# ExtJs 4.X使用详细指南

# console.dir(对象)打印对象

# 弹窗警告

**var** warning\_msg = **function**(title, msg) {

Ext.Msg.show({

title : title,

msg : msg,

y:100,

width : 280,

height:120,

modal : **true**,

icon : Ext.Msg.WARNING,

buttons : Ext.Msg.OK

});

};

**var** error\_msg = **function**(title, msg) {

Ext.Msg.show({

title : title,

msg : msg,

width : 200,

modal : **true**,

icon : Ext.Msg.ERROR,

buttons : Ext.Msg.OK

});

};

# 获取页面可用高度

var availHeight=document.body.clientHeight;

var innerHeight=window.innerHeight; //页面可用高度

# 键盘输入事件监听

|  |
| --- |
| window.onkeyup= **function**(event){  **if**(event.keyCode==13)  //获取查询按钮的dom元素，触发单击事件  Ext.get('query\_button\_id').dom.click();  } |

# 设置轮询执行函数[间隔函数]

|  |
| --- |
| function file\_fun(){…}  window.setInterval(file\_fun, 50) |

# Iframe 用法

把iframe解释成“浏览器中的浏览器“很是恰当

<iframe frameborder=0 width=170 height=100 marginheight=0 marginwidth=0 scrolling=no src=[http://www.163.com](http://www.163.com/)></iframe>

<IFRAME>用于设置文本或图形的浮动图文框或容器。

# Ext4.2

修改button按钮文字

## 修改button文字

**var** button1=***Ext***.*create*(**'Ext.Button'**, {  
 **id**:**'expor'**,  
 **text**: **'导出'**,  
 **width**:100,  
 **tooltip** : **'导出前告警'**,  
 **iconCls** : **'execl'**,  
 **ref** : **'../updateButton'**,  
 handler : **function**() {  
 **if** (***stores***.getCount() >= 1){  
 **if** (***Ext***.**MessageBox**) {  
 ***Ext***.**MessageBox**.**buttonText** = {  
 **ok** : **"确定"**,  
 **cancel** : **"取消"**,  
 **yes** : **"导出当前标题列"**,  
 **no** : **"导出所有标题列"** };  
 }  
 ***Ext***.**Msg**.show({  
 **title** : **'导出'**,  
 **msg** : **'是否导出当前告警列表'**,  
 **buttons** : ***Ext***.**Msg**.**YESNOCANCEL**,  
 **fn** : *save*,  
 **icon** : ***Ext***.**MessageBox**.**QUESTION** });  
 } **else**{  
 ***Ext***.**Msg**.alert(**"提示"**, **"当前表没有记录！"**);  
 }  
  
 }  
  
})

## Ext.grid.Panel

### 操作项

|  |
| --- |
| mainPageColumns.push({  header: "操&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;作",  width: 100,  xtype: "actioncolumn",  align:'center',  items: [{  icon:"image/edit.png" ,  tooltip:'修&nbsp;改',  handler: toShowWindowUpdateScene  },{width:20},{  icon:"image/remove.png" ,  tooltip:'删&nbsp;&nbsp;除',  handler: deleteScene  }]  }); |

### 根据行号取一条记录

var rec = mainPageGridStore.getAt(rowIndex);

### store配置

|  |
| --- |
| Ext.define('AlarmInfo', {  extend: 'Ext.data.Model',  fields: [  {name: 'ERRORDESC'}  ]  });  **var** alarmInfoGridStore = Ext.create('Ext.data.Store', {  storeId:'alarmInfoGridStore',  model: 'AlarmInfo',  pageSize:30,  remoteSort : **true**,  sorters:{ property :'EXECTIME',  direction:'DESC'},  proxy: {  type: 'ajax',  url: 'services/rest/alarmInfoQueryService/queryAlarInfo',  reader: {  type: 'json',  root: 'root',  totalProperty:'total'  },  extraParams:{  startTime:Ext.util.Format.date(**new** Date(),'Y-m-d'),  endTime:Ext.util.Format.date(**new** Date(),'Y-m-d')  }  },  autoLoad: **true**  });  Ext.create('Ext.grid.Panel', {  anchor:'100% 92%',  header:false,//表头不显示  loadMask:**true**,//checkbox  selModel:Ext.create('Ext.selection.CheckboxModel',{listeners:{  deselect:function( RowModel, record,index,eOpts ){  var params = netDetailPanelGrid.getSelectionModel().getSelection();  if(params.length==0){  Ext.getCmp('down').setDisabled(true);  } },select:function(){Ext.getCmp('delete\_button\_id').setDisabled(f alse);}  }}),  store: Ext.data.StoreManager.lookup('alarmInfoGridStore'),  viewConfig:{emptyText:'没有查询数据!'},  columns: [  { xtype:'rownumberer',text:'序号',width:40,align:'left'},  { header: '告警明细', dataIndex: 'ERRORDESC',flex:1,sortable:**false**}  ],  dockedItems: [{  xtype: 'pagingtoolbar',  store:Ext.data.StoreManager.lookup('alarmInfoGridStore'), dock: 'bottom',  displayInfo: **true**  }],  listeners:{  //解决 emptyText:'没有查询数据!' 没有数据，不能显示问题  afterLayout:function(){  netDetailPanelGrid.getStore().clearData();  netDetailPanelGrid.getStore().reload()  }  }  }); |

### store操作

Ext.getCmp('time\_template\_id').setValue(store.getAt(0));//取第一条数据

**for**(**var** i **in** store.data.map){

**var** menu\_id=store.data.map[i].data.menu\_id;

**var** menu\_display=store.data.map[i].data.menu\_display

**for**(**var** j=0;j<firtPage.length;j++){

**if**(menu\_display==firtPage[j]){

newStoreData.push({menu\_id:menu\_id,menu\_display:menu\_display});

