# H3BPM控件的一些使用

## 选人控件

### html页面代码

<**div ui-jq="SheetUser" ui-options="WasAgentOptions" id="shareFile"  
 class="form-horizontal" style="**margin-bottom: 20px;**" data-orgunitvisible="true"**>  
</**div**>

### 代码解读

**ui-jq**此属性是必选，属性值**SheetUser**是H3BPM的一个控件，使用这个控件还有其他方式

data-type="SheetUser"  
data-datafield="SheetUser" //这些属性引入的控件还未研究

最重要的属性便是**ui-options**，给此属性赋值**WasAgentOptions**，**WasAgentOptions**值可以设置选人控件是否可以多选单选（默认单选）。

**data-orgunitvisible**属性，默认false，表示只是选个人。true表示组织和个人可同时显示。

此时已经可以正常显示选人控件了，但是选人控件的后边的复选框的样式是错乱的

**class=“form-horizontal”**，添加这个类样式就可整齐的显示了

id、style根据需要添加

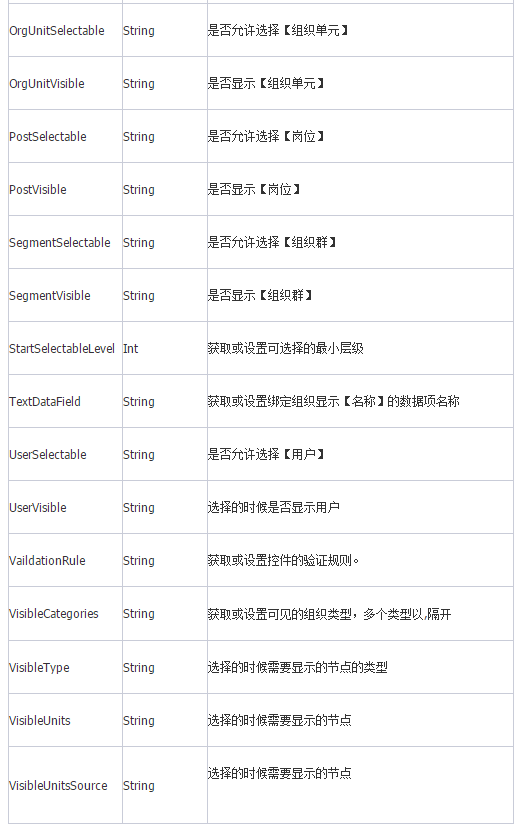
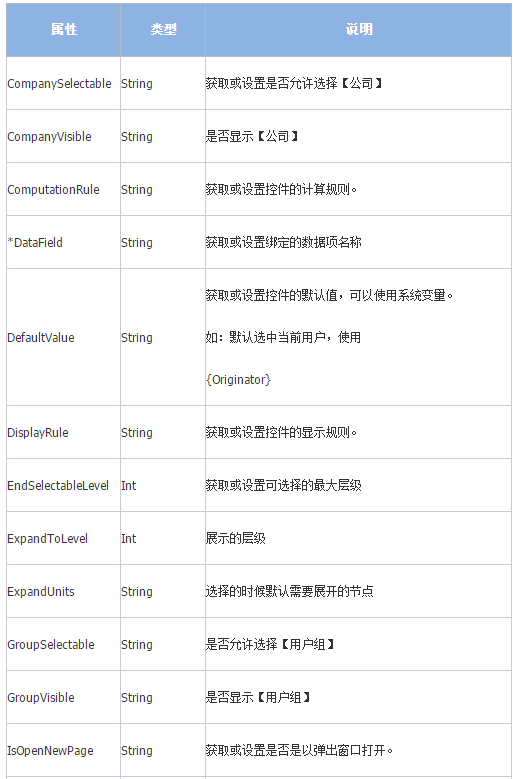
### JavaScript代码

$scope.**WasAgentOptions** = {  
 **Editable**: **true**,  
 **Visiable**: **true**,  
 **IsMultiple**: **true//是否多选单选**

