件有关，具体的文本格式定义在里面。例如默认的英文格式就是“Page {0} of {1}”，其中{0}代表当前载入的页码；{1}代表页码的总数。
* **pagerpos** ：用于定义页码栏在Pager Bar中的位置，默认情况下为center，还可以是left或right。
* **rowNum** ：定义一次性显示多少条记录，默认值为20；这个选项在第一篇文章中已经介绍过了。
* **rowList** ：定义rowNum可供选择的值的数组，默认值为空数组；这个选项在第一篇文章中也已经介绍过了。
* **lastpage** ：只读属性，返回页码总数，也即从response中解析出来的那个对应着“total”项的值（参见第一篇文章jsonReader部分）。
* **viewrecords** ：定义是否在Pager Bar中显示记录数信息。
* **recordpos** ：用于定义记录信息在Pager Bar中的显示位置，默认为right，还可以是left或center。
* **records** ：只读属性，返回记录总数，也即从response中解析出来的那个对应着“record”项的值（参见第一篇文章jsonReader部分）。
* **recordtext** ：记录信息显示时所使用的文本格式，只有当viewrecords为true，且记录总数大于0的时候才会显示。其格式与所选用的language文件有关，具体的文本格式定义在里面。例如默认的英文格式就是“View {0} - {1} of {2}”，其中{0}代表当前页第一条记录在整个记录总数中的位置；{1}代表当前页最后一条记录在整个记录总数中的位置；{2}代表所有记录的总数。

注1：这些属性（选项），并不都是可以用jqGrid的setGridParam方法在初始化之后改变值的。比如pgtext、recordtext、rowNum就是可以修改的，而其余的都是不能修改的，也即只能使用默认值，或者在jqGrid初始化的时候就定义好了。  
注2：在jqGrid创建并初始化之后，修改这些关于Pager的属性，都必须执行trigger("reloadGrid")才能生效。  
注3：在第一篇文章中我曾写到，经过测试jqGrid的caption似乎也不能用setGridParam方法改变并生效。我再次查了一下文档，发现这个选项也明确标注着也是不能改变，除非使用相应的setCaption方法。

**2. Navigator**

默认有6个预定义好的按钮：

* 添加新行
* 编辑选中的行
* 查看选中的行
* 删除选中的行
* 查找记录
* 重载表格

Navigator通过navGrid方法来配置导航栏，用法是：

$("#grid\_id").navGrid('#gridpager',{parameters},prmEdit, prmAdd, prmDel, prmSearch, prmView);

或

$("#grid\_id").jqGrid('navGrid','#gridpager',{parameters},prmEdit, prmAdd, prmDel, prmSearch, prmView);

关于各种参数的具体说明，请参考官方文档(<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:navigator> )

**2.1 关于Navigator的参数**

* **add** ：定义是否启用添加操作的按钮；
* **addicon** ：用于添加操作的按钮上的图标，目前只能使用jQuery UI中的图标样式；
* **addtext** ：用于添加操作的按钮上的文本；
* **addtitle** ：用于添加操作的按钮上的tooltip；
* **del/delicon/deltext/deltitle** ：用于删除按钮的一套属性，可以参照添加按钮的几个属性；
* **edit/editicon/edittext/edittitle** ：用于编辑按钮的一套属性，可以参照添加按钮的几个属性；
* **refresh/refreshicon/refreshtext/refreshtitle** ：用于刷新表格按钮的一套属性，可以参照添加按钮的几个属性；当点击这个按钮时，将执行trigger("reloadGrid")并清除用于查询的参数；
* **refreshstate** ：定义如何重新载入Grid表格，firstpage：重新载入表格的第一页；current：重新载入当前页，并保存当前选中的行；默认为firstpage；
* **afterRefresh** ：定义点击刷新表格按钮之后的事件函数；
* **beforeRefresh** ：定义点击刷新表格按钮之前的事件函数；
* **search/searchicon/searchtext/searchtitle** ：用于查询按钮的一套属性，可以参照添加按钮的几个属性；当点击这个按钮时，将会调用searchGrid方法；
* **view/viewicon/viewtext/viewtitle** ：用于查看记录按钮的一套属性，可以参照添加按钮的几个属性；
* **addfunc** ：如果定义，则用定义的函数替代原有的add函数，这个函数将不接受任何参数；
* **editfunc** ：如果定义，则用定义的函数替代原有的edit函数，编辑的行的id作为参数传入这个函数；
* **delfunc** ：如果定义，则用定义的函数替代原有的del函数，编辑的行的id作为参数传入这个函数；
* **alertcap** ：当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示框的标题；
* **alerttext** ：当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示信息；
* **cloneToTop** ：定义是否“拷贝”一套导航按钮到Grid表格顶端的Pager中，默认为false；

借助这些选项，可以将上一篇中的例子，修改一下，更多的借助jqGrid内置的功能。例如，当载入行数据到对话框时，之前是通过在自定义的loadSelectedRowData方法中，调用jqGrid("getGridParam", "selrow")来检查是否有选中的数据行。现在在点击编辑或删除按钮的时候，可以通过jqGrid自己检查是否有选中的数据行，如果没有则弹出提示。如果有则选中的行，则还会将行id作为参数，传入对应的函数方法中。

将代码修改如下：

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/gengv/article/details/5720707)

