# 简介

每个easyui组件都包含属性、方法和事件，开发者很容易扩展它们。

## 属性

属性被定义在 jQuery.fn.{plugin}.defaults中,例如dialog的属性被定义在 jQuery.fn.dialog.defaults中

## 事件

事件（回调函数）也被定义在jQuery.fn.{plugin}.defaults中

## 方法

调用方法的语法：$('selector').plugin('method', parameter);

其中：

* selector 是jquery选择器。
* plugin是插件名。
* method是该插件里面定义的方法。
* parameter是参数对象，可以是对象、字符串等。

方法被定义在jQuery.fn.{plugin}.methods中，每个方法有两个参数：jq和param。第一个参数”jq”是必须的，指代jquery对象。第二个参数“param”指的是调用该方法实际传递的参数。例如，想给dialog组件扩展一个叫“mymove”的方法，代码可以这样写：

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

mymove: function(jq, newposition){

return jq.each(function(){

$(this).dialog('move', newposition);

});

}

});

现在你可以调用“mymove”方法把一个dialog移动到特定的位置了：

$('#dd').dialog('mymove', {

left: 200,

top: 100

});

## 快速入门

下载easyui压缩包，在你的页面导入EasyUI CSS和JavaScript文件

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

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

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

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

导入必须的EasyUI文件后，你就可以使用html或javascript创建easyui组件。例如，你可以用下面的html创建一个可折叠的面板。

<div id="p" class="easyui-panel" style="width:500px;height:200px;padding:10px;"

title="My Panel" iconCls="icon-save" collapsible="true">

The panel content

</div>

从1.3版本开始，用html创建的组件可以使用“data-options”属性兼容HTML5新增的属性。所以，上面的代码你可以这样写：

<div id="p" class="easyui-panel" style="width:500px;height:200px;padding:10px;"

title="My Panel" data-options="iconCls:'icon-save',collapsible:true">

The panel content

</div>

下面的代码演示了如何创建一个绑定了“onSelect”事件的组合框。

<input class="easyui-combobox" name="language"

data-options="

url:'combobox\_data.json',

valueField:'id',

textField:'text',

panelHeight:'auto',

onSelect:function(record){

alert(record.text)

}">

译者注：

combobox\_data.json的格式如下

[{

"id":1,

"text":"text1"

},{

"id":2,

"text":"text2"

},{

"id":3,

"text":"text3",

"selected":true

},{

"id":4,

"text":"text4"

},{

"id":5,

"text":"text5"

}]

# 基础（Base）

## 解析器（parser）

### 用法

$.parser.parse(); // 解析整个页面

$.parser.parse('#cc'); // 解析某个具体的节点

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| **$.parser.auto** | **boolean** | **定义是否自动解析esayui组件** | **true** |

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| **$.parser.onComplete** | **context** | **解析器完成解析动作时触发** |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| $.parser.parse | context | 解析easyui组件 |

## 加载器（easyloder）

### 用法

#### 加载easyui模块

<script>

easyloader.base = '../'; // 设置easyui基本目录

$(function(){

easyloader.load('messager', function(){ // 加载指定模块

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

});

});

</script>

#### 通过绝对路径加载script脚本

using('http://code.jquery.com/jquery-1.4.4.min.js', function(){

// ...

});

#### 通过相对路径加载script脚本

// 相对easyui路径的脚本

using('./myscript.js', function(){

// ...

});

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| modules | object | 预定义的模块 |  |
| locales | object | 预定义的语言环境 |  |
| base | string | Easyui的基本目录，必须以“/”结尾 | 基本目录自动被设置为相对于easyload.js |
| theme | string | 定义在“themes”目录中的主题名称 | default |
| css | boolean | 定义在加载模块时是否加载css文件 | true |
| locale | string | 语言环境名称 | null |
| timeout | number | 以毫秒计的超时值。超时就触发 | 2000 |

### 预定义的语言环境

* af
* bg
* ca
* cs
* cz
* da
* de
* en
* es
* fr
* nl
* tr
* zh\_CN
* zh\_TW

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| onProgress | name | 一个模块成功加载后触发 |
| onLoad | name | 一个模块和它的依赖文件成功加载后触发 |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| load | module, callback | 加载指定模块，加载成功后一个回调函数会被调用  该模块有效的参数类型是：  一个模块名，  一个模块名数组，  一个css文件，  一个js文件。 |

## 可拖动块（draggable）

使用$.fn.draggable.defaults重写默认设置

### 使用示例

使用html标签创建一个可拖动元素

<div id="dd" class="easyui-draggable" data-options="handle:'#title'" style="width:100px;height:100px;">

