easyUI学习

# easyUi应用和简单的测试

## dialog创建方式1

### 引入一些文件

1 引入文件JQuery的js文件

2 引入easyUi的js文件

3 引入默认主题文件themes/default/easyui.css

4 引入图标文件icon.css

5 中文语言显示文件locale/easyui-lang-zh-Ch

### dialog显示方式1

1. <div id="dd" class="easyui-dialog" title="My Dialog" style="width:400px;height:200px;"
2. data-options="iconCls:'icon-save',resizable:true,modal:true">
3. Dialog Content.
4. </div>

浏览器查看效果.

### dialog显示方式2

页面中加入:

<div id="dd">Dialog Content.</div>

写一个javascript加入如下:

属性可以只写部分在javascript中也可以都写在里面

1. $('#dd').dialog({
2. title: 'My Dialog',
3. width: 400,
4. height: 200,
5. closed: **false**,
6. cache: **false**,
7. href: 'get\_content.php',
8. modal: **true**
9. });
10. $('#dd').dialog('refresh', 'new\_content.php');

# easyLoader使用方式

## 引入文件

1 jquery自己的文件

2 easyLoader.js文件

## 如何使用easyLoader加载dialog或者其他组件

Load EasyUI module

1. easyloader.load('messager', **function**(){ // load the specified module
2. $.messager.alert('Title', 'load ok');
3. });

Load script from an absolute url

1. using('http://code.jquery.com/jquery-1.4.4.min.js', **function**(){
2. // ...
3. });

Load script from relative url

1. // the script path relative to easyui directory
2. using('./myscript.js', **function**(){
3. // ...
4. });

## 同时加载两个组件写法

<script type="text/javascript">

$(**function**(){

easyloader.load([ 'dialog' , 'messager' ],**function**(){

$('#dd').dialog({

model:**true**

});

$.messager.alert( 'Title' , 'load ok' );

});

});

</script>

# panel组件

## 加载一个panel组件

1. <div id="p" class="easyui-panel" title="My Panel"
2. style="width:500px;height:150px;padding:10px;background:#fafafa;"
3. data-options="iconCls:'icon-save',closable:true,
4. collapsible:true,minimizable:true,maximizable:true">
5. <p>panel content.</p>
6. <p>panel content.</p>
7. </div>

## 加载销毁关闭方法

## 如何加载样式

<!DOCTYPE html>

<html>

<head>

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

<script type="text/javascript"

src="js/jquery-easyui-1.3/jquery-1.7.2.min.js"></script>

<script type="text/javascript" src="js/jquery-easyui-1.3/jquery.easyui.min.js"></script>

<link rel="stylesheet" href="js/jquery-easyui-1.3/themes/icon.css" type="text/css"></link>

<link rel="stylesheet" href="js/jquery-easyui-1.3/themes/default/easyui.css" type="text/css"></link>

<script type="text/javascript" src="js/jquery-easyui-1.3/locale/easyui-lang-zh\_CN.js"></script>

*<!-- 如何初始化 -->*

<script type="text/javascript">

$(**function**(){

$('#p').panel({

width:300,

height:150,

});

});

</script>

</head>

<body>

<div id="p" class="easyui-panel" title="My Panel"

style="padding:10px;background:#fafafa;"

data-options="iconCls:'icon-save',closable:true,

collapsible:true,minimizable:true,maximizable:true">

<p>panel content.</p>

<p>panel content.</p>

</div>

*<!-- 方法的加载 -->*

<input type="button" onclick="$('#p').panel('close')" value="关闭"/>

<input type="button" onclick="$('#p').panel('open')" value="打开"/>

<input type="button" onclick="$('#p').panel('destroy')" value="销毁"/>

</body>

</html>

# 用户登陆

## 使用dialog实现用户登陆的样式和json请求

## 设置springmvc不拦截某些请求.因为/的配置会拦截除了.jsp外的所有请求

<mvc:resources location=*"/easyUi/"* mapping=*"/easyUi/\*.html"*/>

<mvc:resources location=*"/js/"* mapping=*"/js/\*\*"*/>

<mvc:resources location=*"/easyUi/"* mapping=*"/easyUi/\*.js"*/>

## 新建text.html文件如下

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

// 在程序运行结束hou加载dialog组件

**var** loginAndRegDialog;

$(**function**(){

loginAndRegDialog=$('#loginAndRegDialog').dialog({

closable: **false**,//右上角小差去掉

modal : **true**, //模式化窗口.只有操作后才能进行其他操作

buttons :

