# 平台改造方案（msBase）

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 版本 | 修改人 | 修改时间 |
| 1 | V0.1 | 谢永文 | 2016-11-24 |
| 2 |  |  |  |
| 3 |  |  |  |
| … |  |  |  |

* 项目背景

公司现有平台基于grails+Dojo架构，存在页面效果差、浏览器兼容性不好、Dojo开发资源少、前后台交互的二次开发量大等问题。为了解决以上问题，用grails+bootstrap+jQuery架构对平台进行改造。

* 平台改造内容

## 代码生成

根据domain里的注解配置动作和页面显示类型.使用脚本生成控制器和相应页面.

@MSAction 用于配置动作

@MSDomain 用于描述该类的显示名字,控制器的包名,控制器的名称,页面的包名

生成命令:

ms-generate-controller +domain类的路径(包名+domain类类名):生成控制器

ms-generate-view + :生成页面

ms-generate-all + :生成页面+控制器

### 1.1 @MSAction 注解

|  |  |  |
| --- | --- | --- |
| 参数名 | 特性 | 注释 |
| actionType | MSActionType 必填 | 设置动作的类型. 保存,删除,修改状态,页面显示方式…. |
| actionName | String 一般不用填 | 方法名 |
| msFileds | MSField 的数组,有些方法并不需要 | 用于指定一个动作里需要用到哪些属性  具体看下面的MSField说明 |

##### 1.1.1 @MSField 注解

|  |  |  |
| --- | --- | --- |
| 对象 | 特性 | 注释 |
| name |  | 字段名 |
| label |  | 方法名 |
| msTemplate |  | 修改状态, msFileds用来指示修改状态之后的值.  赋值 boolean值(或者其他对象):  @MSField(name = "enabled",label="无效",value = "false")  赋值字符串  @MSField(name = "userName",label="用户名",value = "\”aa\”") |
| value |  | 表单查看 |

##### 1.1.2 MSActionType 枚举类

|  |  |  |
| --- | --- | --- |
| 对象 | 特性 | 注释 |
| DELETE | 不需要msFileds | 删除 |
| SAVEORUPDATE |  | 方法名 |
| CHANGEOFSTATE |  | 修改状态, msFileds用来指示修改状态之后的值.  赋值 boolean值(或者其他对象):  @MSField(name = "enabled",label="无效",value = "false")  赋值字符串  @MSField(name = "userName",label="用户名",value = "\”aa\”") |
| TABLEVIEW | 不需要msFileds | 表单查看 |
| TABLEVIEWSEARCH |  | msFileds用来指示需要查询的field |

##### 1.1.3 MSTemplate枚举类

|  |  |  |
| --- | --- | --- |
| 属性 | 特性 | 注释 |
| url |  | 用于指定生成代码的文件名(src\templates\msbase\fields) |

### @MSDomain注解

修饰一个类

|  |  |  |
| --- | --- | --- |
| 属性 | 特性 | 注释 |
| name | 必须 | 指这个类的中文名 |
| controllerName | 除非有特殊需求,一般不需要指定,默认值为:  控制器名字==domain的名字+Controller  控制器包名==domain的包名  页面包名==控制器名 | 控制器名称 |
| controllerPackage | 控制器的包名 |
| viewPackage | 页面包名 |

## 表设计

#### 2.1多对多

Class M1{

static hasMany = [m2s:M2]

static mapping = {

m2s joinTable: [name: "tb\_m1\_m2", key: 'm1\_id']

}

static belongsTo = [M2]

}

Class M2{

static hasMany = [m1s:M1]

static mapping = {

m1s joinTable: [name: "tb\_m1\_m2", key: 'm2\_id']

}

}

持有的那一方(M2)删除的时候会自动删除中间表的数据,但是被持有的那一方(M1)删除之前必须手动删除中间表的数据.(在M1 domain类中添加下列方法)

def beforeDelete(){

M2.withNewSession {

M2.createCriteria().list{m1s{eq("id",id)}}?.each { M2 m2->

m2.m1s.removeAll {m1->

m1.id==id

}

println m2.m1s

m2.save(flush:true)

