

JEECG 微云快速开发平台

开发指南

2017/03/13

www.jeecg.org

张代浩

目 录

1.	前言.....	1
1.1.	1.1. 技术背景.....	1
1.2.	1.2. 平台介绍.....	1
1.3.	1.3. 平台优势.....	1
1.4.	1.4. 功能特点.....	2
1.5.	1.5. 技术支持.....	4
2.	JEECG框架初探.....	4
2.1.	2.1. 演示系统.....	4
2.2.	2.2. 示例代码.....	6
3.	JEECG目录结构.....	7
3.1.	3.1. 配置文件目录结构.....	7
3.2.	3.2. Java源码目录结构.....	8
3.3.	3.3. 单元测试代码结构.....	8
3.4.	3.4. JSP页面目录结构.....	9
4.	代码生成器.....	9
4.1.	4.1. 功能介绍.....	9
4.2.	4.2. 代码生成器配置.....	10
4.3.	4.3. Online代码生成器.....	11
4.3.1.	4.3.1. 配置表单.....	11
4.3.2.	4.3.2. 同步数据库.....	12
4.3.3.	4.3.3. 功能测试.....	12
4.3.4.	4.3.4. 代码生成.....	13
4.4.	4.4. GUI代码生成器.....	14
4.4.1.	4.4.1. 建表.....	14
4.4.2.	4.4.2. 代码生成.....	14
4.4.3.	4.4.3. 配置扫描路径.....	16
4.4.4.	4.4.4. 功能测试.....	17
4.4.5.	4.4.5. 代码生成器使用规则.....	19
4.4.6.	4.4.6. 一对多的代码生成.....	20
5.	Online表单开发.....	21
5.1.	5.1. 原理.....	21
5.2.	5.2. 使用.....	22
5.3.	5.3. Online表单风格.....	23
5.3.1.	5.3.1. 介绍.....	23
5.3.2.	5.3.2. 自定义风格方法.....	23
5.3.3.	5.3.3. 风格模板命名.....	24
5.3.4.	5.3.4. 风格上传.....	24
6.	Online报表配置.....	26
6.1.	6.1. 原理.....	26
6.2.	6.2. 使用.....	26
6.3.	6.3. Online图表配置.....	28
7.	查询过滤器.....	30
7.1.	7.1. 功能描述.....	30
7.2.	7.2. 查询条件如何实现.....	30
7.3.	7.3. 查询过滤器高级特性.....	31
7.3.1.	7.3.1. 组合条件查询.....	31

7.3.2.	字段范围查询.....	32
7.3.3.	日期字段的数据格式化.....	32
7.4.	查询规则.....	32
8.	基础用户权限.....	33
8.1.	权限设计原理.....	33
8.2.	权限设计目标.....	34
8.3.	权限数据表.....	34
8.4.	权限设计实现.....	35
8.4.1.	访问菜单权限.....	35
8.4.2.	列表按钮权限.....	35
8.4.3.	页面表单权限.....	39
8.4.4.	页面表单权限标签.....	40
8.4.5.	数据权限控制.....	41
8.4.6.	数据权限当前用户上下文变量.....	42
9.	多数据源.....	43
9.1.	多数据源背景.....	43
9.2.	多数据源的配置.....	43
9.3.	多数据源的使用.....	44
10.	国际化.....	45
10.1.	国际化背景.....	45
10.2.	国际化语言维护.....	45
10.3.	国际化标签用法 t:mutiLang.....	45
10.3.1.	参数.....	45
10.3.2.	用法.....	45
10.4.	其他标签国际化用法.....	46
10.4.1.	列表datagrid.....	46
10.4.2.	列表字段.....	46
10.4.3.	列表按钮.....	46
11.	定时任务.....	46
11.1.	定时任务配置文件.....	46
11.2.	定时任务在线管理.....	47
12.	消息中心.....	48
12.1.	简介.....	48
12.2.	使用方式.....	48
12.3.	使用详解.....	49
12.4.	系统配置文件.....	52
13.	插件模块集成文档.....	53
13.1.	在线聊天插件.....	53
14.	附录.....	54
14.1.	UI标签规则.....	54
14.2.	列表自定义查询条件.....	54
14.3.	Formvalid新增属性tiptype的使用.....	56
14.4.	使用toolbar 自定义js参数规则.....	57
14.5.	表单字段重复校验方法.....	58
14.6.	数据列表合计功能.....	58
14.7.	登录权限拦截器排除方法.....	60
14.8.	列表拓展字段展示.....	60
14.9.	JEECG常见问题解决贴.....	61