[{

text: '注册',

handler:**function**(){

}

},

{

text: '登陆',

handler: **function**(){

//点击登陆进行ajax传参

$.ajax({

url: sy.bp()+ '/easyUiJson',

data: {

name: $('#loginInputForm input[name=name]').val(),

password: $('#loginInputForm input[name=password]').val()

},

dataType:'json',

success: **function**(r){

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

$.messager.alert('登陆成功','欢迎:'+r.username);

}**else**

$.messager.alert('提示','您的输入有误');

}

});

}

}]//多个组件用方括号表示

});

});//在页面加载完后执行

</script>

## myjs.js中写一个获取项目路径的方法

**function** sy(){ }

sy.bp = **function**(){

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

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

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

**var** localhostPath=curWwwpath.substring(0,pos);

**var** projectName = pathName.substring(0,pathName.substr(1).indexOf('/')+1);

**return** (localhostPath + projectName);

};

## Controller中如下配置

@RequestMapping(value="easyUiJson")

@ResponseBody

**public** User login(String name,String password){

System.*out*.println("name="+name);

System.*out*.println("password="+password);

User user=accountService.getUserByUserName(name);

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

**return** user;

}

**return** **null**;

}

查看是否输出 你好,小明

## 补充,json序列化操作传参方法

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

**var** loginAndRegDialog;

$(**function**(){

loginAndRegDialog=$('#loginAndRegDialog').dialog({

closable : **false**,//把右上角叉去掉

modal :**true**,//显示背景灰色

buttons:

[

{text:'注册',

handler:**function**(){

}},

{text:'登陆',

handler:**function**(){

console.info("我点击的是按钮");

$.ajax({

url : sy.bp() + '/easyUiJson',

**data : $('#loginInputForm').serialize()**,

cache:**false**,

dataType:'json',

success:**function**(r){

// console.info(r.username);

$.messager.alert('登陆成功','欢迎:'+r.username);

}

});

}}

]

});

});

</script>

### 使用easyUi中form提交表单方式

# 用户登陆form表单提交

## Controller:

@RequestMapping(value="uiform")

@ResponseBody

**public** User loginForm(String name,String password){

System.*out*.println("name="+name);

System.*out*.println("password="+password);

User user=accountService.getUserByUserName(name);

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

**return** user;

}

**return** **null**;

}

## html中:

<!DOCTYPE html>

<html>

<head>

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

<script type=*"text/javascript"*

src=*"../js/jquery-easyui-1.3/jquery-1.7.2.min.js"*></script>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/jquery.easyui.min.js"*></script>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/icon.css"* type=*"text/css"*></link>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/default/easyui.css"* type=*"text/css"*></link>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/locale/easyui-lang-zh\_CN.js"*></script>

<script type=*"text/javascript"* src=*"../js/myjs.js"*></script>

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

**var** loginAndRegDialog;

$(**function**(){

loginAndRegDialog=$('#loginAndRegDialog').dialog({

closable : **false**,//把右上角叉去掉

modal :**true**,//显示背景灰色

buttons:

[

{text:'注册',

handler:**function**(){

}},

{text:'登陆',

handler:**function**(){

//提交

loginInputForm.submit();

}}

]

});

loginInputForm=$('#loginInputForm').form({

url: sy.bp()+ '/uiform',

onSubmit: **function**(){

console.info($('#loginInputForm input[name=name]').val());

**if**($('#loginInputForm input[name=name]').val()==""){

$.messager.alert("提示","用户名不能为空");

**return false**;

}

**if**($('#loginInputForm input[name=password]').val()==""){

$.messager.alert("提示","密码不能为空");

**return false**;

}

},

success: **function**(data){

console.info(data);

//需要将data类型转化成object类型

**var** obj = jQuery.parseJSON(data);

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

$.messager.alert("登陆成功","欢迎 你:"+obj.username);

}

**if**(obj==**null**){

$.messager.alert("提示","您的输入有错误!");

}

}

});

});

</script>

</head>

<body>

This is my HTML page. <br>

<div id=*"loginAndRegDialog"* title=*"用户登陆"* style="width:*250px*;height:*200px*;">

<form method=*"post"* id=*"loginInputForm"*>

<table>

<tr>

<th align=*"right"*>用户名:</th>

<td><input type=*"text"* name=*"name"* /></td>

</tr>

<tr>

<th align=*"right"*>密码:</th>

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

</tr>

</table>

</form>

</div>

</body>

</html>

## 说明:

1 使用form表单提交需要注意将返回的字符类型转换成Object对象才能使用.

2 上面使用json方式,因为用了dataType:'json'所以已经转换成字符串的类型

3 转换对象推荐使用**var** obj = jQuery.parseJSON(data);这种方式,这样在返回为空的时候也能输出null

4 如果使用eval(“(“data”)”)当返回的对象为空时,eval不能正常处理字符串格式.