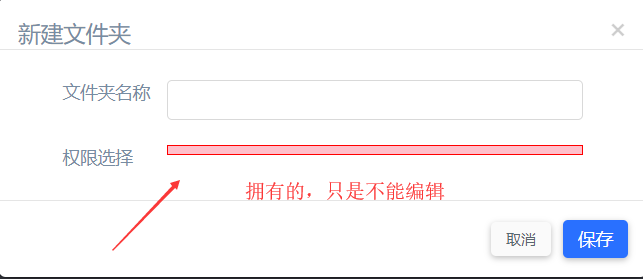
UserVisible:false,//是否显示用

}

具体参数参考 <http://wiki.h3bpm.com/>

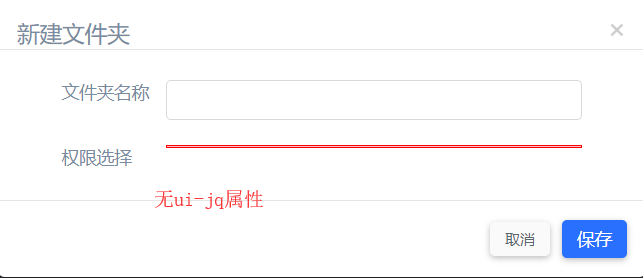


* + Editable: false效果



* + Visiable: false,





## 流程模板控件

### html代码

<**div ui-jq="SheetWorkflow" ui-options="WorkflowOptions"**></**div**>

属性解读参考选人控件

### JavaScript代码

$scope.**WorkflowOptions** = {*//初始化流程模板*  
 **Editable**: **true**,  
 **Visiable**: **true**,  
 **Mode**: **"WorkflowTemplate"**,*//似乎不影响*  
 **IsMultiple**: **true**,  
 **OnChange**: **""**,  
 **PlaceHolder**: $scope.**LanJson**.**WorkFlow**,*//似乎不影响*  
 **IsSearch**:**true** *//是否搜索目录，默认false*}

* + IsSearch:true



## 日历控件

### html代码

<**input ui-jq="wdatePicker" ui-options="StartTimeStartOption"** />

### 代码解读

**ui-jq**此属性是必选，属性值**wdatePicker**是jQuery的一个插件，此处表示引用日期插件

最重要的属性便是**ui-options**，该属性的属性值一般是在controller中定义，给此属性赋值**StartTimeStartOption，**在控制器中可以配置功能，如下所示

### JavaScript代码

$scope.**StartTimeStartOption** = {  
 **dateFmt**: **'yyyy-MM-dd HH:mm:ss'**,//设置日期显示的格式

**realDateFmt**: **"yyyy-MM-dd HH:mm:ss"**,

**minDate**: **'2012-1-1'**, **maxDate**: **'2099-12-31'**,//设置日期范围  
 onpicked: **function** (e) {//当点击日期的时候触发  
 $scope.**StartTimeStart** = e.**el**.**value**;//获取到输入的日期数据  
 }  
}

## table控件

### html代码

**table ui-jq="dataTable" id="tabfinishWorkitem"  
 ui-options="dtFlowOptions"  
 class="table table-striped m-b-none" style="width**:99.5%;**"**>  
 <**thead**>  
 <**tr**>  
 <**th**> <**span**>名称</**span**></**th**>  
 <**th**><**span**>标签</**span**></**th**>  
 <**th class="hide1024"**><**span**>流程模板</**span**></**th**>  
 <**th class="hide414"**><**span**>开始时间</**span**></**th**>  
 <**th class="hide414"**><**span**>结束时间</**span**></**th**>  
 <**th class="hide1024"**><**span**>操作</**span**></**th**>  
 </**tr**>  
 </**thead**>  
 <**tbody**></**tbody**>  
</**table**>

### JavaScript代码

*//ui-options="dtFlowOptions"*

$scope.getColumns = **function** () {  
 **var** columns = [];*//!!注意colums的长度也就是push进去的个数必须与页面中的<th>标签个数一致*  
 columns.push({  
 **"mData"**: **"InstanceName"**,*//响应数据的键*  
 **"mRender"**: **function** (data, type, full) {*//data每个单元格的数据，full每一行的数据* data = $scope.htmlEncode(data);  
 **return "<a target='\_blank' href='WorkItemSheets.html?WorkItemID="** + full.**ObjectID** + **"'>"** + data + **"</a>"**;  
 }  
 });  
 columns.push({  
 **"mData"**: **"DisplayName"**,  
 **"mRender"**: **function** (data, type, full) {  
 *//打开流程状态* data = data != **""** ? data : full.**ActivityCode**;  
 data = $scope.htmlEncode(data);  
  
