**EXT-JS**

## 一、简单介绍

**1 ExtJS的前身是YUI(Yahoo User Interface)。**

经过不断的发展与改进，ExtJS现在已经成功发布到了ExtJS6版本,是一套目前最完整和最成熟的javascript基础库之一,**利用ExtJS构建的WEB应用具有与桌面程序一样的标准用户界面和操作方式，并能够跨不同的浏览器平台使用。**

**2 免费情况**

如果你将EXT用于非营利性目的，就可以在LGPL协议下免费用。如果你将EXT用于营利性目的，就不能再把EXT封装起来当工具库卖，除此之外的领域都可以使用EXT。（注意：如果你使用的是EXT 2.1或更高的版本，开源协议变改成GPL协议。）

**3 ext优化：**

**@不要动不动就ext-all.js，extjs是可以定制的**，如果你用不到tree，就不要包含这个组件了。ExtJs的官网上有详细的定制方法和工具。

**@用的时候再加载，动态加载**，参考使用ExtJs开发MIS系统：Js的动态加载。

**@用Gzip进行js的超强压缩，**解压至某个目录，会看到有一个Gzip.exe文件，然后在命令窗口进入该目录，执行

gzip ext-all.js

ext-all.js马上变成为ext-all.js.gz，大小从原来600多k摇身一变成了160多k

把ext.all.js.gz文件名改为ext.all.gzjs，Filter等一下就会拦截这种文件的访问。

<script type="text/javascript" src="<%=request.getContextPath()%>/ext3/ext-all.gzjs"></script>

<filter>

<filter-name>GzipJsFilter</filter-name>

<filter-class>com.htsoft.core.web.filter.GzipJsFilter</filter-class>

<init-param>

<param-name>headers</param-name>

<param-value>Content-Encoding=gzip</param-value>

</init-param>

</filter>

<filter-mapping>

<filter-name>GzipJsFilter</filter-name>

<url-pattern>\*.gzjs</url-pattern>

<filter-mapping>

<servlet-mapping>

## 二、基础知识

**1 要开始Ext之旅的第一步就是要获得开发包，**

我们可以从官方网站里进行下载: http://www.sencha.com/products/extjs/download

builds目录为ExtJS压缩后的代码

docs目录为ExtJS的文档

examples目录中是官方的演示示例

locale是多国语言的资源文件,其中ext-lang-zh\_CN.js是简体中文

overview是ExtJS的功能概述

packages中是ExtJS各部分功能的打包文件

resource中书ExtJS要用到的图片文件与样式表文件

src目录是未压缩的源码

bootstrap.js是ExtJS库的引导文件,通过参数可以自动切换ext-all.js和ext-debug.js

ext-all.js文件是ExtJS的核心库,是必须要引入的

ext-all-debug.js文件是-all.extjs的调试版,在调试的时候可能需要使用

**2 Ext.Component**

Ext中所有的可视化组件都继承自Ext.Component,这种单继承的模型保证所有组件都拥有相同的通用方法和生命周期,也保证了布局时的便利组件常用的几项功能包含initComponent(),render(),show()与hide(),destroy()无论哪种组件,都是通过初始化,渲染,显示,隐藏来实现生命周期的所有Ext组件的基类,组件下所有的子类都可能参与自动化Ext组件的生命周期执行创建, 渲染和销毁都是由Container容器类提供,组件可以通过容器配置items创建, 也可以通过动态方法 add创建。

**3 Ext.Container**

继承Ext.Component,它提供了两个重要的参数,layout和items,layout参数指定当前组件使用何种布局,items参数(数据类型)中包含的是当前组件中的所有子组件.此处不得不提的一个重要概念是xtype,这个参数来简化配置和延迟布局中组件的初始化,我们在布局中添加几个组件,原始的方法是必须先把这些组件创建好,再加入布局容器中用xtype方式会更加的简单。

**4 所有有效的xtypes列表**

xtype Class------------- ------------------

button Ext.button.Button

buttongroup Ext.container.ButtonGroup

colorpalette Ext.picker.Color

component Ext.Component

container Ext.container.Container

cycle Ext.button.Cycle

dataview Ext.view.View

datepicker Ext.picker.Date

editor Ext.Editor

editorgrid Ext.grid.plugin.Editing

grid Ext.grid.Panel

multislider Ext.slider.Multi

panel Ext.panel.Panel

progressbar Ext.ProgressBar

slider Ext.slider.Single

splitbutton Ext.button.Split

tabpanel Ext.tab.Panel

treepanel Ext.tree.Panel

viewport Ext.container.Viewport

window Ext.window.Window

Toolbar components-------------------------------

pagingtoolbar Ext.toolbar.Paging

toolbar Ext.toolbar.Toolbar

tbfill Ext.toolbar.Fill

tbitem Ext.toolbar.Item

tbseparator Ext.toolbar.Separator

tbspacer Ext.toolbar.Spacer

tbtext Ext.toolbar.TextItem

Menu components---------------------------------------

menu Ext.menu.Menu

menucheckitem Ext.menu.CheckItem

menuitem Ext.menu.Item

menuseparator Ext.menu.Separator

menutextitem Ext.menu.Item

Form components---------------------------------------

form Ext.form.Panel

checkbox Ext.form.field.Checkbox

combo Ext.form.field.ComboBox

datefield Ext.form.field.Date

displayfield Ext.form.field.Display

field Ext.form.field.Base

fieldset Ext.form.FieldSet

hidden Ext.form.field.Hidden

htmleditor Ext.form.field.HtmlEditor

label Ext.form.Label

numberfield Ext.form.field.Number

radio Ext.form.field.Radio

radiogroup Ext.form.RadioGroup

textarea Ext.form.field.TextArea

textfield Ext.form.field.Text

timefield Ext.form.field.Time

trigger Ext.form.field.Trigger

Chart components---------------------------------------

chart Ext.chart.Chart

barchart Ext.chart.series.Bar

columnchart Ext.chart.series.Column

linechart Ext.chart.series.Line

piechart Ext.chart.series.Pie

**5 事件机制**

在ext中,时间分为两种类型,自定义事件和浏览器事件,所有继承ext.util.observable类的控件都可以支持事件,然后为这些继承observable的对象定义一些事件,然后为这些事件配置监听器,当某个事件被触发时,ext会自动调用对应的监听器,这就是ext的事件模型,代码清单