# **ValidateBox的使用和自定义**

## 如何添加一个ValidateBox

<tr>

<th align=*"right"*>密码:</th>

<td><input type=*"password"* name=*"password"* class=*"easyui-validatebox"* required=*"true"* /></td>

</tr>

推荐使用class添加的方法required表示必须填写

## 表单中增加对ValidateBox的验证.

在表单提交前方法中判断对ValidateBox的验证

onSubmit: **function**(){

**if**(!$('#loginInputForm').form('validate')){

**return false**;

}

},

## ajax中增加对ValidateBox的验证.

使用 $('#loginInputForm').form('validate'方法

**if**( $('#loginInputForm').form('validate'))

{

$.ajax({

url: sy.bp()+ '/easyUiJson',

data: $('#loginInputForm').serialize(),

dataType:'json',

success: **function**(r){

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

$.messager.alert('登陆成功','欢迎:'+r.username);

}**else**

$.messager.alert('提示','您的输入有误');

}

});

}

## 如何自定义实现重复密码校验

### js中添加一个方法

$.extend($.fn.validatebox.defaults.rules, {

eqPassword: {

validator: **function**(value, param){

**return** value==$(param[0]).val();

},

message: '密码不一致!'

}

});

### 重复密码中自定义一个类型validType

<tr>

<th align=*"right"*>密码:</th>

<td><input type=*"password"* name=*"password"* class=*"easyui-validatebox"* required=*"true"* /></td>

</tr>

<tr>

<th align=*"right"*>重复密码:</th>

<td><input type=*"password"* name=*"passwordre"* class=*"easyui-validatebox"* required=*"true"* validType=*"eqPassword['#loginInputForm input[name=password]']"*/></td>

</tr>

validType=*"eqPassword['#loginInputForm input[name=password]']"即可以实现*

## 补充

$(function(){

loginInputForm=$(‘loginInputForm’).form({

url:.....,

onSubmit: function(){},

success: function(data){}

}) ----------- loginInputForm中加载的是参数

})

这样写之后loginInputForm就等价于$(‘loginInputForm’)这个表单对象

loginInputForm.submit()也可以写成loginInputForm.form(‘submit’)在用一个javascript中

loginInputForm.submit()在其他地方提交等价于做了

loginInputForm=$(‘loginInputForm’).form(‘submit’,{

url:.....,

onSubmit: function(){},

success: function(data){}

})

其他

$.messager.show({

title: '标题',

msg: '登陆成功'

});

# 增加注册

<!DOCTYPE html>

<html>

<head>

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

<script type=*"text/javascript"*

src=*"../js/jquery-easyui-1.3/jquery-1.7.2.min.js"*></script>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/jquery.easyui.min.js"*></script>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/icon.css"* type=*"text/css"*></link>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/default/easyui.css"* type=*"text/css"*></link>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/locale/easyui-lang-zh\_CN.js"*></script>

<script type=*"text/javascript"* src=*"../js/myjs.js"*></script>

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

**var** loginAndRegDialog;

**var** loginInputForm;

**var** regDialog;

$(**function**(){

loginAndRegDialog=$('#loginAndRegDialog').dialog({

closable : **false**,//把右上角叉去掉

modal :**true**,//显示背景灰色

buttons:

[

{text:'注册',

handler:**function**(){

regDialog.dialog('open');

}},

{text:'登陆',

handler:**function**(){

//提交

loginInputForm.submit();

}}

]

});

loginInputForm=$('#loginInputForm').form({

url: sy.bp()+ '/uiform',

onSubmit: **function**(){

**if**(!$('#loginInputForm').form('validate')){

**return false**;

}

},

success: **function**(data){

console.info(data);

//需要将data类型转化成object类型

**var** obj = jQuery.parseJSON(data);

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

$.messager.alert("登陆成功","欢迎你:"+obj.username);

$.messager.show({

title: '标题',

msg: '登陆成功'

});

}

**if**(obj==**null**){

$.messager.alert("提示","您的输入有错误!");

}

}

});

//给登陆表单绑定事件

loginInputForm.find('input').on('keyup',**function**(event){

**if**(event.keyCode=='13'){

loginInputForm.submit();

}

});

//注册的dialog

regDialog=$('#regDialog').show().dialog({

modal: **true**,

closed: **true**,

title: '注册',

buttons: [{

text: '注册',

handler: **function**(){

regForm.submit();

}

}],

//获取焦点

onOpen: **function**(){

setTimeout(**function**(){

$('#regForm').find('input[name=name]').focus();

