阅读目录

获取最新代码

可以到http://ligerui.googlecode.com下载最新代码。

简介

jQuery LigerUI 是基于jQuery的一系列UI控件组合，简单而又强大，致力于快速打造Web前端界面解决方案。 因为是前端控件，跟服务器无关，可以适合.net,jsp,php等等web服务器环境。目前全部插件的打包压缩JS只有100K左右,很轻巧。使用插件式的开发模式，以“简单”为原则的设计，每个插件尽量独立，并可依赖拓展。

回到顶部

ligerUI是什么

jQuery LigerUI控件丰富，包括基础、导航、布局、表单、表格、树形、窗口等

基础：Resizable、Drag、Tip

导航：Menu、MenuBar、ToolBar

布局：Layout、Tab

表单：Form、TextBox、Button、CheckBox、ComboBox、DateEditor、Radio、Spinner

表格：Grid

树形：Tree

窗口：Dialog、MessageBox、Window

回到顶部

如何使用

jQuery LigerUI是基于jQuery而设计的一系列插件集合。基本上每个插件都是相对独立的。但是彼此之间又紧密地关联着，合理地对插件进行组装，实现出现各种复杂的功能。 使用UI可以帮助你快速地创建友好的用户界面。

第一个例子

```
<head>    <title></title>
    <link
href="http://www.cnblogs.com/lib/ligerUI/skin
s/Aqua/css/ligerui-all.css" rel="stylesheet"
type="text/css" />
    <script
src="http://www.cnblogs.com/lib/jquery/jquery
-1.3.2.min.js" type="text/javascript">
</script>
    <script
src="http://www.cnblogs.com/lib/ligerUI/js/co
re/base.js" type="text/javascript"></script>
    <script
src="http://www.cnblogs.com/lib/ligerUI/js/pl
ugins/ligerTextBox.js"
type="text/javascript"></script>
    <script type="text/javascript">
        $(function ()
```

```
        {
            //我们将一个html文本框对象转换成
ligerui文本框对象,并返回ligerui对象
var g = $("#txt1").ligerTextBox(
            {
                //如果没有输入时,会提示不能为空
                nullText: '不能为空'
            });


            /*
            如何获取属性
            */
            //方式一
            alert('方式一:' +
g.get('disabled'));
            //方式二
            alert('方式二:' +
$("#txt1").ligerTextBox('option',
'disabled'));


            /*
            如何设置属性
            */
            //方式一
            g.set('disabled', true);
            //方式二
            $("#txt1").ligerTextBox('option',
'disabled', false);


            /*
```

```
                如何调用方法
                */
                //方式一
                g.setDisabled();
                //方式二

$("#txt1").ligerTextBox('setEnabled');

                /*
                如何设置事件
                */
                //这里给文本框绑定一个改变值的事件
//也可以设置onChangeValue参数
                g.bind('changeValue', function
(value)
                {
                    alert(value);

                });

        });
    </script>
</head>
<body style="padding:10px">
        <input type="text" id="txt1" value=""
style="width:200px"/>
</body>
```

更多的参数和方法的设置可以查看API：

http://www.ligerui.com/api/

上面是TextBox的使用范例，其他的插件使用方式类

似。

如何使用ligerUI对象

我们应用完插件以后，是返回一个ligerui对象的，可以把这个对象保存在一个全局的变量里面。在后续的操作中可能会用到。如果因为变量作用域的限制等，没有及时保存起来。我们可以用其他方式获取。见如下：

保存到一个全局的**javascript**变量：

```
var g;
$(function ()
{
g = $("#txt1").ligerTextBox();
);
```

使用$.fn.ligerGetTextBoxManager

```
var g = $("#txt1").ligerGetTextBoxManager ();
```

使用$.ligerui.get方法

```
var g =  $.ligerui.get('txt1');
```

- 第三个方式的是使用ligerui对象的id直接获取的，在传入参数没有指定id的情况下，对象的id将会使用html元素的id，如果html元素没有id，将会自动生成一个。所以在这里我们可以用html文本框的id来获取。
- 如果没有指定html元素的id，可以用第一种方式或者第二种方式。
- 其实第二种方式可以用第一种方式来替代，实则上ligerText是可以重复调用的，不同的是第二次以后调用都是直接放回ligerui对象。当我们不确定html元素是否已经应用了插件的情况下可以使用第二种方式。
- 其他插件的命名跟TextBox类同

事件处理

事件处理有两种方式。一种是以参数的形式传入，一种是调用ligerui对象的bind方法。

```
            //方式一
            var g = $("#txt1").ligerTextBox(
            {
                onChangeValue :
function(value){alert(value);}
            });
```

```
            //方式二
             g.bind('changeValue', function
(value)
             {
                 alert(value);
             });
```

- 使用bind方法是没有带”on”的。
- 事件监听是可以多次绑定的。
- 对于某些事件，如果函数的返回值是false，那么后面还没有触发的函数将不会再执行
- 第二种方式(bind)，是V1.1.3使用core机制以后引入的。

方法调用

使用ligerui的接口很方便。只需要调用ligerui对象的方法即可。

```
//这里设置文本框不能编辑
g.setDisabled();
//这里设置文本框可以编辑
g.setEnabled();
```

也可以使用这种方式

```
$(“#grid”).ligerGrid(‘setEnabled’);
```

- 至于这个对象有哪些方法，可以查看API
- 对象的方法是可以扩展的，后面会有一篇ligerui扩展的章节来介绍
- 第二种方式是在V1.1.4加入的

获取参数值

每一个ligerui对象都会有get方法。可以获取参数值

```
var url = g.get(‘url’);
```

或者是：

```
var url =
$(“#grid”).ligerGrid(‘option’,’url’);
```

动态设置参数

每一个ligerui对象都会有set方法。用于动态得设置参数。比如改变Grid的url，那么可以这样写：

```
g.set(‘url’,url);
```

或者是：

```
g.set({url:url});
```

也可以用插件的方式：

```
$(“#grid”).ligerGrid(‘option’,’url’,url);
```

- 第二种方式是允许同时传入多个参数的。
- Set方法是所有插件的统一设置属性的接口
- Set方法是V1.1.3使用core机制以后引入的。
- 插件传参的方式是V1.1.4引入的

回到顶部

如何扩展

Ligerui的默认参数、方法都是可以扩展的，这里我们定义了两个入口: $.ligerDefaults和$.ligerMethods。

比如要改变或者扩展Grid的默认参数，可以改变$.ligerDefaults.Grid

默认参数扩展

只需要扩展对象：$.ligerDefaults.{Plugin}

比如要改变表格默认的头部标题：

```
if($.ligerDefaults.Grid)
{
    $.ligerDefaults.Grid.title = "我的表格";
}
```

本地化支持扩展

只需要扩展对象：$.ligerDefaults.{Plugin}String

比如把表格“加载时”翻译成英文：

```
if($.ligerDefaults.GridString)
{
    $.ligerDefaults.GridString.loadingMessage
```

```
= "loading...";
}
```

方法扩展

只需要扩展对象：$. ligerMethos.{Plugin}

这里给Grid ligerui对象增加一个alert方法：

```
$.extend($.ligerMethods.Grid,
             {
                 alert : function ()
                 {
                      //要注意到一点，这里的this就
是ligerui对象
                     var rowdata =
this.getSelectedRow();
                     if (!rowdata)
                         alert('空');
                     else

alert(rowdata.CustomerID);
                 }
             }
         );

function show()
{
     //后面就可以这样使用
     Var g = $(“#maingrid”).ligerGrid();
     g.alert();
}
```

搜索

欢迎使用Liger UI框架

作者: 谢略
QQ: 175932810
交流QQ群1群: 71664111
交流QQ群2群：104842296
交流QQ群3群：153770480
交流QQ群4群：190391819
交流QQ群5群：37205343
当前版本: V1.1.6
API: http://api.ligerui.com/
演示地址:http://demo.ligerui.com/
源码下载:http://ligerui.googlecode.com/
技术支持：http://www.cnblogs.com/leoxie2011/

事件列表

事件名	参数	描述
onPropertyChanged	(name, value)	属性改变事件
onRender	()	渲染事件
onRendered	()	渲染结束事件

事件详细

onPropertyChanged(name, value)

描述:

属性改变事件

参数列表:

参数名	类型	描述	默认值
name	*{String}*	属性名	
value	*{Object}*	属性值	

---

onRender()

描述:

渲染事件

---

onRendered()

描述:

渲染结束事件

组件方法列表

方法	参数	描述
bind	(name, fn, context)	绑定
get	()	获取属性
set	(arg, value)	设置属性
trigger	(name)	触发事件
unbind	(name)	取消绑定

组件方法详细

{Void} **bind** (name, fn, context)

描述:

绑定

参数列表:

参数名	类型	描述	默认值
name	*{String}*	事件名	''
fn	*{Function}*	监听函数	null
context	*{Object}*	作用域	

返回值:

{Void}

{Object} **get** ()

描述:

获取属性

返回值:

{Object}

{Void} **set** (arg, value)

描述:

设置属性

参数列表:

参数名	类型	描述	默认值
arg	*{String|Object}*	属性名,或者是属性/值列表	
value	*{Object}*	属性值	

返回值:

{Void}

{Void} **trigger** (name)

描述:

触发事件件

参数列表:

参数名	类型	描述	默认值
name	*{String}*	事件名	''

返回值:

{Void}

{Void} **unbind** (name)

描述:

取消绑定

参数列表:

参数名	类型	描述	默认值
name	*{String}*	事件名	''

返回值:

{Void}

插件详细

{$.ligerui.controls.Grid} **ligerGrid**

描述:

1,支持本地数据和服务器数据(配置data或者url)
2,支持排序和分页(包括Javascript排序和分页)
3,支持列的“显示/隐藏”
4,支持明细行(表格内嵌)
5,支持汇总行
6,支持单元格模板
7,支持编辑表格(ligerGrid的一个特色,需要其他表单插件的支持)
8,支持树表格
8,支持分组
8,支持多表头