 **return "<td><a href='index.html#/InstanceDetail/"** + full.**InstanceId** + **"/"** + full.**ObjectID** + **"/"** + **"/' target='\_blank'>"** + data + **"</a></td>"**;  
 }  
 });  
 columns.push({ **"mData"**: **"ReceiveTime"**, **"sClass"**: **"center hide1024"** });  
 columns.push({ **"mData"**: **"FinishTime"**, **"sClass"**: **"center hide414"** });  
 columns.push({  
 **"mData"**: **"OriginatorName"**,  
 **"sClass"**: **"center hide414"**,  
 **"mRender"**: **function** (data, type, full) {  
 data = $scope.htmlEncode(data);  
 **return "<a ng-click=\"showUserInfoModal('"** + full.**Originator** + **"');\" new-Bindcompiledhtml>"** + data + **"</a>"**;  
 }  
 });  
 columns.push({  
 **"mData"**: **"OriginatorOUName"**,  
 **"sClass"**: **"center hide1024"**,  
 **"mRender"**: **function** (data, type, full) {  
 data = $scope.htmlEncode(data);  
 **return** data;  
 }  
 });  
 **return** columns;  
}  
$scope.loadScroll = **function**() {  
 $scope.**myScroll** = **new** IScroll(**'.dataTables\_scrollBody'**, {  
 **scrollbars**: **true**,  
 **bounce**: **false**,  
 **mouseWheel**: **true**,  
 **interactiveScrollbars**: **true**,  
 **shrinkScrollbars**: **'scale'**,  
 **fadeScrollbars**: **true** });  
};  
$scope.**dtFlowOptions** = {  
 **"bProcessing"**: **true**,  
 **"bServerSide"**: **true**, *// 是否读取服务器分页* **"paging"**: **true**, *// 是否启用分页* **"bPaginate"**: **true**, *// 分页按钮* **"bFilter"**: **false**, *// 是否显示搜索栏* **"searchDelay"**: 1000, *// 延迟搜索* **"iDisplayLength"**: 20, *// 每页显示行数* **"bSort"**: **false**, *// 排序* **"singleSelect"**: **true**,  
 **"bInfo"**: **true**, *// Showing 1 to 10 of 23 entries 总记录数没也显示多少等信息* **"bLengthChange"**: **true**, *// 每页显示多少数据* **"aLengthMenu"**: [[10, 20, 50, 100], [10, 20, 50, 100]],*//设置每页显示数据条数的下拉选项* **"pagingType"**: **"full\_numbers"**, *// 设置分页样式，这个是默认的值* **"sScrollY"**: **"500px"**,  
 **"bScrollCollapse"**: **true**,  
 **"iScrollLoadGap"**: 50,  
 **"language"**: { *// 语言设置* **"sLengthMenu"**: $scope.**LanJson**.**sLengthMenu**,  
 **"sZeroRecords"**: **"<div class='no-data'><p class='no-data-img'></p><p>"**+$scope.**LanJson**.**sZeroRecords**+**"</p></div>"**,  
 **"sInfo"**: $scope.**LanJson**.**sInfo**,  
 **"infoEmpty"**: **""**,  
 **"sProcessing"**: **'<div class="loading-box"><i class="icon-loading"></i><p>'**+$scope.**LanJson**.**sProcessing**+**'</p></div> '**,  
 **"paginate"**: {  
 **"first"**: **"<<"**,  
 **"last"**: **">>"**,  
 **"previous"**: **"<"**,  
 **"next"**: **">"** }  
 },  
 **"sAjaxSource"**: ControllerConfig.**WorkItem**.**GetFinishWorkItems**,*//请求的地址，sSource与其保持一致*  
 **"fnServerData"**: **function** (sSource, aDataSet, fnCallback) {  
 $.ajax({  
 **"dataType"**: **'json'**,  
 **"type"**: **'POST'**,  
 **"url"**: sSource,  
 **"data"**: aDataSet,  
 **"success"**: **function** (json) {  
 **if** (json.ExceptionCode == 1 && json.Success == **false**) {  
 json.**Rows** = [];  
 json.**sEcho** = 1;  
 json.**Total** = 0;  
 json.**iTotalDisplayRecords** = 0;  
 json.**iTotalRecords** = 0;  
 $state.go(**"platform.login"**);  
 }  
 fnCallback(json);  
 *//每次搜索成功后清空搜索关键字* $(**".input-text"**).**val**(**""**);  
 }  
 });  
 },  
 **"sAjaxDataProp"**: **'Rows'**,*//后端传回的数据字段，若请求的到是datas，则此处相应的修改*  
 **"sDom"**: **'<"top"f>rt<"row"ipl>'**,  
 **"sPaginationType"**: **"full\_numbers"**,  
 **"fnServerParams"**: **function** (aoData) {  
 *// 增加自定义查询条件  
 //ie9不兼容placeholder属性 ie9下当value为空时，其value取placeholder值* **if**($(**"#StartTime"**).attr(**"placeholder"**)==$(**"#StartTime"**).**val**()){  
 $scope.**StartTime**=**""**;  
 }**else**{  
 $scope.**StartTime** = $(**"#StartTime"**).**val**();  
 }  
 **if**($(**"#EndTime"**).attr(**"placeholder"**)==$(**"#EndTime"**).**val**()){  
 $scope.**EndTime**=**""**;  
 }**else**{  
 $scope.**EndTime** = $(**"#EndTime"**).**val**();  
 }  
 *//将时间转化为时间戳* **var** startTimes = **new *Date***($scope.**StartTime**.replace(/-/g,**"/"**)).getTime();  
 **var** EndTimes = **new *Date***($scope.**EndTime**.replace(/-/g,**"/"**)).getTime();  
 **if**(startTimes>EndTimes){  
 $.notify({**message**:**"时间区间错误"**,**status**:**"danger"**});  
 $(**"#EndTime"**).css(**"color"**,**"red"**);  
 **return false**;  
 }  
 aoData.push(  
 { **"name"**: **"startTime"**, **"value"**: $filter(**"date"**)($scope.**StartTime**, **"yyyy-MM-dd"**) },  
 { **"name"**: **"endTime"**, **"value"**: $filter(**"date"**)($scope.**EndTime**, **"yyyy-MM-dd"**) },  
 { **"name"**: **"workflowCodes"**, **"value"**: $scope.**WorkflowCodes** },  
 { **"name"**: **"instanceName"**, **"value"**: $scope.**InstanceName** }  
 );*//发送给后台的json数据，name表示键，value表示值*  
 },  
 **"aoColumns"**: $scope.getColumns(), *// 字段定义，请求响应回来的数据*