1. 前言

1.1. 技术背景

随着 WEB UI 框架（EasyUI/Jquery UI/Ext/DWZ）等的逐渐成熟,系统界面逐渐实现统一化,代码生成器也可以生成统一规范的界面! 代码生成+手工 MERGE 半智能开发将是新的趋势,单表数据模型和一对多数据模型的增删改查功能直接生成使用,可节省60%工作量,快速提高开发效率。

1.2. 平台介绍

JEECG (J2EE Code Generation) 是一款基于代码生成器的智能开发平台。引领新的开发模式(Online Coding->代码生成器->手工 MERGE 智能开发),可以帮助解决 Java 项目60%的重复工作,让开发更多关注业务逻辑。既能快速提高开发效率,帮助公司节省人力成本,同时又不失灵活性。

JEECG 宗旨是:简单功能由代 Online Coding 配置出功能;复杂功能由代码生成器生成进行手工 Merge;复杂流程业务采用表单自定义,业务流程使用工作流来实现、扩展出任务接口,供开发编写业务逻辑。实现了流程任务节点和任务接口的灵活配置,既保证了公司流程的保密行,又减少了开发人员的工作量。

1.3. 平台优势

- 采用主流框架,容易上手;代码生成器依赖性低,很方便的扩展能力,可完全实现二次开发
- 开发效率很高,采用代码生成器,单表数据模型和一对多(父子表)数据模型,增删改查功能自动生成,菜单配置直接使用
- 页面校验自动生成(必须输入、数字校验、金额校验、时间空间等)
- 封装完善的用户基础权限、强大的数据权限、和数据字典等基础功能,直接使用无需修改
- 常用共通封装,各种工具类(定时任务,短信接口,邮件发送,Excel 导出等),基本满足 80%项目需求
- 集成简易报表工具,图像报表和数据导出非常方便,可极其方便的生成 pdf、excel、word 等报表

- 集成 workflow activiti, 并实现了只需在页面配置流程转向, 可极大的简化 jbpw 工作流的开发; 用 jbpw 的流程设计器画出了流程走向, 一个 workflow 基本就完成了, 只需写很少量的 java 代码
- UI 标签库, 针对 WEB UI 进行标准式封装, 页面统一采用自定义标签实现功能: 列表数据展现、页面校验等, 标签使用简单清晰且便于维护
- 在线流程设计, 采用开源 Activiti 流程引擎, 实现在线画流程, 自定义表单, 表单挂靠, 业务流转
- 查询过滤器: 查询功能自动生成, 后台动态拼 SQL 追加查询条件; 支持多种匹配方式 (全匹配/模糊查询/包含查询/不匹配查询)
- 多数据源: 及其简易的使用方式, 在线配置数据源配置, 便捷的从其他数据抓取数据
- 国际化: 支持多语言, 开发国际化项目非常方便
- 数据权限 (精细化数据权限控制, 控制到行级, 列表级, 表单字段级, 实现不同人看不同数据, 不同人对同一个页面操作不同字段)
- 多种首页风格切换, 支持自定义首页风格。(经典风格、Shortcut 风格、ACE bootstrap 风格、云桌面风格)
- 在线配置报表 (无需编码, 通过在线配置方式, 实现曲线图, 柱状图, 数据等报表)
- 简易 Excel 导入导出, 支持单表导出和一对多表模式导出, 生成的代码自带导入导出功能
- 自定义表单, 支持用户自定义表单布局, 支持单表, 一对多表单、支持 select、radio、checkbox、textarea、date、popup、列表、宏等控件

1.4. 功能特点

- 采用 SpringMVC + Hibernate + Minidao (类 Mybatis) + Easyui (UI 库) + JQuery + Bootstrap + Ehcache + Redis + Ztree 等基础架构
- 采用面向声明的开发模式, 基于泛型编写极少代码即可实现复杂的数据展示、数据编辑、表单处理等功能, 再配合 Online Coding 在线开发与代码生成器的使用, 将 J2EE 的开发效率提高 6 倍以上, 可以将代码减少 80% 以上。
- **JEECG 技术点总结:**