例子:

```
$("#maingrid4").ligerGrid({
                checkbox: true,
                columns: [
                { display: '主键', name: 'CustomerID', align: 'left',
                { display: '公司名', name: 'CompanyName', minWidth: 6
                { display: '联系名', name: 'ContactName', width: 50,a
                { display: '联系名', name: 'ContactName', minWidth: 1
                 { display: '联系名', name: 'ContactName', minWidth:
                 { display: '联系名', name: 'ContactName', minWidth:
                 { display: '联系名', name: 'ContactName', minWidth:
                { display: '城市', name: 'City' }
                ], dataAction: 'server',pageSize:30,
                url: "../../service/NwindDataHandler.ashx?View=Custo
                width: '100%',height:'100%'
            });

var grid;
        $(function ()
        {
            grid = $("#maingrid4").ligerGrid({
                checkbox: true,
                columns: [
                { display: '主键', name: 'CustomerID', align: 'left',
                { display: '公司名', name: 'CompanyName', minWidth: 6
                { display: '联系名', name: 'ContactName', width: 50,
                { display: '城市', name: 'City' }
                ], dataAction: 'server', pageSize: 30,
```

```
            url: "../../service/NwindDataHandler.ashx?View=Custo
            width: '100%', height: '100%',
            onCheckRow: function (checked,data,rowindex,rowobj)
            {
                checked && $.ligerDialog.alert('选择的是'+data.Cu
            }
        });
        $("#pageloading").hide();
    });
    function getCheckedData()
    {
        var rows = grid.getCheckedRows();
        var str = "";
        $(rows).each(function ()
        {
            str += this.CustomerID + ",";
        });
        $.ligerDialog.alert('选择的是' + str);
    }

var jsonObj = {};
        jsonObj.Rows = [
            { id: 3, name: "林三", sex: "男", birthday: "1989-01-
            { id: 43, name: "陈丽", sex: "女", birthday: "1989-01
            { id: 33, name: "啊三", sex: "男", birthday: "1981-12
            { id: 34, name: "表三", sex: "男", birthday: "1983-01
            { id: 43, name: "陈丽", sex: "女", birthday: "1989-01
            { id: 33, name: "成三", sex: "男", birthday: "1989-11
            { id: 34, name: "都三", sex: "女", birthday: "1989-04
            { id: 324, name: "鄂三", sex: "男", birthday: "1999-0
            { id: 344, name: "林三", sex: "男", birthday: "1969-0
            { id: 1, name: "王开", sex: "男", birthday: "1989-01-
        ];
        $("#maingrid").ligerGrid({
            columns: [
            { display: '', width: 30, isAllowHide: false, name:
                render: function (row)
                {
                    var html = '□ ';
                    return html;
                },
                headerRender: function (column)
                {
                    var html = '□ ';
                    return html;
                }
            },
```

```
            { display: '主键', name: 'id', width: 50, type: 'int
            { display: '名字', name: 'name', width: 50 },
            { display: '性别', name: 'sex', width: 50, isSort: fa
            },
            { display: '生日', name: 'birthday', type: 'date', wi
            {
                display: '模板列', isAllowHide: false,
                render: function (row)
                {
                    var html = '编辑';
                    return html;
                }
            }
            ],width:'100%',
            data: jsonObj,
            url: "../Default.aspx", pkName: 'id',
            pageSizeOptions: [5, 10, 15, 20],
            onAfterShowData: function (grid)
            {
                if ($.fn.ligerCheckBox)
                    $(".l-grid-body input:checkbox,.l-grid-hd-ce
            }
        });
```

返回值:

{$.ligerui.controls.Grid} Grid组件管理器

参数列表

参数名	类型	描
width	*{String|Int}*	宽
height	*{String|Int}*	高
columnWidth	*{Int}*	默
resizable	*{String}*	ta
url	*{String}*	a
usePager	*{String}*	是
page	*{Int}*	默
total	*{Int}*	总
pageSize	*{Int}*	每
pageSizeOptions	*{Array}*	可
columns	*{Array}*	列
columns[i].id	*{String}*	列
columns[i].name	*{String}*	表
columns[i].totalSummary	*{Object}*	汇
columns[i].totalSummary.align	*{String}*	汇
columns[i].totalSummary.type	*{String}*	汇 ,(

columns[i].totalSummary.render	*{Function}*	汇至
columns[i].display	*{String}*	表
columns[i].columns	*{Array}*	多
columns[i].headerRender	*{Function}*	头
columns[i].isAllowHide	*{Bool}*	是将在
columns[i].isSort	*{Bool}*	是
columns[i].type	*{String}*	排fl
columns[i].width	*{Int}*	表
columns[i].minWidth	*{Int}*	表大值
columns[i].format	*{String}*	格
columns[i].align	*{String}*	右c
columns[i].hide	*{Int}*	初
columns[i].editor	*{Object}*	编
columns[i].editor.type	*{String}*	编cs

columns[i].editor.data	*{String}*	下
columns[i].editor.valueColumnName	*{String}*	名
columns[i].editor.displayColumnName	*{String}*	性
columns[i].editor.dataValueField	*{String}*	性 v 号
columns[i].editor.dataDisplayField	*{String}*	扇 d 框
columns[i].editor.minValue	*{String}*	最
columns[i].editor.maxValue	*{String}*	最
columns[i].editor.p	*{Function|Object}*	参 正
columns[i].editor.ext	*{Function}*	参 正 p
columns[i].editor.onChange	*{Function}*	绑
columns[i].editor.onChanged	*{Function}*	绑

columns[i].render	*{Function}*	单
detail	*{Object}*	明
detail.height	*{String|Int}*	高
detail.onShowDetail	*{Function}*	明
detail.onExtend	*{Function}*	明
detail.onCollapse	*{Function}*	明
minColToggle	*{Int}*	最
dataAction	*{String}*	拖 (I 尤 持
showTitle	*{Bool}*	是
showTableToggleBtn	*{Bool}*	是 钮
switchPageSizeApplyComboBox	*{Bool}*	切 li
allowAdjustColWidth	*{Bool}*	是
checkbox	*{Bool}*	是
showToggleColBtn	*{Bool}*	是

enabledEdit	*{Bool}*	是
InWindow	*{Bool}*	是 h
statusName	*{String}*	状
method	*{String}*	用
fixedCellHeight	*{Bool}*	是
heightDiff	*{Int}*	高 h 高 个
cssClass	*{String}*	附
dateFormat	*{String}*	默
root	*{String}*	数
record	*{String}*	数
pageParmName	*{String}*	页 务
pagesizeParmName	*{String}*	页 用
sortnameParmName	*{String}*	页
sortorderParmName	*{String}*	页
allowUnSelectRow	*{Bool}*	是
alternatingRow	*{Bool}*	是

mouseoverRowCssClass	*{String}*	鼠
enabledSort	*{Bool}*	是
rowAttrRender	*{Function}*	行 s
groupColumnName	*{String}*	分
groupColumnDisplay	*{String}*	分
groupRender	*{Function}*	分
totalRender	*{Function}*	统
delayLoad	*{Bool}*	初
contentType	*{String}*	A
checkboxColWidth	*{Int}*	复
detailColWidth	*{Int}*	明
where	*{Function}*	数 d it
selectRowButtonOnly	*{Bool}*	复 设
tree	*{Object}*	树
isChecked	*{Function}*	复
whenRClickToSelect	*{Bool}*	右

clickToEdit	*{Bool}*	单
minColumnWidth	*{Int}*	最
detailToEdit	*{Bool}*	是
frozen	*{Bool}*	是
frozenDetail	*{Bool}*	明
frozenCheckbox	*{Bool}*	复
detailHeight	*{Bool}*	明
rownumbers	*{Bool}*	是
frozenRownumbers	*{Bool}*	行
rownumbersColWidth	*{Bool}*	行
colDraggable	*{Bool}*	是
rowDraggable	*{Bool}*	是
rowDraggingRender	*{Function}*	行
autoCheckChildren	*{Bool}*	是

事件列表

事件名	参数	描述
onAfterAddRow	(e)	增加行后事件
onAfterBeginEdit	(e)	开始编辑后事件
onAfterChangeColumnWidth	(column, newwidth)	改变列宽度事件
onAfterShowData	(data)	显示完数据事件
onAfterSubmitEdit	(e)	提交编辑 事件
onBeforeChangeColumnWidth	(column, newwidth)	验证 改变列宽度 是否通过
onBeforeCheckAllRow	(checked, grid element)	选择前事件，可以通过return false阻止操作(复选框 全选/全不选)
onBeforeEdit	(e)	编辑前事件
onBeforeShowData	(data)	显示数据前事件，可以通过reutrn false阻止操作
onBeforeSubmitEdit	(e)	验证编辑器结果是否通过
onBeginEdit	(e)	验证 开始编辑是否通过
onCancelEdit	(e)	取消编辑 事件

onChangeSort	()	改变排序事件
onCheckAllRow	(checked, grid element)	选择事件(复选框 全选/全不选)
onCheckRow	(checked, rowdata, rowindex, rowDomElement)	选择事件(复选框)
onContextmenu	(parm, e)	右击事件
onDblClickRow	(rowdata, rowindex, rowDomElement)	双击行事件
onDragCol	(node)	拖动列事件
onError	()	错误事件
onLoaded	()	加载完函数
onLoading	()	加载时函数
onReload	()	刷新事件，可以通过return false来阻止操作
onRowDragDrop	(e)	行拖拽事件 后事件
onSelectRow	(rowdata, rowindex, rowDomElement)	选择行事件
onSubmit	()	提交前事件
onSuccess	()	成功事件
		第一页，可以

onToFirst	()	通过return false来阻止操作
onToggleCol	()	切换列事件
onToLast	()	最后一页，可以通过return false来阻止操作
onToNext	()	下一页，可以通过return false来阻止操作
onToPrev	()	上一页，可以通过return false来阻止操作
onUnSelectRow	(rowdata, rowindex, rowDomElement)	取消选择行事件

事件详细

onAfterAddRow(e)

描述:

增加行后事件

参数列表:

参数名	类型	描述	默认值
		e(columnname、columnindex、	

e	*{Object}*	column、reocrd、value)	

onAfterBeginEdit(e)

描述:

开始编辑后事件

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(reocrd、rowindex)	

onAfterChangeColumnWidth(column, newwidth)

描述:

改变列宽度事件

参数列表:

参数名	类型	描述	默认值
column	*{Object}*		
newwidth	*{Int}*		

onAfterShowData(data)

描述:

显示完数据事件

参数列表:

参数名	类型	描述	默认值

data	*{Object}*	数据	

onAfterSubmitEdit(e)

描述:

提交编辑 事件

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(reocrd、rowindex、newdata)	

onBeforeChangeColumnWidth(column, newwidth)

描述:

验证 改变列宽度 是否通过

参数列表:

参数名	类型	描述	默认值
column	*{Object}*		
newwidth	*{Int}*		

onBeforeCheckAllRow(checked, grid element)

描述:

选择前事件，可以通过return false阻止操作(复选框 全选/全不选)

参数列表:

参数名	类型	描述	默认值

checked	*{Bool}*		
grid element	*{Dom Element}*		

onBeforeEdit(e)

描述:

编辑前事件

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(columnname、columnindex、column、reocrd、value)	

onBeforeShowData(data)

描述:

显示数据前事件，可以通过reutrn false阻止操作

参数列表:

参数名	类型	描述	默认值
data	*{Object}*	数据	

onBeforeSubmitEdit(e)

描述:

验证编辑器结果是否通过

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(columnname、columnindex、column、reocrd、value)(单单元格编辑状态)或者e(reocrd、rowindex、newdata)	

onBeginEdit(e)

描述:

验证 开始编辑 是否通过

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(reocrd、rowindex)	

onCancelEdit(e)

描述:

取消编辑 事件

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(reocrd、rowindex)	

onChangeSort()

描述:

改变排序事件

onCheckAllRow(checked, grid element)

描述:

选择事件(复选框 全选/全不选)

参数列表:

参数名	类型	描述	默认值
checked	*{Bool}*		
grid element	*{Dom Element}*		

onCheckRow(checked, rowdata, rowindex, rowDomElement)

描述:

选择事件(复选框)

参数列表:

参数名	类型	描述	默认值
checked	*{Bool}*		
rowdata	*{Object}*		
rowindex	*{Int}*		
rowDomElement	*{Dom Element}*		

onContextmenu(parm, e)