}

}

}

### 配置项viewConfig:Object设置行背景色

|  |
| --- |
| viewConfig : {  forceFit : **true**,  emptyText : '没有匹配的数据！',  getRowClass: **function**(record, rowIndex, rowParams, store){  **if**(rowIndex%2==0) {  **return** '*child-row*';  }**else**{  **return** '*adult-row*';  }  }  }  jsp样式配置：  *.child-row* *.x-grid-cell* { background-color: *#ffff00* ; color: *#000000*; }  *.adult-row* *.x-grid-cell* { background-color: *#ECF1F7* ; color: *#000000*; } |

### renderer渲染函数的使用

|  |
| --- |
| renderer:**function**(value, metaData, record, rowIndex, columnIndex, store){  **var** value\_1 = record.get('TX\_VALUE\_1');  **var** value\_2 = record.get('TX\_VALUE\_2');  metaData.tdAttr = 'data-qtip="' + Ext.String.htmlEncode(value) + '"';//增加提示列值  **return** (value\_1-value\_2).toFixed(2);//保留两位小数  } |

### 改变具体列背景色

|  |
| --- |
| *//两个类样式名字间不能有空格*  *.changecellstyle.x-grid-cell*{background-color: *yellow*;}  renderer:**function**(data, metaData) {  metaData.tdCls="changecellstyle";//设置列样式  **return** data.toFixed(2);  } |

### store数据查询

load()方法是重新查询；只会把当前页面带到后台

reload()不会把页码初始化；注意，但会把页面从1-100带到后台

|  |
| --- |
| **var** store = Ext.data.StoreManager.lookup('alarmInfoGridStore');    **var** baseParams = {  startTime:startTime,  endTime:endTime  }  Ext.apply(store.proxy.extraParams,baseParams);//绑定参数  store.currentPage = 1;  store.load();  store.first().get('id') //获取第一条数据 |

### 动态添加删除列,动态设置表头

|  |
| --- |
| var colums=Ext.getCmp('grid\_id').getView().getGridColumns();  colums[2].setText('片区');  reconfigure(store,colums) //reconfigure()重新配置列；  Ext.getCmp('grid\_id').reconfigure(store,mainPageColumns); |

## 表单Ext.form.Panel

### Ext.form.field.ComboBox 单选下拉框

### checkboxgroup 复选框

{xtype: 'checkboxgroup',

fieldLabel: '&nbsp;回复权限',

columns:3,//3列

items: [

{

type:'checkboxfield',

boxLabel: '可回复',

name: 'replyPermission',

inputValue: '3'

}]}

### checkbox

{ xtype: 'checkboxfield',

id:'211',

name: 'replyPermission',

checked:**false**,

fieldLabel: '意见反馈',

boxLabel: '可回复',

inputValue:'1'

}

var chkEditDomain = new Ext.form.CheckboxGroup({  
 xtype: 'checkboxgroup',  
 fieldLabel: '测试',  
 name: 'chkEditDomain',  
 allowBlank: false,  
 id: 'itemsid0',  
 width: 200,  
 columns: 4,  
 items: [  
 { boxLabel: 'IND', name: 'IND', id: '211', inputValue: '5' },  
 { boxLabel: 'CE', name: 'CE', id: '212', inputValue: '6' },

设置默认选中：

Ext.getCmp("itemsid0").setValue("211", true);  
// Ext.getCmp("itemsid0").setValue("212", true);  
// Ext.getCmp("itemsid0").setValue("213", true);  
// Ext.getCmp("itemsid0").setValue("214", true);

另一种写法：

Ext.getCmp("itemsid0").setValue({  
 '211':true,  
 '212':true,   
 '213': false,  
 '214': true  
 });

### 多选下拉框

jsp引入文件

<script type="text/javascript" src="/mapv/plugin/ext-4.2/ux/form/multiComboBox.js"></script>

{

xtype:'multicombobox',

id:'index\_desc\_id',

width:300,

fieldLabel:"指标类型",

name: "index\_desc",

emptyText:'指标类型',

allowBlank: **true**,

editable : **false**,

style:'marginLeft:20px',

store:**new** Ext.data.JsonStore({

fields: [{name:'id',mapping:'INDEX\_DESC'}, {name:'index\_desc',mapping:'INDEX\_DESC'}],

autoLoad:**true**,

proxy: {

type: 'ajax',

url:'/mapv/currentAlarm/queryIndexType.do',

reader: {

type: 'json'

}

},listeners:{

// load:function(store){

// store.insert(0,{id:'',logname:'全部'});

// }

}

}),

queryMode: 'local',

displayField: 'index\_desc',

valueField: 'id',

value:''

}

取值：

**var** index\_desc='';

**var** index\_desc\_id = Ext.getCmp('index\_desc\_id').getValue();

**for**(**var** i=0;i<index\_desc\_id.length;i++){

index\_desc+= '\''+index\_desc\_id[i]+'\''+(index\_desc\_id.length-1==i?'':',');