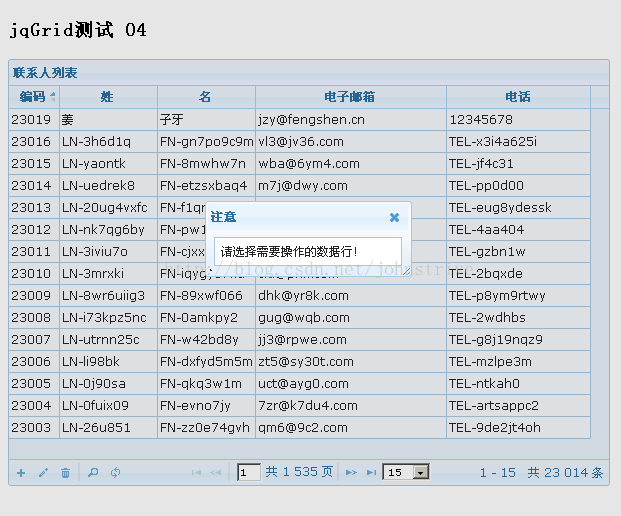
1. $(function(){
2. ...
4. $("#gridTable").jqGrid("navGrid", "#gridPager", {
5. addfunc : openDialog4Adding,    // (1) 点击添加按钮，则调用openDialog4Adding方法
6. editfunc : openDialog4Updating, // (2) 点击编辑按钮，则调用openDialog4Updating方法
7. delfunc : openDialog4Deleting,  // (3) 点击删除按钮，则调用openDialog4Deleting方法
8. alerttext : "请选择需要操作的数据行!"  // (4) 当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示信息
9. });
11. ...
12. });
13. var openDialog4Adding = function() {
14. var consoleDlg = $("#consoleDlg");
15. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
16. consoleDlg.find("input").removeAttr("disabled").val("");
17. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
18. dialogButtonPanel.find("button:contains('创建')").show();
19. consoleDlg.dialog("option", "title", "创建新联系人").dialog("open");
20. };
21. var openDialog4Updating = function(rowid) { // (6) 接受选中行的id为参数
22. var consoleDlg = $("#consoleDlg");
23. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
25. consoleDlg.find("input").removeAttr("disabled");
26. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
27. dialogButtonPanel.find("button:contains('保存')").show();
28. consoleDlg.dialog("option", "title", "修改联系人信息");
30. loadSelectedRowData(rowid); // (7) 将选中行id传入loadSelectedRowData方法
31. };
32. var openDialog4Deleting = function(rowid) { // (8) 接受选中行的id为参数
33. var consoleDlg = $("#consoleDlg");
34. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
36. consoleDlg.find("input").attr("disabled", true);
37. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
38. dialogButtonPanel.find("button:contains('删除')").show();
39. consoleDlg.dialog("option", "title", "删除联系人");
41. loadSelectedRowData(rowid);
42. };
43. var loadSelectedRowData = function(selectedRowId) { // (9) 接受选中行的id为参数
45. var params = {
46. "contact.id" : selectedRowId
47. };
48. var actionUrl = "viewContact.action";
49. // 从Server读取对应ID的JSON数据
50. $.ajax( {
51. url : actionUrl,
52. data : params,
53. dataType : "json",
54. cache : false,
55. error : function(textStatus, errorThrown) {
56. alert("系统ajax交互错误: " + textStatus);
57. },
58. success : function(data, textStatus) {
59. // 如果读取结果成功，则将信息载入到对话框中
60. var rowData = data.contact;
61. var consoleDlg = $("#consoleDlg");
62. consoleDlg.find("#selectId").val(rowData.id);
63. consoleDlg.find("#lastName").val(rowData.lastName);
64. consoleDlg.find("#firstName").val(rowData.firstName);
65. consoleDlg.find("#certificateNo").val(rowData.certificateNo);
66. consoleDlg.find("#email").val(rowData.email);
67. consoleDlg.find("#telNo").val(rowData.telNo);
68. consoleDlg.find("#address").val(rowData.address);
69. consoleDlg.find("#idCardNo").val(rowData.idCardNo);
70. consoleDlg.find("#nationality").val(rowData.nationality);
72. // 根据新载入的数据将表格中的对应数据行一并刷新一下
73. var dataRow = {
74. id : rowData.id,
75. lastName : rowData.lastName,
76. firstName : rowData.firstName,
77. email : rowData.email,
78. telNo : rowData.telNo
79. };
81. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow);
83. // 打开对话框
84. consoleDlg.dialog("open");
85. }
86. });
87. };

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/johnstrive/article/details/42014425)

1. $(function(){
2. ...
4. $("#gridTable").jqGrid("navGrid", "#gridPager", {
5. addfunc : openDialog4Adding,    // (1) 点击添加按钮，则调用openDialog4Adding方法
6. editfunc : openDialog4Updating, // (2) 点击编辑按钮，则调用openDialog4Updating方法
7. delfunc : openDialog4Deleting,  // (3) 点击删除按钮，则调用openDialog4Deleting方法
8. alerttext : "请选择需要操作的数据行!"  // (4) 当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示信息
9. });
11. ...
12. });
13. var openDialog4Adding = function() {
14. var consoleDlg = $("#consoleDlg");
15. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
16. consoleDlg.find("input").removeAttr("disabled").val("");
17. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
18. dialogButtonPanel.find("button:contains('创建')").show();
19. consoleDlg.dialog("option", "title", "创建新联系人").dialog("open");
20. };
21. var openDialog4Updating = function(rowid) { // (6) 接受选中行的id为参数
22. var consoleDlg = $("#consoleDlg");
23. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
25. consoleDlg.find("input").removeAttr("disabled");
26. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
27. dialogButtonPanel.find("button:contains('保存')").show();
28. consoleDlg.dialog("option", "title", "修改联系人信息");
30. loadSelectedRowData(rowid); // (7) 将选中行id传入loadSelectedRowData方法
31. };
32. var openDialog4Deleting = function(rowid) { // (8) 接受选中行的id为参数
33. var consoleDlg = $("#consoleDlg");
34. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
36. consoleDlg.find("input").attr("disabled", true);
37. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
38. dialogButtonPanel.find("button:contains('删除')").show();
39. consoleDlg.dialog("option", "title", "删除联系人");
41. loadSelectedRowData(rowid);
42. };
43. var loadSelectedRowData = function(selectedRowId) { // (9) 接受选中行的id为参数
45. var params = {
46. "contact.id" : selectedRowId
47. };
48. var actionUrl = "viewContact.action";
49. // 从Server读取对应ID的JSON数据
50. $.ajax( {
51. url : actionUrl,
52. data : params,
53. dataType : "json",
54. cache : false,
55. error : function(textStatus, errorThrown) {
56. alert("系统ajax交互错误: " + textStatus);
57. },
58. success : function(data, textStatus) {
59. // 如果读取结果成功，则将信息载入到对话框中
60. var rowData = data.contact;
61. var consoleDlg = $("#consoleDlg");
62. consoleDlg.find("#selectId").val(rowData.id);
63. consoleDlg.find("#lastName").val(rowData.lastName);
64. consoleDlg.find("#firstName").val(rowData.firstName);
65. consoleDlg.find("#certificateNo").val(rowData.certificateNo);
66. consoleDlg.find("#email").val(rowData.email);
67. consoleDlg.find("#telNo").val(rowData.telNo);
68. consoleDlg.find("#address").val(rowData.address);
69. consoleDlg.find("#idCardNo").val(rowData.idCardNo);
70. consoleDlg.find("#nationality").val(rowData.nationality);
72. // 根据新载入的数据将表格中的对应数据行一并刷新一下
73. var dataRow = {
74. id : rowData.id,
75. lastName : rowData.lastName,
76. firstName : rowData.firstName,
77. email : rowData.email,
78. telNo : rowData.telNo
79. };
81. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow);
83. // 打开对话框
84. consoleDlg.dialog("open");
85. }
86. });
87. };



说明：(2)(3)处意味着，当点击编辑或删除按钮时，如果有数据行被选中，则将行id作为参数传入对应的openDialog4Updating和openDialog4Deleting方法中；如果没有选中的行，则将弹出窗口，提示信息。



**2.2 关于prmEdit、prmAdd、prmDel、prmSearch、prmView的参数**

jqGrid本身为这些参数定义了默认值，而且不同的language文件也有各自的定义来覆盖原始的默认值。我们也可以在jqGrid重新定义新的值，以覆盖默认的值。  
  
例如：

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/gengv/article/details/5720707)

1. $("#gridTable").jqGrid("navGrid", "#gridPager",
2. {
3. addfunc : openDialog4Adding,
4. editfunc : openDialog4Updating,
5. delfunc : openDialog4Deleting,
6. alerttext : "请选择需要操作的数据行!"
7. },
8. {}, // prmEdit
9. {}, // prmAdd
10. {}, // prmDel
11. {       // prmSearch
12. caption: "查找",
13. Find: "Let's go!",
14. multipleSearch : true
15. },
16. {}  // prmView
17. );

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/johnstrive/article/details/42014425)

1. $("#gridTable").jqGrid("navGrid", "#gridPager",
2. {
3. addfunc : openDialog4Adding,
4. editfunc : openDialog4Updating,
5. delfunc : openDialog4Deleting,
6. alerttext : "请选择需要操作的数据行!"
7. },
8. {}, // prmEdit
9. {}, // prmAdd
10. {}, // prmDel
11. {       // prmSearch
12. caption: "查找",
13. Find: "Let's go!",
14. multipleSearch : true
15. },
16. {}  // prmView
17. );