描述:

右击事件

参数列表:

参数名	类型	描述	默认值
parm	*{Object}*	parm(data、row、rowindex)	
e	*{Object}*		

---

onDblClickRow(rowdata, rowindex, rowDomElement)

描述:

双击行事件

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*		
rowindex	*{Int}*		
rowDomElement	*{Dom Element}*		

---

onDragCol(node)

描述:

拖动列事件

参数列表:

参数名	类型	描述	默认值
node			

---

onError()

描述:

错误事件

---

onLoaded()

描述:

加载完函数

---

onLoading()

描述:

加载时函数

---

onReload()

描述:

刷新事件，可以通过return false来阻止操作

---

onRowDragDrop(e)

描述:

行拖拽事件 后 事件

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(rows、parent、near、after)	

---

onSelectRow(rowdata, rowindex, rowDomElement)

描述:

选择行事件

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*		
rowindex	*{Int}*		
rowDomElement	*{Dom Element}*		

---

onSubmit()

描述:

提交前事件

---

onSuccess()

描述:

成功事件

---

onToFirst()

描述:

第一页，可以通过return false来阻止操作

---

onToggleCol()

描述:

切换列事件

---

onToLast()

描述:

最后一页，可以通过return false来阻止操作

onToNext()

描述:

下一页，可以通过return false来阻止操作

onToPrev()

描述:

上一页，可以通过return false来阻止操作

onUnSelectRow(rowdata, rowindex, rowDomElement)

描述:

取消选择行事件

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*		
rowindex	*{Int}*		
rowDomElement	*{Dom Element}*		

组件方法列表

方法	参数	描述
addEditRow	(rowdata)	增加一个编辑行
addRow	(rowdata, rowParm, isBefore, parentRow)	增加新行
addRows	(rowdataArr)	一次性增加多行
append	(rowData, targetRow, nearRow, isBefore)	appendRow同名方法
appendRange	(rows, targetRow, nearRow, isBefore)	一次性附加多行
appendRow	(rowData, targetRow, nearRow, isBefore)	附加新行(树模式)
beginEdit	(rowParm)	进入编辑状态
cancelEdit	(rowParm)	取消编辑
changeCol	(from, to, isAfter)	改变列的位置
changeHeaderText	(columnparm, headerText)	改变表头文本
changePage	(ctype)	改变分页
changeSort	(columnName, sortOrder)	改变排序
collapse	(targetRow)	收缩(树模式)

collapseDetail	(rowParm)	收缩明细
deleteRange	(rows)	一次性删除多行
deleteRow	(rowParm)	选择行
deleteSelectedRow	()	删除选择的行
demotion	(targetRow)	降级(树模式)
down	(targetRow)	下移
enabledTotal	()	isTotalSummary同名方法
endEdit	(rowParm)	结束编辑
existRecord	(rowdata)	是否存在某记录
expand	(targetRow)	展开(树模式)
extendDetail	(rowParm)	展开明细
formatRecord	(record)	格式化数据,删除系统字段
getAdded	()	获取新增的数据
getCheckedRowObjs	()	获取选中的行 DOM对象集合
getCheckedRows	()	获取选中的行数据(复选框)
getChidren	(rowParm)	获取子节点数据(树模式)
getColumn	(columnpam)	获取列信息
getColumns	(columnLevel)	获取指定层级的Columns
getColumnType	(columnname)	根据列名获取列类型
getData	(status, removeStatus)	获取数据
getDeleted	()	获取删除过的数据
getParent	(rowParm)	获取父节点数据(树模式)

getRowObj	(rowParm)	行DOM转换为行数据
getSelected	()	获取选中的行数据(同getSelectedRow)
getSelectedRow	()	获取选中的行数据
getSelectedRowObj	()	获取选中的行 DOM对象
getSelectedRowObjs	()	获取选中的行 DOM对象集合
getSelectedRows	()	获取选中的行数据集合(支持Ctrl多选)
getSelecteds	()	获取选中的行数据集合(支持Ctrl多选)(同getSelectedRows)
getUpdated	()	获取修改过的数据
hasChildren	(rowParm)	是否包括子节点(树模式)
isLeaf	(rowParm)	是否叶节点(树模式)
isSelected	(rowdata)	判断是否选中
isTotalSummary	()	是否包含汇总
loadData	(loadDataParm)	刷新数据
loadServerData	(param, clause)	加载数据(服务器)
move	(from, to, isAfter)	移动行
moveRange	(rows)	移动多行
remove	(targetRow)	移除行
removeRange	(rows)	移除多行
reRender	(e)	重新加载html
setColumnWidth	(columnparm, value)	调整列宽
setOptions	(parms)	重新设置参数(同名方法set)

SubmitEdit	(rowParm)	提交编辑
toggle	(targetRow)	伸展/收缩节点(树模式)
toggleCol	(columnparm, visible)	显示/隐藏列
up	(targetRow)	上移
update	(rowDom, newRowData)	
updateCell	(cell, value, rowParm)	更新单元格
updateRow	(newRowData, rowDom)	更新行
upgrade	(targetRow)	升级(树模式)

组件方法详细

{Object} **addEditRow** (rowdata)

描述:

增加一个编辑行

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*		

返回值:

{Object}

{Void} **addRow** (rowdata, rowParm, isBefore, parentRow)

描述:

增加新行

例子:

```
function addRow()
{
    var manager = $("#maingrid").ligerGetGridManager();
    manager.addRow();
}
function addRowWithData()
{
    var manager = $("#maingrid").ligerGetGridManager();
    var row = manager.getSelectedRow();
    manager.addRow({
        DepartmentID: 3,
        DepartmentName: '销售部',
        RealName: "分为" + newrowid,
        ID: newrowid++,
        Sex : 1,
        Age : 25,
        IncomeDay: new Date(1306108800000),
        Phone : "2343434",
        Address: "wwrere4"
    }, row, true);
}
```

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*	要附加的数据(非必填)	
rowParm	*{Object}*	插入的位置 可以是DOM对象，也可以是Row Data(非必填)	
isBefore	*{Object}*	是否在指定Dom对象的前方插入行(非必填)	
parentRow	*{Object}*	作为子节点加入到这个行(树模式可用)(非必填)	

返回值:

{Void}

{Void} **addRows** (rowdataArr)

描述:

一次性增加多行

参数列表:

参数名	类型	描述	默认值
rowdataArr	*{Array}*	要附加的数据	

返回值:

{Void}

append (rowData, targetRow, nearRow, isBefore)

描述:

appendRow同名方法

参数列表:

参数名	类型	描述	默认值
rowData			
targetRow			
nearRow			
isBefore			

appendRange (rows, targetRow, nearRow, isBefore)

描述:

一次性附加多行

参数列表:

参数名	类型	描述	默认值
rows	*{Array}*	要附加的数据	
targetRow	*{Object}*	父节点	
nearRow	*{Object}*	插入的位置 rowid或者 rowdata	
isBefore	*{Object}*	是否在之前附加(非必填)	

{Void} **appendRow** (rowData, targetRow, nearRow, isBefore)

描述:

附加新行(树模式)

例子:

```
function addRow(parent)
{
    var dept = $("#txtDept").val();
    var parentRow;
    var selectRow = manager.getSelectedRow();
    if (parent) parentRow = selectRow;
    else parentRow = manager.getParent(selectRow);

    if (manager.isLeaf(parentRow))
    {
        alert('叶节点不能增加子节点');
        return;
    }
    var data = {
        name: dept,
        id: dept,
        remark: dept
    };
    if (parent)
    {
        manager.appendRow(data, parentRow);
    }
```

```
    else
    {
        manager.appendRow(data, parentRow,selectRow,false);
    }
}
```

参数列表:

参数名	类型	描述	默认值
rowData	*{Object}*	要附加的数据	
targetRow	*{Object}*	父节点	
nearRow	*{Object}*	插入的位置 rowid或者rowdata	
isBefore	*{Object}*	是否在之前附加(非必填)	

返回值:

{Void}

{Object} **beginEdit** (rowParm)

描述:

进入编辑状态

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

{Object} **cancelEdit** (rowParm)

描述:

取消编辑

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

---

changeCol (from, to, isAfter)

描述:

改变列的位置

参数列表:

参数名	类型	描述	默认值
from	*{Object}*	来源表头	
to	*{Object}*	目标位置表头	
isAfter	*{Bool}*	是否附加到后面	

{Void} **changeHeaderText** (columnparm, headerText)

描述:

改变表头文本

参数列表:

参数名	类型	描述	默认值
columnparm	*{Int|String}*	列参数,可以是列索引、列ID,也可以是列名	
headerText	*{String}*	表头文本	

返回值:

{Void}

{Void} **changePage** (ctype)

描述:

改变分页

例子:

```
function changePage(ctype)
        {
            var manager = $("#maingrid").ligerGetGridManager();
            manager.changePage(ctype);
        }
```

参数列表:

参数名	类型	描述	默认值
ctype	*{String}*	类型：first/prev/next/last/input	

返回值:

{Void}

{Void} **changeSort** (columnName, sortOrder)

描述:

改变排序

例子:

```
function changeSort(columnName, sortOrder)
        {
            var manager = $("#maingrid").ligerGetGridManager();
            manager.changeSort(columnName, sortOrder);
        }
```

参数列表:

参数名	类型	描述	默认值
columnName	*{String}*	列名	
sortOrder	*{String}*	排序方向,asc或者desc	

返回值:

{Void}

{Bool} **collapse** (targetRow)

描述:

收缩(树模式)

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Object} **collapseDetail** (rowParm)

描述:

收缩明细

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

---

deleteRange (rows)

描述:

一次性删除多行

参数列表:

参数名	类型	描述	默认值
rows			

{Void} deleteRow (rowParm)

描述:

选择行

例子:

```
function deleteRow()
        {
            var manager = $("#maingrid").ligerGetGridManager();
            manager.deleteRow($("tr.l-grid-row",manager.gridbody)[
        }
```

参数列表:

参数名	类型	描述	默认值
	{Object}	行Dom对象或者行对应的row	

rowParm		data	

返回值:

{Void}

{Void} **deleteSelectedRow** ()

描述:

删除选择的行

例子:

```
function deleteSelectedRow()
        {
            var manager = $("#maingrid").ligerGetGridManager();
            manager.deleteSelectedRow();
        }
```

返回值:

{Void}

{Bool} **demotion** (targetRow)

描述:

降级(树模式)

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Bool} **down** (targetRow)

描述:

下移

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

---

enabledTotal ()

描述:

isTotalSummary同名方法

---

endEdit (rowParm)

描述:

结束编辑

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata(行编辑模式)(单元格编辑模式为空)	

---

existRecord (rowdata)

描述:

是否存在某记录

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*	节点 数据	

{Bool} **expand** (targetRow)

描述:

展开(树模式)

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Object} **extendDetail** (rowParm)

描述:

展开明细

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

{Object} **formatRecord** (record)

描述:

格式化数据,删除系统字段

参数列表:

参数名	类型	描述	默认值
record	*{Object}*		

返回值:

{Object} record

---

getAdded ()