}

}

}

#### 2.2一对多

Class M1{

static hasMany = [m2s:M2]

static mapping = {

m2s cascade: 'all-delete-orphan'

}

}

Class M2{

static belongsTo = [m1:M1]

}

持有的那一方(M1)删除的时候会自动删除m2的对象,被持有的那一方(M2)删除之前必须手动删除(在m2中定义如下方法即可)

def beforeDelete(){

M1.withNewSession {

M1.createCriteria().list{m2s{eq("id",id)}}?.each { M1 m1->

m1. m2s.removeAll {m2->

m2.id==id

}

m1.save(flush:true)

}

}

}

#### 2.3一对一

#### 2.3.1 双向一对一

class P{

C c

}

class C{

static belongsTo = [p:P]

}

当C的对象要删除的时候需要先将关联的p里的c属性设置为null ,因此需要修改在控制器的delete

def cDelete(){

def model=[:]

def result=true

params.ids.split(',').grep()?.each{

def c =C.get(it)

if(c){

c.P?c.p.c=null:null

c.delete()

}else{

result=false

}

}

model["result"]=result

render ( model as JSON)

}

#### 2.3.2单向一对一

class A{

P p

}

def beforeInsert() {

TestP.withNewSession {

p?.save(flush: true)

}

}

如果需要用A来创建p的话 那么在beforeInsert 中需要先将p进行保存

## 3.数据列表交互

使用datatables表格插件进行实现http://datatables.club/

在domain类使用GridColumn注解对属性和方法进行修饰用来实现数据列表的每个 列 的显示和设置

在view层使用ms:dataTable 标签实现数据列表的显示和设置

在controller层使用GridDataUtil工具类的gridList()方法

### 3.1@GridColumn注解

|  |  |  |
| --- | --- | --- |
| 参数名 | 特性 | 注释 |
| name | String 必填 | 设置列的标签名.(中文显示名) |
| colIdx | Int 默认值100 | 列的排序顺序 |
| unusableTableIds | Array | 设置不需要这个列的表格id.  例子:  unusableTableIds=[‘a’,’b’]  那么id=’a’ 和 id=’b’ 的表格不会存在该字段 |
| target | String | **当GridColumn修饰方法的时候必须拥有该属性**.用于指定该方法排序和搜索时的字段名称.支持model.modelName 这样的写法..  *如果target指定的字段并不在该显示列表中,该column的name值将会设置为该target值* |
| htmlEncode | Boolean 默认值为true | 是否转义,如果返回的值会是一个标签,请将该属性设置为false |
| defaultOrder | Boolean 默认值为false | 该列是否为默认的排序字段 |
| defaultOrderType | String 默认为 desc | defaultOrder 为true时用来选择默认的排序方法 |
| defaultOrderIndex | Int 默认为100 | 默认排序的时候的顺序(值小的先排序) |

其余属性值去看datatables官网的关于Columns的描述<https://datatables.net/reference/option/>

### 3.2ms:dataTable

|  |  |  |
| --- | --- | --- |
| 参数名 | 特性 | 注释 |
| domainClass | Class 必填 | 根据所填的Class去生成表格 |
| id | String 必填 | Table的id |
| multiSelect | 默认为true | 是否拥有多选框 |
| orderNumber | 默认为true | 是否拥有序列 |
| actionList | List<Map> | 用于产生最右边的操作按钮.  例子:  [ [ action:'编辑', class:'edit', [url:createLink(controller:'user',action:'userSaveView')](file:///D:\IdeaGrails2.5.5\svn\msbase\doc\createLink(controller:'user',action:'userSaveView')) ,  method:’ enabledShow’  ]]  Map 里的action属性代表显示按钮的值  url:请求路径  Class 代表显示按钮的class属性 其中edit 和del是已经写好的都会携带一个id参数.edit 代表请求一个url,该url会返回一个页面作为编辑页面,del则是要返回一个json对象,该json对象里有一个result属性(boolean)用于判断是否成功.  method:指domain类的一个方法名,该方法  如果返回的是boolean类型 true代表显示,false代表不显示  如果返回的是 String ,返回值将会替代action里的值,如果返回null或者’’按钮将会不显示 |
| serverSide | 默认值为true | 用于决定是否使用datatables的serverSide模式  true:代表使用服务器端,那么查询,排序和分页是向后台进行请求  false:则会一次性读取所有数据,查询,排序都是使用前台处理 |
| defaultScript | 默认值为false | 使用默认的Script(不需要再页面里写关于datatables的js代码)  注意:必须有url |
| url | String | 当使用defaultScript时 必须填写该属性 |