<div id="title" style="background:#ccc;">title</div>

</div>

使用javascript创建一个可拖动元素

<div id="dd" style="width:100px;height:100px;">

<div id="title" style="background:#ccc;">title</div>

</div>

$('#dd').draggable({

handle:'#title'

});

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| proxy | string,function | 该属性设置为‘clone’时，会产生当前元素的代理元素，拖动时拖动的是代理元素。如果该属性设置为一个函数，那么这个函数的返回值应该是一个jQuery对象。  下面的例子演示了如何创建一个简单的代理对象  $('.dragitem').draggable({  proxy: function(source){  var p = $('<div style="border:1px solid #ccc;width:80px"></div>');  p.html($(source).html()).appendTo('body');  return p;  }  }); | null |
| revert | boolean | 如果设置为true，当停止拖动时，元素会回到初始位置 | false |
| cursor | string | 拖动时光标的css样式 | move |
| deltaX | number | 拖动的元素相对于光标在x轴的偏移量 | null |
| deltaY | number | 拖动的元素相对于光标在y轴的偏移量 | null |
| handle | selector | 开启拖动的处理器 | null |
| disabled | boolean | 设置为true，禁止拖动 | false |
| edge | number | The drag width in which can start draggable. | 0 |
| axis | string | 定义拖动元素沿x轴还是沿y轴拖动，设置为“v”，沿y轴拖动，设置为“h”，沿x轴拖动。 | null |
| delay | number | 定义开始拖动的延迟毫秒值 | 100 |

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| onBeforeDrag | e | 拖动前触发，返回false取消拖动 |
| onStartDrag | e | 开始拖动时触发 |
| onDrag | e | 拖动中触发，返回false取消拖动 |
| onEndDrag | e | 拖动停止时触发。此事件在onStopDrag之前触发。从1.5.3版本开始可用。 |
| onStopDrag | e | 拖动停止时触发 |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| options | none | 返回此插件对象 |
| proxy | none | 如果设置了proxy属性，返回拖动元素的代理元素 |
| enable | none | 启用元素的拖动 |
| disable | none | 禁用元素的拖动 |

## 可放置域（droppable）

使用$.fn.droppable.defaults重写默认设置

### 使用示例

使用html标签创建一个可放置块区域

<div class="easyui-droppable" data-options="accept:'#d1,#d3'" style="width:100px;height:100px;">

</div>

使用javascript创建一个可放置块区域

<div id="dd" style="width:100px;height:100px;"></div>

$('#dd').droppable({

accept:'#d1,#d3'

});

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| accept | selector | 确定哪些可拖动元素能被放置 | null |
| disabled | boolean | 设置为true，取消放置 | false |

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| onDragEnter | e,source | 当可拖动元素拖入时触发，参数source表示拖动的DOM元素 |
| onDragOver | e,source | 当可拖动元素在此放置区拖动时触发，参数source表示拖动的DOM元素 |
| onDragLeave | e,source | 当可拖动元素拖离时触发。参数source表示拖动的DOM元素 |
| onDrop | e,source | 当可拖动元素被放置时触发。参数source表示拖动的DOM元素 |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| options | none | 返回此插件对象 |
| enable | none | 启用放置区功能 |
| disable | none | 禁用放置区功能 |

## 可变尺寸域（resizable）

使用$.fn.resizable.defaults重写默认设置

### 使用示例

使用html标签创建可变尺寸域

<div class="easyui-resizable" style="width:100px;height:100px;border:1px solid #ccc;"

data-options="maxWidth:800,maxHeight:600">

</div>

使用javascript创建可变尺寸域

<div id="rr" style="width:100px;height:100px;border:1px solid #ccc;"></div>

$('#rr').resizable({

maxWidth:800,

maxHeight:600

});

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| disabled | boolean | 设置为true，禁用可变尺寸 | false |
| handles | string | 表示在哪个方向调节尺寸，“n”表示在北部，“e”表示在东部等等 | n, e, s, w, ne, se, sw, nw, all |
| minWidth | number | 可调节的最小宽度 | 10 |
| minHeight | number | 可调节的最小高度 | 10 |
| maxWidth | number | 可调节的最大宽度 | 10000 |
| maxHeight | number | 可调节的最大高度 | 10000 |
| edge | number | 调节时边框的有效选中宽度 | 5 |

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| onStartResize | e | 开始调节时触发 |
| onResize | e | 正在调节时触发。设置为false，此DOM元素不会实时地显示拖动的过程。 |
| onStopResize | e | 当停止拖动时触发 |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| options | none | 返回此可变尺寸域对象 |
| enable | none | 启用可调节功能 |
| disable | none | 禁用可调节功能 |