描述:

获取新增的数据

{Array} **getCheckedRowObjs** ()

描述:

获取选中的行 DOM对象集合

返回值:

{Array} Row Dom Object

{Array} **getCheckedRows** ()

描述:

获取选中的行数据(复选框)

返回值:

{Array} Row Array Data

{Array} **getChidren** (rowParm)

描述:

获取子节点数据(树模式)

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Array}

{Object} **getColumn** (columnpam)

描述:

获取列信息

参数列表:

参数名	类型	描述	默认值
columnpam	*{String}*	列ID、列索引	

返回值:

{Object} Column，对应于g.columns[columnindex]

{Array} **getColumns** (columnLevel)

描述:

获取指定层级的Columns

参数列表:

参数名	类型	描述	默认值
columnLevel			

返回值:

{Array} 返回指定层级的Columns

{String} **getColumnType** (columnname)

描述:

根据列名获取列类型

例子:

```
function getColumnType(columnname)
        {
            var manager = $("#maingrid").ligerGetGridManager();
            return manager.getColumnType(columnname);
        }
```

参数列表:

参数名	类型	描述	默认值
columnname	*{String}*	列名	

返回值:

{String} Column Type

{Array} **getData** (status, removeStatus)

描述:

获取数据

例子:

```
function getData()
{
var manager = $("#maingrid").liger();
return manager.getData();
}
function getData()
{
return $("#maingrid").liger('getData')
}
```

参数列表:

参数名	类型	描述	默认值
status	*{Object}*	状态名,可以为空	
removeStatus	*{Object}*	是否移除状态字段,可以为空	

返回值:

{Array} 返回的是一个数组

---

getDeleted ()

描述:

获取删除过的数据

{Object} getParent (rowParm)

描述:

获取父节点数据(树模式)

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

{Object} **getRowObj** (rowParm)

描述:

行DOM转换为行数据

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object} Row Object

{Object} **getSelected** ()

描述:

获取选中的行数据(同getSelectedRow)

返回值:

{Object} Row Data

{Object} **getSelectedRow** ()

描述:

获取选中的行数据

例子:

```
function getSelectedRow()
        {
            var manager = $("#maingrid").ligerGetGridManager();
            return manager.getSelectedRow();
        }
```

返回值:

{Object} Row Data

{Array} getSelectedRowObj ()

描述:

获取选中的行 DOM对象

返回值:

{Array} Row Dom Object

{Array} getSelectedRowObjs ()

描述:

获取选中的行 DOM对象集合

返回值:

{Array} Row Dom Object

{Object} getSelectedRows ()

描述:

获取选中的行数据集合(支持Ctrl多选)

返回值:

{Object} Row Data

{Object} getSelecteds ()

描述:

获取选中的行数据集合(支持Ctrl多选)(同getSelectedRows)

返回值:

{Object} Row Data

---

getUpdated ()

描述:

获取修改过的数据

{Bool} hasChildren (rowParm)

描述:

是否包括子节点(树模式)

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Bool} isLeaf (rowParm)

描述:

是否叶节点(树模式)

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Object} **isSelected** (rowdata)

描述:

判断是否选中

参数列表:

参数名	类型	描述	默认值
rowdata	*{Object}*	行数据	

返回值:

{Object} Row Data

{Bool} **isTotalSummary** ()

描述:

是否包含汇总

例子:

```
function isTotalSummary()
        {
            var manager = $("#maingrid").ligerGetGridManager();
            return manager.isTotalSummary();
        }
```

返回值:

{Bool} 是否包含汇总

{Void} **loadData** (loadDataParm)

描述:

刷新数据

例子:

```
function f_reload()
        {
            var manager = $("#maingrid").ligerGetGridManager();
            manager.loadData();
        }
```

参数列表:

参数名	类型	描述	默认值
loadDataParm	*{Function|Boolean|Object}*	是否重新提交服务器,或者是筛选的函数,也可以指定data	

返回值:

{Void}

{Void} **loadServerData** (param, clause)

描述:

加载数据(服务器)

参数列表:

参数名	类型	描述	默认值
param	*{Object}*	加载参数	
clause	*{Function}*	是筛选的函数	

返回值:

{Void}

---

move (from, to, isAfter)

描述:

移动行

参数列表:

参数名	类型	描述	默认值
from	*{Object}*	来源表头	
to	*{Object}*	目标位置表头	
isAfter	*{Bool}*	是否附加到后面	

{Bool} **moveRange** (rows)

描述:

移动多行

参数列表:

参数名	类型	描述	默认值
rows	*{Array}*		

返回值:

{Bool}

{Bool} **remove** (targetRow)

描述:

移除行

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Bool} **removeRange** (rows)

描述:

移除多行

参数列表:

参数名	类型	描述	默认值
rows	*{Array}*		

返回值:

{Bool}

{Object} **reRender** (e)

描述:

重新加载html

参数列表:

参数名	类型	描述	默认值
e	*{Object}*	e(rowdata、column)	

返回值:

{Object}

{Void} **setColumnWidth** (columnparm, value)

描述:

调整列宽

参数列表:

参数名	类型	描述	默认值
columnparm	*{Int|String}*	列参数,可以是列索引、列ID,也可以是列名	
value	*{Int}*	宽度	

返回值:

{Void}

{Void} **setOptions** (parms)

描述:

重新设置参数(同名方法set)

例子:

国家：

搜索

```
$("#searchbtn").ligerButton({ click: function ()
            {
                var gridManager = $("#maingrid").ligerGetGridManager
                var Country = $("#ddlCountry").val();
                gridManager.setOptions(
                    { parms: [{ name: 'Country', value: Country}] }
                );
                gridManager.loadData(true);
            }
```

参数列表:

参数名	类型	描述	默认值
parms	*{Object}*	参数列表	

返回值:

{Void}

{Object} **SubmitEdit** (rowParm)

描述:

提交编辑

参数列表:

参数名	类型	描述	默认值
rowParm	*{Object}*	rowindex或者rowdata	

返回值:

{Object}

{Bool} **toggle** (targetRow)

描述:

伸展/收缩节点(树模式)

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

{Void} **toggleCol** (columnparm, visible)

描述:

显示/隐藏列

参数列表:

参数名	类型	描述	默认值
columnparm	*{Int|String}*	列参数,可以是列索引、列ID,也可以是列名	
visible	*{Boolean}*	是否可见	

返回值:

{Void}

{Bool} **up** (targetRow)

描述:

上移

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

---

update (rowDom, newRowData)

描述:

参数列表:

参数名	类型	描述	默认值
rowDom			

newRowData			

{Void} **updateCell** (cell, value, rowParm)

描述:

更新单元格

参数列表:

参数名	类型	描述	默认值
cell	*{String|Object}*	单元格Dom对象或者列名	
value	*{String|Int|Float|Date}*	新值	
rowParm	*{Object}*	行Dom对象或者行对应的row data	

返回值:

{Void}

{Void} **updateRow** (newRowData, rowDom)

描述:

更新行

参数列表:

参数名	类型	描述	默认值
newRowData	*{Object}*	要附加的数据(非必填)	
rowDom	*{Object}*	DOM对象	

返回值:

{Void}

{Bool} **upgrade** (targetRow)

描述:

升级(树模式)

参数列表:

参数名	类型	描述	默认值
targetRow	*{Object}*	rowindex或者rowdata	

返回值:

{Bool}

插件详细

{$.ligerui.controls.Tree} **ligerTree**

描述:

1,使一段html配置为树结构。
2,通过data json对象配置为树结构。
3,通过ajax json对象配置为树结构。

例子:

```
$("#tree1").ligerTree({
data: [
{ text: '节点1', children: [
{ text: '节点1.1' },
{ text: '节点1.2' },
{ text: '节点1.3', children: [
{ text: '节点1.3.1' },
{ text: '节点1.3.2' }
]
},
{ text: '节点1.4' }
]
},
{ text: '节点2' },
{ text: '节点3' },
{ text: '节点4' }
]
});
```

返回值:

{$.ligerui.controls.Tree} Tree 组件管理器

参数列表

参数名	类型	描述	默认值
url	*{String}*	设置一个url用于加载数据	null
method	*{String}*	提交数据的方式	'POST
data	*{String}*	设置一个本地数据data用于加载数据	null
checkbox	*{Bool}*	是否显示复选框	true
parentIcon	*{String}*	非叶节点的图标	'folder'
childIcon	*{String}*	叶节点的图标	'leaf'
textFieldName	*{String}*	text字段名	'text'
treeLine	*{Bool}*	是否显示节点连接线	true
attribute	*{Array}*	属性,获取行数据时很有作用	['id','ur
nodeWidth	*{Int}*	节点的宽度	70
statusName	*{String}*	状态名	'__stat
single	*{Boolean}*	是否单选	false
isLeaf	*{Function}*	是否子节点的判断函数 参数1：Tree Node Data	null
		V1.02增加，支持ID、PID这种线性数	

idFieldName	*{String}*	据结构，只需要同时配置idFieldName和parentIDFieldName	null
parentIDFieldName	*{String}*	V1.02增加，支持ID、PID这种线性数据结构，只需要同时配置idFieldName和parentIDFieldName	null
slide	*{Bool}*	是否以动画的形式显示(展开/收缩节点)	true
iconFieldName	*{String}*	按钮的字段名	'icon'
nodeDraggable	*{Bool}*	是否允许节点拖拽	true
nodeDraggingRender	*{Function}*	节点拖拽时提示自定义函数 参数1：nodes data 参数2：draggable manager 参数3：grid manager	true
btnClickToToggleOnly	*{Bool}*	是否只在点击 展开/收缩 按钮时才 展开/收缩节点	true

事件列表

事件名	参数	描述
onAfterAppend	(parentNode, newdata)	加载数据完事件
onAppend	(parentNode, newdata)	加载数据时事件，对数据进行预处理以后
onBeforeAppend	(parentNode, newdata)	加载数据前事件，可以通过return false取消操作
onBeforeCollapse	(node)	折叠前事件,可以通过返回false来阻止继续折叠
onBeforeExpand	(node)	展开前事件,可以通过返回false来阻止继续展开
onBeforeSelect	(node)	选择前事件,可以通过返回false来阻止继续选择
onCheck	(node, checked)	选择事件
onClick	(node)	点击事件
onCollapse	(node)	折叠事件
onContextmenu	(node)	右击事件
onError	()	异步加载数据失败事件
onExpand	(node)	展开事件
onSelect	(node)	选择事件
onSuccess	()	异步加载数据成功事件

事件详细

onAfterAppend(parentNode, newdata)

描述:

加载数据完事件

参数列表:

参数名	类型	描述	默认值
parentNode	*{Object}*		
newdata	*{Object}*		

---

onAppend(parentNode, newdata)

描述:

加载数据时事件，对数据进行预处理以后

参数列表:

参数名	类型	描述	默认值
parentNode	*{Object}*		
newdata	*{Object}*		

---

onBeforeAppend(parentNode, newdata)

描述:

加载数据前事件，可以通过return false取消操作

参数列表:

参数名	类型	描述	默认值
parentNode	*{Object}*		
newdata	*{Object}*		

onBeforeCollapse(node)