具体的参数如下，也可参考官方文档(<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:navigator> )

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/gengv/article/details/5720707)

1. $.jgrid = {
2. ...
3. search : {
4. caption: "Search...",
5. Find: "Find",
6. Reset: "Reset",
7. odata : ['equal', 'not equal', 'less', 'less or equal','greater','greater or equal', 'begins with','does not begin with','is in','is not in','ends with','does not end with','contains','does not contain'],
8. groupOps: [ { op: "AND", text: "all" }, { op: "OR", text: "any" } ],
9. matchText: " match",
10. rulesText: " rules"
11. },
12. edit : {
13. addCaption: "Add Record",
14. editCaption: "Edit Record",
15. bSubmit: "Submit",
16. bCancel: "Cancel",
17. bClose: "Close",
18. saveData: "Data has been changed! Save changes?",
19. bYes : "Yes",
20. bNo : "No",
21. bExit : "Cancel",
22. },
23. view : {
24. caption: "View Record",
25. bClose: "Close"
26. },
27. del : {
28. caption: "Delete",
29. msg: "Delete selected record(s)?",
30. bSubmit: "Delete",
31. bCancel: "Cancel"
32. },
33. nav : {
34. edittext: "",
35. edittitle: "Edit selected row",
36. addtext:"",
37. addtitle: "Add new row",
38. deltext: "",
39. deltitle: "Delete selected row",
40. searchtext: "",
41. searchtitle: "Find records",
42. refreshtext: "",
43. refreshtitle: "Reload Grid",
44. alertcap: "Warning",
45. alerttext: "Please, select row",
46. viewtext: "",
47. viewtitle: "View selected row"
48. },
49. ...

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/johnstrive/article/details/42014425)

1. $.jgrid = {
2. ...
3. search : {
4. caption: "Search...",
5. Find: "Find",
6. Reset: "Reset",
7. odata : ['equal', 'not equal', 'less', 'less or equal','greater','greater or equal', 'begins with','does not begin with','is in','is not in','ends with','does not end with','contains','does not contain'],
8. groupOps: [ { op: "AND", text: "all" }, { op: "OR", text: "any" } ],
9. matchText: " match",
10. rulesText: " rules"
11. },
12. edit : {
13. addCaption: "Add Record",
14. editCaption: "Edit Record",
15. bSubmit: "Submit",
16. bCancel: "Cancel",
17. bClose: "Close",
18. saveData: "Data has been changed! Save changes?",
19. bYes : "Yes",
20. bNo : "No",
21. bExit : "Cancel",
22. },
23. view : {
24. caption: "View Record",
25. bClose: "Close"
26. },
27. del : {
28. caption: "Delete",
29. msg: "Delete selected record(s)?",
30. bSubmit: "Delete",
31. bCancel: "Cancel"
32. },
33. nav : {
34. edittext: "",
35. edittitle: "Edit selected row",
36. addtext:"",
37. addtitle: "Add new row",
38. deltext: "",
39. deltitle: "Delete selected row",
40. searchtext: "",
41. searchtitle: "Find records",
42. refreshtext: "",
43. refreshtitle: "Reload Grid",
44. alertcap: "Warning",
45. alerttext: "Please, select row",
46. viewtext: "",
47. viewtitle: "View selected row"
48. },
49. ...





**3. 自定义按钮**

jqGrid还允许将自定义的按钮加入Page Bar，具体API是：

$("#grid\_id").navGrid("#pager",...).navButtonAdd("#pager",{parameters});

或者

$("#grid\_id").jqGrid('navGrid',"#pager",...).jqGrid('navButtonAdd',"#pager",{parameters});

关于navButtonAdd的属性：

* **caption** ：按钮上的文本，可以是空值；
* **buttonicon** ：按钮上的图标，如果设为“none”，则只显示按钮上的文本；
* **onClickButton** ：当点击按钮时所调用的方法函数，默认为null；
* **position** ：添加新按钮的位置，first或last；默认为last；
* **title** ：新按钮的tooltip
* **cursor** ：当鼠标滑过按钮时的光标样式，默认为pointer；
* **id** ：为按钮设置id。

**4. 自定义分隔符**

$("#grid\_id").navSeparatorAdd('#pager',{separator\_parameters});   
  
相关属性：

* **sepclass** ：分隔符的CSS样式；
* **sepcontent** ：分隔符中的内容；

附：相关代码  
  
javascript部分：

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/gengv/article/details/5720707)