*// 初始化完成事件,这里需要用到 JQuery ，因为当前表格是 JQuery 的插件*

**"initComplete"**: **function** (settings, json) {  
 **var** filter = $(**".searchContainer"**);  
 filter.find(**"button"**).unbind(**"click.DT"**).bind(**"click.DT"**, **function** () {  
 $scope.**WorkflowCodes** = $(**"#sheetWorkflow"**).SheetUIManager().GetValue();  
 $(**"#tabfinishWorkitem"**).**dataTable**().fnDraw();  
 });  
 $scope.loadScroll();  
 },  
 *// 创建行，未绘画到屏幕上时调用* **"fnRowCallback"**: **function** (nRow, aData, iDisplayIndex, iDisplayIndexFull) {*//其中aData 存放的是每一行的数据  
 //将添加的angular事件添加到作用域中* $compile(nRow)($scope);  
 *setTimeout*(**function**(){  
 $scope.**myScroll**.refresh();  
 },300);  
 },  
 **"fnDrawCallback"**: **function** () {  
 jqdatables.trcss();  
 }}

## 标签控件

### Html代码

<**input autocomplete="off" ng-model="tagName" id="tagName" maxlength="32" class="input form-control" placeholder="标签"**/>

### JavaScript代码

$scope.renderTag = **function**(idName) {*//input的元素的id名称*  
 **var** obj = $(**"#"**+idName);**var** sltCategoryComBox;*//拿到所有数据，然后再重新加载页面数据* **var** initId = **""**;  
 **var** initValue = **""**;  
 **var** tmpData = [];  
  
 *//渲染数据的地方* sltCategoryComBox = obj.ligerComboBox({  
 **initValue**: initId,  
 **initText**: initValue,  
 **data**: tmpData,*//返回的下拉框的数据*  
 **valueFieldID**: **'category'**,  
 **url**: **'/Portal/tag/listKnowledgeTagByName'**,*//请求的后台地址* **ajaxType**: **'GET'**,  
 **valueField**: **'id'**,*//返回的数据的键必须有id*  
 **textField**: **'text'，***//返回的数据的键必须有test*  
 **autocomplete**: **true**,  
 **setTextBySource**: **true**,  
 **keySupport**: **true** });  
  
 *//设置想要的样式* obj.parent().removeClass();  
 **var** inputHeight = obj.outerHeight();  
 **var** inputWidth = obj.outerWidth();  
 obj.css({**"border"**:**"1px solid #d9d9d9"**,**"border-radius"**:**"4px"**});  
 *//下拉框的内容样式设置,模态框样式有点* $(**"div.l-box-select-inner"**).parent().css({**"margin-top"**: inputHeight + **'px'**});  
 $(**"div.l-box-select"**).css({**"width"**: inputWidth - 2 + **'px'**});

*//鼠标离开输入框，下拉框隐藏*  
 obj.mouseleave(**function** (e) {  
 **var** x = e.**pageX** - obj.offset().**left**;  
 **var** y = e.**pageY** - obj.offset().**top**;  
 **if**(x < 0 || y < 0 || x - inputWidth > 0 ){  
 $(**"div.l-box-select"**).hide();  
 }  
 });  
  
 *//update by zhangj* obj.change(**function** () {*//按回车键触发该函数* **var** id = $(**"#category"**).**val**();*//获取的是数据中的，选中数据条的id* });  
}

## 已办任务中详情页导航栏按钮自定义

### Html代码

<**ul** class**="nav navbar-nav" style="float**: **right**;**" id="collect"**>  
 <**li**>  
 <**a ng-click="toCollectFinishedFlow()"  
 class="btn-action" title="收藏到我的知识"**>  
 <**span class="toolText"**>收藏到我的知识</**span**>  
 </**a**>  
 </**li**>  
</**ul**>

### 代码解读

由于formSheet.html没有使用angular语法，要使得这个formSheet.html页面与app关联，则需要在<html>根标签，加入属性data-ng-app="app"

关联对应的模块app。

引入的模块app，同时也需要引入相应的js，css文件等。

在需要控制的地方，加入对应的控制器formSheetServiceCtrl，控制器的新建可以参考项目中原来的控制器的新建。

这样在ng-click的方法就可以在formSheetServiceCtrl控制器定义且生效了。

然后，自定义的导航栏按钮，需求是在点击已办任务的详情页显示，而待办任务等等中不需要显示，解决方法只能是jQuery。

在MvcSheet.js文件中根据WorkItemType属性的值不同，进行隐藏



### JavaScript代码

*/\*如果不是已办任务，就隐藏分享和收藏按钮\*/***if**(*$*.**MvcSheetUI**.**SheetInfo**.**WorkItemType** !== **'Fill'**){  
 *$*(**"#collect"**).hide();  
 *$*(**"#share"**).hide();  
}