描述:

折叠前事件,可以通过返回false来阻止继续折叠

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onBeforeExpand(node)

描述:

展开前事件,可以通过返回false来阻止继续展开

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onBeforeSelect(node)

描述:

选择前事件,可以通过返回false来阻止继续选择

参数列表:

参数名	类型	描述	默认值

node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onCheck(node, checked)

描述:

选择事件

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null
checked	*{bool}*		

onClick(node)

描述:

点击事件

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onCollapse(node)

描述:

折叠事件

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

---

onContextmenu(node)

描述:

右击事件

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

---

onError()

描述:

异步加载数据失败事件

---

onExpand(node)

描述:

展开事件

参数列表:

参数名	类型	描述	默认值

node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onSelect(node)

描述:

选择事件

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	node(node.data和node.target) target是DOM对象	null

onSuccess()

描述:

异步加载数据成功事件

组件方法列表

方法	参数	描述
append	(parentNode, newdata)	增加节点集合
cancelSelect	(domNode)	反选择节点
clear	()	清空
collapseAll	()	全部节点都折叠
demotion	(treenode)	降级为叶节点级别
expandAll	()	全部节点都展开
getChecked	()	获取选择的行(复选框)
getData	()	获取树的数据源
getDataByID	(id)	getDataByID
getNodeDom	(treenode)	获取节点 Dom对象
getParent	(treenode, level)	获取父节点数据
getParentTreeItem	(treenode, level)	获取父节点
getSelected	()	获取选择的行
getTextByID	(id)	getTextByID
hasChildren	(treenode)	是否包含子节点
loadData	(node, url, param)	加载数据
remove	(node)	删除节点
selectNode	(selectNodeParm)	选择节点
update	(domnode, newnodedata)	更新节点
upgrade	(treenode)	升级为父节点级别

组件方法详细

append (parentNode, newdata)

描述:

增加节点集合

参数列表:

参数名	类型	描述	默认值
parentNode	*{Object}*	节点(DOM 对象 标签为 li)、节点数据或者节点ID, 加载的数据将增加到这个节点下面	
newdata	*{Array}*	节点数据的集合,该参数为数组	

---

cancelSelect (domNode)

描述:

反选择节点

参数列表:

参数名	类型	描述	默认值
domNode	*{Object}*	Dom节点	

{Void} **clear** ()

描述:

清空

例子:

```
var manager = null;
        $(function ()
        {
            $("#tree1").ligerTree({ url: 'treeData.ashx' });
            manager = $("#tree1").ligerGetTreeManager();
        });
        function clear()
        {
            manager.clear();
        }
```

返回值:

{Void}

{Void} collapseAll ()

描述:

全部节点都折叠

例子:

```
var manager = null;
        $(function ()
        {
            $("#tree1").ligerTree({ url: 'treeData.ashx' });
            manager = $("#tree1").ligerGetTreeManager();
        });
        function collapseAll()
        {
            manager.collapseAll();
        }
```

返回值:

{Void}

---

demotion (treenode)

描述:

降级为叶节点级别

参数列表:

参数名	类型	描述	默认值

treenode	*{Object}*	节点(DOM 对象 标签为li) 、 节点数据或者节点ID	

{Void} **expandAll ()**

描述:

全部节点都展开

例子:

```
var manager = null;
        $(function ()
        {
            $("#tree1").ligerTree({ url: 'treeData.ashx' });
            manager = $("#tree1").ligerGetTreeManager();
        });
        function expandAll()
        {
            manager.expandAll();
        }
```

返回值:

{Void}

{Array, Void} **getChecked ()**

描述:

获取选择的行(复选框)

例子:

```
var manager = null;
        $(function ()
        {
            $("#tree1").ligerTree({ url: 'treeData.ashx' });
            manager = $("#tree1").ligerGetTreeManager();
        });
        function getCheckedData()
        {
            manager.getChecked();
        }
```

返回值:

{Array} Nodes 每一个Node包括的参数有：data(数据源),target(DOM 对象 标签为li)
{Void}

{Void, Array} **getData ()**

描述:

获取树的数据源

例子:

```
var manager = null;
        $(function ()
        {
            $("#tree1").ligerTree({ url: 'treeData.ashx' });
            manager = $("#tree1").ligerGetTreeManager();
        });
        function getData()
        {
            manager.getData();
        }
```

返回值:

{Void}
{Array} Tree Data Object

---

getDataByID (id)

描述:

getDataByID

参数列表:

参数名	类型	描述	默认值
id	*{String}*	ID	

---

getNodeDom (treenode)

描述:

获取节点 Dom对象

参数列表:

参数名	类型	描述	默认值
treenode	*{Object}*	节点数据或者节点ID	

{Object} **getParent** (treenode, level)

描述:

获取父节点数据

参数列表:

参数名	类型	描述	默认值
treenode	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID	
level	*{Object}*	获取第N级别的父节点(选填，不填时表示上一级父节点)	

返回值:

{Object} parentData

{Bool} **getParentTreeItem** (treenode, level)

描述:

获取父节点

参数列表:

参数名	类型	描述	默认值
treenode	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID	
level	*{Object}*	获取第N级别的父节点(选填，不填时表示上一级父节点)	

返回值:

{Bool} hasChildren

{Object} **getSelected** ()

描述:

获取选择的行

返回值:

{Object} Node 节点包括的参数有：data(数据源),target(DOM 对象 标签为li)

---

getTextByID (id)

描述:

getTextByID

参数列表:

参数名	类型	描述	默认值
id	*{String}*	ID	

{Bool, Void} **hasChildren** (treenode)

描述:

是否包含子节点

参数列表:

参数名	类型	描述	默认值
treenode	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID	

返回值:

{Bool} hasChildren
{Void}

---

loadData (node, url, param)

描述:

加载数据

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID,加载的数据将增加到这个节点下面	
url	*{String}*	要加载数据的URL	
param	*{String}*	提交数据的附件的参数	

---

remove (node)

描述:

删除节点

参数列表:

参数名	类型	描述	默认值
node	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID	

---

selectNode (selectNodeParm)

描述:

选择节点

参数列表:

参数名	类型	描述
selectNodeParm	*{Function|String}*	条件函数(treenodedata,treedata Dom节点或ID值

---

update (domnode, newnodedata)

描述:

更新节点

参数列表:

参数名	类型	描述	默认值
domnode	*{Object}*	节点(DOM 对象 标签为li)、节点数据或者节点ID	

newnodedata	*{Object}*	节点数据	

upgrade (treenode)

描述:

升级为父节点级别

参数列表:

参数名	类型	描述	默认值
treenode	*{Object}*	节点(DOM 对象 标签为li) 、节点数据或者节点ID	

插件详细

{$.ligerui.controls.ComboBox} ligerComboBox

描述:

1,可以将一个Select的表单对象转换成自定义下拉框。
2,可以将一个type=text的表单对象转换成自定义下拉框(需要配置数据源)。

返回值:

{$.ligerui.controls.ComboBox} 组件管理器

参数列表

参数名	类型	描述	默认值
data	*{Array}*	数据源.JSON格式	null
tree	*{Object}*	下拉框以树的形式显示，tree的参数跟ligerTree的参数一致	null
grid	*{Object}*	下拉框以Grid的形式显示，grid的参数跟ligerGrid的参数一致	null
resize	*{Bool}*	是否调整大小	true
slide	*{Bool}*	是否以动画的形式显示	true
isMultiSelect	*{Bool}*	是否多选	false
isShowCheckBox	*{Bool}*	是否选择复选框	false
selectBoxWidth	*{Bool}*	下拉框的宽度	false
selectBoxHeight	*{Bool}*	下拉框的高度	false
textField	*{String}*	text对应的字段名	'text'
valueField	*{String}*	value对应的字段名	'id'

valueFieldID	*{String}*	隐藏域的控件ID，如果页面不存在，自动创建一个	null
lable	*{String}*	添加标签	null
labelWidth	*{Int|Object}*	标签宽度	null
labelAlign	*{String}*	标签内文字水平位置	null
initValue	*{String}*	初始化Value	null
initText	*{String}*	初始化Text	null
split	*{String}*	分割符号,多选时有效	","
columns	*{Array}*	将下拉框转换为表格的模式	null
columns[i].name	*{String}*	表格列名	
columns[i].header	*{String}*	表格列标题	
columns[i].width	*{String}*	表格列宽度	
disabled	*{Bool}*	是否只读	false
hideOnLoseFocus	*{Bool}*	失去焦点时是否隐藏	true
url	*{String}*	数据源URL(需返回JSON)	null
render	*{Function}*	文本框显示html函数	null

事件列表

事件名	参数	描述
onBeforeOpen	()	打开下拉框前事件，可以通过return false来阻止继续操作，利用这个参数可以用来调用其他函数，比如打开一个新窗口来选择值
onBeforeSelect	(value, text)	选择前事件,可以通过返回false来阻止选择
onEndResize	()	下拉框停止调整大小事件
onError	()	Ajax读取失败事件
onHide	()	关闭 下拉框 事件
onSelected	(value, text)	选择后事件
onShow	()	打开 下拉框 事件
onStartResize	()	下拉框开始调整大小事件
onSuccess	()	Ajax读取成功事件

事件详细

onBeforeOpen()

描述:

打开下拉框前事件，可以通过return false来阻止继续操作，利用这个参数可以用来调用其他函数，比如打开一个新窗口来选择值

---

onBeforeSelect(value, text)

描述:

选择前事件,可以通过返回false来阻止选择

参数列表:

参数名	类型	描述	默认值
value	*{Object}*	值	
text	*{Object}*	文本	

---

onEndResize()

描述:

下拉框停止调整大小事件

---

onError()

描述:

Ajax读取失败事件

---

onHide()

描述:

关闭 下拉框 事件

---

onSelected(value, text)

描述:

选择后事件

参数列表:

参数名	类型	描述	默认值
value	*{Object}*	值	
text	*{Object}*	文本	

onShow()

描述:

打开 下拉框 事件

onStartResize()

描述:

下拉框开始调整大小事件

onSuccess()

描述:

Ajax读取成功事件

组件方法列表

方法	参数	描述
bulidContent	()	创建内容
clearContent	()	清空内容
findTextByValue	(value)	查找Text
findValueByText	(text)	查找Value
selectValue	(value)	选择值
setData	()	重新创建下拉框选项内容(针对于text input)，可用于data的内容改变时,或绑定data
setSelect	()	重新创建下拉框选项内容(针对于select)，可用于select的内容改变时

组件方法详细

{Void} **bulidContent ()**

描述:

创建内容

返回值:

{Void}

{Void} **clearContent ()**

描述:

清空内容

返回值:

{Void}

{Void} **findTextByValue** (value)

描述:

查找Text

参数列表:

参数名	类型	描述	默认值
value	*{String}*	值	

返回值:

{Void}

{Void} **findValueByText** (text)

描述:

查找Value

参数列表:

参数名	类型	描述	默认值
text	*{String}*	文本值	

返回值:

{Void}

---

selectValue (value)

描述:

选择值

参数列表:

参数名	类型	描述	默认值
value	*{String}*	值	

{Void} **setData ()**

描述:

重新创建下拉框选项内容(针对于text input)，可用于data的内容改变时,或绑定data

返回值:

{Void}

{Void} **setSelect ()**

描述:

重新创建下拉框选项内容(针对于select)，可用于select的内容改变时

返回值:

{Void}

插件详细

{$.ligerui.controls.Drag} **ligerDrag**

描述:

使目标对象可以拖动。

返回值:

{$.ligerui.controls.Drag} ligerui对象

参数列表

参数名	类型	描述	默认值
handler	*{String|jQuery}*	拖动的作用区域，在这个区域才可以触发拖动。可以是字符串(jQuery selector)，也可以是一个Dom jQuery对象	null
proxy	*{String|jQuery}*	拖动时的副本,可以是'clone'或者是函数,放回jQuery 对象	null
revert	*{Bool}*	设置为true的时候,副本将会恢复,而不应用拖动效果	false
animate	*{Bool}*	应用拖动的时候是否使用动画效果	false
disabled	*{Bool}*	是否不可用	false
proxyX	*{Int}*	代理是否按设置的值相对于鼠标定位	null
proxyY	*{Int}*	代理是否按设置的值相对于鼠标定位	null

事件列表

事件名	参数	描述
onDrag	(current, e)	拖动事件(返回值为false时阻止继续拖动)
onDragEnter	(current, e)	进入区域
onDragLeave	(current, e)	离开区域
onDragOver	(current, e)	在区域移动
onDrop	(current, e)	在区域释放
onRevert	(current, e)	正在Revert
onRevertd	(current, e)	Revert完成
onStartDrag	(current, e)	开始拖动事件
onStopDrag	(current, e)	结束拖动事件

事件详细

onDrag(current, e)

描述:

拖动事件(返回值为false时阻止继续拖动)

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

---

onDragEnter(current, e)

描述:

进入区域

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

---

onDragLeave(current, e)

描述:

离开区域

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

---

onDragOver(current, e)

描述:

在区域移动

参数列表:

			默认

参数名	类型	描述	值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

---

onDrop(current, e)

描述:

在区域释放

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

---

onRevert(current, e)

描述:

正在Revert

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

onRevertd(current, e)

描述:

Revert完成

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

onStartDrag(current, e)

描述:

开始拖动事件

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY)	
e	*{Object}*		

onStopDrag(current, e)

描述:

结束拖动事件

参数列表:

参数名	类型	描述	默认值
current	*{Object}*	current(Object 包括 target,left,top,startX,startY,diffX,diffY)	
e	*{Object}*		

参数列表

参数名	类型	描述	默认值
handles	*{String}*	调整大小的作用区域，在这个区域才可以触发调整大小。字符串。包括n , e, s, w, ne, se, sw, nw这八个方向，可自由选择一个或多个，多个时用逗号隔开(jQuery selector)，也可以是一个Dom jQuery对象	"n, e, s, w, ne, se, sw, nw"
maxWidth	*{Int}*	最大宽度	2000
maxHeight	*{Int}*	最大高度	2000
minWidth	*{Int}*	最小宽度	20
minHeight	*{Int}*	最小高度	20
animate	*{Bool}*	应用调整的时候是否使用动画效果	false

事件列表

事件名	参数	描述
onResize	(current, e)	调整大小事件
onStartResize	(current, e)	开始调整大小事件
onStopResize	(current, e)	结束调整大小事件

事件详细

onResize(current, e)

描述:

调整大小事件

参数列表:

参数名	类型	描述
current	*{Object}*	current(Object 包括 dir,left,top,width,height,newWidth,newHeight,d
e	*{Object}*	

---

onStartResize(current, e)

描述:

开始调整大小事件

参数列表:

参数名	类型	描述	默认值
current		current(Object 包括	

	{Object}	dir,left,top,width,height)	
e	*{Object}*		

---

onStopResize(current, e)

描述:

结束调整大小事件

参数列表:

参数名	类型	描述
current	*{Object}*	current(Object 包括 dir,left,top,width,height,newWidth,newHeight,d
e	*{Object}*	

插件详细

{$.ligerui.controls.Radio} **ligerRadio**

描述:

可以将一个单选框的表单对象转换成自定义单选框。

返回值:

{$.ligerui.controls.Radio} Radio 组件管理器

参数列表

参数名	类型	描述	默认值
disabled	*{Bool}*	是否只读	false

组件方法列表

方法	参数	描述
getValue	()	获取值
setDisabled	()	设置为不可用
setEnabled	()	设置为可用
setValue	(value)	设置值

组件方法详细

{Void} **getValue ()**

描述:

获取值

返回值:

{Void}

{Void} **setDisabled ()**

描述:

设置为不可用

返回值:

{Void}

{Void} **setEnabled ()**

描述:

设置为可用

返回值:

{Void}

{Void} **setValue** (value)

描述:

设置值

参数列表:

参数名	类型	描述	默认值
value	*{Bool}*	值	

返回值:

{Void}

插件详细

{$.ligerui.controls.Spinner} **ligerSpinner**

描述:

可以将一个'文本框'的表单对象转换成可调整

返回值:

{$.ligerui.controls.Spinner} Spinner 组件管理器

参数列表

参数名	类型	描述	默认值
type	*{Int}*	类型 float:浮点数 int:整数 time:时间	
decimalplace	*{Int}*	小数位 type=float时起作用	2
isNegative	*{Bool}*	是否是否负数	false
disabled	*{Bool}*	是否只读	false

事件列表

事件名	参数	描述
onChangeValue	(value)	改变值事件

事件详细

onChangeValue(value)

描述:

改变值事件

参数列表:

参数名	类型	描述	默认值
value	*{String}*	值	

组件方法列表

方法	参数	描述
getValue	()	获取值
setDisabled	()	设置为不可用
setEnabled	()	设置为可用
setValue	(value)	设置值

组件方法详细

{Void} **getValue ()**

描述:

获取值

返回值:

{Void}

{Void} **setDisabled ()**

描述:

设置为不可用

返回值:

{Void}

{Void} **setEnabled ()**

描述:

设置为可用

返回值:

{Void}

{Void} **setValue** (value)

描述:

设置值

参数列表:

参数名	类型	描述	默认值
value	*{Bool}*	值	

返回值:

{Void}

插件详细

{$.ligerui.controls.Layout} **ligerLayout**

描述:

应用Layout

返回值:

{$.ligerui.controls.Layout} Layout组件管理器

参数列表

参数名	类型	描述	默认值
height	*{Int|String}*	高度	'100%'
InWindow	*{Bool}*	是否以窗口的高度为准 height设置为百分比时可用	true
topHeight	*{Int}*	Top高度	50
bottomHeight	*{Int}*	Bottom宽度	50
leftWidth	*{Int}*	左侧宽度	110
centerWidth	*{Int}*	中侧宽度	300
rightWidth	*{Int}*	右侧宽度	170
isLeftCollapse	*{Bool}*	初始化时，左边是否折叠	false
isRightCollapse	*{Bool}*	初始化时，右边是否折叠	false
allowLeftCollapse	*{Bool}*	是否允许 左边可以隐藏	false
allowRightCollapse	*{Bool}*	是否允许 右边可以隐藏	false
allowLeftResize	*{Bool}*	是否允许 左边可以调整大小	false

allowRightResize	*{Bool}*	是否允许 右边可以调整大小	false
allowTopResize	*{Bool}*	是否允许 头部可以调整大小	false
allowBottomResize	*{Bool}*	是否允许 底部可以调整大小	false
space	*{Int}*	间隔	3

事件列表

事件名	参数	描述
onHeightChanged	(layoutHeight, diffHeight, middleHeight)	高度改变事件

事件详细

onHeightChanged(layoutHeight, diffHeight, middleHeight)

描述:

高度改变事件

参数列表:

参数名	类型	描述	默认值
layoutHeight	*{Int}*	整个layout高度	
diffHeight	*{Int}*	跟上一次比，高度的偏差	
middleHeight	*{Int}*	中间区域的高度	

组件方法列表

方法	参数	描述
setLeftCollapse	(isCollapse)	设置左边展开/收缩
setRightCollapse	(isCollapse)	设置右边展开/收缩

组件方法详细

{Void} **setLeftCollapse** (isCollapse)

描述:

设置左边展开/收缩

参数列表:

参数名	类型	描述	默认值
isCollapse	*{Bool}*		

返回值:

{Void}

{Void} **setRightCollapse** (isCollapse)

描述:

设置右边展开/收缩

参数列表:

参数名	类型	描述	默认值
isCollapse	*{Bool}*		

返回值:

{Void}

插件详细

{$.ligerui.controls.Accordion} **ligerAccordion**

描述:

应用面板

返回值:

{$.ligerui.controls.Accordion} 组件管理器

参数列表

参数名	类型	描述	默认值
height	*{Int}*	高度	null
speed	*{String}*	折叠/张开的速度	"normal"
changeHeightOnResize	*{Bool}*	改变窗口大小时是否自动调整	false
heightDiff	*{Int}*	高度补差	0

组件方法列表

方法	参数	描述
onResize	()	重新调整大小
setHeight	(height)	设置高度

组件方法详细

{Void} **onResize** ()

描述:

重新调整大小

返回值:

{Void}

{Void} **setHeight** (height)

描述:

设置高度

参数列表:

参数名	类型	描述	默认值
height			

返回值:

{Void}

插件详细

{$.ligerui.controls.Tab} **ligerTab**

描述:

应用Tab

返回值:

{$.ligerui.controls.Tab} Tab 组件管理器

参数列表

参数名	类型	描述	默认值
height	*{Int}*	高度	
heightDiff	*{Int}*	高度补差	
contextmenu	*{Bool}*	是否显示右键菜单	true
changeHeightOnResize	*{Bool}*	改变高度时是否调整大小	false
dblClickToClose	*{Bool}*	是否双击时关闭	true
dragToMove	*{Bool}*	是否允许拖动时改变tab项的位置	true

事件列表

事件名	参数	描述
onAfterAddTabItem	(TabID)	增加tab项后事件
onAfterOverrideTabItem	(TabID)	覆盖tab项后事件
onAfterRemoveTabItem	(TabID)	删除tab项后事件
onAfterSelectTabItem	(TabID)	选择tab项后事件
onBeforeAddTabItem	(TabID)	增加tab项前事件，可以通过return false阻止操作
onBeforeOverrideTabItem	(TabID)	覆盖tab项前事件，可以通过return false阻止操作
onBeforeRemoveTabItem	(TabID)	删除tab项前事件，可以通过return false阻止操作
onBeforeSelectTabItem	(TabID)	选择tab项前事件，可以通过return false阻止操作

事件详细

onAfterAddTabItem(TabID)

描述:

增加tab项后事件

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onAfterOverrideTabItem(TabID)

描述:

覆盖tab项后事件

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onAfterRemoveTabItem(TabID)

描述:

删除tab项后事件

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onAfterSelectTabItem(TabID)

描述:

选择tab项后事件

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onBeforeAddTabItem(TabID)

描述:

增加tab项前事件，可以通过return false阻止操作

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onBeforeOverrideTabItem(TabID)

描述:

覆盖tab项前事件，可以通过return false阻止操作

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onBeforeRemoveTabItem(TabID)