- ① 技术点一：Online Coding 在线开发(通过在线配置实现一个表模型的增删改查功能，无需一行代码，支持用户自定义表单布局)
- ② 技术点二：代码生成器，支持多种数据模型，根据表生成对应的 Entity, Service, Dao, Action, JSP 等，增删改查功能生成直接使用
- ③ 技术点三：UI 快速开发库，针对 WEB UI 进行标准封装，页面统一采用 UI 标签实现功能：数据 datagrid, 表单校验, Popup, Tab 等，实现 JSP 页面零 JS，开发维护非常高效
- ④ 技术点四：在线流程定义，采用开源 Activiti 流程引擎，实现在线画流程，自定义表单, 表单挂接, 业务流转，流程监控，流程跟踪，流程委托等
- ⑤ 技术点五：自定义表单, 支持用户自定义表单布局，支持单表、列表、Select\Radio\Checkbox\PopUP\Date 等特殊控件
- ⑥ 技术点六：查询过滤器：查询功能自动生成，后台动态拼 SQL 追加查询条件；支持多种匹配方式（全匹配/模糊查询/包含查询/不匹配查询）
- ⑦ 技术点七：移动平台支持，对 Bootstrap(兼容 Htm15) 进行标准封装
- ⑧ 技术点八：动态报表功能（用户输入一个 sql，系统自动解析生成报表）
- ⑨ 技术点九：数据权限（精细化数据权限控制，控制到行级，列表级，表单字段级，实现不同人看不同数据，不同人对同一个页面操作不同字段）
- ⑩ 技术点十：国际化（支持多语言，国际化的封装为多语言做了便捷支持）
- 11 技术点十一：多数据源（在线配置数据源，数据源工作类封装）
- 12 技术点十二：多种首页风格切换, 支持自定义首页风格。（经典风格、Shortcut 风格、ACE bootstrap 风格、云桌面风格）
- 13 技术点十三：在线配置报表（无需编码，通过在线配置方式，实现曲线图，柱状图，数据等报表）
- 14 技术点十四：简易 Excel 导入导出，支持单表导出和一对多表模式导出，生成的代码自带导入导出功能
- 15 技术点十五：移动 OA，移动 OA 审批功能，采用 H5 技术，实现手机移动办公，无缝对接微信、钉钉、微信企业号、也可以做 APP
- 16 技术点十六：移动图表，在线配置移动报表，采用 H5 技术，可以手机端查看
- 17 技术点十七：插件开发，业务功能组件以插件方式集成平台，也可以单独部署发发布，有力支撑了 SAAS 云应用系统需求

- JEECG V3.7,经过了专业压力测试,性能测试,保证后台数据的准确性和页面访问速度
- 支持多种浏览器: IE, 谷歌, 火狐等
- 支持数据库: Mysql,Oracle11g,Postgre,SqlServer 等
- 基础权限: 用户, 角色, 菜单权限, 按钮权限, 数据权限, 表单权限
- 智能报表集成: 简易的图像报表工具和 Excel 导入导出
- Web 容器测试通过的有 Jetty 和 Tomcat6,Weblogic
- 即将推出功能: 分布式部署, 云平台, 插件式开发平台, 移动平台开发, 规则引擎
- 要求 JDK1.6+

1.5. 技术支持

- 作者: 张代浩
- QQ 交流群: ③289782002、④176031980 (满)、②106838471 (满)、①106259349 (满)
- 邮箱: jeecg@sina.com
- 技术论坛: www.jeecg.org

2. JEECG框架初探

2.1. 演示系统

打开浏览器, 输入JEECG演示环境地址: <http://demo.jeecg.org>, 可以看到如图 2-1 所示的登录界面。



图 2-1 演示系统登录界面

点击【登陆】按钮，进入演示系统的主界面，如图 2-2 所示。



图 2-2 演示系统主界面

在 JEECG 演示系统中的功能模块包括:ONLINE 开发、自定义表单、在线报表开发、在线图表开发、在线移动报表开发、消息推送、系统监控、统计管理、系统管理、常用示例、数据权限、工作流等。其中，流程设计器、表单设计器的界面截图如图 2-3 和图 2-4 所示。