}

### 多选下拉框

|  |
| --- |
| {xtype:'multicombobox',id:'modify\_work\_order\_permission\_id',fieldLabel:"工单权限",name: "work\_order",emptyText:'工单权限',allowBlank: true,editable : false,style:'marginLeft:20px',store:new Ext.data.JsonStore({fields: [{name:'work\_id',mapping:'WORK\_ID'}, {name:'work\_name',mapping:'WORK\_NAME'}],autoLoad:true,proxy: {type: 'ajax',url:'/mapv/roleManage/getWorkOrderPermission.do',reader: {type: 'json'}}}),queryMode: 'local',displayField: 'work\_name',valueField: 'work\_id',value:''}]}设置值var word\_order=[]Ext.getCmp('modify\_work\_order\_permission\_id').setValue(word\_order);取值var work\_order = Ext.getCmp('modify\_work\_order\_permission\_id').getValue();//工单权限try{for(var i=0;i<work\_order.length;i++){work\_order\_permission+=work\_order[i]+(work\_order.length-1==i?'':',');}}catch(e){} |

### Ext.form.RadioGroup单选

#### 配置本地数据

|  |
| --- |
| { xtype:'combobox',  width:180,  id:'check\_cycle\_id',  editable : **false**,  style:'marginLeft:20px',  labelWidth:60,  fieldLabel: '核查周期:',  store:**new** Ext.data.ArrayStore({  fields: ['value', 'name'],  data : [  ['all','全部'],  ['day','一天'],  ['week','一周'],  ['month','一个月']  ]  }),  queryMode: 'local',  name: 'check\_cycle',  displayField: 'name',  valueField: 'value',  allowBlank: **false**,  value:'all'  } |

#### 配置

|  |
| --- |
| {  xtype: 'radiogroup',  id:'status',  style:'marginLeft:30px',  fieldLabel: '状态',  labelWidth:40,  vertical: **false**,  items: [  { boxLabel: '启用', name: 'status', inputValue: '启用',checked:**true** },  { boxLabel: '停用', name: 'status', inputValue: '停用'}  ]  }  取值：  var countType = Ext.getCmp('status').getValue().status; |

#### radiogroup根据值设置选中

**if**(statiType=='0'){// 数据库取到的值statiType

Ext.getCmp('statiType\_1\_id').items.items[0].setValue(**true**)

//家庭

}**else** **if**(statiType=='2'){

Ext.getCmp('statiType\_1\_id').items.items[1].setValue(**true**)

//cookie

}**else** **if**(statiType=='1'){

Ext.getCmp('statiType\_1\_id').items.items[2].setValue(**true**)

}

### 获取form表单

var fd = new FormData(document.forms.namedItem("fileinfo")); // name=’fileinfo’

#### 配置

|  |
| --- |
| anchor:'100% 8%',//宽高为屏幕 100% 8%  layout:'column',  border:**false**,//去除边框线  bodyStyle:'padding:20 0 10 10;background:#EDEDED;opacity:1;',  items: [{  xtype:'textfield',  width:180,  id:'bts\_type',  editable : **false**,  labelSeparator:'',//去掉冒号  labelWidth:60,  fieldLabel: '基站类型:',  name: 'bts\_type',  allowBlank: **true**,  value:''  }，{  width:100,  xtype:'button',  style:'marginLeft:30px',  text:'查&nbsp;&nbsp;询',  handler:**function**(){}  }] |

### 多列布局

{

defaults:{labelWidth: 70,width:240,xtype : "textfield",border:**false**,margin:'10 10'},

xtype:'form',

layout:'column',

baseCls : 'my-panel-no-border',  //去掉边框

items:[{

fieldLabel:"手&nbsp;&nbsp;机&nbsp;&nbsp;号",

name: "phone",

emptyText:'手机号',

maxLength : 25,

maxLengthText : '不能超过25个字符',

regex : **/^1\d{10}$/**,

regexText : '手机号格式不正确!',

allowBlank: **true**,

value:phone

},{

fieldLabel:"邮&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;箱",

name: "email",

emptyText:'邮箱',

regex : **/^([a-z0-9\_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/**,

regexText : '邮箱格式不正确!',

allowBlank: **true**,

value:email

}]

}

### fieldset布局

{

defaults:{labelWidth: 70,xtype : "textfield",border:**false**,listConfig:{resizable:**true**}},

xtype:'fieldset',

title: '<strong>基本信息</strong>',

items:baseInfo

},

### 操作form

|  |
| --- |
| //获取表单所有字段封装成一个对象，可以通过  //取值  regex : /^([a-z0-9\_\.-]+)@([\da-z\.-]+)\.([a-z\.]{2,6})$/,  regexText : '邮箱格式不正确!',  formObject .字段名  var formObject = boardcardInfoQueryForm.getForm().getValues(); |

### form表单提交

|  |
| --- |
| Ext.getCmp('imgform').getForm().submit(  {  url : '#/addUser.do',  waitMsg : '数据在处理中,请稍后......',  waitTitle : '提示',  params : { //携带的参数  photo : photo,  dept\_id : parent\_id,  card\_id : rows  },  method : 'POST',  success : function(form, options) {//成功后要做的事情  var returnValue = options.result.yourPropertyName|| ""; //不需要解码  },  failure : function(form, options) {}//失败要做的事情}  ) |

### Ajax提交

|  |
| --- |
| Ext.Ajax.request({  url : '#/ReadCardInfo.do',  params : {},//携带的参数  method : 'POST',  success : function(response, opts) {  var cardNo=Ext.decode(response.responseText).cardNo||'';  Ext.getCmp('card\_index\_id').setValue(cardNo);  },  failure : function(response, opts) { }  })  //Ajax请求用回调函数做的事情,回调函数不管是成功还是失败都会执行  Ext.Ajax.request({  url : '#/validEmployee.do',  params : {employee\_id : employee\_id}, //携带的参数  method : 'POST',  callback : function(options, success, response) {  var m = Ext.decode(response.responseText).success;//解码返回的json数据  if (m) {  //写请求成功后要做的事情  }  else {//失败后要做的事情 }  }  }) |