### 3.3 GridDataUtil工具类

#### 3.3.1 gridList()

|  |  |  |
| --- | --- | --- |
| 参数名 | 特性 | 注释 |
| domainClass | Class | 需要搜索的表 |
| params | Map | params |
| query | Closure | 一个闭包  例子:看 grails 2.3手册的 第六章 4.3 的内容  闭包拥有的方法:  http://gorm.grails.org/latest/api/org/grails/datastore/mapping/query/api/Criteria.html  闭包内所添加的参数是一个map对象有如下key.  searchs :map对象 里面会有搜索的条件,key为column的name属性 value为搜索的值  orders : map 对象 排序依据 同上  offset : 从第几条开始显示 ,不用管  max :每页的最大值 ,不用管 |
| autoSort | 默认为true | 自动排序 ,如果没有特殊情况的话 就开着 |

备注:*主键字段名称必须为id*

1. 前端布局
2. 前端控件
   1. msTree数据

4.1.1 msTree相关依赖文件

使用该插件请先引入jquery、bootstarp(js和css)、ztree(js和css)文件，同时根据需要选择引入不同tree插件，该封装插件考虑网络、等待时间相关因素，点击父节点加载子节点形式。Demo请下载cylw项目后启动，访问地址为/cylw/tree/testTree：具体请看以下介绍：

1. 如需使用下拉选形式树，请引用msInputTree.js文件。
2. 如需使用勾选形式树，请引用msCheckboxTree.js文件。
3. 如需进行增删改操作的树，请引用msOperateTree.js文件。
4. 注意：在引用msTree插件css时请将tree文件夹放入到web-app/css文件夹下，或将tree下的metroStyle.css文件和img文件夹统一放到web-app/css文件夹下

4.1.2 msTree下拉选(msInputTree.js)属性及方法

1. 属性：

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 属性值 | 属性值类型 | 应用对象 |
| data-ms-tree | msTree实例容器名称 | String | input |

1. 举例说明：

<input id="text1" type="text" data-ms-tree="treeDemo"/>，在此段html代码中，data-ms-tree属性后的容器名称必须是页面不存在的容器名称，如果存在则无法实例化msTree插件。

1. 方法：

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数 | 返回值 | 方法解释 |
| msTreeInput(holder,url) | holder：实例容器名称对应data-ms-tree中的容器名称、url：获取数据后台url | 无 | 调用该方法初始化下拉选树，需传入容器名称及获取数据url请求地址。双击msTree节点（地区名、机构名、部门名）将节点名称写入带有data-ms-tree属性input框。 |
| getInputCode(treeId) | treeId：实例容器名称 | String | 使用该方法能够获取到data-ms-tree属性input框中节点（地区名、机构名、部门名）对应的代码。 |

1. 举例：

树测试文本框1：<input id="text1" type="text" data-ms-tree="treeDemo"/>

<button type="button" id="get1">获取文本框1中名称对应的代码</button>

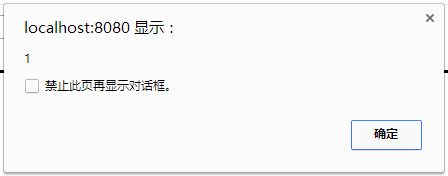
$("#get1").click(function(){

var value = getInputCode("treeDemo");

alert(value);

});