});

},

onClose: **function**(){

loginTabs.tabs('select',0);

}

});

//注册的表单

regForm=$('#regForm').form({

url: sy.bp()+ '/regForm',

onSubmit: **function**(){

**if**(!$('#regForm').form('validate')){

**return false**;

};

},

success: **function**(data){

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

$.messager.show({

title: '标题',

msg: '注册成功'

});

regDialog.dialog('close');

}

}

});

//给注册表单绑定时间

regForm.find('input').on('keyup',**function**(event){

**if**(event.keyCode=='13'){

regForm.submit();

}

});

});

</script>

</head>

<body>

This is my HTML page. <br>

<div id=*"loginAndRegDialog"* title=*"用户登陆"* style="width:*250px*;height:*200px*;">

<form method=*"post"* id=*"loginInputForm"*>

<table>

<tr>

<th align=*"right"*>用户名:</th>

<td><input type=*"text"* name=*"name"* class=*"easyui-validatebox"* required=*"true"*/></td>

</tr>

<tr>

<th align=*"right"*>密码:</th>

<td><input type=*"password"* name=*"password"* class=*"easyui-validatebox"* required=*"true"* /></td>

</tr>

<tr>

<th align=*"right"*>重复密码:</th>

<td><input type=*"password"* name=*"passwordre"* class=*"easyui-validatebox"* required=*"required"* validType=*"eqPassword['#loginInputForm input[name=password]']"*/></td>

</tr>

</table>

</form>

</div>

<div id=*"regDialog"* style="width:*250px*;height:*200px*; display: *none*">

<form method=*"post"* id=*"regForm"*>

<table>

<tr>

<th align=*"right"*>用户名:</th>

<td><input type=*"text"* name=*"name"* class=*"easyui-validatebox"* required=*"true"*/></td>

</tr>

<tr>

<th align=*"right"*>密码:</th>

<td><input type=*"password"* name=*"password2"* class=*"easyui-validatebox"* required=*"true"* /></td>

</tr>

<tr>

<th align=*"right"*>重复密码:</th>

<td><input type=*"password"* name=*"passwordaa"* class=*"easyui-validatebox"* required=*"required"* validType=*"eqPassword['#regForm input[name=password2]']"*/></td>

</tr>

</table>

</form>

</div>

</body>

</html>

# layout的使用

<!DOCTYPE html>

<html>

<head>

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

<script type=*"text/javascript"*

src=*"../js/jquery-easyui-1.3/jquery-1.7.2.min.js"*></script>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/jquery.easyui.min.js"*></script>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/icon.css"* type=*"text/css"*></link>

<link rel=*"stylesheet"* href=*"../js/jquery-easyui-1.3/themes/default/easyui.css"* type=*"text/css"*></link>

<script type=*"text/javascript"* src=*"../js/jquery-easyui-1.3/locale/easyui-lang-zh\_CN.js"*></script>

<script type=*"text/javascript"* src=*"../js/myjs.js"*></script>

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

**var** cc;

$(**function**(){

cc=$('#layout').layout();//使用js加载在这里不能写其他的方法,否则无法加载成功

cc.layout('collapse','west');//左边的面板隐藏

});

**function** getCenterPane(){

**var** centerPanel=cc.layout('panel','center');//如何使用panel方法.返回一个中间面板.

console.info(centerPanel);

//如何获取中间面板中title或者其他的属性

console.info(centerPanel.panel('options').title);

**if**(!centerPanel.panel('options').title){

centerPanel.panel({

title: '显示标题'

});

}**else**{

centerPanel.panel({

title: **null**

});

}

}

**function** addPanel(){

$('#layout').layout('add',{

region: 'east',

width: 180,

title: 'east Title',

split: **true**,

tools: [{

iconCls:'icon-add',

handler:**function**(){alert('add')}

},{

iconCls:'icon-remove',

handler:**function**(){alert('remove')}

}]

});

}

</script>

</head>

<body id=*"layout"* >

<div data-options=*"region:'north',title:'aaaaa',split:true"* style="height:*100px*;"></div>

<div data-options=*"region:'south',title:'South Title',split:true"* style="height:*100px*;">

<input type=*"button"* onclick="getCenterPane();" value=*"显示隐藏标题"*/>

<input type=*"button"* onclick="$('#layout').layout('expand','west')" value=*"展开左侧面板"*/>

<input type=*"button"* onclick="addPanel()" value=*"右侧添加面板"*/>

</div>

<div data-options=*"region:'west',title:'West',split:true"* style="width:*100px*;"></div>