## 分页（pagination）

使用$.fn.pagination.defaults重写默认设置

此分页插件能帮助用户按页获取数据。支持页码输入和每页显示的数据条数设置。开发者可以在此分页栏右侧添加自定义按钮来增加自己的功能。

http://www.jeasyui.com/documentation/images/pagination.png

### 依赖

* 链接按钮（linkbutton）

### 使用示例

使用html标签创建一个分页栏

<div id="pp" class="easyui-pagination" style="background:#efefef;border:1px solid #ccc;"

data-options="total:2000,pageSize:10">

</div>

使用javascript创建一个分页栏

<div id="pp" style="background:#efefef;border:1px solid #ccc;"></div>

$('#pp').pagination({

total:2000,

pageSize:10

});

让我们用面板和分页插件来创建一个ajax分页栏。当用户选择一个新的页码时，面板将会显示该指定页的相应内容。

<div id="content" class="easyui-panel" style="height:200px"

data-options="href:'show\_content.php?page=1'">

</div>

<div class="easyui-pagination" style="border:1px solid #ccc;"

data-options="

total: 2000,

pageSize: 10,

onSelectPage: function(pageNumber, pageSize){

$('#content').panel('refresh', 'show\_content.php?page='+pageNumber);

}">

</div>

上面的面板默认显示第一页的内容。当用户输入页码时，“onSelectPage”事件会被触发，从而调用面板的“refresh”方法，传递页码参数，加载新一页的内容。

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| total | number | 总记录数。分页栏创建时就应该设置 | 1 |
| pageSize | number | 每页记录数 | 10 |
| pageNumber | number | 分页栏创建时显示的页码 | 1 |
| pageList | array | 用户可以改变每页记录数。pageList属性定义了每页记录数的选择项。  代码示例:  $('#pp').pagination({  pageList: [10,20,50,100]  }); | [10,20,30,50] |
| loading | boolean | 定义数据是否正在加载中 | false |
| buttons | array,selector | 定义功能按钮，可能的值： 1) 一个数组,每个按钮包含两个属性: iconCls: 用于显示背景图片的css类名 handler: 当按钮被点击时触发的处理函数 2) 表示一个按钮的选择器  用html标签声明的按钮：  <div class="easyui-pagination" style="border:1px solid #ccc" data-options="  total: 114,  buttons: [{  iconCls:'icon-add',  handler:function(){alert('add')}  },'-',{  iconCls:'icon-save',  handler:function(){alert('save')}  }]">  </div>  用javascript声明的按钮：  $('#pp').pagination({  total: 114,  buttons: [{  iconCls:'icon-add',  handler:function(){alert('add')}  },'-',{  iconCls:'icon-save',  handler:function(){alert('save')}  }]  }); | null |
| layout | array | 分页栏布局。从1.3.5版本开始可用。布局条目包含一个或多个下面的值： 1) list: 每页记录数列表。 2) sep: 页码按钮分隔符。 3) first: 首页按钮。 4) prev: 上一页按钮。 5) next: 下一页按钮。 6) last: 尾页按钮。 7) refresh: 刷新按钮。 8) manual: 手工的页码输入框，可以输入页码。 9) links: 页码链接。 10) info: 页面信息。  代码示例：  $('#pp').pagination({  layout:['first','links','last']  }); |  |
| links | number | 页码链接数量。仅当“layout”包含了“links”时可用。从1.3.5版本开始可用。 | 10 |
| showPageList | boolean | 定义是否显示每页记录数列表 | true |
| showRefresh | boolean | 定义是否显示刷新按钮 | true |
| showPageInfo | boolean | 定义是否显示页面信息。从1.5.2版本开始可用。 | true |
| beforePageText | string | 在页码输入框前显示一个标签 | Page |
| afterPageText | string | 在页码输入框后显示一个标签 | of {pages} |
| displayMsg | string | 展示页面信息 | Displaying {from} to {to} of {total} items |

**译者注:**

分页栏的中文化，也可以通过重写默认值实现

if($.fn.pagination) {

$.fn.pagination.defaults.beforePageText = "第 ";

$.fn.pagination.defaults.afterPageText = "页 共 {pages} 页";

$.fn.pagination.defaults.displayMsg = "当前{from}-{to} 条 共{total}条记录";

}

### 事件

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| onSelectPage | pageNumber, pageSize | 当用户选择一个新页码时触发。回调函数包含两个参数： pageNumber: 新的页码 pageSize: 每页记录数  代码示例：  $('#pp').pagination({  onSelectPage:function(pageNumber, pageSize){  $(this).pagination('loading');  alert('pageNumber:'+pageNumber+',pageSize:'+pageSize);  $(this).pagination('loaded');  }  }); |
| onBeforeRefresh | pageNumber, pageSize | 刷新按钮点击前触发，返回false会取消刷新操作 |
| onRefresh | pageNumber, pageSize | 刷新后触发 |
| onChangePageSize | pageSize | 修改每页记录数时触发 |