4.1.3 msTree勾选(msCheckboxTree.js)方法

1. 方法：

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数 | 返回值 | 方法解释 |
| msTreeCheckbox(holder,url) | holder：实例容器名称，url：获取数据后台url | 无 | 初始化带有复选框的msTree实例。需要在页面提供容器。格式为div+ul格式 |
| getMsTreeCheck(treeId) | treeId：实例容器名称 | Json字符串 | 获取页面中已展开节点并勾选的节点。返回勾选节点的id、name、pId、pName属性。 |
| getMsTreeNoCheck(treeId) | treeId：实例容器名称 | Json字符串 | 获取页面中展开节点未勾选节点，返回勾选节点的id、name、pId、pName属性。 |

1. 详情请看/cylw/tree/testTree demo演示案例

4.1.4 msTree操作(msOperateTree.js)属性及方法

1. 属性

|  |  |  |
| --- | --- | --- |
| 属性名称 | 应用对象 | 属性解释 |
| data-ms-tree-operate-id | Input | 内嵌页面input应用此属性后，编辑时将所点击节点的id写入到该input，新增时将所点击节点id写入到内嵌页面input属性为data-ms-tree-operate-pId的元素中。 |
| data-ms-tree-operate-name | Input | 内嵌页面input应用此属性后，编辑时将所点击节点的name写入到该input，新增时将所点击节点name写入到内嵌页面input属性为data-ms-tree-operate-pName的元素中。 |
| data-ms-tree-operate-pid | Input | 内嵌页面input应用此属性后，编辑时将所点击节点的pId写入到该input，新增时将所点击节点id写入到内嵌页面input属性为data-ms-tree-operate-pId的元素中。 |
| data-ms-tree-operate-pname | Input | 内嵌页面input应用此属性后，编辑时将所点击节点的pName写入到该input，新增时将所点击节点name写入到内嵌页面input属性为data-ms-tree-operate-pName的元素中。 |
| newPageOperateHolder | Div | 内嵌页面容器divid名称 |
| editMsTreeNodeUrl |  | 此属性为后台javaBean必须提供属性，该属性用于保存点击编辑按钮时，向所点击节点获取后台返回数据地址 |
| delMsTreeNodeUrl |  | 此属性为后台javaBean必须提供属性，该属性用于保存点击删除按钮时，向所点击节点获取是否删除成功后台地址。 |
| loadMsTreeNodeUrl |  | 此属性为后台javaBean必须提供属性，该属性用于保存内嵌页面地址。 |

1. 方法:

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名 | 参数 | 返回值 | 方法解释 |
| msTreeOperate(holder,url) | holder：实例容器名称，url：获取数据后台url | 无 | 初始化操作的msTree实例。需要在页面提供容器。格式为div+ul格式 |
| msTreeOperateUpdateNode(treeId,isSilent) | treeId：需要更新的实例容器名称，isSilent：更新后是否展开节点 | 无 | 该方法如果只传入treeid则更新根节点，如传入treeId、true表示更新容器中选中节点把展开子节点，反之则展开子节点。 |

4.1.5 msTree后台工具类

1. 该工具类名称：MsTree.java

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 类型 | 是否必须 | 属性值 |
| id | string | Y | 节点id |
| name | string | Y | 节点名称 |
| pId | string | Y | 节点pid |
| parent | int | Y | 是否为父节点，0：父，1：子 |
| chked | int | 初始化复选框树时必须 | 是否勾选，0：勾选，1：不勾选 |
| loadMsTreeNodeUrl | string | 初始化为操作树时必须 | 存放内嵌页面到指定容器url地址，相对路径。"../abc.html" |
| editMsTreeNodeUrl | string | 初始化为操作树时必须 | 点击编辑时用于向后台获取数据的action地址，填写相对地址。"../boatArea/testEditaArea" |
| delMsTreeNodeUrl | string | 初始化为操作树时必须 | 点击删除时用于删除节点的后台action地址，请使用相对地址。"../boatArea/testDelArea" |
|  |  |  |  |
|  |  |  |  |