### 时间时分

jsp导入文件：

<script type=*"text/javascript"* src=*"../plugin/ext-4.2/ux/dateTime/picker/DateTime\_yy.js"*></script>

<script type=*"text/javascript"* src=*"../plugin/ext-4.2/ux/dateTime/form/field/DateTime.js"*></script>

{

xtype:"datetimefield",

id :"startdate\_id",

fieldLabel : "开始时间",

editable:**false**,

name:'startdate',

format:'Y-m-d H:i',

value : Ext.Date.add(**new** Date(**new** Date().getFullYear(),**new** Date().getMonth(),**new** Date().getDate(),**new** Date().getHours(),0),Ext.Date.DAY,-7)

,listeners:{

select:**function**(){

**var** startDate = **this**.getValue();

startDate=Ext.Date.add(startDate,Ext.Date.DAY,7);

**var** today=**new** Date();

**if**(startDate.getTime()>today.getTime()){

Ext.getCmp('enddate\_id').setValue(**new** Date());

}**else**{

Ext.getCmp('enddate\_id').setValue(startDate);

}

}

}

}

### 精确到时

修改插件

### Ext.form.field.Date 日期控件

#### 配置

|  |
| --- |
| xtype:"datefield",  labelAlign:'right',  fieldLabel : "结束日期",  labelWidth:65,  id :"enddate",  value : Ext.util.Format.date(Ext.Date.add(**new** Date(),Ext.Date.DAY,-2),"Y-m-d"),  style : 'margin:10px 5px 5px 5px;' ,  maxValue : Ext.Date.add(**new** Date(),Ext.Date.DAY,-0)// 限制当前日期  ,listeners:{//时间联动  select:function(){  var startDate = this.getValue();  Ext.getCmp('enddate\_id').setValue(Ext.Date.add(startDate,Ext.Date.DAY,5));  }  } |

## 日期转换

**var** timeString = ‘2016-12-12 08:20’.substring(0,16).replace(**/-/g**,"/");

**var** timeFlag = **new** Date(timeString);

//时间相加

timeFlag.setHours(timeFlag.getHours()+6);

//日期相加

Ext.Date.add(startDate,Ext.Date.DAY,5) //返回相加后的结果

## 时间处理

var date=new Date();

if(amTime[0].length>1){

var b=amTime[0].substring(0,1);

var c=amTime[0].substring(1);

if(b==0){

date.setHours(parseInt(c)-1);

date.setMinutes(amTime[1]);

}else{

date.setHours(parseInt(amTime[0]));

date.setMinutes(parseInt(amTime[1]));

}

}else{

date.setHours(parseInt(amTime[0])-1);

date.setMinutes(parseInt(amTime[1]));

}

var am\_1=date.getHours()+":"+(date.getMinutes()?date.getMinutes():'00');

if(amTime[0].length>1){

var b=amTime[0].substring(0,1);

var c=amTime[0].substring(1);

if(b==0){

date.setHours(parseInt(c));

date.setMinutes(amTime[1]+30);

}else{

date.setHours(parseInt(amTime[0]));

date.setMinutes(parseInt(amTime[1])+30);

}

}else{

date.setHours(parseInt(amTime[0]));

date.setMinutes(parseInt(amTime[1])+30);

}

var am\_2=date.getHours()+":"+(date.getMinutes()?date.getMinutes():'00');

if(pmTime[0].length>1){

var b=pmTime[0].substring(0,1);

var c=pmTime[0].substring(1);

if(b==0){

date.setHours(parseInt(c)-1);

date.setMinutes(pmTime[1]);

}else{

date.setHours(parseInt(pmTime[0])-1);

date.setMinutes(parseInt(pmTime[1]));

}

}else{

date.setHours(parseInt(pmTime[0])-1);

date.setMinutes(parseInt(pmTime[1]));

}

var pm\_1=date.getHours()+":"+(date.getMinutes()?date.getMinutes():'00');

if(pmTime[0].length>1){

var b=pmTime[0].substring(0,1);

var c=pmTime[0].substring(1);

if(b==0){

date.setHours(parseInt(c));

date.setMinutes(pmTime[1]+30);

}else{

date.setHours(parseInt(pmTime[0]));

date.setMinutes(parseInt(pmTime[1])+30);

}

}else{

date.setHours(parseInt(pmTime[0]));

date.setMinutes(parseInt(pmTime[1])+30);

}

var pm\_2=date.getHours()+":"+(date.getMinutes()?date.getMinutes():'00');

## ExtJs布局

### Absolute布局

|  |
| --- |
| Ext.create('Ext.form.Panel', {  title: 'Absolute布局',  anchor: '20%',  frame:**true**,  height: 200,  x: '76%',  y: 0,  layout: 'absolute',  defaultType: 'textfield',  items: [{  x: 10,  y: 10,  xtype: 'label',  text: '名称:'  }, {  x: 50,  y: 10,  anchor: '20%',  name: 'name'//输入框  }, {  x: 10,  y: 40,  xtype: 'label',  text: '项目:'  }, {  x: 50,  y: 40,  name: 'project',//输入框  anchor: '20%'  }, {  x: 10,  y: 80,  xtype: 'textareafield',  name: 'msg',  anchor: '20% 30%' //控件从居左0px处拉长100%，空间从居上80px处，拉长至余下控件高度的80%处  }]    }) |