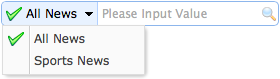
### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| options | none | 返回此插件对象 |
| loading | none | 让分页栏变为加载状态 |
| loaded | none | 让分页栏变为加载成功状态 |
| refresh | options | 刷新并展示页面信息。从1.3版本可用  代码示例：  $('#pp').pagination('refresh'); // 刷新页码条信息  $('#pp').pagination('refresh',{ // 修改并刷新页码条信息  total: 114,  pageNumber: 6  }); |
| select | page | 选择一个新页码。页码索引从1开始。从1.3版本可用。  代码示例：  $('#pp').pagination('select'); // refresh the current page  $('#pp').pagination('select', 2); // select the second page |

## 搜索框（SearchBox）

扩展自$.fn.textbox.defaults。使用$.fn.searchbox.defaults重写默认设置。

搜索框能提示用户输入要搜索的内容。它能让用户从绑定的菜单中选择不同的搜索种类。当用户敲击回车键或者点击右边的搜索按钮时，该组件会执行搜索。



### 依赖

* 文本框（textbox）
* 菜单按钮（menubutton）

### 使用示例

#### 创建搜索框

1.使用html标签创建。在<input/>标签上添加class “easyui-searchbox”。

<script type="text/javascript">

function qq(value,name){

alert(value+":"+name)

}

</script>

<input id="ss" class="easyui-searchbox" style="width:300px"

data-options="searcher:qq,prompt:'Please Input Value',menu:'#mm'"></input>

<div id="mm" style="width:120px">

<div data-options="name:'all',iconCls:'icon-ok'">All News</div>

<div data-options="name:'sports'">Sports News</div>

</div>

2. 通过编程创建

<input id="ss"></input>

<div id="mm" style="width:120px">

<div data-options="name:'all',iconCls:'icon-ok'">All News</div>

<div data-options="name:'sports'">Sports News</div>

</div>

$('#ss').searchbox({

searcher:function(value,name){

alert(value + "," + name)

},

menu:'#mm',

prompt:'Please Input Value'

});

### 属性

|  |  |  |  |
| --- | --- | --- | --- |
| **名称** | **类型** | **描述** | **默认值** |
| width | number | 该组件的宽度 | auto |
| height | number | 该组件的高度。从1.3.2版本可用 | 22 |
| prompt | string | 展示在输入框中的提示信息 | '' |
| value | string | 输入的值 | '' |
| menu | selector | 搜索类型菜单。菜单的每一项可以有下面这些属性： name: 搜索类型名 selected: 当前应该选中的类型名  下面的例子展示了如何定义一个选中的类型名：  <input class="easyui-searchbox" style="width:300px" data-options="menu:'#mm'" />  <div id="mm" style="width:150px">  <div data-options="name:'item1'">Search Item1</div>  <div data-options="name:'item2',selected:true">Search Item2</div>  <div data-options="name:'item3'">Search Item3</div>  </div> | null |
| searcher | function(value,name) | 当用户按下回车键或点击搜索按钮是，searcher函数会被触发 | null |
| disabled | boolean | 定义是否禁用搜索框。从1.3.6版本可用 | false |

### 方法

|  |  |  |
| --- | --- | --- |
| **名称** | **参数** | **描述** |
| options | none | 返回此插件对象 |
| menu | none | Return the search type menu object. The example below shows how to change the menu item icon.  var m = $('#ss').searchbox('menu'); // get the menu object  var item = m.menu('findItem', 'Sports News'); // find the menu item  // change the menu item icon  m.menu('setIcon', {  target: item.target,  iconCls: 'icon-save'  });  // select the searching type name  $('#ss').searchbox('selectName', 'sports'); |
| textbox | none | Return the text box object. |
| getValue | none | Return the current searching value. |
| setValue | value | Set a new searching value. |
| getName | none | Return the current searching type name. |
| selectName | name | Select the current searching type name.  Code example:  $('#ss').searchbox('selectName', 'sports'); |
| destroy | none | Destroy this component. |
| resize | width | Reset the component width. |
| disable | none | Disable the searchbox. Available since version 1.3.6. |
| enable | none | Enable the searchbox. Available since version 1.3.6. |
| clear | none | Clear the searching value. Available since version 1.3.6. |
| reset | none | Reset the searching value. Available since version 1.3.6. |