1. $(function(){
2. // 配置jqGrid组件
3. $("#gridTable").jqGrid({
4. url: "jqGrid03.action",
5. datatype: "json",
6. mtype: "GET",
7. height: 350,
8. width: 600,
9. colModel: [
10. {name:"id",index:"id",label:"编码",width:40},
11. {name:"lastName",index:"lastName",label:"姓",width:80,sortable:false},
12. {name:"firstName",index:"firstName",label:"名",width:80,sortable:false},
13. {name:"email",index:"email",label:"电子邮箱",width:160,sortable:false},
14. {name:"telNo",index:"telNo",label:"电话",width:120,sortable:false}
15. ],
16. viewrecords: true,
17. rowNum: 15,
18. rowList: [15,50,100],
19. prmNames: {search: "search"},
20. jsonReader: {
21. root:"gridModel",
22. records: "record",
23. repeatitems : false
24. },
25. pager: "#gridPager",
26. caption: "联系人列表",
27. hidegrid: false,
28. shrikToFit: true
29. });
31. var alertText = "<div style="margin:0.3em; background:#FFFFFF; border:1px solid #A6C9E2; padding: 0.5em;" mce\_style="margin:0.3em; background:#FFFFFF; border:1px solid #A6C9E2; padding: 0.5em;">请选择需要操作的数据行!</div>";
33. $("#gridTable").jqGrid("navGrid", "#gridPager", {
34. addfunc : openDialog4Adding,    // (1) 点击添加按钮，则调用openDialog4Adding方法
35. editfunc : openDialog4Updating, // (2) 点击添加按钮，则调用openDialog4Updating方法
36. delfunc : openDialog4Deleting,  // (3) 点击添加按钮，则调用openDialog4Deleting方法
37. alerttext : alertText   // (4) 当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示信息
38. },{},{},{},{    // (5) 修改于查询相关的prmSearch参数
39. caption: "查找",
40. Find: "Let's go!",
41. multipleSearch: true,
42. groupOps: [{ op: "AND", text: "全部" }],
43. },{});
45. // 配置对话框
46. $("#consoleDlg").dialog({
47. autoOpen: false,
48. modal: true,    // 设置对话框为模态（modal）对话框
49. resizable: true,
50. width: 480,
51. buttons: {  // 为对话框添加按钮
52. "取消": function() {$("#consoleDlg").dialog("close")},
53. "创建": addContact,
54. "保存": updateContact,
55. "删除": deleteContact
56. }
57. });
58. });
59. var openDialog4Adding = function() {
60. var consoleDlg = $("#consoleDlg");
61. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
62. consoleDlg.find("input").removeAttr("disabled").val("");
63. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
64. dialogButtonPanel.find("button:contains('创建')").show();
65. consoleDlg.dialog("option", "title", "创建新联系人").dialog("open");
66. };
67. var openDialog4Updating = function(rowid) { // (6) 接受选中行的id为参数
68. var consoleDlg = $("#consoleDlg");
69. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
71. consoleDlg.find("input").removeAttr("disabled");
72. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
73. dialogButtonPanel.find("button:contains('保存')").show();
74. consoleDlg.dialog("option", "title", "修改联系人信息");
76. loadSelectedRowData(rowid); // (7) 将选中行id传入loadSelectedRowData方法
77. }
78. var openDialog4Deleting = function(rowid) { // (8) 接受选中行的id为参数
79. var consoleDlg = $("#consoleDlg");
80. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
82. consoleDlg.find("input").attr("disabled", true);
83. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
84. dialogButtonPanel.find("button:contains('删除')").show();
85. consoleDlg.dialog("option", "title", "删除联系人");
87. loadSelectedRowData(rowid);
88. }
89. var loadSelectedRowData = function(selectedRowId) { // (9) 接受选中行的id为参数
90. var params = {
91. "contact.id" : selectedRowId
92. };
93. var actionUrl = "viewContact.action";
94. // 从Server读取对应ID的JSON数据
95. $.ajax( {
96. url : actionUrl,
97. data : params,
98. dataType : "json",
99. cache : false,
100. error : function(textStatus, errorThrown) {
101. alert("系统ajax交互错误: " + textStatus);
102. },
103. success : function(data, textStatus) {
104. // 如果读取结果成功，则将信息载入到对话框中
105. var rowData = data.contact;
106. var consoleDlg = $("#consoleDlg");
107. consoleDlg.find("#selectId").val(rowData.id);
108. consoleDlg.find("#lastName").val(rowData.lastName);
109. consoleDlg.find("#firstName").val(rowData.firstName);
110. consoleDlg.find("#certificateNo").val(rowData.certificateNo);
111. consoleDlg.find("#email").val(rowData.email);
112. consoleDlg.find("#telNo").val(rowData.telNo);
113. consoleDlg.find("#address").val(rowData.address);
114. consoleDlg.find("#idCardNo").val(rowData.idCardNo);
115. consoleDlg.find("#nationality").val(rowData.nationality);
117. // 根据新载入的数据将表格中的对应数据行一并刷新一下
118. var dataRow = {
119. id : rowData.id,
120. lastName : rowData.lastName,
121. firstName : rowData.firstName,
122. email : rowData.email,
123. telNo : rowData.telNo
124. };
126. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow);
128. // 打开对话框
129. consoleDlg.dialog("open");
130. }
131. });
132. };
133. var addContact = function() {
134. var consoleDlg = $("#consoleDlg");
136. var pLastName = $.trim(consoleDlg.find("#lastName").val());
137. var pFirstName = $.trim(consoleDlg.find("#firstName").val());
138. var pEmail = $.trim(consoleDlg.find("#email").val());
139. var pTelNo = $.trim(consoleDlg.find("#telNo").val());
140. var pAddress = $.trim(consoleDlg.find("#address").val());
141. var pIdCardNo = $.trim(consoleDlg.find("#idCardNo").val());
142. var pNationality = $.trim(consoleDlg.find("#nationality").val());
144. var params = {
145. "contact.lastName" : pLastName,
146. "contact.firstName" : pFirstName,
147. "contact.email" : pEmail,
148. "contact.telNo" : pTelNo,
149. "contact.address" : pAddress,
150. "contact.idCardNo" : pIdCardNo,
151. "contact.nationality" : pNationality
152. };
154. var actionUrl = "createContact.action"
156. $.ajax( {
157. url : actionUrl,
158. data : params,
159. dataType : "json",
160. cache : false,
161. error : function(textStatus, errorThrown) {
162. alert("系统ajax交互错误: " + textStatus);
163. },
164. success : function(data, textStatus) {
165. if(data.ajaxResult == "success") {
166. var dataRow = {
167. id : data.contact.id,   // 从Server端得到系统分配的id
168. lastName : pLastName,
169. firstName : pFirstName,
170. email : pEmail,
171. telNo : pTelNo
172. };
174. var srcrowid = $("#gridTable").jqGrid("getGridParam", "selrow");
176. if(srcrowid) {
177. $("#gridTable").jqGrid("addRowData", data.contact.id, dataRow, "before", srcrowid);
178. } else {
179. $("#gridTable").jqGrid("addRowData", data.contact.id, dataRow, "first");
180. }
181. consoleDlg.dialog("close");
183. alert("联系人添加操作成功!");
185. } else {
186. alert("添加操作失败!");
187. }
188. }
189. });
190. };
191. var updateContact = function() {
192. var consoleDlg = $("#consoleDlg");
194. var pId = $.trim(consoleDlg.find("#selectId").val());
195. var pLastName = $.trim(consoleDlg.find("#lastName").val());
196. var pFirstName = $.trim(consoleDlg.find("#firstName").val());
197. var pEmail = $.trim(consoleDlg.find("#email").val());
198. var pTelNo = $.trim(consoleDlg.find("#telNo").val());
199. var pAddress = $.trim(consoleDlg.find("#address").val());
200. var pIdCardNo = $.trim(consoleDlg.find("#idCardNo").val());
201. var pNationality = $.trim(consoleDlg.find("#nationality").val());
202. var params = {
203. "contact.id" : pId,
204. "contact.lastName" : pLastName,
205. "contact.firstName" : pFirstName,
206. "contact.email" : pEmail,
207. "contact.telNo" : pTelNo,
208. "contact.address" : pAddress,
209. "contact.idCardNo" : pIdCardNo,
210. "contact.nationality" : pNationality
211. };
212. var actionUrl = "updateContact.action";
213. $.ajax( {
214. url : actionUrl,
215. data : params,
216. dataType : "json",
217. cache : false,
218. error : function(textStatus, errorThrown) {
219. alert("系统ajax交互错误: " + textStatus);
220. },
221. success : function(data, textStatus) {
222. if (data.ajaxResult == "success") {
223. var dataRow = {
224. id : data.contact.id,
225. lastName : pLastName,
226. firstName : pFirstName,
227. email : pEmail,
228. telNo : pTelNo
229. };
230. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow, {color:"#FF0000"});
232. alert("联系人信息更新成功!");
234. consoleDlg.dialog("close");
236. } else {
237. alert("修改操作失败!");
238. }
239. }
240. });
241. };
242. var deleteContact = function() {
243. var consoleDlg = $("#consoleDlg");
245. var pId = $.trim(consoleDlg.find("#selectId").val());
246. var params = {
247. "contact.id" : pId
248. };
249. var actionUrl = "deleteContact.action";
250. $.ajax({
251. url : actionUrl,
252. data : params,
253. dataType : "json",
254. cache : false,
255. error : function(textStatus, errorThrown) {
256. alert("系统ajax交互错误: " + textStatus);
257. },
258. success : function(data, textStatus) {
259. if (data.ajaxResult == "success") {
260. $("#gridTable").jqGrid("delRowData", pId);
262. consoleDlg.dialog("close");
263. alert("联系人删除成功!");
264. } else {
265. alert("删除操作失败!");
266. }
267. }
268. });
269. };