### accordion手风琴布局

|  |
| --- |
| Ext.create("Ext.panel.Panel", {  title: "west",  layout: "accordion", //设置为手风琴布局  layoutConfig: {  animate: **true**  },  width: 250,  minWidth: 90,  region: "west", //设置方位  split: **true**,  collapsible: **true**,  items: [  { title: "嵌套面板一", html: "嵌套面板一", iconCls: "save" },  { title: "嵌套面板二", html: "嵌套面板二", iconCls: "search" },  { title: "嵌套面板三", html: "嵌套面板三", iconCls: "back" },  { title: "嵌套面板四", html: "嵌套面板四", iconCls: "12" }  ]  }); |

### Anchor布局

|  |
| --- |
| Ext.create('Ext.Panel', {  width: 800,  height: 400,  title: "Anchor布局",  layout: 'anchor',  x: '20%',  y: 80,  renderTo: Ext.getBody(),  items: [{  xtype: 'panel',  title: '75% Width and 25% Height',  //可以使用百分比或者是-数字来表示。配置字符串使用空格隔开  //表示宽度为父容器width的75%，高度为父容器height的25%  anchor: '75% 25%'  }, {  xtype: 'panel',  title: 'Offset -300 Width & -200 Height',  //表示组件【相对于父容器右边距为250】，【相对于父容器(bottom)的底部位300】  anchor: '-300 50%'  }, {  xtype: 'panel',  title: 'Mixed Offset and Percent',  //父容器右边为250，【高度为父容器的30%】  anchor: '-250 50%'  }]  }); |

### border布局

### 添加自适应布局

Ext.create('Ext.container.Viewport', {

layout: 'border',

autoScroll:**true**,

items: [{

region:'north',

height:120,

title:'查询条件',

items:queryForm // queryForm 需要配置 region:’north’ 指向父组件

},{

xtype:'panel',

region:'center',

layout:'border',

items:mainPageGrid // mainPageGrid需要配置 region:'center' 指向父组件

}],

listeners:{

afterrender:**function**(){

}

}

});

|  |
| --- |
| Ext.create('Ext.container.Viewport', {  layout: 'border',  items: [  {  title: '北边',  height: 100,  region: 'north',  html: '<h1 class="x-panel-header">北边的</h1>',  border: **false**,  margins: '0 0 5 0'  },  {  region: 'west',  collapsible: **true**,  title: '西边',  width: 200,  html:'西边的'  // 这里通常可以使用 TreePanel 或者 AccordionLayout布局的导航菜单  }, {  region: 'center',  xtype: 'tabpanel', // TabPanel本身没有title属性  activeTab: 0, // 配置默认显示的激活面板  items: {  title: 'Default Tab',  html: 'The first tab\'s content. Others may be added dynamically'  }  }, {  region: 'east',  title: '东边的',  collapsible: **true**,  split: **true**,  width: 300  }  ,  {  region: 'south',  title: '南边',  collapsible: **true**,  html: '南边',  split: **true**,  height: 300,  minHeight: 100  }  ]  }); |

### column布局

|  |
| --- |
| Ext.create('Ext.panel.Panel', {  title: 'Column Layout - 按比例',  width: 600,  height: 250,  x: '20%',  y: 100,  layout: 'column',  items: [{  title: 'Column 1',  columnWidth: .25  }, {  title: 'Column 2',  columnWidth: .55  }, {  title: 'Column 3',  columnWidth: .20  }],  renderTo: Ext.getBody()  }); |

### card布局

|  |
| --- |
| **var** navigate = **function** (panel, next) {  **var** layout = panel.getLayout();  layout[next]();  Ext.getCmp('move-prev').setDisabled(!layout.getPrev());  Ext.getCmp('move-next').setDisabled(!layout.getNext());  };  Ext.create('Ext.panel.Panel', {  title: 'Card布局示例',  width: 600,  height: 300,  renderTo: Ext.getBody(),  layout: 'card',  activeItem: 0,  x: '20%',  y: 60,  bodyStyle: 'padding:100px',  defaults: { border: **false** },  bbar: [{  id: 'move-prev',  text: '上一步',  handler: **function** (btn) {  navigate(btn.up("panel"), "prev");  },  disabled: **true**  },  '->',  {  id: 'move-next',  text: '下一步',  handler: **function** (btn) {  navigate(btn.up("panel"), "next");  }  }  ],  items: [{  id: 'card-0',  html: '<h1>第一步</h1>'  },  {  id: 'card-1',  html: '<h1>第二步</h1>'  },  {  id: 'card-2',  frame:**true**,  html: '<h1>最后一步</h1>'  }]    }); |

## 工具函数

### 字符转int parseInt()

### 数组排序

**var** page\_id = Ext.getCmp('page\_ids\_id').getValue(),page\_ids='';

**var** page=[];

**for**(**var** i **in** page\_id){

page.push(parseInt(page\_id[i]));

}

//数组默认按字符排序

得加个排序规则：

**function**(a,b){**return** a>b?1:-1} //从小到大

**function(a,b){return a<b?1:-1}//从大到小**

page.sort(**function**(a,b){**return** a>b?1:-1});

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

page\_ids+= page[i]+(page.length-1==i?'':';');

}

### Ext遍历函数

Ext.each(list, **function**(Obj,index){

console.log(Obj);

console.log(index);

pieData.push([Obj.CFCQ,Obj.CNT]);

})

### 参数序列化

$.param({obj});

### 查看对象

JSON.stringify(obj);