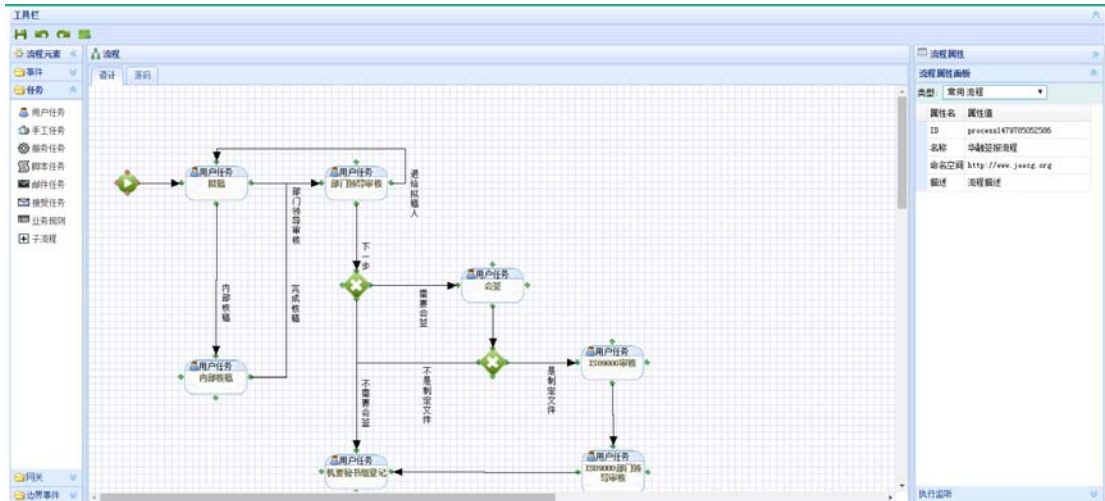


图 2-3 流程设计器界面

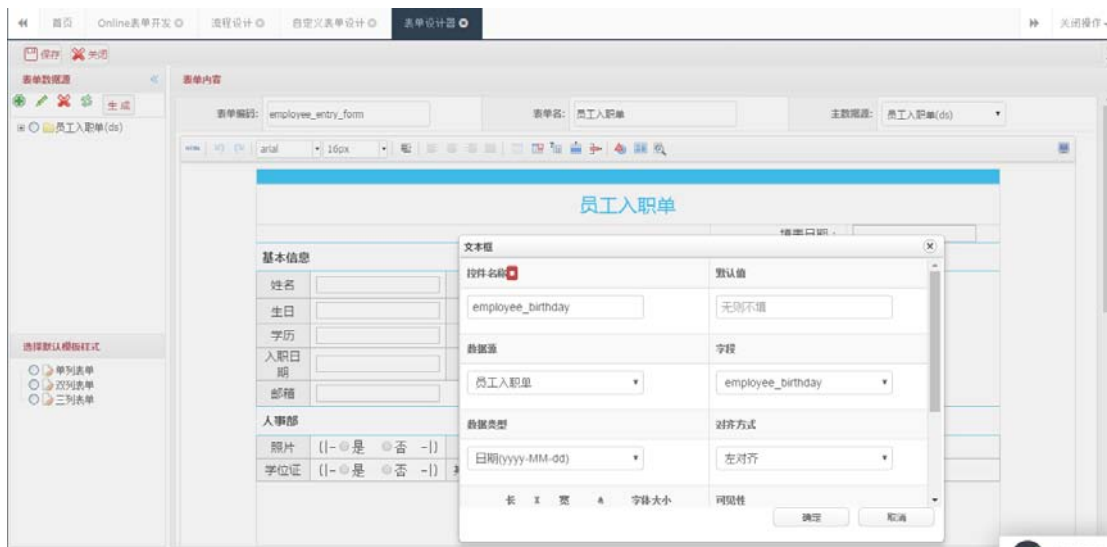


图 2-4 表单设计器界面

2.2. 示例代码

用户管理中的用户列表和用户表单所用的 jsp 页面代码分别如图2-5和图2-6所示。

```
<div class="easyui-layout" fit="true">
<div region="center" style="padding:0px;border:0px">
<datagrid name="jeecgDemoList" title="DEMO示例列表" autoLoadData="true" actionUrl="jeecgDemoController.do?datagrid" fitColumns="true"
idField="id" fit="true" queryMode="group" checkBox="true" queryBuilder="true">
<tdgCol title="编号" field="id" hidden="true"></tdgCol>
<tdgCol title="用户名" field="userName" query="true" frozenColumn="true" extend="{style:{width:300px;color:red};datatype:''}" defaultVal="小强" width="120"></tdgCol>
<tdgCol title="电话号码" sortable="false" field="mobilePhone" query="true" width="120"></tdgCol>
<tdgCol title="办公电话" field="officePhone" width="120"></tdgCol>
<tdgCol title="创建日期" field="createDate" editor="datebox" formatter="yyyy-MM-dd hh:mm:ss" query="true" queryMode="group" width="200"></tdgCol>
<tdgCol title="邮箱" field="email" width="200"></tdgCol>
<tdgCol title="年龄" sortable="true" editor="numberbox" field="age" width="80"></tdgCol>
<tdgCol title="工资" field="salary" width="120" queryMode="group" query="true"></tdgCol>
<tdgCol title="生日" field="birthday" formatter="yyyy/MM/dd" width="120"></tdgCol>
<tdgCol title="性别" field="sex" dictionary="sex" query="true" width="60"></tdgCol>
<tdgCol title="状态" field="status" replace="未处理,已处理" width="60" query="true" defaultVal="1"></tdgCol>
<tdgCol title="操作" field="opt" width="150"></tdgCol>
<tdgFuncOpt exp="status$eq$0" operationCode="ssm" funname="ssm(id)" title="审核" />
<tdgDelOpt operationCode="del" title="删除" url="jeecgDemoController.do?del{id={id}" exp="status$eq$0" urlStyle="color:red;padding-left:5px;"/>
<tdgDelOpt operationCode="del" title="删除" url="jeecgDemoController.do?del{id={id}" exp="status$eq$1" urlStyle="color:green;padding-right:5px;"/>
<tdgToolBar operationCode="add" title="添加" icon="icon-add" url="jeecgDemoController.do?addorupdate" funname="add"></tdgToolBar>
<tdgToolBar operationCode="edit" title="编辑" icon="icon-edit" url="jeecgDemoController.do?addorupdate" funname="update"></tdgToolBar>
<tdgToolBar operationCode="mobileAdd" title="Mobile添加" icon="icon-add" url="jeecgDemoController.do?addorupdatemobile" funname="addMobile"></tdgToolBar>
<tdgToolBar operationCode="mobileEdit" title="Mobile编辑" icon="icon-edit" url="jeecgDemoController.do?addorupdatemobile" funname="updateMobile"></tdgToolBar>