**[javascript]** [view plaincopyprint?](http://blog.csdn.net/johnstrive/article/details/42014425)

1. $(function(){
2. // 配置jqGrid组件
3. $("#gridTable").jqGrid({
4. url: "jqGrid03.action",
5. datatype: "json",
6. mtype: "GET",
7. height: 350,
8. width: 600,
9. colModel: [
10. {name:"id",index:"id",label:"编码",width:40},
11. {name:"lastName",index:"lastName",label:"姓",width:80,sortable:false},
12. {name:"firstName",index:"firstName",label:"名",width:80,sortable:false},
13. {name:"email",index:"email",label:"电子邮箱",width:160,sortable:false},
14. {name:"telNo",index:"telNo",label:"电话",width:120,sortable:false}
15. ],
16. viewrecords: true,
17. rowNum: 15,
18. rowList: [15,50,100],
19. prmNames: {search: "search"},
20. jsonReader: {
21. root:"gridModel",
22. records: "record",
23. repeatitems : false
24. },
25. pager: "#gridPager",
26. caption: "联系人列表",
27. hidegrid: false,
28. shrikToFit: true
29. });
31. var alertText = "<div style="margin:0.3em; background:#FFFFFF; border:1px solid #A6C9E2; padding: 0.5em;" mce\_style="margin:0.3em; background:#FFFFFF; border:1px solid #A6C9E2; padding: 0.5em;">请选择需要操作的数据行!</div>";
33. $("#gridTable").jqGrid("navGrid", "#gridPager", {
34. addfunc : openDialog4Adding,    // (1) 点击添加按钮，则调用openDialog4Adding方法
35. editfunc : openDialog4Updating, // (2) 点击添加按钮，则调用openDialog4Updating方法
36. delfunc : openDialog4Deleting,  // (3) 点击添加按钮，则调用openDialog4Deleting方法
37. alerttext : alertText   // (4) 当未选中任何行而点击编辑、删除、查看按钮时，弹出的提示信息
38. },{},{},{},{    // (5) 修改于查询相关的prmSearch参数
39. caption: "查找",
40. Find: "Let's go!",
41. multipleSearch: true,
42. groupOps: [{ op: "AND", text: "全部" }],
43. },{});
45. // 配置对话框
46. $("#consoleDlg").dialog({
47. autoOpen: false,
48. modal: true,    // 设置对话框为模态（modal）对话框
49. resizable: true,
50. width: 480,
51. buttons: {  // 为对话框添加按钮
52. "取消": function() {$("#consoleDlg").dialog("close")},
53. "创建": addContact,
54. "保存": updateContact,
55. "删除": deleteContact
56. }
57. });
58. });
59. var openDialog4Adding = function() {
60. var consoleDlg = $("#consoleDlg");
61. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
62. consoleDlg.find("input").removeAttr("disabled").val("");
63. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
64. dialogButtonPanel.find("button:contains('创建')").show();
65. consoleDlg.dialog("option", "title", "创建新联系人").dialog("open");
66. };
67. var openDialog4Updating = function(rowid) { // (6) 接受选中行的id为参数
68. var consoleDlg = $("#consoleDlg");
69. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
71. consoleDlg.find("input").removeAttr("disabled");
72. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
73. dialogButtonPanel.find("button:contains('保存')").show();
74. consoleDlg.dialog("option", "title", "修改联系人信息");
76. loadSelectedRowData(rowid); // (7) 将选中行id传入loadSelectedRowData方法
77. }
78. var openDialog4Deleting = function(rowid) { // (8) 接受选中行的id为参数
79. var consoleDlg = $("#consoleDlg");
80. var dialogButtonPanel = consoleDlg.siblings(".ui-dialog-buttonpane");
82. consoleDlg.find("input").attr("disabled", true);
83. dialogButtonPanel.find("button:not(:contains('取消'))").hide();
84. dialogButtonPanel.find("button:contains('删除')").show();
85. consoleDlg.dialog("option", "title", "删除联系人");
87. loadSelectedRowData(rowid);
88. }
89. var loadSelectedRowData = function(selectedRowId) { // (9) 接受选中行的id为参数
90. var params = {
91. "contact.id" : selectedRowId
92. };
93. var actionUrl = "viewContact.action";
94. // 从Server读取对应ID的JSON数据
95. $.ajax( {
96. url : actionUrl,
97. data : params,
98. dataType : "json",
99. cache : false,
100. error : function(textStatus, errorThrown) {
101. alert("系统ajax交互错误: " + textStatus);
102. },
103. success : function(data, textStatus) {
104. // 如果读取结果成功，则将信息载入到对话框中
105. var rowData = data.contact;
106. var consoleDlg = $("#consoleDlg");
107. consoleDlg.find("#selectId").val(rowData.id);
108. consoleDlg.find("#lastName").val(rowData.lastName);
109. consoleDlg.find("#firstName").val(rowData.firstName);
110. consoleDlg.find("#certificateNo").val(rowData.certificateNo);
111. consoleDlg.find("#email").val(rowData.email);
112. consoleDlg.find("#telNo").val(rowData.telNo);
113. consoleDlg.find("#address").val(rowData.address);
114. consoleDlg.find("#idCardNo").val(rowData.idCardNo);
115. consoleDlg.find("#nationality").val(rowData.nationality);
117. // 根据新载入的数据将表格中的对应数据行一并刷新一下
118. var dataRow = {
119. id : rowData.id,
120. lastName : rowData.lastName,
121. firstName : rowData.firstName,
122. email : rowData.email,
123. telNo : rowData.telNo
124. };
126. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow);
128. // 打开对话框
129. consoleDlg.dialog("open");
130. }
131. });
132. };
133. var addContact = function() {
134. var consoleDlg = $("#consoleDlg");
136. var pLastName = $.trim(consoleDlg.find("#lastName").val());
137. var pFirstName = $.trim(consoleDlg.find("#firstName").val());
138. var pEmail = $.trim(consoleDlg.find("#email").val());
139. var pTelNo = $.trim(consoleDlg.find("#telNo").val());
140. var pAddress = $.trim(consoleDlg.find("#address").val());
141. var pIdCardNo = $.trim(consoleDlg.find("#idCardNo").val());
142. var pNationality = $.trim(consoleDlg.find("#nationality").val());
144. var params = {
145. "contact.lastName" : pLastName,
146. "contact.firstName" : pFirstName,
147. "contact.email" : pEmail,
148. "contact.telNo" : pTelNo,
149. "contact.address" : pAddress,
150. "contact.idCardNo" : pIdCardNo,
151. "contact.nationality" : pNationality
152. };
154. var actionUrl = "createContact.action"
156. $.ajax( {
157. url : actionUrl,
158. data : params,
159. dataType : "json",
160. cache : false,
161. error : function(textStatus, errorThrown) {
162. alert("系统ajax交互错误: " + textStatus);
163. },
164. success : function(data, textStatus) {
165. if(data.ajaxResult == "success") {
166. var dataRow = {
167. id : data.contact.id,   // 从Server端得到系统分配的id
168. lastName : pLastName,
169. firstName : pFirstName,
170. email : pEmail,
171. telNo : pTelNo
172. };
174. var srcrowid = $("#gridTable").jqGrid("getGridParam", "selrow");
176. if(srcrowid) {
177. $("#gridTable").jqGrid("addRowData", data.contact.id, dataRow, "before", srcrowid);
178. } else {
179. $("#gridTable").jqGrid("addRowData", data.contact.id, dataRow, "first");
180. }
181. consoleDlg.dialog("close");
183. alert("联系人添加操作成功!");
185. } else {
186. alert("添加操作失败!");
187. }
188. }
189. });
190. };
191. var updateContact = function() {
192. var consoleDlg = $("#consoleDlg");
194. var pId = $.trim(consoleDlg.find("#selectId").val());
195. var pLastName = $.trim(consoleDlg.find("#lastName").val());
196. var pFirstName = $.trim(consoleDlg.find("#firstName").val());
197. var pEmail = $.trim(consoleDlg.find("#email").val());
198. var pTelNo = $.trim(consoleDlg.find("#telNo").val());
199. var pAddress = $.trim(consoleDlg.find("#address").val());
200. var pIdCardNo = $.trim(consoleDlg.find("#idCardNo").val());
201. var pNationality = $.trim(consoleDlg.find("#nationality").val());
202. var params = {
203. "contact.id" : pId,
204. "contact.lastName" : pLastName,
205. "contact.firstName" : pFirstName,
206. "contact.email" : pEmail,
207. "contact.telNo" : pTelNo,
208. "contact.address" : pAddress,
209. "contact.idCardNo" : pIdCardNo,
210. "contact.nationality" : pNationality
211. };
212. var actionUrl = "updateContact.action";
213. $.ajax( {
214. url : actionUrl,
215. data : params,
216. dataType : "json",
217. cache : false,
218. error : function(textStatus, errorThrown) {
219. alert("系统ajax交互错误: " + textStatus);
220. },
221. success : function(data, textStatus) {
222. if (data.ajaxResult == "success") {
223. var dataRow = {
224. id : data.contact.id,
225. lastName : pLastName,
226. firstName : pFirstName,
227. email : pEmail,
228. telNo : pTelNo
229. };
230. $("#gridTable").jqGrid("setRowData", data.contact.id, dataRow, {color:"#FF0000"});
232. alert("联系人信息更新成功!");
234. consoleDlg.dialog("close");
236. } else {
237. alert("修改操作失败!");
238. }
239. }
240. });
241. };
242. var deleteContact = function() {
243. var consoleDlg = $("#consoleDlg");
245. var pId = $.trim(consoleDlg.find("#selectId").val());
246. var params = {
247. "contact.id" : pId
248. };
249. var actionUrl = "deleteContact.action";
250. $.ajax({
251. url : actionUrl,
252. data : params,
253. dataType : "json",
254. cache : false,
255. error : function(textStatus, errorThrown) {
256. alert("系统ajax交互错误: " + textStatus);
257. },
258. success : function(data, textStatus) {
259. if (data.ajaxResult == "success") {
260. $("#gridTable").jqGrid("delRowData", pId);
262. consoleDlg.dialog("close");
263. alert("联系人删除成功!");
264. } else {
265. alert("删除操作失败!");
266. }
267. }
268. });
269. };