### 封装json数据

|  |
| --- |
| **var** formObject = otherparamInfoQueryForm.getForm().getValues();  **var** ParamArray=[];  **if**(addManyConditionWin)  {  **var** conditionForm = Ext.getCmp('conditionForm\_id').getForm().getValues()  **var** msc = conditionForm.MSC\_CODE;  **var** sm = conditionForm.SM;  **if**(msc **instanceof** Array){  **for**(**var** k **in** msc){  **if**(msc[k].length>=sm[k].length){  //ParamArray  **var** mscArray=msc[k].split(",")  **var** smArray=sm[k].split(",")  **for**(**var** i **in** mscArray){  **for**(**var** j **in** smArray){  ParamArray.push({MSC:mscArray[i],SM:smArray[j]})  }  }    }**else**{  **var** mscArray=msc[k].split(",")  **var** smArray=sm[k].split(",")  **for**(**var** i **in** mscArray){  **for**(**var** j **in** smArray){  ParamArray.push({MSC:mscArray[i],SM:smArray[j]})  }  }  }  }  }**else** {  **if**(msc.length>=sm.length){  //ParamArray  **var** mscArray=msc.split(",")  **var** smArray=sm.split(",")  **for**(**var** i **in** mscArray){  **for**(**var** j **in** smArray){  ParamArray.push({MSC:mscArray[i],SM:smArray[j]})  }  }    }**else**{  **var** mscArray=msc.split(",")  **var** smArray=sm.split(",")  **for**(**var** i **in** mscArray){  **for**(**var** j **in** smArray){  ParamArray.push({MSC:mscArray[i],SM:smArray[j]})  }  }  }  }    }  把数组对象编码成字符对象Ext.encode(ParamArray) |

### js删除指定元素

Array.prototype.indexOf = **function**(val) {

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

**if** (**this**[i] == val) **return** i;

}

**return** -1;

};

Array.prototype.remove = **function**(val) {

**var** index = **this**.indexOf(val);

**if** (index > -1) {

**this**.splice(index, 1);

}

};

### 字符串数组互转

var aColors = new Array("red","green","blue");

aColors.toString();

var sColors = "red,green,blue";

     var aColors = sColors.split(',');   // 字符串就转换成Array数组了。

### 对象转对象转为字符串

**var** jsObj = {};

jsObj.testArray = [1,2,3,4,5];

jsObj.name = 'CSS3';

jsObj.date = '8 May, 2011';

**var** str = JSON.stringify(jsObj);

### JSON字符串转为对象

**var** jsObj = {};

jsObj.testArray = [1,2,3,4,5];

jsObj.name = 'CSS3';

jsObj.date = '8 May, 2011';

**var** str = JSON.stringify(jsObj);

**var** str1 = JSON.parse(str);

### map对象创建

var map ={1:20};

取值 map[1]

赋值map[1]=10；

### 转换大小写

LinkIndexTitile[i].toUpperCase()

sn.toLowerCase()

### 数组操作

var a=[10,20,30,40,50]

## //删除数组最后一个元素

a.pop();

document.write('<center>'+a+'</center>');

var b=a.slice(1,3);//删除数组下标 1到3的元素,原数组元素个数不变

document.write('<center>'+b+'</center>');

## //删除数组第一个元素

a.shift();

document.write('<center>'+a+'</center>');

## //添加元素到第一位

a.unshift(8);

document.write('<center>'+a+'</center>');

## //向数组添加多个元素

a.push(10,11,12); document.write('<center>'+a+'</center>');

var c =a.splice(0,3,7,8,9);//splice(start,deleteCount,val1,val2,...)删除指定下标位置的元素【第二个参数是删除的个数】，并返回被删除的元素,并从该位置起插入val1,val2,...

document.write('<center>'+a+'</center>');

## //删除指定下标内的元素,原数组元素不受影响

var b=a.slice(2,3); //

## //删除前三个元素

var b=a.slice(3);

document.write('<center>'+b+'</center>');

document.write('<center>--------------------------</center>');

a.reverse();//反向排序

document.write('<center>'+a+'</center>');

document.write('<center>---------------------------</center>');

a.sort();//反向排序

document.write('<center>'+a+'</center>');

## //合并数组

var h1=[10,20,30];

var h2=[40,50,60];

var h3=h1.concat(h2);//合并没有长度限制，千万级别的数组合并大约30毫秒 Array.prototype.push.apply(h1, h2);// 数组长度有限制，不同浏览器不同，一般不能超过十万, 一万数据左右性能和concat差不多，大约2毫秒

document.write('<center>'+h1+'</center>');

## 比较数组元素最大，最小值

**var** h=Math.max.apply(**null**,c);

**var** j=Math.min.apply(**null**,c);

## 树

### 递归回去树的所有节点

//----------------递归调用-----------

getNode(id);

**if**(personalStore.getNodeById(id).hasChildNodes()){

deleteids+=''+id+",",tagName+=''+text+",";

}

**function** getNode(node){

**var** thisNode=personalStore.getNodeById( node )

**if**(!thisNode.hasChildNodes()){

deleteids+=thisNode.data.id;

tagName+=thisNode.data.text; **return** **false**;

}**else**{

**var** childNodes=thisNode.childNodes;

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

deleteids+=childNodes[i].data.id+","

tagName+=childNodes[i].data.text+','

**if**(childNodes[i].hasChildNodes()){

getNode(childNodes[i].data.id);

}

}

}

}

**if**(tagName.charAt(tagName.length-1)==','){

deleteids=deleteids.substring(0,deleteids.length-1);

tagName=tagName.substring(0,tagName.length-1);

}

### 修复多选下拉树不能全部勾选bug

**var** rootNodes = Ext.getCmp('modify\_menuCombox\_id').picker.getRootNode();

**var** checkValue='';

**var** parentNodes=rootNodes.childNodes;

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

**var** parent=parentNodes[i];

**var** nodes=parent.childNodes;

//----子节点-----------

**for**(**var** k=0;k<nodes.length;k++){

**var** record2=nodes[k];

**if**(record2.get('id')==35){

record2.set('checked', **true**);