自定义事件

person=function(name){

this.name=name;

this.addEvents(“walk”,”eat”,”sleep”);

}

Ext.extend(person,Ext.util.Observable,{

info:function(event){

return this.name

}

})

itemclick:选项的单击事件 单击选项时触发。

beforeexpand:在Panel被展开之前触发.监听器可以返回false取消展开操作。

beforeload:在一个新数据对象请求发出前触发此事件

**6 EXT mvc**

**1），每个应用都有一个实体，就是Application对象**，而每个应用同样采用单一入口结构，有个快捷函数就是Ext.application({config})，创建一个Application对象实例，并且运行它。Application在创建之初，会去加载Controller类

**2），Application在lunch的时候，**会创建一个Viewport实例，这个东西就像一个骨架一样，上面可以拼装各种View，具体说，就是各种布局形式和窗体控件，可以说是view界面的载体，一个页面只能有一个Viewport实例。

**3），View纯粹是一个界面组件，**或者说窗体控件的集合（比如form,grid和window）。它其实就是利用窗体控件panel,grid或form等进行用户界面展示,表格可以通过Ext.grid.GridPanel的getView()函数获取当前表格使用的视图实例,当我们希望在创建Gridview初始化一些参数可以使用Ext.grid.GridPanel的viewConfig参数,具体属性可以查看api.

通过Store来加载数据并且展现到界面上，界面控件的响应都写在Controller里面，View对Controller的存在全无所知，也没有代码上的依赖。

**4），Controller的角色完全是个粘合剂**，它在加载之初，会帮忙加载跟其有关的Model，Store，View类，而其真正的作用，是通过一系列的事件处理函数(比如点击保存按钮)，确定了每个View上面界面组件对用户交互行为的响应方法，可以说是一堆事件处理器函数的集合；这里面主要通过一个control成员函数来进行事件绑定，通过另一个叫ComponentQuery的组件，使用类似css selector的语法来定位界面上的组件，并为其绑定事件处理器。

**5），Model是对抽象数据的具体化，**简单理解就是数据库里面的一行记录。

**6），Store是对通过网络加载数据的过程的一个抽象，**Store通过data发送请求(一般为ajax请求)到后台获取数据(一般返回json格式)，Store依赖于Model，通过关联的Model对象才知道如何将取回的数据对象化。不管是grid(表格),tree(树),form(表单)都可以通过model格式化字段,这样就可以把后台字段转化为Ext想要的字段,有时需要用到mapping属性

**a）.Model模型：**模型是字段和它们的数据的集合，例如User模型带有username和password字段，模型知道如何持久化自己的数据，并且可以和其他模型关联，模型跟Ext JS 3 中的Record类有点像（区别是，Record只是单纯的扁平结构，而Model可以nest），通常都用在Store中去展示grid和其他组件的数据。

Store就是多个model实例的集合。这里的model实例相当于 Java中的对象，Model就相当于Class

Store如果看成是 一张数据库表，

record(s)就是 表中的记录 （行），

Model 代表（封装）了表的字段信息（属性）。

Ext.define('FWY.store.Students', {

extend: 'Ext.data.Store',

fields: ['id','name', 'age','sex'],

data: [

{id:1,name: '张三', age: 30,sex:'男'},

{id:2,name: '李四', age: 20,sex:'女'}

]

});

Ext.define('FWY.model.Student', {

extend: 'Ext.data.Model',

fields: ['id','name','age','sex']

});

**b）.View视图：**视图是组件的一种，专注于界面展示 – grid, tree, panel 都是view。

Ext.define('FWY.view.student.List' ,{

extend: 'Ext.grid.Panel',

alias : 'widget.studentlist',

title : '学生信息列表',

initComponent: function() {

this.store = {

fields: ['id','name', 'age','sex'],

data : [

{id:1,name: 'zhangsan', age: 18,sex:'boy'},

{id:2,name: 'lishi', age: 20,sex:'girl'}

]};

this.columns = [

{header: '编号', dataIndex: 'id', flex: 1},

{header: '姓名', dataIndex: 'name', flex: 1},

{header: '年龄', dataIndex: 'age', flex: 1},

{header: '性别', dataIndex: 'sex', flex: 1}

];

this.callParent(arguments);

}

});

**c）.Controllers控制器：**一个安放所有使你的app正确工作的代码的位置，具体一点应该是所有动作，例如如何渲染view，如何初始化model，和app的其他逻辑。

ExtJS 4 应用都遵循一个统一的目录结构，每个应有都相同 控制器是应用的粘合剂，它们所作的事情就是监听事件并执行动作，继续我们的应用，创建一个控制器。创建app/controller/Students.js这个文件，并添加如下代码Ext.define('FWY.controller.Students', {

extend: 'Ext.app.Controller',

views:

[

'student.List',

'student.Edit'

],

stores: ['Students'],//加载store

model: 'FWY.model.Student',

init: function() {

this.control({

'viewport > panel': {

render: this.onPanelRendered

}

});

},

editStudent: function(grid, record) {

console.log('Double clicked on ' + record.get('name'));

}

});

MVC中，所有类都放在app目录里面，这个目录可以有子目录，代表的是命名空间（一个子目录对应一个命名空间），使用不同的目录存放views,models,controllers,stores。当我们完成例子的时候，目录结构应该和下图一样

**7 通过Ext.Loader开启动态加载功能**

Ext4.x版本提供的一大亮点就是Ext.Loader这个类的动态加载机制！只要遵循路径规范，即可动态加载js文件，方便把自己扩展组件动态加载进来，并且减轻浏览器的压力通过Loader对象我们可以开启动态加载的依赖加载功能，我们一般通过它的setConfig方法开启动态加载，通过setPath设置加载路径，然后通过Ext.require方法加载相关类库。代码如下。

/\*\*

开启动态加载的依赖加载功能和设置插件的路径，并加载相关类库

\*\*/

Ext.Loader.setConfig({ enabled: true });//开启动态加载的依赖加载功能，默认为false不开启

Ext.Loader.setPath('Ext.ux', '/Scripts/ux');//设置命名空间"Ext.ux"的路径

Ext.require(['Ext.ux.CheckColumn', //加载相关类库和它的直接依赖

'Ext.ux.grid.FiltersFeature',

'Ext.ux.RowExpander',

'Ext.ux.grid.SPrint']);

**8 常用的组件**

@Ext.Panel

Ext.Panel是Ext经常使用的一个组件,它直接继承自Ext.Container,Ext.Panel无须继承就可以直接使用,使用collapseFirst,collapsed,collapsedCls和collapsible设置与面板折叠相关的配置,除此之外,还可以使用floating和shadow设置浮动阴影效果,以及使用html直接设置面板内容.