# [jqGrid 与Struts2的结合应用（二） —— 操作Grid数据](http://blog.csdn.net/tecie/article/details/5773108)

2010-07-29 10:08 637人阅读 [评论](http://blog.csdn.net/tecie/article/details/5773108#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/tecie/article/details/5773108#report)

[struts](http://www.csdn.net/tag/struts)[function](http://www.csdn.net/tag/function)[email](http://www.csdn.net/tag/email)[stylesheet](http://www.csdn.net/tag/stylesheet)[search](http://www.csdn.net/tag/search)[methods](http://www.csdn.net/tag/methods)

上一篇中，jqGrid已经可以从Server端获得数据，并显示在Grid表格中了。下面说一下，如何操作Grid表格及其数据。

jqGrid有很多方法函数，用来操作数据或者操作Grid表格本身。jqGrid的方法有两种调用方式：

$("#grid\_id").jqGridMethod( parameter1,...,parameterN );

或者

$("#grid\_id").jqGrid('method', parameter1,...,parameterN );

首先介绍一下jqGrid的几个最常用的方法函数，具体的方法API也可以参考官方文档（<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:methods> ）

1. getGridParam

这个方法用来获得jqGrid的选项值。它具有一个可选参数name，name即代表着jqGrid的选项名，例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  id = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var id = $("#gridTable").jqGrid("getGridParam",

"selrow");

  
即可获得当前选中的行的ID。

注：selrow 是jqGrid选项之一，默认值是null。这是一个只读选项，代表最后选中行的ID。如果执行翻页或者排序后，此选项将被设为null。关于其他选项， 后续会有介绍。

如果不传入name参数，则会返回jqGrid整个选项options。

2. getRowData

这个方法用来获得某行的数据。它具有一个rowid参数，jqGrid会根据这个rowid返回对应行的数据，返回的是name:value类型的 数组。例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  getContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  rowData = $( "#gridTable" ).jqGrid( "getRowData" , selectedId);

    alert("First Name: "  + rowData.firstName);

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var getContact

 = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var rowData = $("#gridTable").jqGrid("getRowData", selectedId);

    alert("First Name: " + rowData.firstName);

};



如果rowid未能被找到，则返回一个空数组；如果未设置rowid参数，则以数组的形式返回Grid的所有行数据。

3. addRowData

这个方法用于向Grid中插入新的一行。执行成功返回true，否则返回false。它具有4个参数：

rowid ：新行的id号；

data ：新行的数据对象，形式为{name1:value1,name2: value2…}，其中name为colModel中定义的列名称name；

position ：插入的位置（first：表格顶端；last：表格底端；before：srcrowid之前；after：srcrowid之后）；

srcrowid ：新行将插入到srcrowid指定行的前面或后面。

例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  addContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  dataRow = {

        id : 99,

        lastName : "Zhang" ,

        firstName : "San" ,

        email : "zhang\_san@126.com" ,

        telNo : "0086-12345678"

    };

    if  (selectedId) {

        $("#gridTable" ).jqGrid( "addRowData" , 99, dataRow,  "before" , selectedId);

    } else  {

        $("#gridTable" ).jqGrid( "addRowData" , 99, dataRow,  "first" );

    }

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var addContact

 = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var dataRow = {

        id : 99,

        lastName : "Zhang",

        firstName : "San",

        email : "zhang\_san@126.com",

        telNo : "0086-12345678"

    };

    if (selectedId) {

        $("#gridTable").jqGrid("addRowData", 99, dataRow, "before",

selectedId);

    } else {

        $("#gridTable").jqGrid("addRowData", 99, dataRow, "first");

    }

};



这个方法可以一次性插入多行，data参数必须是[{name1:value1,name2: value2…}, {name1:value1,name2: value2…}]这样的数组形式，而且rowid参数也应该设为data参数对象中代表id的field名称。不过，此时的rowid不用必须是 colModel中的一部分。  
例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  addContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  dataRow = [{

        id : 99,

        lastName : "Zhang" ,

        firstName : "San" ,

        email : "zhang\_san@126.com" ,

        telNo : "0086-12345678"

    },

    {

        id : 100,

        lastName : "Li" ,

        firstName : "Si" ,

        email : "li\_si@126.com" ,

        telNo : "0086-12345678"

    },

    {

        id : 101,

        lastName : "Wang" ,

        firstName : "Wu" ,

        email : "wang\_wu@126.com" ,

        telNo : "0086-12345678"

    }];

    if  (selectedId) {

        $("#gridTable" ).jqGrid( "addRowData" ,  "id" , dataRow,  "before" , selectedId);

    } else  {

        $("#gridTable" ).jqGrid( "addRowData" ,  "id" , dataRow,  "first" );

    }

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var addContact

 = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var dataRow = [{

        id : 99,

        lastName : "Zhang",

        firstName : "San",

        email : "zhang\_san@126.com",

        telNo : "0086-12345678"