}

**if**(record2.get('checked')){

checkValue+=record2.get('text')+";"

}

}

**for**(**var** k=0;k<nodes.length;k++){

**var** record2=nodes[k];

**if**(!record2.get('checked')){

parent.set('checked',**false**);

**break**;

}**else**{parent.set('checked',**true**);}

}

**if**(parent.get('id')==1||parent.get('id')==2){

parent.set('checked', **true**);

}

**if**(parent.get('checked'))checkValue+=parent.get('text')+";";

### 多选下拉树：

|  |
| --- |
| 导入：**<script type="text/javascript" src="../plugin/ext-4.2/ux/comboxtree.js"></script>**  **var** tagStore = Ext.create('Ext.data.TreeStore',{  autoLoad:true,  root:{id:'-1',text:'节点1'},  proxy:{type:'ajax',url:'/videoTag/userManage/getTags.do'},  listeners:{  }  });  **var** tags=Ext.create('Ext.ux.comboboxtree',{  id:'tags\_id',  fieldLabel:"家庭标签",  name: "tagname",  cascade:'child',  emptyText:'家庭标签',  selectNodeModel : 'leaf',  allowBlank: **true**,  editable : **false**,  style:'marginLeft:20px',  store:tagStore,  valueField:'id',  displayField:'text',  hiddenName:'id',  editable:**false**,  labelWidth:70,  width:700,  rootId:'-1',  rootText:'节点1'  });  取值：  **var** root=Ext.getCmp('tags\_id').picker.getView().getChecked(),id = "";  Ext.Array.each(root,**function**(rec,index) {  id+=rec.get('id')+(root.length-1==index?'':",");  }); |

### 单选下拉树

|  |
| --- |
| **var** KQItreeStore = Ext.create('Ext.data.TreeStore',{  autoLoad:**true**,  proxy:{type:'ajax',url:'qualityQrill/getKQI.do'}  });  //var KQItree=Ext.create('Redm.commons.TreeCombox',{  **var** KQItree=Ext.create('Ext.ux.comboboxtree',{  fieldLabel:"KQI指标",  name: "KQI",  emptyText:'KQI指标',  cascade:'child',  mode:'local',  lines: **true**,  checkModel:'single',  allowBlank: **false**,  editable : **false**,  style:'marginLeft:20px',  store:KQItreeStore,  valueField:'id',  displayField:'text',  labelWidth:70,  width:250,  rootId:'-1',  rootText:'节点1'  }); |

**Ext.data.NodeInterface** 接口

### 获取父节点

**record.parentNode.data.text**

**record.hasChildNodes()**

### 默认选中树节点

**var** nav = ***Ext***.*create*(**'Ext.tree.Panel'**, {  
 **title**:**'阀值配置菜单'**,  
 **region**:**'west'**, *//放在西边，即左侧* **split**:**true**,  
 **store**:store,  
 **width**:200,  
 **collapsible**:**false**, *//允许伸缩* **margins**:**'3 0 3 3'**,  
 **cmargins**:**'3 3 3 3'**,  
 **listeners**:{  
 afterrender:**function**(){  
 nav.**selModel**.select(nav.getRootNode().getChildAt(0));  
 }  
  
 }  
});

tree.selModel.select(node)

获取选中节点

var selectTreeNode = tree.selModel.getSelectedNode();