## angular的模态框

### Html代码

*<!--我的文件删除-->*<**script type="text/ng-template" id="myDeleteFile.html"**>  
 *<!--此处的所有方法，参数都是在ModalController里面-->*  
 <**form ng-submit="myRemoveFile(temp)" style="**color:#778CA2;**"**>  
 <**div class="modal-header"**>  
 <**button type="button" class="close" data-dismiss="modal"**>  
 <**span aria-hidden="true" ng-click="***cancel***();"**>**&times;**</**span**>  
 </**button**>  
 <**h4 class="modal-title"**>确认框</**h4**>  
 </**div**>  
 <**div class="modal-body"**>  
 确定删除 <**b**>{{**temp**.**model**.**name**}}</**b**> ?</**div**>  
 <**div class="modal-footer"**>  
 <**button type="button" class="btn btn-default" data-dismiss="modal" ng-disabled="temp.inprocess" ng-click="***cancel***();"**>取消</**button**>  
 <**button type="submit" class="btn btn-primary" ng-disabled="temp.inprocess" autofocus="autofocus"**>确定</**button**>  
 </**div**>  
 </**form**>  
</**script**>

### JavaScript代码

* + 弹出框

$scope.modal = **function** (data) {  
 *//处理参数，这里得参数的值不影响弹出框* **if** (data == undefined) **AgencyID** = **""**;  
 **else AgencyID** = data;  
  
 *//请求代理人控件的数据* $http({  
 **url**: ControllerConfig.**Agents**.**GetAgency**,  
 **params**: {  
 **agentID**: **AgencyID**,  
 **random**: **new *Date***().getTime()  
 }  
 })  
 .success(**function** (result, header, config, status) {  
 **var** Agency = result.**Rows**[0];  
 *// 弹出模态框* **var** modalInstance = $modal.open({  
 **templateUrl**: **'myDeleteFile.html'**, *// 指向上面创建的视图，弹框页面的内容，对应html代码里的script标签里面的id* **controller**: **'ModalController'**,*// 初始化模态范围，在这个controller里面处理模态框上的方法，例如保存的方法等* **size**: **"md"**,  
 **resolve**: {  
 params: **function** () {  
 **return** {  
 **user**: $scope.**user**,  
 **Agency**: Agency,  
 **AgencyID**: **AgencyID** };  
 },  
 **deps**: [**'$ocLazyLoad'**, **function** ($ocLazyLoad) {*//* **return** $ocLazyLoad.load([  
 **'WFRes/\_Content/themes/ligerUI/Aqua/css/ligerui-all.min.css'**,*//这个文件已经在主页面加载了，此处可以不需要* **'WFRes/assets/stylesheets/sheet.css'**,*//选人框的的样式* **'WFRes/\_Scripts/jquery/jquery.lang.js'**,*//这个文件已经在主页面加载了，此处可以不需要* **'WFRes/\_Scripts/ligerUI/ligerui.all.min.js'**,*//这个文件已经在主页面加载了，此处可以不需要* **'WFRes/\_Scripts/MvcSheet/SheetControls.js'**,*//这个文件已经在主页面加载了，此处可以不需要* **'WFRes/\_Scripts/MvcSheet/MvcSheetUI.js'***//这个文件已经在主页面加载了，此处可以不需要* ]).then(**function** () {  
 **return** $ocLazyLoad.load([  
 **'WFRes/\_Scripts/MvcSheet/Controls/SheetWorkflow.js'**,*//这个文件已经在主页面加载了，此处可以不需要* **'WFRes/\_Scripts/MvcSheet/Controls/SheetUser.js'***//这个文件已经在主页面加载了，此处可以不需要* ])  
 });  
 }]  
 }  
 });  
 modalInstance.opened.then(**function**() {  
 *//****TODO not work*** });  
   
 })  
   
 **var** times = *setInterval*(**function**() {*//等待控件加载完毕后，进行样式更改*  
 **if**(*$*(**"#WorkflowCodes"**).**length**>0){  
 *clearInterval*(times);  
 *$*(**".select2-search-field"**).find(**"input"**).css(**"z-index"**, 0);  
 *//console.log($("#WorkflowCodes"));  
 $*(**"#WorkflowCodes"**).css(**"width"**,**"246px"**);  
 }  
 }, 50)  
}

## 8.提示框

提示用户信息提示框

在控制器的定义处依赖注入notify。

*$*.notify({ **message**: **"请选择权限！"**, **status**: **"danger"** });*//提示message的内容，status表示提示的状态*