<tdgToolBar operationCode="detail" title="查看" icon="icon-search" url="jeecgDemoController.do?addorupdate" funname="detail"></tdgToolBar>
<tdgToolBar operationCode="print" title="打印" icon="icon-print" url="jeecgDemoController.do?print" funname="detail"></tdgToolBar>
<tdgToolBar title="彻底删除" icon="icon-remove" url="jeecgDemoController.do?deleteAllSelect" funname="deleteAllSelect"></tdgToolBar>
<tdgToolBar title="导入" icon="icon-out" url="transdata.do?doImportData" funname="doImportData"></tdgToolBar>
<tdgToolBar title="导出" icon="icon-outout" url="transdata.do?doExportData" funname="doExportData"></tdgToolBar>
<tdgToolBar title="加载百度" icon="icon-print" url="#" funname="testReloadPage"></tdgToolBar>
</datagrid>
</div>
</div>
```

图 2-5 列表页面代码（采用标签封装，简化页面代码）

```
1 <!-- 引入 jQuery 和 EasyUI 的 CSS 文件 -->
2 <!-- 引入 jQuery 和 EasyUI 的 JS 文件 -->
3 <!-- 引入 EasyUI 的 CSS 文件 -->
4 <html>
5 <head>
6 <title>DEMO示例</title>
7 <base href="jeecgDemoController.do?datagrid" />
8 </head>
9 <body>
10 <div style="border:1px solid #ccc; padding:5px;">
11 <input type="text" name="id" value="${jeecgDemo.id}" />
12 <table border="1" style="width:100%; border-collapse:collapse;">
13 <tr>
14 <td align="right" width="150px" nowrap>用户名: </td>
15 <td align="left" class="value" width="350px">
16 <input type="text" value="${jeecgDemo.userName}" />
17 <input type="text" value="${jeecgDemo.userName}" />
18 <input type="text" value="${jeecgDemo.userName}" />
19 </td>
20 </tr>
21 <tr>
22 <td align="right" nowrap>手机号码: </td>
23 <td align="left" class="value" width="350px">
24 <input type="text" value="${jeecgDemo.mobilePhone}" />
25 </td>
26 </tr>
27 <tr>
28 <td align="right" nowrap>办公电话: </td>
29 <td align="left" class="value" width="350px">
30 <input type="text" value="${jeecgDemo.officePhone}" />
31 </td>
32 </tr>
33 <tr>
34 <td align="right" nowrap>常用邮箱: </td>
35 <td align="left" class="value" width="350px">
36 <input type="text" value="${jeecgDemo.email}" />
37 </td>
38 </tr>
39 <tr>
40 <td align="right" nowrap>年龄: </td>
41 <td align="left" class="value" width="350px">
42 <input type="text" value="${jeecgDemo.age}" />
43 </td>
44 </tr>
45 <tr>
46 <td align="right" nowrap>工资: </td>
47 <td align="left" class="value" width="350px">
48 <input type="text" value="${jeecgDemo.salary}" />
49 </td>
50 </tr>
51 <tr>
52 <td align="right" nowrap>生日: </td>
```