@表格控件

Ext表格功能非常的强大,包括排序,缓存,拖动,隐藏某一列,自动显示行号,,单元格编辑等实用功能.表格由Ext.grid.Gridpanel定义,继承于Ext.panel,其xtype为grid,在ext中,表格控件必须包含列定义信息,并指定表格的数据存储器,表格的列信息由数组columns定义,而表格的数据存储器由Ext.data.Store定义表格包含的列由columns配置属性来描述,columns是一个数组,每一行数据元素描述表格的列信息,表格的列信息包含首部显示文本(header),列对应的记录集字段(dataIndex),列是否可排序(sortable),列的渲染函数(renderer),宽度(width),格式化信息(format)等,主要用的是显示文本和记录集字段

**9 表单**

Ext.form.FormPanel继承自Ext.Panel,可以对其执行各种panel的操作,实际上,表单的功能是Ext.form.BasicForm中实现的,在获取Ext.form.FormPanel之后,随时可以使用getForm()方法获得BasicForm对象,我们可以在得到的BasicForm上执行”提交表单数据”和”复位表单初始值”等操作,Ext.form.FormPanel最大的好处就是利于布局,Ext.form.FormPanel继承了Ext.Panel,我们可以把Ext.form.FormPanel放在Ext.Viewport中作为整个页面布局的一部分,同时也可以利用items指定Ext.form.FormPanel内部的子组件,如果其他panel一样,可以通过xtype来指定每个子组件的类型

**10 表单布局**

默认布局

表单中的标签默认使用左对齐的方式,一共有left,top,right,通过配置lableAlign:’right’,我们可以使用右对齐标签,标签文字的宽度也可以使用labelWidth设置,formPanel的默认布局是layout:’form’(自上而下布局)平行分列布局. 我们使用layout=’column’来说明使用的是列布局,然后在items指定的每列中使用columnWidth指定每列所占总宽度的百分比,如果使用列布局,不能使用defaultType只能默认的xtype了,否则会影响布局,在每一列中,我们也手动指定layout:’form’这样才能在每列中正常显示输入框和对应的标签

**11 数据校验**

1.输入不能为空:allowBlank:false

2.最大长度和最小长度:maxLength ,minLength

3.借助vtype,属性值alpha:只能输入英文

alphanum:只能输入字母和数字

email:电子邮箱

url:网址

4.自定义校验规则,这种规则其实就是应许自定义正则表达式,

regex:正则表达式,regexText:提示语

**12 布局**

5.1 fit布局

它自动适应页面大小,适应页面大小变化,页面变大表格变大,页面变小表格也变小,缺点是每次只能使用一个子组件,

5.2 border布局

它将整个布局区域划分了东西南北中五个部分,

5.3 accordion布局

伸缩菜单,展开折叠分组

5.4 card布局

Card布局：这种布局用来管理多个子组件，并且在任何时刻只能显示一个子组件。这种布局最常用的情况是向导模式，也就是我们所说的分布提交。比如上一步,下一步

5.5 anchor布局

anchor布局将使组件固定于父容器的某一个位置，既可以为items中的每个组件指定与总体布局大小的差值,也可以设置一个比例使子组件可以根据整体自行计算本身的大小,即父容器容器的大小发生变化时，使用anchor布局的组件会根据规定的规则重新渲染位置和大小,是一个比较灵活的布局

5.6 absolute布局

Absolute布局继承Ext.layout.container.Anchor布局方式，并增加了X/Y配置选项对子组件进行定位，Absolute布局的目的是为了扩展布局的属性，使得布局更容易使用

5.7 form布局

form布局也是anchor的一个子类,是表单专用布局,可以在它里面使用anchor设置宽和高的比例,但主要还是布局表单

5.8 column布局

它将整个容器进行竖直切分的布局方式,一般被称为列布局，这种布局的目的是为了创建一个多列的格式

5.9 table布局

一般被border布局和column布局代替

5.10 HBox布局

水平布局,它可以实现一行中排列多个组件的功能,实现组件的平行排列,column显得过于复杂.

5.11 VBox布局

竖直布局,用法和HBox用法类似

**13 实用工具**

6.1Ext提供的常用函数

6.1.1 onReady()函数

这个方法是Ext的准备函数，也就是Ext相关的代码都会在这个函数里书写，它比较类似于window的onload方法，但是注意其执行时机是在页面的DOM对象加载完毕之后立即执行。

6.2.2 get()函数

用来获取一个Ext元素,也就是类型为Ext.Element的对象,Ext.Element类是对Ext对DOM的封装,每个Element对象都对应着HTML中的一个DOM元素,我们可以为每一个DOM创建一个对应的Element对象,并通过Element对象中的函数实现Dom的指定操作,例如可以使用hide()函数隐藏元素get函数是Ext.Element.get的简写形式,参数可以是DOM节点的id,也可以是一个element,或者是一个DOM节点对象

6.3.3 getCmp()函数

该函数用来获得一个Ext组件,也就是已经在页面中被初始化的Component或者其子类的对象,getCmp()函数可以根据指定的id获得对应的Ext.Component,实际上Ext.getCmp()是对Ext.ComponentMgr.get()的简写形式,Ext每个组件都会注册到ComponentMgr中,只要得到他的id,就可以得到对应的组件