## 9.运维表单代码解析

后台管理子表单js代码解析

$(function () {

$.MvcSheet.PreInit = function () {

//表单预加载方法

var load\_status = [{

undo: 0,

liquidation\_night\_problem\_finish: 1,

liquidation\_monitor\_stmarket\_finish: 2,

}]//夜间清算及开市和日间核对标志位枚举以及方进入页面默认标志位为0

$.MvcSheetUI.GetControlValue('node\_load\_status');

$("#div312865").hide();//隐藏状态标志位

$("#DayByCollect").css("cursor", "pointer");//日间核对展开收起按钮鼠标点击小手样式

$("#NightCollect").css("cursor", "pointer");

$("#NightOpenCollect").css("cursor", "pointer");

$("#DayDealCollect").css("cursor", "pointer");

DealRestart = function () {

//投资交易系统刷新方法

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {//当ActivityCode == "Activity3"时触发，ActivityCode是节点编码名

$("#div605882").hide();//隐藏股转作市表单

$.ajax({

dataType: 'json',

type: 'GET',

url: '/Portal/kingdom/findDealNodeList',

success: function (data) {

var itemValue = data.data;//后端响应的数据

if (itemValue.length > 0) {

var diGrace = $.MvcSheetUI.GetSheetDataItem("liquidation\_monitor\_deal", 0);

//获取表单具体数据项内容，liquidation\_monitor\_deal为表单名， $.MvcSheetUI.GetSheetDataItem为控件获取数据项方法

for (i = 0; i < itemValue.length; i++) {

$("table[data-datafield=liquidation\_monitor\_deal]").SheetGridView(diGrace);//控件解析每一列数据项

$.MvcSheetUI.SetControlValue("liquidation\_monitor\_deal.node\_name", itemValue[i].name, i + 1);

//SetControlValue为控件对具体数据项赋值方法，共三个参数第一个为数据项名，第二项赋值的值，第三项为赋值所在行数。

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_name", i + 1).prop("disabled", true);

//GetElement方法用于获取表单dom元素，第一个参数为获取的数据项名，第二个为行数，prop方法为增加属性，此处为所需要的disable是不可编辑效果

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_name", i + 1).prop("title", itemValue[i].name);

//方法同上，此处是为了给数据项增加鼠标浮动上去时显示数据项内容

$.MvcSheetUI.SetControlValue("liquidation\_monitor\_deal.node\_status", itemValue[i].executeResult, i + 1);

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_status", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_status", i + 1).prop("title", itemValue[i].executeResult);

$.MvcSheetUI.SetControlValue("liquidation\_monitor\_deal.node\_execute\_result", itemValue[i].statusStr, i + 1);

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_execute\_result", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_monitor\_deal.node\_execute\_result", i + 1).prop("title", itemValue[i].statusStr);

}

}

},

error: function () {

alert("Failed");

},

});

setTimeout(function () {

$("#div605882").show();//延时函数达到展开收起的刷新效果

}, 1000)

}

}

restart = function () {

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

//股转作市刷新方法

}

NightOpen = function () {

//夜盘开市刷新方法

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity8") {

var NodataVal = document.getElementById("liquidation\_monitor\_moan\_noData");

if (NodataVal.innerHTML == "加载中" || NodataVal.innerHTML == "流程暂无执行数据")//获取页面数据，

{

$("#liquidation\_monitor\_moan\_noData").html("加载中");//为页面点击刷新时显示加载中

$("#liquidation\_monitor\_moan\_noData").show()

}

$("#div857555").hide();

$.ajax({

dataType: 'json',

type: 'GET',

url: '/Portal/kingdom/findMarketOpenAtNightNodeList?flowId=' + $.MvcSheetUI.SheetInfo.BizObjectID,

success: function (data) {

var itemValue = data.data;

var liquidation\_monitor\_moan\_refresh\_diGrace = $.MvcSheetUI.GetSheetDataItem("liquidation\_monitor\_moan", 0);

var digraceLength=$("table[data-datafield=liquidation\_monitor\_moan]").children().children().length-2

if (itemValue.length > 0) {

if (itemValue.length > digraceLength) {

/判断返回数据和原有数据的长度

var linePlus = itemValue.length – digraceLength

/设置变量用于判断循环次数

for (j = 0; j < linePlus; j++) {

$("table[data-datafield=liquidation\_monitor\_moan]").SheetGridView(liquidation\_monitor\_moan\_refresh\_diGrace).\_AddRow();

//添加行，注意选择器内填入正确的表名

digraceLength++;

}

}

if (itemValue.length < digraceLength) {

var lineMinus = digraceLength- itemValue.length

for (k = 0; k < lineMinus; k++) {

$("table[data-datafield=liquidation\_monitor\_moan]").SheetGridView(liquidation\_monitor\_moan\_refresh\_diGrace).\_Deleterow(

$("table[data-datafield=liquidation\_monitor\_moan]").children().children()[2 + k]);