### 树添加右键菜单

|  |
| --- |
| **var** familyStore = Ext.create("Ext.data.TreeStore", {  autoLoad : **true**,  defaultRootId : 1,  root: {  text:"家庭标签",  id:"1",  // children: 'children',  nodeType: 'async',  expanded:**true**  },  fields: [  { name: 'id'},  { name: 'text'},  { name: 'pid'},  { name: 'delete\_flag'},  { name: 'expanded'}  ],  proxy: {  type: 'ajax',  url: "/videoTag/tagConfig/getFamilyTree.do"  },  listeners:{  load:**function**(store){  **try**{  store.getNodeById(1).expandChildren(**false**,**true**);  }**catch**(e){}  }  }  });  **var** familyTreePanel=Ext.create("Ext.tree.Panel", {  store: familyStore,  header:**false**,  region: 'west',  anchor:'100% 100%',  useArrows: **true**,  animate: **true**,  lines:**true**,  autoScroll: **true**,  iconCls: 'User',  rootVisible : **true**,  listeners:{  cellcontextmenu:**function**(node,td,cellIndex,record,tr,rowIndex,e){  e.preventDefault();  node.select();  **var** isdiabl=**false**;  **if**(record.get('delete\_flag')!=1){  isdiabl=**true**;  }  family\_nodemenu = **new** Ext.menu.Menu({  floating : **true**,  items : [  {  text : "添加子标签",  icon : '../images/add.png',  handler : **function**() {    **var** id = record.get('id');  **var** pid = record.get('pid');  **var** thisNode=familyStore.getNodeById( id )  addChildTag(id,pid,thisNode)  }  },  {  text : "重命名",  icon : '../images/cog\_edit.png',  handler : **function**() {  **var** id = record.get('id');  **if**(id==1){  warning\_msg('系统提示','根标签不能修改!');  **return**;  }  **var** thisNode=familyStore.getNodeById(id);  renameTag(id,thisNode)  }  },  {  text : "删除标签",  disabled:isdiabl,  icon : '../images/page\_delete.png',  handler : **function**() {  **var** id = record.get('id');  **var** text = record.get('text');  **var** pid = record.get('pid');  **if**(id==1){  warning\_msg('系统提示','根标签不能删除!');  **return**;  }  **var** thisNode=familyStore.getNodeById( id );  **var** deleteids='',tagName='';  **if**(thisNode.hasChildNodes()){  getNode(id);  **if**(familyStore.getNodeById(id).hasChildNodes()){  deleteids+=''+id+",",tagName+=''+text+",";  }  **function** getNode(node){  **var** thisNode=familyStore.getNodeById( node )  **if**(!thisNode.hasChildNodes()){  deleteids+=thisNode.data.id;  tagName+=thisNode.data.text;  **return** **false**;  }**else**{  **var** childNodes=thisNode.childNodes;  **for**(**var** i=0;i<childNodes.length;i++){  deleteids+=childNodes[i].data.id+","  tagName+=childNodes[i].data.text+','  **if**(childNodes[i].hasChildNodes()){  getNode(childNodes[i].data.id);  }  }  }  }    **if**(tagName.charAt(tagName.length-1)==','){  deleteids=deleteids.substring(0,deleteids.length-1);  tagName=tagName.substring(0,tagName.length-1);  }    //-----------------------------------------------    Ext.MessageBox.confirm('系统提示', '删除该标签将连同子标签一起删除，确定删除吗?',**function**(btn){  //---------------执行删除-----------------------  **if**(btn=="yes"){  Ext.Ajax.request({  url : '/videoTag/tagConfig/deleteTag.do',  params:{tagid:deleteids,tagName:tagName},  success : **function**(resp, opts) {  **var** json=Ext.decode(resp.responseText);  **if**(json.flag>0){  thisNode.remove();  }  },  failure : **function**(resp, opts) {  error\_msg('系统提示','删除失败！');  }  })  }  //---------------执行删除-----------------------  });  }**else**{  deleteids=''+id,tagName=''+text;  Ext.MessageBox.confirm('系统提示', '确定删除该标签吗?',**function**(btn){  //---------------执行删除-----------------------  **if**(btn=="yes"){  Ext.Ajax.request({  url : '/videoTag/tagConfig/deleteTag.do',  params:{tagid:id,tagName:tagName},  success : **function**(resp, opts) {  **var** json=Ext.decode(resp.responseText);  **if**(json.flag>0){  thisNode.remove();  // win\_alert('系统提示','已成功删除！');  }  },  failure : **function**(resp, opts) {  error\_msg('系统提示','删除失败！');  }  })  }  //---------------执行删除-----------------------  });  };    //    }  }  ]  })  //-------------------------------------------------------------------  family\_nodemenu.showAt(e.getXY());  //-------------------------------------------------------------------  }  } }); |

# 获取鼠标当前元素的位置

**top:$(this).position().top+70,**

**left:$(this).position().left+50**

# jQuery查找元素

## 查找元素

**var imgElement=$('#modify\_image\_box').contents().find("img")**

## 过滤属性元素

**imgElement.filter("[resolution='']").length**

# cookie

### 一、Ext.util.Cookies.set('saveacct',saveacct);

取cookie中数据如下

|  |
| --- |
| var validStatus = Ext.util.Cookies.get("saveacct"); |

备注：在同一界面中生效，跨越界面是取不到cookie中的值

### 二、Ext.state.CookieProvider();

|  |
| --- |
| var cookie = new Ext.state.CookieProvider();   Ext.state.Manager.setProvider(cookie);    cookie.set('saveacct',saveacct); |

|  |
| --- |
| var cookie1 = Ext.state.Manager.getProvider();    var getsaveacct = cookie1.get('saveacct'); |

备注：全局都可使用，但是在设置之前需要clear一下。

*TreePanelState*.**prototype**.init = **function**() {  
  
 **this**.**cookie** = **new *Ext***.**state**.CookieProvider({  
 **expires**: **new** *Date*(**new** *Date*().getTime()+(1000\*60\*60\*24\*1))  
 });  
 ***Ext***.**state**.Manager.setProvider(**this**.**cookie**);  
 */\*清楚cookie数据\*/* **this**.**cookie**.clear(**'tr\_'** + **this**.**treePanel**.id + **'\_state'**);  
 **this**.**cookie**.clear(**'TreePanelState\_'** + **this**.**treePanel**.id + **'\_Station'**);  
 **this**.**cookie**.clear(**'TreePanelState\_'** + **this**.**treePanel**.id + **'\_SR'**);  
 **this**.**cookie**.clear(**'TreePanelState\_'** + **this**.**treePanel**.id + **'\_Switch'**);  
 **this**.**cookie**.clear(**'TreePanelState\_'** + **this**.**treePanel**.id + **'\_Comp'**);  
 **this**.**cookie**.clear(**'TreePanel\_'** + **this**.**treePanel**.id + **'\_sort\_state'**);  
};

# word2013操作快捷键

## 快捷键

Ctrl+U——为字符添加（或取消）下划线

Ctrl+B——使字符变为（或取消）粗体

Ctrl+I——使字符变为（或取消）斜体

## 字号调整

Ctrl+] ——增大字号

Ctrl+[ ——减小字号

## 段落行距调整

Ctrl +L —— 左对齐

 Ctrl+R —— 右对齐

Ctrl+1 —— 单倍行距

Ctrl+2 —— 双倍行距

Ctrl+5 —— 1.5 倍行距

Ctrl+E —— 段落居中

Ctrl+J —— 两端对齐 C

Ctrl+Shift+C —— 复制格式 （格式刷效果）Ctrl+Shift+V —— 粘贴格式 （格式刷效果）

## 选中区域移动

Tab （右移动）或 Shift+Tab（左移动）

Alt+shift+向上键 （所选段上移）

Alt+shift+向下键（所选段下移）

## 展开或折叠功能区

ctrl+F1

## 大小写切换

shift+f3