<div data-options=*"region:'center'"* style="padding:*5px*;background:*#eee*;" href=*"5.html"*>

</div>

</body>

</html>

# dataGrid数据表格

## html中

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

**var** datagrid;

$(**function**(){

datagrid=$('#datagrid').datagrid({

url: sy.bp()+ '/datagrid',

title: 'User数据测试', //标题

iconCls: 'icon-save',//图标

pagination: **true**,//是否标记页数页码,在下方会出现页数页码显示

pageSize: 5, //每页显示多少条记录

pageList: [5,10,15,20], //分页列表

fit:**true**, //随着窗体的改变而改变大小

fitColumns: **true**,//是否去掉x轴滚动

nowarp: **false**, //文字是否可以自动换行

border: **false**, //边界是否有像素

idField: 'id', //对每行进行的标识

sortName: 'id', //默认的排列id

sortOrder: 'desc',//默认的排列方式

columns: [[

{title:'编号',field:'id',width:100,sortable:**true**},//sortable表示编号可以点击并切换升序降序

{title:'姓名',field:'username',width:100,sortable:**true**},

{title:'密码',field:'password',width:100}

]]

});

});

</script>

<div class=*"easyui-tabs"* fit=*"true"* border=*"false"*>

<div title=*"用户管理"* border=*"false"*>

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

</div>

</div>

## Controller中

@RequestMapping(value="datagrid")

@ResponseBody

**public** Map dataGrid(**int** page,**int** rows,String sort,String order){

List<User> list=accountService.getAllUser(page,rows,sort,order);

Long num=accountService.getUserCount();

Map map=**new** HashMap();

map.put("total", num);

map.put("rows", list);

**return** map;

}

## Service方法

/\*\*

\* 返回待条件的用户列表

\*/

**public** List<User> getAllUser(Integer page,Integer rows,String sort, String order) {

String hql="from User order by "+sort+" "+order ;

List<User> list=dao.findPage(hql, page, rows);

**return** list;

}

/\*\*

\* 查询用户总数

\*/

**public** Long getUserCount() {

String hql="select count(\*) from User";

**return** dao.findCounts(hql);

}

## dao层方法

/\*\*

\* 查询总数

\*/

**public** Long findCounts(String hql,Object...params) {

Query query=sessionFactory.getCurrentSession().createQuery(hql);

setParamter(query, params)

**return** (Long) query.uniqueResult();

}

**public** List findPage(String hql, **int** page, **int** size) {

Query query=sessionFactory.getCurrentSession().createQuery(hql);

**return** query.setFirstResult((page-1)\*size).setMaxResults(size).list();

}

# dataGrid实现数据交互和增删改查

## 实现统一查询方法和查询功能

datagridTest = $('#datagridTest').datagrid({

url : sy.bp() + '/datagrid',

title : '用户列表', //标题

iconCls : 'icon-save',//图标

pagination : **true**,//是否标记页数页码,在下方会出现页数页码显示

pageSize : 5, //每页显示多少条记录

pageList : [ 5, 10, 15, 20 ], //分页列表

fit : **true**, //随着窗体的改变而改变大小

fitColumns : **true**,//是否去掉x轴滚动

nowarp : **false**, //文字是否可以自动换行

border : **false**, //边界是否有像素

idField : 'id', //对每行进行的标识

sortName : 'id', //默认的排列id

sortOrder : 'desc',//默认的排列方式

### 如何增加工具栏

toolbar : [ {

text : '增加',

iconCls : 'icon-add',

handler : **function**() {

}

}, '-', {

text : '修改',

iconCls : 'icon-edit',

handler : **function**() {

}

}, '-', {

text : '删除',

iconCls : 'icon-remove',

handler : **function**() {

}

} ]

### 如何在一个窗口中再创建多个layout

<div class=*"easyui-tabs"* fit=*"true"* border=*"false"*>

<div title=*"用户管理"* border=*"false"*>

<div class=*"easyui-layout"* fit=*"true"* border=*"false"*>

<div region=*"north"* border=*"false"* title=*"过滤"*

style="height:*100%*;width:*100%*">

</div>

<div border=*"false"* region=*"center"*>

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

</div>

</div>

</div>

</div>

### dataGrid创建表单以及如何后台传值

表格创建在添加的layout中即*north*

<table class=*"tableForm"* style="width:*100%*;height:*100%*">

<tr>

<th>用户名</th>

<td><input name=*"name"* style="width:*315px*;" /></td>

</tr>

<tr>

<th>创建时间</th>

<td><input name=*"createdatatimeStart"*

class=*"easyui-datetimebox"* style="width:*155px*;" editable=*"false"* />

至<input name=*"createdatatimeEnd"* class=*"easyui-datetimebox"*

editable=*"false"* style="width:*155px*;" /> <a

href=*"javascript:void(0)"* class=*"easyui-linkbutton"*

onclick="my.serch();">查询</a> <a href=*"javascript:void(0)"*

class=*"easyui-linkbutton"* onclick="cleanSerch();">清空</a></td>

</tr>

</table>

### javascript中写上查询的方法

**var** usertableForm=$('#usertableForm').form();

my.serch=**function**(){

//可以序列化所有的表单对象

datagrid.datagrid('load',sy.serializeObject(usertableForm));