图 2-6 用户管理页面代码（通用校验，简单易用）

3. JEECG 目录结构

3.1. 配置文件目录结构

JEECG 中的配置文件目录结构如图 3-1 所示。



图 3-1 JEECG 配置文件目录结构

3.2. Java源码目录结构

JEECG 中的 Java 源码目录结构如图 3-2 所示。



图 3-2 Java 源码目录结构

3.3. 单元测试代码结构

JEECG 中的单元测试代码存放的目录结构如图 3-3 所示。

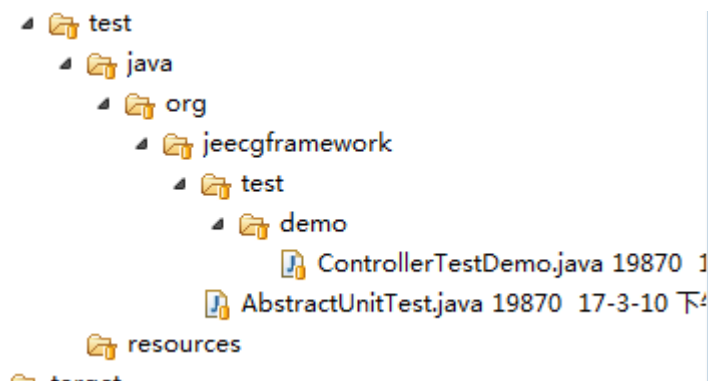


图 3-3 单元测试代码结构

3.4. JSP 页面目录结构

JEECG 中的 JSP 文件存放的目录结构如图 3-4 所示。

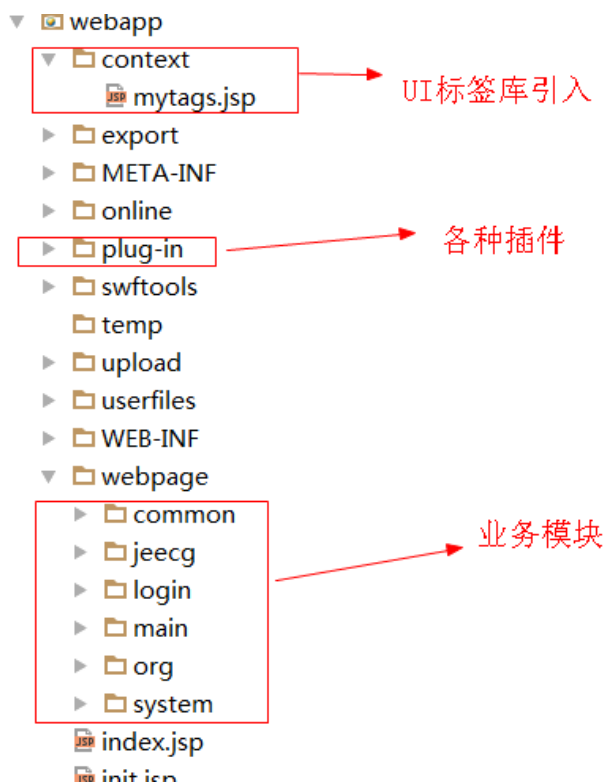


图 3-4 JSP 页面目录结构

4. 代码生成器

4.1. 功能介绍

jeecg 代码生成器可以一键生成全部代码，包括：jsp、control、service、dao、包括页面复杂控件，时间控件，页面校验等等，实现一键生成，直接使用，不需要改造代码；目前 jeecg 代码生成器分两种模式：Online 代码生成器、GUI 代码生成器，两种模式各有特点。

Online 代码生成器特点：更智能可视化操作，优于 GUI 模式，更多组件支持，支持 Popup 组件、字典组件、UE 编辑器等；

Online 模型支持：支持单表模型、一对一模型、一对多模型；

GUI 代码生成器特点：代码简洁；

GUI 模型支持：单表模型、一对多模型；

4.2. 代码生成器配置

代码生成器两个配置文件：

代码生成器参数配置	src/main/resources/jeecg/jeecg_config.properties
数据源配置	src/main/resources/jeecg/jeecg_database.properties

jeecg_config.properties：生成器参数配置文件，各参数说明如**错误!未找到引用源**。所示：

表 4-1 代码生成器参数说明

参数	参数说明	默认值	取值
source_root_package	Source folders on build path (JAVA 文件的根目录)	src.main.java	
webroot_package	WEB 应用文件的根目录（例如：jsp）	src.main.webapp	
bussi_package	业务包（举例：比如 ERP 中的一个大的模块销售模块目录） 特点：支持多级目录例如 [com.sys]	com.jeecg	
templatepath	代码生成器使用的模板文件目录	jeecg/template	
system_encoding	项目编码	utf-8	
jeecg_generate_table_id	自定义主键命名	id	目前表主键只能命名 ID
jeecg_ui_search_file_d_num	配置代码生成器生成的 JSP 页面，默认前几个字段生成查询条件	1	

jeecg_filed_convert	数据库表字段转换为实体字段是采用原生态, 还是采用驼峰写法转换	true	true/false
ui_filter_fields	根据过滤器自动在表中生成创建人、创建时间、修改人、修改时间等值（映射的字段参照“表 4 2 建表模板”）	create_date, create_by, create_name, update_date, update_by, update_name	
project_path	项目路径	D:\\workspace\\jeecg	