6.4.4 getDom()函数

该函数可以获得文档中的Dom节点,他只包含一个参数,该参数可以是Dom节点的id,Dom节点的对象,或者Dom节点对应的Ext元素,Ext.getDom()可以看做是Ext.get().dom的等同形式,

6.5.5 getBody()函数

该函数直接获取文档中与document.body这个Dom节点对应的Ext元素,实际上就是把document.body对象封装成Ext元素对象并作为结果返回,没有参数

6.6.6 getDoc()函数

getDoc可以将当前HTML文档对象(也就是document对象)封装成Ext的element对象并返回,没有参数,

6.7.7 encode()函数与decode()函数

encode()和decode()函数是专门用来对json数据进行编码和解码的函数,

encode()函数的作用是对对想,数组,或者其他值进行编码,将对象转换为json字符串的形式,目的是把javascript中的对象通过HTTP协议发送到后台服务器并进行相应的处理,因为Http协议只能发送字符串形式的参数,无法将javascript中的对象直接传递到后台,http只能发送iso-8859-1编码的字符,像中文这种无法使用iso-8859-1编码的字符,还需要先转换为iso8859-1编码格式才能通过http传输,

Ext.decode()函数的作用是将字符串解析为JSON对象,在ajax应用中,我们经常让服务器返回json数据,由于json数据是字符串,所以在程序中需要先把它解析成javascript对象,我们利用XmlHttpRequest对象responseText或者responseXML属性获得服务器返回的数据,利用decode进行json转换为javascript对象的操作.

6.8.8 apply()函数与applyIf()函数

Ext.apply()函数的作用是将一个对象中所有属性都复制到另一个对象中,applyIf()的作用与apply的作用类似,区别在于如果某个属性在目标对象中已经存在,则不会覆盖原属性

Ext.apply(属性接受者,属性提供者)

6.9.9 namespace()函数

该函数的作用是将能够传入的参数转换为对象,使用该方法的主要目的是区分名称相同的类,这与Java中package的作用类似

6.10.Ext.each()函数

当需要对数组中每个元素进行同一种操作时,可以利用each()函数,它会迭代循环数组,将每个元素传入预先定义的回调函数中进行处理

6.11 Ext之DOM

6.11.1 Ext.Element（几乎对DOM的一切进行了彻底封装）可以查询Ext.dom.Element

6.11.2 EXt.DomHelper（操控UI界面的工具类）

Element这个类，无疑是非常强大的，里面提供了丰富的方法供我们使用，但是Ext为了更加的方便我们去操作DOM元素，特提供了DomHelper这个辅助的工具类。下面是DomHelper提供的几个常用的函数

DomHelper.append()函数会将生成的节点放到指定节点的最后边

DomHelper.insertBefore()函数会将新生成的节点插入指定节点的前面

DomHelper.insertAfter()函数会将新生成的节点插入指点节点的后边

DomHelper.overwrite()会替换指定节点的insertHTML内容

DomHelper.applyStyles()为指定的DOM元素设置指定的css样式

DomHelper.createTemplate()可以创建一个模板,具体参阅Api

6.11.3 Ext.DomQuery（用来进行DOM节点查询）参阅Ext.dom.Query

下面是此方法的详细使用规则：

基本元素选择器 id选择器css的类选择器 标签选择器（简单选择器）

属性选择器

伪类选择器（也可以说是相当于JQ过滤选择器）

Ext.query基本使用形式：

Ext.query('span') 返回整个文档的span标签

Ext.query('span' , 'root') 根据跟节点进行查询

Ext.query('#id') 根据id进行查询，但返回数组

Ext.query('.class') 根据样式进行查询，返回数组

Ext.query('div span') 根据标签进行包含选择器过滤

Ext.query('\*') 匹配所有元素

Ext.query('input[value\*=val]') 进行一个属性的选择匹配

Ext.query('E>F') 进行一个层次查找父节点为E的F节点

## 三、知识总结

**1 优点**

<1>.UI组件丰富，外观漂亮。

ExtJS库有着丰富且漂亮的UI组件，大大缩短了我们的开发周期，而且组件拥有漂亮的布局，经过简单的调用与配置就可以实现不错的界面布局。

ExtJS提供的各种组件可以用更加标准的方式展示数据降低了开发难度。

<2>.浏览器兼容性好。

使用ExtJS对浏览器没有任何要求。可以说是一种绿色的富客户端实现方式，ExtJs基本可以运行于现在主流的浏览器。

<3>.有很多动画效果做得很不错，提高了用户的感知度。

<4>.和后台代码无关。

不管后台用什么语言开发的都不会受影响，不管你是用C#也好 JAVA也好 还是PHP都和它没关系。

<5>.将Web程序向桌面系统转化。

ExtJS最大的优势在于它将Web应用程序的操作方式向传统桌面应用程序的操作方式进行转化甚至消除了这种差异，从根本上提高了用户的使用体验，

这是ExtJS应用前景广阔的主要原因。

<6>.相对丰富的文档和示例。

毫无疑问，刚刚接触到ExtJS的人多数都是被它附带的例子和开发文档吸引过去的，它的文档做的确实不错。

**2 缺点**

<1>.体积较大，速度稍慢。

由于使用了大量的UI组件，所以体积较大，导致页面加载速度比较慢。

<2>.收费,好像不免费。

因为它太优秀了，所以从Ext JS 2.0以后的版本都是收费的。也许这一点不能算是它的缺点，但这确实阻碍了它的推广与应用。

<3>.没有合适的开发利器。

毫无疑问，一个好的开发工具可以大大的提高编码的速度，但是对于ExtJS，始终没有一个完美的开发工具，可以推荐的有Aptana Studio， Spket IDE，和Spket 提供的提示文件，但是都是各有优缺点，都不完美，只能一边看SDK一边写代码。

<4>.没有界面设计工具。

虽然有人提供了一个在线的界面设计工具，但是和Visual Studio提供的ASP.Net设计工具来说，真的可以说是天壤之别。因此，只能一边预览，一边写代码。

<5>.文档不全。

虽然ExtJS提供的文档很丰富，但是还是跟不上源代码的更新速度，所以，经常要通过看源代码，调试才能真正解决问题。

<6>.不能编译。

这一点可以说是JavaScript的缺点（如果能编译，就不叫JavaScript了），在实际的开发中，经常会敲错一些代码，比如大小写错误等，不能通过编译得到反馈，