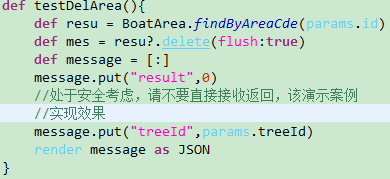
1. 初始化树举例(后台编码)：

初始化操作树，首先引入相关依赖文件后引入msOperateTree.js文件，前面有所介绍这里不一一介绍了。

初始化时，页面会向后台传入两个参数id、name。后台只需判断是否有参数传入，根据传入情况做相关操作，最后将从数据库中获得的数据集合用工具类转换为定义的格式即可。如下图所示：



1. editMsTreeNodeUrl、delMsTreeNodeUrl地址对应方法中，页面会向该方法中传入nodeId、treeId，返回数据时请将返回treeId返回，该插件中前端成功回调判断成功标志"result",0。如下图所示；



## 4.2 input数据校验

### 4.2.1 相关依赖文件

1. 引入jquery、bootstarp后，需要引入checks.js。
2. 属性介绍

|  |  |  |  |
| --- | --- | --- | --- |
| 属性名称 | 属性值类型 | 应用对象 | 解释 |
| data-ms-validate | 无 | input | 该属性用于为元素绑定获得焦点失去失去焦点事件 |
| data-ms-validate-phone | 无 | input | 验证手机号 |
| data-ms-validate-idcard | 无 | input | 验证身份证号 |
| data-ms-validate-email | 无 | input | 验证邮箱 |
| data-ms-validate-max | String必须 | input | 验证最大长度 |
| data-ms-validate-min | String必须 | input | 验证最小长度 |
| data-ms-validate-leftblank | 无 | input | 去除左空格 |
| data-ms-validate-rightblank | 无 | input | 去除右空格 |
| data-ms-validate-allblank | 无 | input | 去除所有空格 |
| data-ms-validate-isnotnull | 无 | input | 非空验证 |
| data-ms-validate-integral | 无 | input | 整数验证 |
| data-ms-validate-small | 无 | input | 小数验证 |

1. 方法

|  |  |  |  |
| --- | --- | --- | --- |
| 方法名称 | 参数 | 参数类型 | 解释 |
| validate（） | 无 | 无 | 为选择区域内的元素校验 |
| formSubmitCheck(formElem) | 有 | html元素 | 就该方法用于校验选定区域是否可以提交 |

## 4.3日期插件

<http://www.bootcss.com/p/bootstrap-datetimepicker/index.htm>

初始化代码都在msInit.js里了

调用时

<input type="text" readonly class="date-picker form-control search-input"/>

class:

“date-picker”初始化的是日期选择器

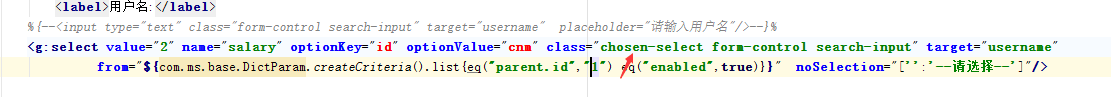
“time-picker”初始化的是时间选择器(精确到分钟)

## 4.4下拉选择框

<http://harvesthq.github.io/chosen/>

初始化代码在msInit.js

调用时



加一个chosen-select 的class 即可

## 4.5 表格编辑器

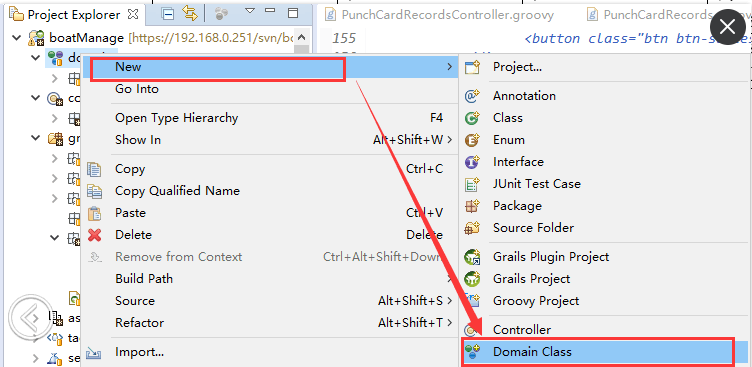
https://editor.datatables.net/

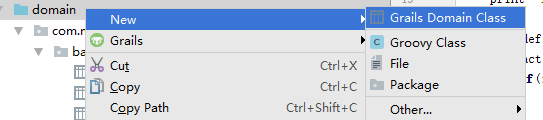
例子:

Msbase->views->zmd->tableAdd

## 6.示例

GGTS的

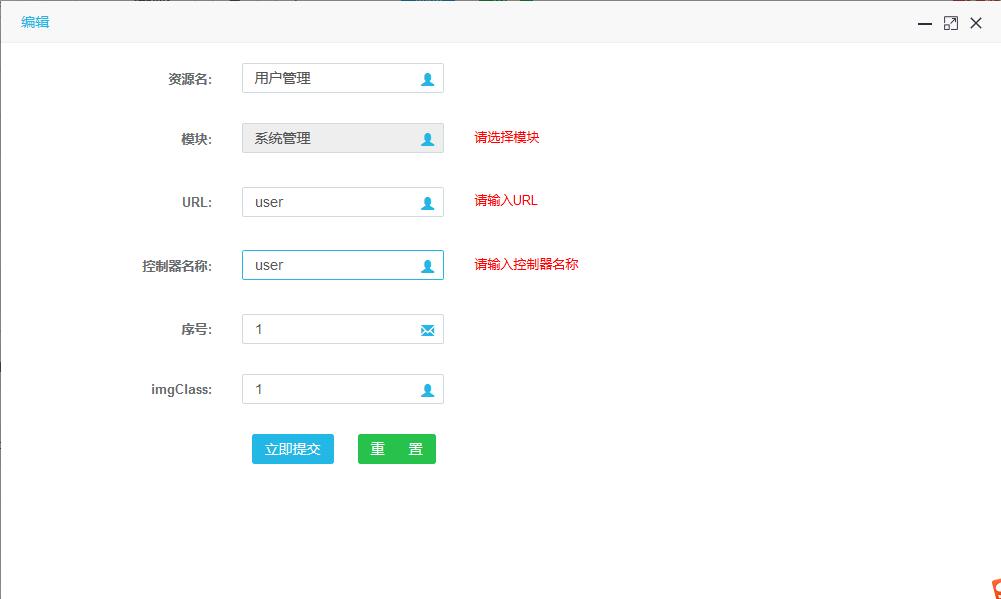


Idea: 

创建User类,配置好@ MSActions 和@ MSDomain后.

运行命令行ms-generate-controller com.ms.base.User.

然后添加resource即可



## 7.代码规则

#### 7.1 domain

1.包名必须从 com.ms 开始//配置文件中对com.ms开始的domain类做了特殊处理

2.必须要有 String id 字段

3.一个类中作为显示效果的字段推荐使用 类名(首字母小写)+Name

//不然要在调用多选或者树工具类的方法的时候需要多加一个配置显示字段名的参数

#### 7.2 Controller

## 8.其他工具类

### 8.1 时间工具类

Date. msParse(String date)

将字符串解析为Date对象 支持多种转换格式

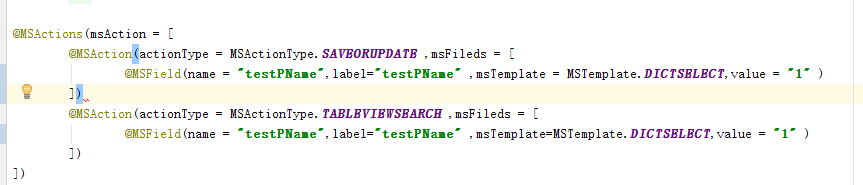
Date对象. msToDateStart() 将日期对象转换成当天的0点0分0秒

Date对象. msToDateEnd() 将日期对象转换成当天的23点59分59秒

## 9.版本更新说明

0.2.0版本更新说明

1.MSTemplate 添加了DICTSELECT对象 用于描述字典表的字段例子:



其中value用来指定字典表的上级id

0.2.9 更新:

添加可搜索的下拉选



在select中添加一个 chosen-select 的class