};

**sy.serializeObject为将表单对象中的所有元素转换为对象的方法如下:**

sy.serializeObject=**function**(form){/\*将form表单元素的值序列化成对象\*/

**var** o={};

$.each(form.serializeArray(),**function**(index){

//console.info(form.serializeArray());

**if**(o[**this**['name']]){

o[**this**['name']]=o[**this**['name']] + "," + **this**['value'];

}**else**{

o[**this**['name']]=**this**['value'];

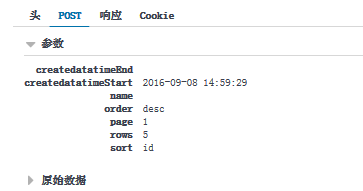
}

});

**return** o;

};

### 通过浏览器验证点击是否传入表单中的元素到后台



### 后台UserDao重写通用查询方法.

**public** **class** UserDaoImpl **extends** BaseDaoImpl **implements** UserDao{

**public** List<User> findPage(String hql, Integer page, Integer rows,

JsonUser userd) {

**final** String finalHql=**this**.setHql(hql,userd);

Query query=**this**.sessionFactory.getCurrentSession().createQuery(finalHql);

**return** query.setFirstResult((page-1)\*rows).setMaxResults(rows).list();

}

**private** String setHql(String hql, JsonUser userd) {

String finalHql=hql;

**if**(StringUtil.*isNotEmpty*(userd.getName())){

finalHql=finalHql+" and username like '%"+userd.getName()+"%'";

}

**if**(StringUtil.*isNotEmpty*(userd.getCreatedatatimeStart())){

finalHql=finalHql+" and createDate > '" + userd.getCreatedatatimeStart()+"'";

}

**if**(userd.getCreatedatatimeEnd()!=**null**&&userd.getCreatedatatimeEnd()!=""){

finalHql=finalHql+" and createDate < '" + userd.getCreatedatatimeEnd()+"'";

}

**if**(StringUtil.*isNotEmpty*(userd.getCreatedatatimeEnd())){

finalHql=finalHql+userd.getOrderby();

}

**return** finalHql;

}

}

### Controller

List<User> list=accountService.getAllUser(page,rows,sort,order,userd);就能实现所有的条件查询

### 实现清空按钮

my.cleanSerch=**function**(){

console.info($('#usertableForm'));

//清空的操作

datagrid.datagrid('load',{});//重新加载组件

//替换表单中inout中所有内容为空

$('#usertableForm input').val('');

}

# dataGrid添加对象

## 如何增加一行

### 增加按钮的事件中调用dataGrid中appendRow方法

handler : **function**() {

datagrid.datagrid('appendRow',{

});

### 如何使增加的一行可以编辑调用appendRow中beginEdit的方法

在增加的绑定事件中添加

handler : **function**() {

datagrid.datagrid('appendRow',{

});

**var** rows=datagrid.datagrid('getRows');

datagrid.datagrid('beginEdit',rows.length-1);

}

'getRows'是datagrid中方法获取所有行对象

'beginEdit'方法是当前行可以编辑

### 指定参数编辑属性

在对应的属性中增加editor属性

editor:{

type:'validatebox',

options:{

required:**true**

}

}

type属性是我需要将编辑框变成什么样.option表示editor的属性

### insertRow添加第一行

## datagrid中onAfterEdit进行传参

onAfterEdit会在事件进行了 datagridTest.datagrid('endEdit',editRow);时候触发操作

### 新建一个保存的toolbar

{

text : '保存',

iconCls : 'icon-save',

handler : **function**() {

//点击保存的时候结束当前编辑的行editRow为当前编辑的行

datagridTest.datagrid('endEdit',editRow);

}

当用户点击保存结束当前行编辑状态

### 在datagrid初始化中使用onAfterEdit方法

//在用户编辑结束后触发

onAfterEdit :**function**(rowIndex,rowData,changes){

console.info(rowData);

//重置editRow

//$.ajax

editRow=**undefined**;