描述:

删除tab项前事件，可以通过return false阻止操作

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

---

onBeforeSelectTabItem(TabID)

描述:

选择tab项前事件，可以通过return false阻止操作

参数列表:

参数名	类型	描述	默认值
TabID	*{String}*	TabID	

组件方法列表

方法	参数	描述
addHeight	(heightDiff)	增加或缩小Tab内容区域的高度
addTabItem	(options)	增加一个tab
getNewTabid	()	生成一个TabID
getSelectedTabItemID	()	获取tab seelcted item 的tabid
getTabidList	(notabid, noclose)	获取tab item 的tabid集合(数组)
getTabItemCount	()	获取tab item 的数量
isTabItemExist	(tabid)	判断tab是否存在
moveTabItem	(fromTabItemID, toTabItemID)	切换两个tab项的位置
moveToLastTabItem	()	移动到最后一个tab
moveToNextTabItem	()	切换到下一个tab
moveToPrevTabItem	()	切换到上一个tab
overrideSelectedTabItem	(options)	覆盖选中的Tab项
overrideTabItem	(options)	覆盖指定的Tab项
reload	()	刷新指定的Tab项
removeAll	(tabid)	删除掉所有Tab项
removeOther	(tabid)	删除掉其他tabitem
removeSelectedTabItem	(tabid)	删除选中的tabitem
removeTabItem	(tabid)	移除tab项
selectTabItem	(tabid)	选中tab项

setHeight	(heightDiff)	设置Tab的高度

组件方法详细

{Bool} **addHeight** (heightDiff)

描述:

增加或缩小Tab内容区域的高度

参数列表:

参数名	类型	描述	默认值
heightDiff	*{Int}*	高度补差	

返回值:

{Bool}

{Void} **addTabItem** (options)

描述:

增加一个tab

参数列表:

参数名	类型	描述	默认值
options	*{Object}*	参数	
options.tabid	*{String}*	Tab ID	
options.url	*{String}*	传入整个参数时，将根据这个url创建一个iframe，并显示出来	
options.text	*{String}*	Tab Text	

options.showClose	*{Bool}*	是否允许关闭	true
options.height	*{Int}*	内容区域的高度	

返回值:

{Void}

{String} **getNewTabid ()**

描述:

生成一个TabID

返回值:

{String}

{Int} **getSelectedTabItemID ()**

描述:

获取tab seelcted item 的tabid

返回值:

{Int}

{Array} **getTabidList (notabid, noclose)**

描述:

获取tab item 的tabid集合(数组)

参数列表:

参数名	类型	描述	默认值
notabid	*{String}*	过滤掉的tabid	

noclose	*{String}*	是否过滤掉没有关闭按钮的	

返回值:

{Array}

{Int} **getTabItemCount** ()

描述:

获取tab item 的数量

返回值:

{Int}

{Bool} **isTabItemExist** (tabid)

描述:

判断tab是否存在

参数列表:

参数名	类型	描述	默认值
tabid	*{String}*		

返回值:

{Bool}

{Void} **moveTabItem** (fromTabItemID, toTabItemID)

描述:

切换两个tab项的位置

参数列表:

参数名	类型	描述	默认值
fromTabItemID	*{String}*		
toTabItemID	*{String}*		

返回值:

{Void}

{Void} **moveToLastTabItem** ()

描述:

移动到最后一个tab

返回值:

{Void}

{Void} **moveToNextTabItem** ()

描述:

切换到下一个tab

返回值:

{Void}

---

moveToPrevTabItem ()

描述:

切换到上一个tab

{Void} **overrideSelectedTabItem** (options)

描述:

覆盖选中的Tab项

参数列表:

参数名	类型	描述	默认值
options	*{Object}*	参数	
options.tabid	*{String}*	Tab ID	
options.url	*{String}*	传入整个参数时，将根据这个url创建一个iframe，并显示出来	
options.text	*{String}*	Tab Text	
options.showClose	*{Bool}*	是否允许关闭	true
options.height	*{Int}*	内容区域的高度	

返回值:

{Void}

{Void} **overrideTabItem** (options)

描述:

覆盖指定的Tab项

参数列表:

参数名	类型	描述	默认值
targettabid	*{String}*	TabID	
options	*{Object}*	Tab参数	

options.tabid	*{String}*	Tab ID	
options.url	*{String}*	传入整个参数时，将根据这个url创建一个iframe，并显示出来	
options.text	*{String}*	Tab Text	
options.showClose	*{Bool}*	是否允许关闭	true
options.height	*{Int}*	内容区域的高度	

返回值:

{Void}

{Void} **reload** ()

描述:

刷新指定的Tab项

返回值:

{Void}

{Void} **removeAll** (tabid)

描述:

删除掉所有Tab项

参数列表:

参数名	类型	描述	默认值
tabid			

返回值:

{Void}

{Void} **removeOther** (tabid)

描述:

删除掉其他tabitem

参数列表:

参数名	类型	描述	默认值
tabid	*{String}*		

返回值:

{Void}

{Void} **removeSelectedTabItem** (tabid)

描述:

删除选中的tabitem

参数列表:

参数名	类型	描述	默认值
tabid	*{String}*		

返回值:

{Void}

{Bool} **removeTabItem** (tabid)

描述:

移除tab项

参数列表:

参数名	类型	描述	默认值
tabid	*{String}*		

返回值:

{Bool}

{Void} **selectTabItem** (tabid)

描述:

选中tab项

参数列表:

参数名	类型	描述	默认值
tabid	*{String}*		

返回值:

{Void}

{Bool} **setHeight** (heightDiff)

描述:

设置Tab的高度

参数列表:

参数名	类型	描述	默认值
heightDiff	*{Int}*	高度补差	

返回值:

{Bool}

插件详细

{$.ligerui.controls.Menu} **ligerMenu**

描述:

显示菜单

例子:

```
menu = $.ligerMenu({ top: 100, left: 100, width: 120, items:
            [
            { text: '增加', click: onclick11,icon:'add' },
            { text: '修改', click: onclick11  },
            { line: true },
            { text: '查看', click: onclick11, children:
            [
            { text: '报表', click: onclick11 },
            { text: '导出', children: [{ text: 'Excel', click: onc
            }
            ] },
            { text: '关闭', click: onclick112 }
            ]
            });
```

返回值:

{$.ligerui.controls.Menu} Menu组件管理器

参数列表

参数名	类型	描述	默认值
pheight	*{Int}*	高度	120
pleft	*{Int}*	位置left	0
ptop	*{Int}*	位置top	0
pshadow	*{Bool}*	是否显示阴影	true
pitems	*{Array}*	菜单项的集合	nul
pitems[i].line	*{bool}*	是否是Line	false
pitems[i].text	*{String}*	菜单项名称	null
pitems[i].id	*{String}*	菜单项ID	null
pitems[i].icon	*{String}*	菜单项图标	null
pitems[i].disable	*{Bool}*	菜单项是否不可用	false
pitems[i].children	*{Array}*	菜单项的子菜单，参数跟pitems相同，递归结构	null
pitems[i].click	*{Function}*	菜单项点击事件 参数1：item 当前点击菜单项的信息 参数2：itemcount 菜单项的个数	null
pitems[i].dblclick	*{Function}*	菜单项双击事件 参数1：item 当前点击菜单项的信息 参数2：	null

		itemcount 菜单项的个数	

组件方法列表

方法	参数	描述
addItem	(options)	增加菜单项
getItemCount	()	获取菜单项的数目
hide	()	隐藏菜单
isEnable	(itemid)	获取菜单项是否可用
removeItem	(itemid)	删除菜单项
setDisable	(itemid)	设置菜单项不可用
setEnable	(itemid)	设置菜单项可用
show	(options)	显示菜单
toggle	()	显示/隐藏菜单

组件方法详细

{Void} **addItem** (options)

描述:

增加菜单项

例子:

```
var menuitem = {
            text: '动态项' + menu.getItemCount(),
            click : onclick11
        };
        menu.addItem(menuitem);
```

参数列表:

参数名	类型	描述	默认值
item	*{Object}*	菜单项信息	

item.line	*{bool}*	是否是Line	false
item.text	*{String}*	菜单项名称	null
item.id	*{String}*	菜单项ID	null
item.icon	*{String}*	菜单项图标	null
item.disable	*{Bool}*	菜单项是否不可用	false
item.children	*{Array}*	菜单项的子菜单	null
item.click	*{Function}*	菜单项点击事件 参数1：item 当前点击菜单项的信息 参数2：itemcount 菜单项的个数	null
item.dblclick	*{Function}*	菜单项双击事件 参数1：item 当前点击菜单项的信息 参数2：itemcount 菜单项的个数	null

返回值:

{Void}

{Int} **getItemCount ()**

描述:

获取菜单项的数目

返回值:

{Int}

{Void} **hide ()**

描述:

隐藏菜单

例子:

```
menu.hide();
```

返回值:

{Void}

{Bool} **isEnable** (itemid)

描述:

获取菜单项是否可用

参数列表:

参数名	类型	描述	默认值
itemid	*{String}*	菜单项ID	

返回值:

{Bool}

{Void} **removeItem** (itemid)

描述:

删除菜单项

参数列表:

参数名	类型	描述	默认值
itemid	*{String}*	菜单项ID	

返回值:

{Void}

{Void} **setDisable** (itemid)

描述:

设置菜单项不可用

参数列表:

参数名	类型	描述	默认值
itemid	*{String}*	菜单项ID	

返回值:

{Void}

{Void} **setEnable** (itemid)

描述:

设置菜单项可用

参数列表:

参数名	类型	描述	默认值
itemid	*{String}*	菜单项ID	

返回值:

{Void}

{Void} **show** (options)

描述:

显示菜单

例子:

```
menu = $.ligerMenu({ top: 100, left: 100, width: 120, items:
            [
            { text: '增加', click: onclick11,icon:'add' },
            { text: '修改', click: onclick11  },
            { line: true },
            { text: '查看', click: onclick11, children:
            [
            { text: '报表', click: onclick11 },
            { text: '导出', children: [{ text: 'Excel', click: onc
            }
            ] },
            { text: '关闭', click: onclick112 }
            ]
            });
            $(document).bind("contextmenu", function (e)
            {
                menu.show({ top: e.pageY, left: e.pageX });
                return false;
            });
```

参数列表:

参数名	类型	描述	默认值
options	*{Object}*	位置信息	
options.top	*{Int}*	位置信息Top	
options.left	*{Int}*	位置信息Left	

返回值:

{Void}

{Void} **toggle ()**

描述:

显示/隐藏菜单

返回值:

{Void}

组件方法列表

方法	参数	描述
ligerDialogAlert	(content, title, type, callback(不必填))	显示提示弹出框,
ligerDialogCloseWaitting	(content)	关闭对话框
ligerDialogconfirm	(title, content, callback)	确定框
ligerDialogError	(title, content, callback)	显示失败弹出框
ligerDialogOpen	(options)	弹窗打开方法，核心方法，弹窗所有方法都需要调用这个方法
ligerDialogPrompt	(title, value, multi, callback)	输入框
ligerDialogQuestion	(title, content, callback)	显示问题弹出框
ligerDialogSuccess	(title, content, callback)	显示成功弹出框
ligerDialogWaitting	(content)	等待对话框
ligerDialogWarn	(title, content, callback)	显示警告弹出框
ligerDialogWarning	(title, content, callback)	warning