4.3. Online代码生成器

4.3.1. 配置表单

点击菜单：在线开发 -> Online 表单开发，配置对应参数

[1]、数据库属性配置



[2]、页面属性配置



[3]、校验字典配置



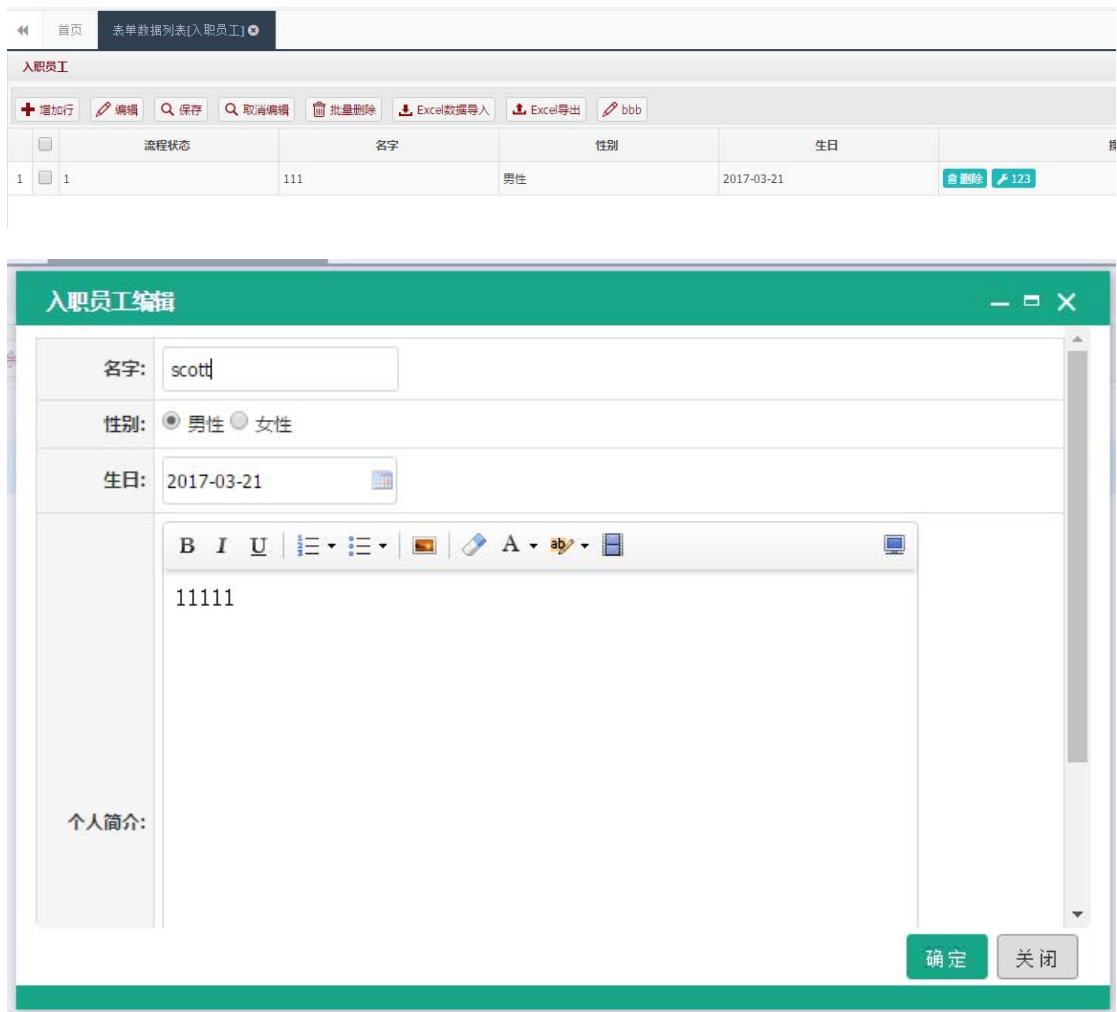
4.3.2. 同步数据库

执行同步操作，会在数据库中生成该表



4.3.3. 功能测试

点击功能测试，测试表单的列表功能、添加功能、修改功能、查看功能



4.3.4. 代码生成

选中需要生成的表单，点击代码生成，弹出代码生成器配置页面；

第一个参数可选择代码生成目录，其他参数可选择需要的风格等等，点击确认即可生成代码，代码生成可直接使用；





4.4. GUI代码生成器

本章通过一个实际的示例来讲解 JEECG 代码生成器的使用

4.4.1. 建表

现在有一张员工表 person，其建表 SQL 为：

```
CREATE TABLE `person` (  
  `ID` varchar(32) NOT NULL default '' COMMENT '主键',  
  `NAME` varchar(32) default NULL COMMENT '用户名',  
  `AGE` int(11) default NULL COMMENT '年龄',  
  `SALARY` decimal(10,2) default NULL COMMENT '工资',  
  `createDt` datetime default NULL COMMENT '创建时间',  
  PRIMARY KEY (`ID`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8;
```