只能在运行时排错，导致开发的效率比较低下。

**3 使用场景**

1） 富客户端，管理系统

2） Extjs有着非常牛逼的UI设计。所以当你做像后台管理，企业内部流程，或其他管理系统的时候，这个东西无疑是你的最佳选择。为什么呢？ 因为它几行代码就可以生成非常漂亮的全界面UI，功能强大，样式可选。 做过一个CMS的后台和一个企业内部的系统都是用这个做的，节省了很多人力和时间，而且广受好评。 而它的缺点是，在一个页面加载很多tab的时候，js可能会要同时下载若干个文件。简单说就是js会比较大。 但对于后台管理和企业内部系统来说，这个会成为瓶颈。

**4 Ext和Jquery比较**

1. 首先根据你的需求，你需要这些框架具有你需要的UI组件，ExtJS、easy ui都具备你需要用到的那几款ui组件，比如messagebox、tree、grid。相比较jQuery UI的ui组件不够全面，比如grid，但是jquery有很多各种插件，包括grid,但它不在jQuery UI内。而Ext JS和easy ui最新版提供的GRID都有20种以上实现不同功能grid。

2）轻重比较，说ExtJS比较大，jquery较小，那只是浅显的理解。首先ExtJS是一个完整的Framework,是重量级别的，easy ui 是基于jquery库的一套UI组件库，是轻量级的，ExtJS是应用application级的，而jquery是page页面级的。当然application也是由page组成的，那就需要你自己去完成了，考虑你的需求，和使用框架的初衷，选择使用哪一种。同时ExtJs由于是重量级框架，完全面向对象风格，提供API非常完备也非常庞大，所以学习成本也想相对较大。

3）兼容性ExtJS兼容IE全系列浏览器和其他非IE现代浏览器，jquery UI向来不太考虑ie低版本浏览器的兼容，从态度上的鄙视。easy UI是基于jquery的，jquery2.X以上的版本不再支持IE6、7、8,，已郑重声明，请看官方网站，easyUI最新版本1.3.3使用jQuery2.0,由于又很多HTML5特性，不再支持IE6\78，低版本由一些小部分的兼容不够好，请自己做技术选型的时候去测试，你要使用那个版本。在兼容问题上，他们都有瑕疵，看你的接收程度。

4）使用许可license. EXTJS 2.1以上版本，商用需要购买商业授权，jquery UI 使用MIT协议，开源。 jquery easyUI如果商用需遵循license commercial商业许可，也就是要购买使用权.

## 四、代码练习

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

<HTML>

<HEAD>

<meta http-equiv="Content-Type" content="text/html; charset=gb2312" />

<TITLE>EXTJS练习</TITLE>

</HEAD>

<link rel="stylesheet" type="text/css" href="extjs/resources/css/ext-all.css"/>

<script language="javascript" src="extjs/adapter/ext/ext-base.js"></script>

<script language="javascript" src="extjs/ext-all.js"></script>

<BODY>

<div style="margin:5px 15px 5px 15px;width:100%;height:2000px;" >

<p> <p/>

<p> <p/>

<table style="width:100%;border-width:1px;border-color:blue;" >

<tr>

<td><input type="button" value="模态窗口" onclick="openWindow()"></td>

<td><input type="button" value="不能关闭的模态窗口" onclick="openWindowNoClose()"></td>

<td><input type="button" value="面 板" onclick="openPanel()"></td>

<td><input type="button" value="column布局" onclick="openPanelLayout()"></td>

<td><input type="button" value="form布局" onclick="openPanelForm()"></td>

</tr>

<tr>

<td><input type="button" value="Accordion布局" onclick="openPanelAccordion()"></td>

<td><input type="button" value="Table布局" onclick="openPanelTable()"></td>

<td><input type="button" value="面 板" onclick="openPanel()"></td>

<td><input type="button" value="可编辑表格" onclick="openEditorGridPanel()"></td>

<td> <input type="button" value="树形控件" onclick="openTreePanel()"></td>

</tr>

<tr>

<td><input type="button" value="选项面板" onclick="openTabPanel()"></td>

<td><input type="button" value="点击事件" id="clickId"></td>

<td><input type="button" value="点击事件延迟" id="clickIdDelay"></td>

<td><input type="button" value="viewPort效果" onclick="showViewPort()"></td>

<td><input type="button" value="后台管理系统" onclick="showManagerView()"></td>

</tr>

<tr>

<td><input type="button" value="component属性" onclick="showComponentAttr()"></td>

<td><button id="mb1">确认框弹出来</button></td>

<td><button id="mb2">输入text再确认</button></td>

<td><button id="mb3">输入textArea再确认</button></td>

<td><button id="mb4">Yes/No/Cancel</button></td>

</tr>

<tr>

<td><input type="button" value="component属性" onclick="showComponentAttr()"></td>

<td><button id="mb6"> 进度条</button></td>

<td></td>

<td></td>

<td></td>

</tr>

</table>

<div id="panelDiv"></div><br><br>

<div id="accordionDiv"></div><br><br>

<div id="treeDiv"></div><br><br>

<div id="formDiv"></div> <br><br>

<div id="GridPanelDiv"></div> <br><br>

<div id="EditorGridPanelDiv"></div><br><br>

<div id="TableDiv"></div><br><br>

<div id="compocontDiv"></div><br><br>

<div id="tabPanelDiv"></div>

</div>

<select id="positionDiv">

<option>小前锋</option>

<option>大前锋</option>

<option>中锋</option>

<option>得分后卫</option>

<option>组织后卫</option>

</select>

<script language="javascript">

showComponentAttr();

function showComponentAttr()

{

new Ext.Panel(

{

allowDomMove:true,

renderTo:"compocontDiv",

title:"compocont属性",

width:800,

height:500,

html:"<p style='text-align:center;margin-top:20px;font-weight:700;font-family:arial;color:red;'>大家好，我是总决赛MVP、全明星MVP、最佳阵容一队、全明星技巧大赛冠军、得分王德维恩-韦德！</p>",

tools:

[{ id:"save"},

{id:"help",handler:function (){Ext.MessageBox.alert("help","帮助信息请查看http://www.bangzhu.com");}},

{ id:"close",handler:function (){Ext.MessageBox.alert("close","关闭窗口请查看http://www.guanbi.com");}} ],

tbar:[{text:'顶部工具栏'},{pressed:false,text:'刷新'}],

bbar:[{text:'底部工具栏'}],

buttons:[{text:'按钮1'},{text:'按钮2'}]

}

);

}

Ext.onReady(function ()

{

//Ext.MessageBox.alert("","欢迎光临，韦德先生！");

Ext.get("clickId").addListener("click",function (){alert("监控！");});

Ext.get("clickIdDelay").on("click",function (){alert("监控！");},this,{delay:2000});

Ext.get('mb1').on('click', function(e)

{

Ext.MessageBox.confirm('确认框', '你真的想跳，考虑清楚啦?', function (btn)

{

Ext.MessageBox.alert('',btn);

});

});

});

Ext.get('mb2').on('click', function(e)

{

Ext.MessageBox.prompt('姓名', '请输入姓名:', function (btn,text)

{

Ext.MessageBox.alert('',btn+"："+text);

});

});

Ext.get('mb3').on('click', function(e){

Ext.MessageBox.show({

title: '地址',

msg: '请输入地址:',

width:300,

buttons: Ext.MessageBox.OKCANCEL,

multiline: true,

fn: function (btn,text)

{

Ext.MessageBox.alert('',btn+"："+text);

},

animEl: 'mb3'

});

});

Ext.get('mb4').on('click', function(e){

Ext.MessageBox.show({

title:'保存修改',

msg: '你已经修改了你的生日信息. <br />确认要修改吗?',

buttons: Ext.MessageBox.YESNOCANCEL,

fn: function (btn)

{

Ext.MessageBox.alert('',btn);

},

animEl: 'mb4',

icon: Ext.MessageBox.QUESTION

});

});

Ext.get('mb6').on('click', function(){

Ext.MessageBox.show({

title: '请稍等',

msg: '正在处理...',

progressText: 'Initializing...',

width:300,

progress:true,

closable:false,

animEl: 'mb6'

});

var f = function(v)

{

return function(){

if(v == 12){

Ext.MessageBox.hide();

Ext.MessageBox.alert('','加载已完成');

}else{

var i = v/11;

Ext.MessageBox.updateProgress(i, Math.round(100\*i)+'% completed');

}

};

};

for(var i = 1; i < 13; i++){

setTimeout(f(i), i\*500);

}

});

});

function showViewPort()

{

new Ext.Viewport(

{

enableTabScroll:true,

layout:'fit',

items:[{title:'viewPort',html:'韦德+詹姆斯+波什=总冠军 ',bbar:[{text:'我的待办'},{text:'我的工作'}]}]

});

}

function showManagerView()

{

new Ext.Viewport(

{

enableTabScroll:true,

layout:'border',

items:

[{title:'',region:'north',height:100,html:'<br><br><b style=font-size:20px;color:red;>吉安青山国际机场运营系统</b>',bbar:[{text:'我的待办'},{text:'我的审批'},{text:'我的收藏'},{text:'我的设置'},{text:'注销'}]},

{title:'菜单',region:'west',html:'菜单栏 ',width:200,collapsible:true},

{xtype:'tabpanel',region:'center',items:[{title:'韦德'},{title:'詹姆斯'}]}

]

});

/\*new Ext.Panel(

{

renderTo:"panelDiv",

width:700,

height:800,

enableTabScroll:true,

layout:'border',

items:

[{title:'',region:'north',height:100,html:'<br><br><b style=font-size:20px;color:red;>吉安青山国际机场运营系统</b>',bbar:[{text:'我的待办'},{text:'我的审批'},{text:'我的收藏'},{text:'我的设置'},{text:'注销'}]},

{title:'菜单',region:'west',html:'菜单栏 ',width:200,collapsible:true},

new Ext.TabPanel({region:'center',items:[{title:'韦德',html:'rrrrwadewade+wade'},{title:'詹姆斯',html:'james+sdfsdg'}]}),

{title:'菜单2',region:'east',html:'菜单栏2 ',width:150,collapsible:false}

]

});\*/

}

var winIndex = 0;

function openWindow()

{

var winContext =

{

title:"窗体"+winIndex++,

width:400,

height:300,

maximizable:true,

minimizable:true,

resizable:true,

draggable:true,

closable:true,

html:"<p>大家好，我是MVP韦德！</p>"

};

var win = new Ext.Window(winContext);

win.show();

}

function openWindowNoClose()

{

var winContext = {title:"我是不能关的窗体",width:800,height:500,html:"<p>大家好，我是MVP韦德！</p>"};

var win = new Ext.Window(winContext);

win.on("beforedestroy",function (obj)

{

alert("别费力气了，关不掉的。");

obj.show();

return false;

});

win.show();

}

function openPanel()

{

var panCon =

{

renderTo:"panelDiv",

title:"我是面板",

width:800,

height:500,

html:"<p>大家好，我是MVP韦德！</p>",

tools:

[{ id:"save"},

{id:"help",handler:function (){Ext.MessageBox.alert("help","帮助信息请查看http://www.bangzhu.com");}},

{ id:"close",handler:function (){Ext.MessageBox.alert("close","关闭窗口请查看http://www.guanbi.com");}} ],

tbar:[{text:'顶部工具栏'},{pressed:false,text:'刷新'}],

bbar:[{text:'底部工具栏'}],

buttons:[{text:'按钮1'},{text:'按钮2'}]

};

new Ext.Panel(panCon);

//var panCon = {title:"我是面板",width:800,height:500,html:"<p>大家好，我是MVP韦德！</p>"};

//var pan = new Ext.Panel(panCon);

//pan.render("hello");

}

function openPanelLayout()

{

var panCon =

{

renderTo:"panelDiv",

title:"我是面板",

width:800,

height:500,

//html:"<p>大家好，我是MVP韦德！</p>",

layout:'column',

tools:

[{ id:"save"},

{id:"help",handler:function (){Ext.MessageBox.alert("help","帮助信息请查看http://www.bangzhu.com");}},

{ id:"close",handler:function (){Ext.MessageBox.alert("close","关闭窗口请查看http://www.guanbi.com");}} ],

tbar:[{text:'顶部工具栏'},{pressed:false,text:'刷新'}],

bbar:[{text:'底部工具栏'}],

buttons:[{text:'按钮1'},{text:'按钮2'}],

items:[{title:'1111',html:'1111',columnWidth:0.3},

{title:'2222',html:'2222',columnWidth:0.3},

{title:'3333',html:'3333',columnWidth:0.4}]

};

new Ext.Panel(panCon);

}

function openPanelForm()

{

/\*var panCon =

{

renderTo:"formDiv",

title:"form",

width:800,

height:500,

layout:'form',

labelAlign:'right',

buttons:[{text:'按钮1'},{text:'按钮2'}],

defaultType:'textfield',

items:[{fieldLabel:'姓名',id:'name'},

{fieldLabel:'地址',id:'addr'},

{fieldLabel:'电话',id:'tel'}]

};

new Ext.Panel(panCon);\*/

var panCon =

{

renderTo:"formDiv",

title:"form",

width:800,

height:500,

labelAlign:'right',

buttons:[{text:'按钮1'},{text:'按钮2'}],

defaultType:'textfield',

items:[{fieldLabel:'姓名',id:'name'},

{fieldLabel:'地址',id:'addr'},

{fieldLabel:'电话',id:'tel'}]

};

new Ext.FormPanel(panCon);

}

function openPanelAccordion()

{

var panCon =

{

renderTo:"accordionDiv",

title:"accordion",

width:300,

height:500,

layout:'accordion',

layoutConfig:{animate:true},

buttons:[{text:'按钮1'},{text:'按钮2'}],

items:[{title:'频道管理',html:'name'},

{title:'信息发布管理',html:'addr'},

{title:'应用信息',html:'tel'}]

};

new Ext.Panel(panCon);

}

function openPanelTable()

{

var panCon =

{

renderTo:"TableDiv",

title:"table",

width:600,

height:500,

layout:'table',

layoutConfig:{columns:3},

items:[{title:'1111',html:'1111',rowspan:2,height:100},

{html:'2222',colspan:2,height:100},

{html:'3333',height:100}, {html:'4444',height:100}]

};

new Ext.Panel(panCon);

}

function openGridPanel()

{

/\*

//普通的数据存储

var data = [

[1,'韦德','得分后卫','28.6','http://www.nba.com/players/id=wade'],[2,'杜兰特','小前锋','28.2','http://www.nba.com/players/id=dulat'],

[3,'詹姆斯','小前锋','27.8','http://www.nba.com/players/id=james'],[4,'麦克格雷迪','得分后卫/小前锋','27.6','http://www.nba.com/players/id=mcgrady']];\*/

//var store = new Ext.data.SimpleStore({data:data,fields:['id','name','station','point','url']});

/\*

//xml类型的数据存储

var store = new Ext.data.Store({url:'nba.xml',reader:new Ext.data.XmlReader({recode:'row'},['id','name','station','point','url'])});\*/

//Json类型的数据存储

var data = [

{id:1,name:'韦德',station:'得分后卫',point:'28.6',url:'http://www.nba.com/players/id=wade'},

{id:2,name:'杜兰特',station:'小前锋',point:'28.2',url:'http://www.nba.com/players/id=dulat'},

{id:3,name:'詹姆斯',station:'小前锋',point:'27.8',url:'http://www.nba.com/players/id=james'},

{id:4,name:'麦克格雷迪',station:'得分后卫/小前锋',point:'27.6',url:'http://www.nba.com/players/id=mcgrady'}];

var store = new Ext.data.JsonStore({data:data,fields:['id','name','station','point','url']});

var columnMo = new Ext.grid.ColumnModel([

{header:'排名',dataIndex:'id',sortable:true,width:200,height:50},

{header:'姓名',dataIndex:'name',sortable:true,width:200},

{header:'位置',dataIndex:'station',sortable:true,width:200},

{header:'得分',dataIndex:'point',sortable:true,width:200},

{header:'个人黄页',dataIndex:'url',sortable:true,width:200,renderer:showUrl}

]);

var con =

{

renderTo:"GridPanelDiv",

title:"2011-2012赛季NBA球员评分榜",

width:1020,

height:200,

store:store,

cm:columnMo

};

new Ext.grid.GridPanel(con);

//store.load();

}

function openEditorGridPanel()

{

var data = [

{id:1,name:'韦德',station:'得分后卫',point:'28.6',url:'http://www.nba.com/players/id=wade',birth:'2011-01-05'},

{id:2,name:'杜兰特',station:'小前锋',point:'28.2',url:'http://www.nba.com/players/id=dulat',birth:'2011-11-05'},

{id:3,name:'詹姆斯',station:'小前锋',point:'27.8',url:'http://www.nba.com/players/id=james',birth:'2012-01-05'},

{id:4,name:'麦克格雷迪',station:'得分后卫/小前锋',point:'27.6',url:'http://www.nba.com/players/id=mcgrady',birth:'2001-10-05'}];

var store = new Ext.data.JsonStore({data:data,fields:['id','name','station','point','url',{name:'birth',type:'date',dateFormat:'Y-n-j'}]});

var columnMo = new Ext.grid.ColumnModel([

{header:'排名',dataIndex:'id',sortable:true,width:200,height:50},

{header:'姓名',dataIndex:'name',sortable:true,width:200,editor:new Ext.form.TextField()},

{header:'位置',dataIndex:'station',sortable:true,width:200,editor:new Ext.form.ComboBox(

{

transform:'positionDiv',

triggerAction:'all'

}

)},

{header:'得分',dataIndex:'point',sortable:true,width:200,editor:new Ext.form.TextField()},

{header:'个人黄页',dataIndex:'url',sortable:true,width:200,renderer:showUrl,editor:new Ext.form.TextField()},