}

使用此方法可以进行后台传参

### 限定只出现一个编辑状态

**if**(editRow!=**undefined**){

datagridTest.datagrid('endEdit',editRow);

}

**if**(editRow==**undefined**){

datagridTest.datagrid('appendRow',{

id: sy.uuid(),

username:'请输入用户名',

password:'请输入密码',

});

datagridTest.datagrid('beginEdit',rows.length-1);

editRow=rows.length-1;

}

## 生成uuid的js方法

//自动生成uuid

sy.uuid=**function**() {

**var** d = **new** Date().getTime();

**var** uuid = 'xxxxxxxx-xxxx-4xxx-yxxx-xxxxxxxxxxxx'.replace(/[xy]/g, **function**(c) {

**var** r = (d + Math.random()\*16)%16 | 0;

d = Math.floor(d/16);

**return** (c=='x' ? r : (r&0x3|0x8)).toString(16);

});

**return** uuid;

};

## 扩展datagrid可以使用datetimebox

### js中添加方法

$.extend($.fn.datagrid.defaults.editors, {

datetimebox: {

init: **function**(container, options){

**var** editor = $('<input />').appendTo(container);

options.editable=**false**;

editor.datetimebox(options);//

**return** editor;

},

getValue: **function**(target){ //获取datetimebox的值

**return** $(target).datetimebox('getValue');

},

setValue: **function**(target, value){

$(target).datetimebox('setValue',value);

},

resize: **function**(target, width){

$(target).datetimebox('resize',width);

},

destroy: **function**(target){

$(target).datetimebox('destroy');

}

}

### 对时间类型的参数加上属性editor并声明类型为..

editor: {

type:'datetimebox',

options:{

required: **true**

}

}

## 取消编辑关闭所有行的实现

{

text : '取消编辑',

iconCls : 'icon-relo',

handler : **function**() {

editRow=**undefined**;

//取消编辑

datagridTest.datagrid('rejectChanges');

datagridTest.datagrid('unselectAll');

}

### 调用rejectChanges方法

回滚所有从创建或者上一次调用acceptChanges函数后更改的数据。

datagridTest.datagrid('rejectChanges');

### 调用unselectAll方法

datagridTest.datagrid('unselectAll');

在用户取消选择一行的时候触发，参数包括：  
index：选择的行的索引值，索引从0开始。  
row：对应于取消选择行的记录。

# datagrid修改的实现

方法:onDblClickRow:

|  |  |
| --- | --- |
|  | 在用户双击一行的时候触发，参数包括： index：点击的行的索引值，该索引值从0开始。 row：对应于点击行的记录。 |

## 双击开启编辑行

datagrid中调用方法onDblClickRow,双击开启当前行的编辑状态

onDblClickRow: **function**(index, row){

**if**(editRow!=**undefined**){

datagridTest.datagrid('endEdit',editRow);

}

**if**(editRow==**undefined**){

datagridTest.datagrid('beginEdit',index);

//设置editRow

editRow=index;

}

}

## 点击修改进入编辑状态

点击修改的时候判断选择了哪些行 方法:

|  |  |  |
| --- | --- | --- |
| getSelections |  | 返回所有被选中的行，当没有记录被选中的时候将返回一个空数组。 |

判断选中的行的长度是否等于1如果等于1开启编辑状态,

如何获取选中的行.

|  |  |  |
| --- | --- | --- |
| getRowIndex |  | 返回指定行的索引号，该行的参数可以是一行记录或一个ID字段值。 |
|  |  |  |
|  |  |  |

//返回选中的行.只允许选择一行.才继续修改

**var** rows=datagridTest.datagrid('getSelections');

//判断长度1

**if**(rows.length==1){

**if**(editRow!=**undefined**){

//关闭编辑行

datagridTest.datagrid('endEdit',editRow);

}

**if**(editRow==**undefined**){

//开启编辑状态,选获取到要开启的行的索引,使用方法getRowIndex

editRow=datagridTest.datagrid('getRowIndex',rows[0]);

datagridTest.datagrid('beginEdit',editRow);

//每次开启就取消所有的选中状态

datagridTest.datagrid('unselectAll');

}

}

# datagrid中使用删除

## 补充.添加id字段类型变成checkbox

columns : [ [ {

title : '编号',

field : 'id',

width : 100,

checkbox:**true**

},

## 批量删除实现步骤

1获得选中的项

2判断长度必须大于0不然提示错误提示框.$.message.alert('提示',...,'error');

3 提示用户是否确定删除..回调方法

4如果=true表示确定,

5定义一个数组记录要传的参数.