//删除行方法(所传参数为删除行整行，是一个对象，此处获取整行的方法为

//$("table[data-datafield=liquidation\_monitor\_moan]").children().children()[2+k])

digraceLength--; }

} }

for (i = 0; i < itemValue.length; i++) {

//加载数据同下请求数据方法

}

else {//当没有数据时

var NodataVal = document.getElementById("liquidation\_monitor\_moan\_noData");

if (NodataVal.innerHTML == "" || NodataVal.innerHTML == "加载中") {

$("#liquidation\_monitor\_moan\_noData").html("流程暂无执行数据");

//先显示加载中若没有数据返回又显示回流程暂无数据

}

}

},

error: function () {

alert("Failed");

},

});

setTimeout(function () {

$("#div857555").show();

}, 1000)

}

}

NightCollect = function (obj) {

//夜间清算展开收起方法

var val = document.getElementById("NightCollect");

if (val.innerHTML == "展开") {

obj.innerHTML = "收起";//将展开替换为收起

$("#div761844").show();//展示表单

}

else if (val.innerHTML == "收起") {

obj.innerHTML = "展开";//将收起替换为展开

$("#div761844").hide();//收起表单

}

}

NightOpenCollect = function (obj) {

//夜盘开市展开收起方法

}

DayCollect = function (obj) {

//股转作市收起展开

}

DealCollect = function (obj) {

//投资交易系统展开收起

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity2") {

//隐藏其他节点的表单和删除格

$(".rowOption").hide();

$("#div294300").hide();

$("#div2943001").hide();

$("#div857555").hide();

$("#div605882").hide();

$("#div800237").hide();

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

$(".rowOption").hide();

//隐藏其他节点表单

$("#Control12\_Option").show();

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity8") {

$(".rowOption").hide();

$("#div2943001").hide();

$("#control3").hide();

$("#div605882").hide();

$("#div800237").hide();

//隐藏其他节点表单

$.MvcSheet.SubmitAction.OnActionPreDo = function () {

//.SubmitAction.OnActionPreDo此方法为点击提交后执行

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

$.MvcSheetUI.SetControlValue('node\_load\_status', load\_status[0].liquidation\_monitor\_stmarket\_finish)

//对状态标志位进行赋值，方便点击保存和提交后表单不再请求

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity8") {

$.MvcSheetUI.SetControlValue('node\_load\_status', load\_status[0].liquidation\_monitor\_moan\_finish)

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity2") {

$.MvcSheetUI.SetControlValue('node\_load\_status', load\_status[0].liquidation\_night\_problem\_finish)

$(".rowOption").hide();

}

};

$.MvcSheet.SaveAction.OnActionPreDo = function () {

//SaveAction.OnActionPreDo点击保存后方法，作用同上点击提交后方法，用于设置状态位的值

}

if ($.MvcSheetUI.GetControlValue('node\_load\_status') == 1) {

//获取状态标志位的值，

var liquidation\_night\_problem\_diGrace = $.MvcSheetUI.GetSheetDataItem("liquidation\_night\_problem", 0);

for (i = 0; i < liquidation\_night\_problem\_diGrace.V.R.length; i++) {

$(".InvalidText").hide()

$("table[data-datafield=liquidation\_night\_problem]").SheetGridView(liquidation\_night\_problem\_diGrace);

var night\_problem\_sys\_name = $.MvcSheetUI.GetControlValue("liquidation\_night\_problem.problem\_sys\_name", i + 1);

//GetControlValue方法用于获取数据项的值，此处获取值是为了方便对保存后的表单进行设置

var night\_problem\_ip\_address = $.MvcSheetUI.GetControlValue("liquidation\_night\_problem.problem\_ip\_address", i + 1);

var night\_problem\_operation\_step = $.MvcSheetUI.GetControlValue("liquidation\_night\_problem.problem\_operation\_step", i + 1);

var night\_problem\_check\_time = $.MvcSheetUI.GetControlValue("liquidation\_night\_problem.problem\_check\_time", i + 1);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_sys\_name", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_sys\_name", i + 1).prop("title", night\_problem\_sys\_name);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_ip\_address", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_ip\_address", i + 1).prop("title", night\_problem\_ip\_address);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_operation\_step", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_operation\_step", i + 1).prop("title", night\_problem\_operation\_step);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_check\_time", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_check\_time", i + 1).prop("title", night\_problem\_check\_time);

}

}

if ($.MvcSheetUI.GetControlValue('node\_load\_status') == 2) {

//参考标志位为2时的代码

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity2") {

$("#control3").hide();

$(".rowOption").hide();

if ($.MvcSheetUI.GetControlValue('node\_load\_status') == 0) {

$.ajax({

data: {

sEcho: 1,

iDisplayStart: 1,

iDisplayLength: 999,

status: 1,

},//封装的请求头

dataType: 'json',

type: 'POST',

url: '/Portal/liquidation/listLiquidationByPage',

success: function (data) {

var itemValue = data.datas;

if (itemValue.length > 0) {

var liquidation\_night\_problem\_diGrace = $.MvcSheetUI.GetSheetDataItem("liquidation\_night\_problem", 0);