    },

    {

        id : 100,

        lastName : "Li",

        firstName : "Si",

        email : "li\_si@126.com",

        telNo : "0086-12345678"

    },

    {

        id : 101,

        lastName : "Wang",

        firstName : "Wu",

        email : "wang\_wu@126.com",

        telNo : "0086-12345678"

    }];

    if (selectedId) {

        $("#gridTable").jqGrid("addRowData", "id", dataRow, "before",

selectedId);

    } else {

        $("#gridTable").jqGrid("addRowData", "id", dataRow, "first");

    }

};



注： 我测试了一下，一次插入多行的情况下，用于设置插入位置的后两个参数，似乎没有起作用。插入的几行数据都被置于Grid的底端。

4. setRowData

这个方法用于为某行数据设置数据值。执行成功返回true，否则返回false。它具有3个参数：

rowid ：更新数据的行id；

data ：更新的数据对象，形式为{name1:value1,name2: value2…}，其中name为colModel中定义的列名称name；这个数据对象，不必设置完全，需要更新哪列，就设置哪列的 name:value对；

cssprop ：如果cssprop为String类型，则会使用jQuery的addClass为行增加相应名称的CSS类；如果为object类型，则会使用 html的css属性，为行添加样式。如果只想增加css样式而不更新数据，可以将data参数设为false。

例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  updateContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  dataRow = {

        lastName : "Li" ,

        firstName : "Si" ,

        email : "li\_si@126.com"

    };

    var  cssprop = {

        color : "#FF0000"

    };

    $("#gridTable" ).jqGrid( 'setRowData' , selectedId, dataRow, cssprop);

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var

updateContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var dataRow = {

        lastName : "Li",

        firstName : "Si",

        email : "li\_si@126.com"

    };

    var cssprop = {

        color : "#FF0000"

    };

    $("#gridTable").jqGrid('setRowData', selectedId, dataRow, cssprop);

};



5. delRowData

这个方法用于删除某行数据。执行成功返回true，否则返回false。具有一个参数rowid，代表要删除的行id。例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  deleteContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    $("#gridTable" ).jqGrid( 'delRowData' , selectedId);

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var

deleteContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    $("#gridTable").jqGrid('delRowData', selectedId);

};



6. setGridParam

这个方法与getGridParam对应，用于设置jqGrid的options选项。返回jqGrid对象。参数为 {name1:value1,name2: value2…}形式的对象（name来自jqGrid的options选项名）。某些选项在设置之后需要trigger("reloadGrid")， 才能显示出效果。  
  
例如：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

var  changeGridOptions =  function () {

    $("#gridTable" ).jqGrid( "setGridParam" , {

        rowNum: 50,

        page: 16

    }).trigger('reloadGrid' );

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

var

changeGridOptions = function() {

    $("#gridTable").jqGrid("setGridParam", {

        rowNum: 50,

        page: 16

    }).trigger('reloadGrid');

};



注： 我测试了一下，这个方法对于设置jqGrid的caption选项似乎无效。大概因此有了setCaption方法。

7. setGridWidth

为Grid动态地设定一个新的宽度。两个参数：

new\_width ：以px为单位的新宽度值；

shrink ：作用与jqGrid的shrinkToFit选项相同；如果此参数未设置，则沿用jqGrid的shrinkToFit选项的值。

8. trigger("reloadGrid")

根据当前设置，重新载入Grid表格，即意味着向Server发送一个新的请求。此方法只能用于已经构建好的Grid。此外，此方法不会使对 colModel所做出的改变生效。应该使用gridUnload来重新载入对colModel的新设置。

9. 其他方法

除了以上介绍的的方法外，jqGrid还有其他有用的方法，例如：  
addJSONData、clearGridData、hideCol、resetSelection、setCaption、 setGridHeight、setLabel、showCol等  
以及增强模块提供的方法，例如：  
filterGrid、GridDestroy、GridUnload、setColProp等。  
  
这些方法的具体用法，或浅显易懂，或不是非常常用。都可以参考官方文档（<http://www.trirand.com/jqgridwiki/doku.php?id=wiki:methods> ），得到具体指示。

10. 额外的考虑

在上面介绍的增删改数据行的操作中，jqGrid实际上完成的只是客户端的操作，主要是DOM的更改工作。但是如果传到Server的请求失败了， 或者没有得到想要的返回结果，那jqGrid的处理还继续吗？会不会和Server端的数据不同步了呢？  
这个问题在下一篇中再讨论吧。

注： 本篇中介绍的方法在jqGrid的Demo中也有介绍，但是我觉得还是自己的实例看起来更直观，更适合自己理解。  
  
附上代码：

javascript部分：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

$( function (){

    // 配置jqGrid组件

    $("#gridTable" ).jqGrid({

        url: "jqGrid01.action" ,

        datatype: "json" ,

        mtype: "GET" ,

        height: 350,

        width: 600,

        colModel: [

              {name:"id" ,index: "id" ,label: "编码" ,width:40},

              {name:"lastName" ,index: "lastName" ,label: "姓" ,width:80,sortable: false },

              {name:"firstName" ,index: "firstName" ,label: "名" ,width:80,sortable: false },

              {name:"email" ,index: "email" ,label: "电子邮箱" ,width:160,sortable: false },

              {name:"telNo" ,index: "telNo" ,label: "电话" ,width:120,sortable: false }

        ],

        viewrecords: true ,

        rowNum: 15,

        rowList: [15,50,100],

        prmNames: {search: "search" },

        jsonReader: {

            root:"gridModel" ,

            records: "record" ,

            repeatitems : false

        },

        pager: "#gridPager" ,

        caption: "联系人列表" ,

        hidegrid: false ,

        shrikToFit: true

    });

});

var  echoSelRow =  function () {

    var  id = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    alert("当前选中行ID："  + id);

};

var  getContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  rowData = $( "#gridTable" ).jqGrid( "getRowData" , selectedId);

    alert("First Name: "  + rowData.firstName);

};

var  addContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  dataRow = {

        id : 99,

        lastName : "Zhang" ,

        firstName : "San" ,

        email : "zhang\_san@126.com" ,

        telNo : "0086-12345678"

    };

    if  (selectedId) {

        $("#gridTable" ).jqGrid( "addRowData" , 99, dataRow,  "before" , selectedId);

    } else  {

        $("#gridTable" ).jqGrid( "addRowData" , 99, dataRow,  "first" );

    }

};

var  updateContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    var  dataRow = {

        lastName : "Li" ,

        firstName : "Si" ,

        email : "li\_si@126.com"

    };

    var  cssprop = {

        color : "#FF0000"

    };

    $("#gridTable" ).jqGrid( 'setRowData' , selectedId, dataRow, cssprop);

};

var  deleteContact =  function () {

    var  selectedId = $( "#gridTable" ).jqGrid( "getGridParam" ,  "selrow" );

    $("#gridTable" ).jqGrid( 'delRowData' , selectedId);

};

var  changeGridOptions =  function () {

    $("#gridTable" ).jqGrid( "setGridParam" , {

        rowNum: 50,

        page: 16

    }).trigger('reloadGrid' );

    $("#gridTable" ).jqGrid( "setCaption" ,  "Contact List" ).trigger( 'reloadGrid' );

    alert($("#gridTable" ).jqGrid( "getGridParam" ,  "caption" ));

    alert($("#gridTable" ).jqGrid( "getGridParam" ,  "rowNum" ));