6 数组中添加数据.push(rows[i].id)

7 进行分割 .join(',')

handler : **function**() {

**var** rows=datagridTest.datagrid('getSelections');

**if**(rows.length==0){

$.messager.alert('提示','请选中一行','error');

}**else**{

$.messager.confirm('提示','您是否要删除选中的行',**function**(b){

**if**(b==**true**){

**var** ids=[];

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

ids.push(rows[i].id);

}

console.info(ids.join(','));

};

});

};

}

## 扩展动态改变editor

### js中方法

//扩展动态改变editor

$.extend($.fn.datagrid.methods,{

addEditor: **function**(jq,param){

**if**(param **instanceof** Array){

$.each(param,**function**(){

**var** e=$(jq).datagrid('getColumnOption',item.field);

e.editor=item.editor;

});

}**else**{

**var** e=$(jq).datagrid('getColumnOption',param.field);

e.editor=param.editor;

}

},

removeEditor: **function**(jq,param){

**if**(param **instanceof** Array){

$.each(param,**function**(index,item){

**var** e=$(jq).datagrid('getColumnOption', item);

e.editor={};

});

}**else**{

**var** e=$(jq).datagrid('getColumnOption',param);

e.editor={};

}

}

});

### 当点击增加的时候对密码增加一个editor

**if**(editRow==**undefined**){

//使用扩展的方法实现点击增加的时候密码不能修改

datagridTest.datagrid('addEditor',{

field:'password',

editor:{

type: 'validatebox',

options: {

required: **true**

}

}

});

### 当点击修改的时候移除editor

**if**(rows.length==1){

//使用扩展进行修改操作的时候不能进行密码编辑

datagridTest.datagrid('removeEditor','password');

……….

# datagrid的后台传值(交互)

## datagrid使用onAfterEdit操作增加和修改

1 在方法中使用getChanges方法能返回我上一次提交的增加的行或者修改删除的行,

2 判断返回的增加的还是修改的数据.来判断url

3 使用ajax返回数据到后台

4 添加成功则acceotChanges

5 添加失败则回滚.rejectChanges,并设置当前行状态undefined

6 最后要把editRow编辑状态设置为空,

### Controller中增加的方法

@RequestMapping(value="insert")

@ResponseBody

**public** Json insert(String username,String password,Date createdate){

//进行保存

User user=**new** User();

user.setUsername(username);

user.setPassword(password);

user.setCreateDate(createdate);

Json json=**new** Json();

**try** {

accountService.saveUser(user);

} **catch** (Exception e) {

json.setSuccess(**false**);

json.setMsg("失败");

**return** json;

}

json.setSuccess(**true**);

json.setMsg("保存成功,欢迎:"+username);

**return** json;

}

json为创建的一个返回信息类

### 页面中增加操作的处理

onAfterEdit :**function**(rowIndex,rowData,changes){

//在这里进行更新和增加的操作

//使用getChanges返回我需要返回的最后一次提交的属性对象

**var** changeInsert=datagridTest.datagrid('getChanges','inserted');

**var** changeUpdate=datagridTest.datagrid('getChanges','updated');

**var** url = '';

//判断我要指明的url

**if**(changeInsert.length>0){

url=sy.bp()+'/insert';

}

**if**(changeUpdate.length>0){

url=sy.bp()+'/update';

}

//使用ajax提交数据

$.ajax({

url: url,

data: rowData,

dataType: 'json',

success: **function**(r){

//判断返回值

**if**(r && r.success){

//接受改变acceptChanges,表示取消编辑状态不会改变该行数据

datagridTest.datagrid('acceptChanges');

$.messager.show({

msg: r.msg,

title: '成功'

});

}**else**

{

//保存没有成功,回滚所有的更改

datagridTest.datagrid('rejectChanges');

$.messager.alert('错误',r.msg,'error');

}

editRow=**undefined**;

}

});

### 实现删除的传值

handler : **function**() {

**var** rows=datagridTest.datagrid('getSelections');

**if**(rows.length==0){

$.messager.alert('提示','请选中一行','error');

}**else**{

$.messager.confirm('提示','您是否要删除选中的行',**function**(b){

**if**(b==**true**){

**var** ids=[];

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

ids.push(rows[i].id);

}

$.ajax({

url:sy.bp()+'/delete',

data: {

ids: ids.join(',')

},

dataTyte: 'json',

success: **function**(r){

**if**(r && r.success){

$.messager.show({

msg: r.msg,

title: '成功'

});

datagridTest.datagrid('load');

datagridTest.datagrid('unselectAll');

}**else**

$.messager.alert('失败','删除失败','warning');

}

});

};

});

};

}