组件方法详细

{$.ligerui.controls.Dialog} **ligerDialogAlert**

(content, title, type, callback(不必填))

描述:

显示提示弹出框,

例子:

```
$.ligerDialog.alert('内容');
```

参数列表:

参数名	类型	描述	默认值
content	*{String}*	内容	
title	*{String}*	标题(不必填,第二个参数，可替换为callback)	
type	*{String}*	类型 (不必填,第三个参数，可替换为callback)	
callback(不必填)	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog}
ligerDialogCloseWaitting (content)

描述:

关闭对话框

例子:

```
$.ligerDialog.closeWaitting();
```

参数列表:

参数名	类型	描述	默认值
content			

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} ligerDialogconfirm (title, content, callback)

描述:

确定框

例子:

```
$.ligerDialog.confirm(content,function (r)
{
alert(r ? '您点击了Y' : '您点击了N');
});
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：是否点击了确定	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} ligerDialogError (title, content, callback)

描述:

显示失败弹出框

例子:

```
$.ligerDialog.error('内容');
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} ligerDialogOpen (options)

描述:

弹窗打开方法，核心方法，弹窗所有方法都需要调用这个方法

例子:

```
$.ligerDialog.open(
{
```

```
width: 280,
type: 'error',
title:'对话框弹出演示标题',
content: '对话框弹出演示',
buttons: [{ text: '确定', onclick: btnClick }, { text: '取消', oncl
});
```

参数列表:

参数名	类型	描述	默认值
type	*{String}*	类型，包括success、donne、ok、error、warn、question	""
cls	*{String}*	给dialog附加css class	null
id	*{String}*	给dialog附加id	null
isDrag	*{Bool}*	是否拖动	true
isResize	*{Bool}*	是否调整大小	true
allowClose	*{Bool}*	是否允许关闭	true
width	*{Int}*	宽度	280
height	*{Int}*	高度，默认自适应	null
title	*{String}*	标题	
target	*{Object}*	目标对象，指定它将以appendTo()的方式载入	null
url	*{String}*	目标页url，默认以iframe的方式载入	null
		是否以load()的方	

load	*{Bool}*	式加载目标页的内容	false
modal	*{Bool}*	是否模态对话框	true
name	*{String}*	创建iframe时 作为iframe的name和id	null
content	*{String}*	内容	null
closeWhenEnter	*{String}*	按回车是否关闭对话框	null
isHidden	*{Bool}*	关闭对话框时是否只是隐藏，还是销毁对话框	true
show	*{Bool}*	初始化时是否马上显示	true
title	*{String}*	头部标题	
buttons	*{Array}*	按钮	
buttons[i].width	*{Int}*	宽度	
buttons[i].text	*{String}*	文本	
buttons[i].onclick	*{Function}*	显示完数据事件 参数1：button 参数2：dialog Dom object 参数3：button index	
showMax	*{Bool}*	是否显示最大化按钮	true
showToggle	*{Bool}*	是否显示收缩按钮	true
showMin		是否显示最小化按	true

	{Bool}	钮	
slide	*{Bool}*	是否启动动画效果	false
fixedType	*{String}*	是否固定在右下角	null
showType	*{String}*	是否动态显示	null

返回值:

{$.ligerui.controls.Dialog} 组件管理器

{$.ligerui.controls.Dialog} ligerDialogPrompt (title, value, multi, callback)

描述:

输入框

例子:

```
$.ligerDialog.prompt('提示内容', function (yes, value)
                    {
                        if (yes) alert(value);
                    });
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
value	*{String}*	初始化值(不必填,第二个参数，可替换为callback或multi)	
multi	*{String}*	是否多行文本框(不必填,第三个参数，可替换为callback)	

callback	*{Function}*	点击确定事件 参数1：yes 是否点击了是 参数2：value 文本框输入的值	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} ligerDialogQuestion(title, content, callback)

描述:

显示问题弹出框

例子:

```
$.ligerDialog.question('内容');
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} ligerDialogSuccess

(title, content, callback)

描述:

显示成功弹出框

例子:

```
$.ligerDialog.success('内容');
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} **ligerDialogWaitting** (content)

描述:

等待对话框

例子:

```
$.ligerDialog.waitting(content);
```

参数列表:

参数名	类型	描述	默认值
content	*{String}*	内容	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} **ligerDialogWarn** (title, content, callback)

描述:

显示警告弹出框

例子:

```
$.ligerDialog.warn('内容');
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

{$.ligerui.controls.Dialog} **ligerDialogWarning** (title, content, callback)

描述:

warning

例子:

```
$.ligerDialog.warning(content, function (action)
{
alert(action);
});
```

参数列表:

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容(不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：action 包括ok、no、cancel	

返回值:

{$.ligerui.controls.Dialog} jQuery对象

参数列表

参数名	类型	描述	默认值
content	*{String}*	内容	
title	*{String}*	标题(不必填,第二个参数，可替换为callback)	
type	*{String}*	类型 (不必填,第三个参数，可替换为callback)	
callback(不必填)	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

参数列表

参数名	类型	描述	默认值
content			

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：是否点击了确定	

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

参数列表

参数名	类型	描述	默认值
type	*{String}*	类型，包括success、donne、ok、error、warn、question	""
cls	*{String}*	给dialog附加css class	null
id	*{String}*	给dialog附加id	null
isDrag	*{Bool}*	是否拖动	true
isResize	*{Bool}*	是否调整大小	true
allowClose	*{Bool}*	是否允许关闭	true
width	*{Int}*	宽度	280
height	*{Int}*	高度，默认自适应	null
title	*{String}*	标题	

target	*{Object}*	目标对象，指定它将以appendTo()的方式载入	null
url	*{String}*	目标页url，默认以iframe的方式载入	null
load	*{Bool}*	是否以load()的方式加载目标页的内容	false
modal	*{Bool}*	是否模态对话框	true
name	*{String}*	创建iframe时 作为iframe的name和id	null
content	*{String}*	内容	null
closeWhenEnter	*{String}*	按回车是否关闭对话框	null
isHidden	*{Bool}*	关闭对话框时是否只是隐藏，还是销毁对话框	true
show	*{Bool}*	初始化时是否马上显示	true
title	*{String}*	头部标题	
buttons	*{Array}*	按钮	
buttons[i].width	*{Int}*	宽度	
buttons[i].text	*{String}*	文本	
buttons[i].onclick	*{Function}*	显示完数据事件 参数1：button 参数2：dialog	

		Dom object 参数3：button index	
showMax	*{Bool}*	是否显示最大化按钮	true
showToggle	*{Bool}*	是否显示收缩按钮	true
showMin	*{Bool}*	是否显示最小化按钮	true
slide	*{Bool}*	是否启动动画效果	false
fixedType	*{String}*	是否固定在右下角	null
showType	*{String}*	是否动态显示	null

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
value	*{String}*	初始化值(不必填,第二个参数，可替换为callback或multi)	
multi	*{String}*	是否多行文本框(不必填,第三个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：yes 是否点击了是 参数2：value 文本框输入的值	

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

参数列表

参数名	类型	描述	默认值
content	*{String}*	内容	

参数列表

参数名	类型	描述	默认值

title	*{String}*	标题	
content	*{String}*	内容 (不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：button 参数2：dialog Dom object 参数3：button index	

参数列表

参数名	类型	描述	默认值
title	*{String}*	标题	
content	*{String}*	内容(不必填,第二个参数，可替换为callback)	
callback	*{Function}*	点击确定事件 参数1：action 包括ok、no、cancel	

组件方法列表

方法	参数	描述
active	()	设置为最前端显示
close	()	关闭对话框
collapse	()	收缩
enter	()	按下回车
esc	()	按下esc
extend	()	展开
hidden	()	隐藏对话框
mask	()	遮罩
max	()	最大化
min	()	最小化
recover	()	还原
setUrl	()	设置URL
show	()	显示对话框
toggle	()	收缩/展开
unmask	()	取消遮罩

组件方法详细

{Void} **active ()**

描述:

设置为最前端显示

返回值:

{Void}

{Void} **close ()**

描述:

关闭对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.close();
```

返回值:

{Void}

{Void} **collapse ()**

描述:

收缩

返回值:

{Void}

{Void} **enter ()**

描述:

按下回车

返回值:

{Void}

{Void} **esc ()**

描述:

按下esc

返回值:

{Void}

{Void} **extend ()**

描述:

展开

返回值:

{Void}

{Void} **hidden ()**

描述:

隐藏对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.hidden();
```

返回值:

{Void}

{Void} **mask ()**

描述:

遮罩

返回值:

{Void}

{Void} **max ()**

描述:

最大化

返回值:

{Void}

{Void} **min ()**

描述:

最小化

返回值:

{Void}

{Void} **recover ()**

描述:

还原

返回值:

{Void}

{Void} **setUrl ()**

描述:

设置URL

返回值:

{Void}

{Void} **show ()**

描述:

显示对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.show();
```

返回值:

{Void}

{Void} **toggle ()**

描述:

收缩/展开

返回值:

{Void}

{Void} **unmask ()**

描述:

取消遮罩

返回值:

{Void}

插件详细

{$.ligerui.controls.Window} **ligerWindow**

描述:

显示窗口

返回值:

{$.ligerui.controls.Window} Window 组件管理器

参数列表

参数名	类型	描述	默认值
url	*{Bool}*	url 将url载入iframe并转换为窗口	
content	*{Bool}*	将HTML内容转换为窗口	
target	*{jQuery}*	将jQuery对象转换为窗口	
showClose	*{Bool}*	是否显示关闭	true
showMax	*{Bool}*	是否显示最大化窗口	true
showToggle	*{Bool}*	是否显示显示/隐藏窗口	true
showMin	*{Bool}*	是否显示最小化按钮	true
modal	*{Bool}*	是否模态对话框	true

组件方法列表

方法	参数	描述
active	()	设置为最前端显示
close	()	关闭对话框
hidden	()	隐藏对话框
mask	()	遮罩
max	()	最大化
min	()	最小化
setUrl	()	设置URL
show	()	显示对话框
unmask	()	取消遮罩

组件方法详细

{Void} **active ()**

描述:

设置为最前端显示

返回值:

{Void}

{Void} **close ()**

描述:

关闭对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.close();
```

返回值:

{Void}

{Void} **hidden ()**

描述:

隐藏对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.hidden();
```

返回值:

{Void}

{Void} **mask ()**

描述:

遮罩

返回值:

{Void}

{Void} **max ()**

描述:

最大化

返回值:

{Void}

{Void} **min ()**

描述:

最小化

返回值:

{Void}

{Void} **setUrl ()**

描述:

设置URL

返回值:

{Void}

{Void} **show ()**

描述:

显示对话框

例子:

```
var m = $.ligerDialog.open({ url: '../../welcome.htm', height: 200
m.show();
```

返回值:

{Void}

{Void} **unmask ()**

描述:

取消遮罩

返回值:

{Void}