};

var  resetWidth =  function () {

    $("#gridTable" ).jqGrid( "setGridWidth" , 300,  false );

};

[javascript] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

$(function(){

    // 配置jqGrid组件

    $("#gridTable").jqGrid({

        url: "jqGrid01.action",

        datatype: "json",

        mtype: "GET",

        height: 350,

        width: 600,

        colModel: [

              {name:"id",index:"id",label:"编码",width:40},

{name:"lastName",index:"lastName",label:"姓",width:80,sortable:false},

{name:"firstName",index:"firstName",label:"名",width:80,sortable:false},

{name:"email",index:"email",label:"电子邮箱",width:160,sortable:false},

              {name:"telNo",index:"telNo",label:"电话",width:120,sortable:false}

        ],

        viewrecords: true,

        rowNum: 15,

        rowList: [15,50,100],

        prmNames: {search: "search"},

        jsonReader: {

            root:"gridModel",

            records: "record",

            repeatitems : false

        },

        pager: "#gridPager",

        caption: "联系人列表",

        hidegrid: false,

        shrikToFit: true

    });

});

var echoSelRow = function() {

    var id = $("#gridTable").jqGrid("getGridParam", "selrow");

    alert("当前选中行ID：" + id);

};

var getContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var rowData = $("#gridTable").jqGrid("getRowData", selectedId);

    alert("First Name: " + rowData.firstName);

};

var addContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var dataRow = {

        id : 99,

        lastName : "Zhang",

        firstName : "San",

        email : "zhang\_san@126.com",

        telNo : "0086-12345678"

    };

    if (selectedId) {

        $("#gridTable").jqGrid("addRowData", 99, dataRow, "before",

selectedId);

    } else {

        $("#gridTable").jqGrid("addRowData", 99, dataRow, "first");

    }

};

var updateContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    var dataRow = {

        lastName : "Li",

        firstName : "Si",

        email : "li\_si@126.com"

    };

    var cssprop = {

        color : "#FF0000"

    };

    $("#gridTable").jqGrid('setRowData', selectedId, dataRow, cssprop);

};

var deleteContact = function() {

    var selectedId = $("#gridTable").jqGrid("getGridParam", "selrow");

    $("#gridTable").jqGrid('delRowData', selectedId);

};

var changeGridOptions = function() {

    $("#gridTable").jqGrid("setGridParam", {

        rowNum: 50,

        page: 16

    }).trigger('reloadGrid');

    $("#gridTable").jqGrid("setCaption", "Contact

List").trigger('reloadGrid');

    alert($("#gridTable").jqGrid("getGridParam", "caption"));

    alert($("#gridTable").jqGrid("getGridParam", "rowNum"));

};

var resetWidth = function() {

    $("#gridTable").jqGrid("setGridWidth", 300, false);

};



html部分：

[view plain](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [copy to clipboard](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [print](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx) [?](http://blog.csdn.net/gengv/archive/2010/07/06/5716347.aspx)

<!DOCTYPE html

PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

< html   xmlns = "http://www.w3.org/1999/xhtml"   xml:lang = "zh-CN"   lang = "zh-CN" >

    < head >

        < title > jqGrid02 </ title >

        < link   rel = "stylesheet"   type = "text/css"   media = "screen"   href = "../css/themes/aero/jquery-ui-1.8.1.custom.css"   mce\_href = "css/themes/aero/jquery-ui-1.8.1.custom.css"   />

        < link   rel = "stylesheet"   type = "text/css"   media = "screen"   href = "../css/themes/aero/ui.jqgrid.custom.css"   mce\_href = "css/themes/aero/ui.jqgrid.custom.css"   />

        < mce:script   type = "text/javascript"   src = "../js/jquery-1.4.2.min.js"   mce\_src = "js/jquery-1.4.2.min.js" > </ mce:script >

        < mce:script   src = "../js/jquery-ui-1.8.1.custom.min.js"   mce\_src = "js/jquery-ui-1.8.1.custom.min.js"   type = "text/javascript" > </ mce:script >

        < mce:script   src = "../js/i18n/grid.locale-zh-CN.js"   mce\_src = "js/i18n/grid.locale-zh-CN.js"   type = "text/javascript" > </ mce:script >

        < mce:script   src = "../js/jquery.jqGrid.min.js"   mce\_src = "js/jquery.jqGrid.min.js"   type = "text/javascript" > </ mce:script >

        < mce:script   src = "js/jqGrid02.js"   mce\_src = "js/jqGrid02.js"   type = "text/javascript" > </ mce:script >

    </ head >

    < body >

        < h3 >

            jqGrid 测试 02

        </ h3 >

        < div >

            < table   id = "gridTable" > </ table >

            < div   id = "gridPager" > </ div >

        </ div >

        < br   />

        < div >

            < button   onclick = "echoSelRow()" > 当 前行ID </ button >

            < button   onclick = "getContact()" > 当 前联系人 </ button >

            < button   onclick = "addContact()" > 添 加行 </ button >

            < button   onclick = "updateContact()" > 修 改行 </ button >

            < button   onclick = "deleteContact()" > 删 除行 </ button >

            < button   onclick = "changeGridOptions()" > 改 变Grid选项 </ button >

            < button   onclick = "resetWidth()" > 改 变Grid宽度 </ button >

        </ div >

    </ body >

</ html >

[xhtml] [view plaincopyprint?](http://blog.csdn.net/tecie/article/details/5773108)

<!DOCTYPE

 html

PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"

"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="zh-CN"

lang="zh-CN">

    <head>

        <title>jqGrid02</title>

        <link rel="stylesheet" type="text/css" media="screen"

href="../css/themes/aero/jquery-ui-1.8.1.custom.css"

mce\_href="css/themes/aero/jquery-ui-1.8.1.custom.css" />

        <link rel="stylesheet" type="text/css" media="screen"

href="../css/themes/aero/ui.jqgrid.custom.css"

mce\_href="css/themes/aero/ui.jqgrid.custom.css" />

        <mce:script type="text/javascript" src="../js/jquery-1.4.2.min.js"

mce\_src="js/jquery-1.4.2.min.js"></mce:script>

        <mce:script src="../js/jquery-ui-1.8.1.custom.min.js"

mce\_src="js/jquery-ui-1.8.1.custom.min.js"

type="text/javascript"></mce:script>

        <mce:script src="../js/i18n/grid.locale-zh-CN.js"

mce\_src="js/i18n/grid.locale-zh-CN.js"

type="text/javascript"></mce:script>

        <mce:script src="../js/jquery.jqGrid.min.js"

mce\_src="js/jquery.jqGrid.min.js"

type="text/javascript"></mce:script>

        <mce:script src="js/jqGrid02.js" mce\_src="js/jqGrid02.js"

type="text/javascript"></mce:script>

    </head>

    <body>

        <h3>

            jqGrid测试 02

        </h3>

        <div>

            <table id="gridTable"></table>

            <div id="gridPager"></div>

        </div>

        <br />

        <div>

            <button onclick="echoSelRow()">当前行ID</button>

            <button onclick="getContact()">当前联系人</button>

            <button onclick="addContact()">添加行</button>

            <button onclick="updateContact()">修改行</button>

            <button onclick="deleteContact()">删除行</button>

            <button onclick="changeGridOptions()">改变Grid选项</button>

            <button onclick="resetWidth()">改变Grid宽度</button>

        </div>

    </body>

</html>