$("table[data-datafield=liquidation\_night\_problem]").SheetGridView(liquidation\_night\_problem\_diGrace).\_Clear();

for (i = 0; i < itemValue.length; i++) {

var $pre = document.getElementsByClassName('pre');//操作步骤自适应实现插入的span的id名为pre，

$pre[0].innerHTML = itemValue[i].operateStep;//将操作步骤的值赋值给这个span达到撑开这个td的效果、同理要在保存后对这个span也进行赋值，但是在已办里面要删除

$("table[data-datafield=liquidation\_night\_problem]").SheetGridView(liquidation\_night\_problem\_diGrace).\_AddRow();

$.MvcSheetUI.SetControlValue("liquidation\_night\_problem.problem\_sys\_name", itemValue[i].name, i + 1);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_sys\_name", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_sys\_name", i + 1).prop("title", itemValue[i].name);

$.MvcSheetUI.SetControlValue("liquidation\_night\_problem.problem\_ip\_address", itemValue[i].ipAddress, i + 1);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_ip\_address", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_ip\_address", i + 1).prop("title", itemValue[i].ipAddress);

$.MvcSheetUI.SetControlValue("liquidation\_night\_problem.problem\_operation\_step", itemValue[i].operateStep, i + 1);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_operation\_step", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_operation\_step", i + 1).prop("title", itemValue[i].operateStep);

$.MvcSheetUI.SetControlValue("liquidation\_night\_problem.problem\_check\_time", itemValue[i].doTime, i + 1);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_check\_time", i + 1).prop("disabled", true);

$.MvcSheetUI.GetElement("liquidation\_night\_problem.problem\_check\_time", i + 1).prop("title", itemValue[i].doTime);;

}

$(".InvalidText").hide()//隐藏必填项多出来的星号符

}

},

error: function () {

alert("Failed");

},

});

}

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

if ($.MvcSheetUI.GetControlValue('node\_load\_status') == 2) {

//当节点到日间核对时请求

}

}

$.MvcSheet.Loaded = function () {

//表单加载后方法

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

$("#NightOpenRestart").hide();

//隐藏夜盘开市刷新按钮，之所以放在表单加载后方法是因为，异步加载的表单直接隐藏是不起效果的

}

$("#div761844").hide();

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity2") {

$("#DashedBlank").hide();//虚线分割线多余部分隐藏

$("#DashedLine").hide();

var val = document.getElementById("NightCollect");

if (val.innerHTML == "展开") {

$("#NightCollect").html("收起");

$("#div761844").show();//设置默认收起或者展开

}

}

if (!$.MvcSheetUI.SheetInfo.Close) {

//$.MvcSheetUI.SheetInfo.Close此参数可以知道此表单是已办还是代办

$("#refresh").hide();

$("#NightOpenRestart").hide();

$("#DealRefresh").hide();

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity3") {

$("#div857555").hide();

$("#NightOpenRestart").hide();

}

if ($.MvcSheetUI.SheetInfo.ActivityCode == "Activity8") {

$("#DashedLine").hide();

$("#DashedBlank").hide();

var val = document.getElementById("NightOpenCollect");

if (val.innerHTML == "展开") {

$("#NightOpenCollect").html("收起");

$("#div857555").show();

}

}

}

## Boolstrap多选下拉框

项目里面本身就有Boolstrap的多选下拉框组件，只需在config.router.js中的对应app中加载以下两个文件

'vendor/jquery/bootstrap-select-1.12.4/dist/css/bootstrap-select.min.css', 'vendor/jquery/bootstrap-select-1.12.4/dist/js/bootstrap-select.min.js',

除此之外，在所开发功能的js中需要加上这两句代码才能保证正常运行，

//使用refresh方法更新UI以匹配新状态

$('#usertype').selectpicker('refresh');

//render方法强制重新渲染引导程序 - 选择ui。

$('#usertype').selectpicker('render')

除此之外html页面需要这么写

<select id="usertype" class="selectpicker show-tick form-control"style="width:80px;height:29.6px" multiple ></select>

多选的属性是multiple，而下拉框的基础样式是由class名决定的（此class名是由加载的下拉框css决定的不能随意修改，可以百度或者查看下拉框的css去改成自己想要的样式）

对下拉框动态赋值选项（js实现方法）

var optionMulti=[

{ name: '全部', id: "0" },

{ name: '已完成', id: "1" },

{ name: '未完成', id: "2" },

{ name: '待完成', id: "3" },

]//定义一个对象数组，用于储存所需选项

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

$("#usertype").append($("<option value=\"" + optionMulti[i].id + "\">" + optionMulti[i].name + "</option>"));}

遍历定义的对象数组，利用jquery的append方法将选项动态加入

获取所选选项的值由以下方法实现

$("#usertype").val() ，返回的值为一个有字符串构成的数组，字符串内容是option的value值