{header:'生日',dataIndex:'birth',sortable:true,width:200,renderer:Ext.util.Format.dateRenderer('Y年m月d日'),editor:new Ext.form.DateField({format:'Y年m月d日'})}

]);

var con =

{

renderTo:"EditorGridPanelDiv",

title:"2011-2012赛季NBA球员评分榜",

width:1220,

height:200,

store:store,

cm:columnMo,

clicksToEdit:1

};

new Ext.grid.EditorGridPanel(con);

}

function showUrl(value)

{

return "<a href="+value+" target='\_blank'>"+value+"</a>";

}

function openTabPanel()

{

var tabJson ={ renderTo:"tabPanelDiv",title:"我是面板",width:800,height:500,items:[{title:'面板1',html:'<p>大家好，我是面板1！</p>'},

{title:'面板2',html:'<p>大家好，我是面板2！</p>'},{title:'面板3',html:'<p>大家好，我是面板3！</p>'}]};

new Ext.TabPanel(tabJson);

}

function openTreePanel()

{

var root1 = new Ext.tree.TreeNode({id:'广东',text:'广东'});

var shenzhen = new Ext.tree.TreeNode({id:'深圳',text:'深圳'});

shenzhen.appendChild(new Ext.tree.TreeNode({id:'宝安区',text:'宝安区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

shenzhen.appendChild(new Ext.tree.TreeNode({id:'南山区',text:'南山区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

shenzhen.appendChild(new Ext.tree.TreeNode({id:'福田区',text:'福田区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

shenzhen.appendChild(new Ext.tree.TreeNode({id:'罗湖区',text:'罗湖区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

shenzhen.appendChild(new Ext.tree.TreeNode({id:'盐田区',text:'盐田区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

shenzhen.appendChild(new Ext.tree.TreeNode({id:'龙岗区',text:'龙岗区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

root1.appendChild(shenzhen);

var guangzhou = new Ext.tree.TreeNode({id:'广州',text:'广州'});

guangzhou.appendChild(new Ext.tree.TreeNode({id:'天河区',text:'天河区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

guangzhou.appendChild(new Ext.tree.TreeNode({id:'越秀区',text:'越秀区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

guangzhou.appendChild(new Ext.tree.TreeNode({id:'白云区',text:'白云区',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

root1.appendChild(guangzhou);

root1.appendChild(new Ext.tree.TreeNode({id:'东莞',text:'东莞',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

root1.appendChild(new Ext.tree.TreeNode({id:'佛山',text:'佛山',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

root1.appendChild(new Ext.tree.TreeNode({id:'珠海',text:'珠海',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

root1.appendChild(new Ext.tree.TreeNode({id:'湛江',text:'湛江',href:'http://www.guangdong.com/id=shenzhen-baoan',hrefTarget:'\_blank'}));

var tree = new Ext.tree.TreePanel(

{

renderTo:'treeDiv',

autoScroll:true,

animate:true,

root:root1,

width:300

});

tree.on('click',function(node,event)

{

alert(node.id);

});

/\*加载url的树形菜单

var loader = new Ext.tree.TreeLoader({url:'http://w3-beta.huawei.com/w3admin/infohome.do?pcatalog=catalog'});

var root = new Ext.tree.TreeNode({id:'广东',text:'广东',loader:loader});

var tree = new Ext.tree.TreePanel(

{

renderTo:'treeDiv',

autoScroll:true,

animate:true,

root:root1,

width:300

});

\*/

}

/\*

//获取id为div1的第一个元素

var c1 = Ext.get("wade1");

//获取id为div1的元素集合

var c3 = Ext.select("#wade1");

//获取title=wade的Ext.CompositeElement类型集合

var c4 = Ext.select("#div[title=wade]");

//获取title=wade的dom类型集合

var c5 = Ext.query("#div[title=wade]");

//获得title属性值

var e1 = div6.first();

var c7 = e1.getAttribute("title");

//设置title属性值

e1.set('title','NewTitle');

//增加样式

c6.addCls("class1");

//删除样式

c6.removeCls("class1");

//是否存在样式

c6.hasCls("class1");

//获取html值

var c7 = e1.dom.innerHTML;

//修改html的值

e1.update("<b>更醒后的</b>");

//获得元素的value值

var c8 = e1.getValue();

//设置元素的value值

e1.set({value: 150 });

//查找class为c1的元素集合

var c9 = Ext.select(".c1");

//筛选第2个元素返回

var c10 = c9.item(2);

//筛选第一个和最后一个元素返回

var c10 = c9.first();

var c10 = c9.last();

//进一步过滤

var c10 = c9.filter("div[title=t1]");

//e元素结束之前插入

var e = Ext.select("div .c1");

var ee = e.first();

ee.insertHtml("beforeEnd", "<b>新内容</b>");

//e元素结束之后插入

ee.insertHtml("afterEnd", "<b>新内容</b>");

//e元素开始之前插入

ee.insertHtml("beforeBegin", "<b>新内容</b>");

//e元素开始之后插入

ee.insertHtml("afterBegin", "<b>新内容</b>");

//移除集合中第三个元素

e.removeElement(2, true);

//删除

e.first().remove();

//设置样式

e.setStyle("width", "550px");

e.applyStyles

({

height: "500px",

color: "red",

position: "absolute"

});

//设置高度，带动画效果

e.setHeight(100, true);

//设置定位

e.setLeft("50px");

e.setTop("10px");

e.setLeftTop("100px", "50px");

//获取宽度

e.getWidth();

//给元素绑定click事件：

var clickhandler = function ()

{

Ext.Msg.alert("消息", "click事件被触发！");

};

e.on("click", clickhandler);

//解除绑定click事件

e.un("click", clickhandler);

//hover事件

e.hover(function () {

e.setStyle("background-color", "Red");

}, function () {

e.setStyle("background-color", "Aqua");

});

常用的组件或控件----------------------------------------------------------------------------------------

对话框、窗口、面板、多选项面板、viewPort面板、事件处理、layout布局、表格、可编辑表格、树形菜单

layout布局----form,fit,table,border,accordion,column,card

\*/

</script>

</BODY>

</HTML>