注意：建表时，必须给每个字段加上注释，代码生成器会根据注释去生成页面字段对应的显示文本。

4.4.2. 代码生成

运行“src/main/java/test/JeecgOneGUI.java”文件，打开代码生成器并输入相应的参数如图 4-1 所示。



图 4-1 员工信息维护的代码生成器

执行【生成】之后，可以在源代码目录 src 中看到新生成的 java 代码文件，如图 4-2 所示。

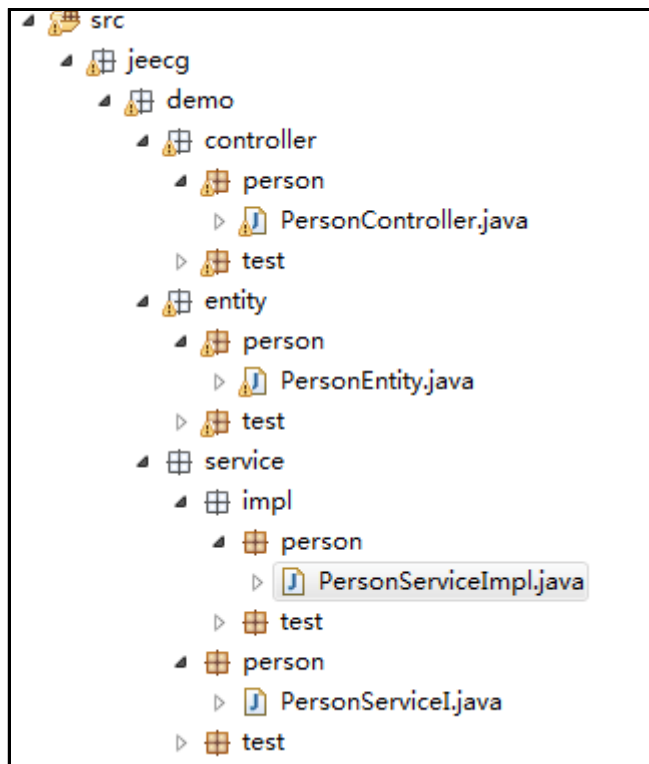


图 4-2 生成的 java 文件

同样地，可以在 WebRoot/webpage 中看到新生成的 jsp 页面，如图 4-3 所示。

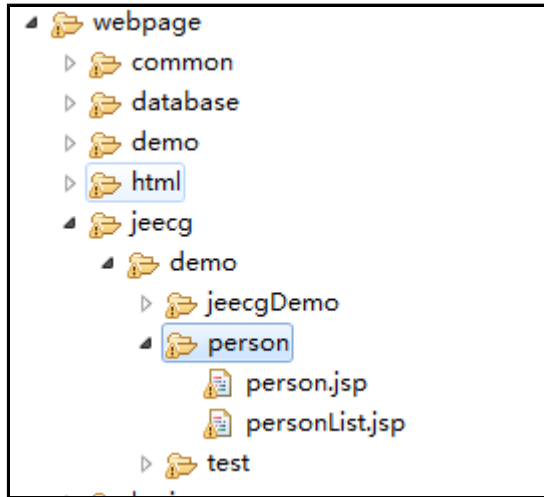


图 4-3 生成的 JSP 文件

生成代码结构说明

1. 添加和修改页面在一个 JSP 页面中
2. service 层接口和实现都继承父类

4.4.3. 配置扫描路径

用代码生成器生成代码后, 需要进行相关配置配置, 扫描注入 control、service、entity 等;

详细操作步骤如下:

- a. control 扫描配置, 在 spring-mvc.xml 文件里面

```
<!-- 加载controller的时候, 不加载service, 因为此时事务并未生效, 若此时加载了service, 那么事物无法对service进行拦截 -->  
<context:component-scan base-package="jeecg.*, com.chenhb.*">  
    <context:exclude-filter type="annotation" expression="org.springframework.stereotype.Service" />  
</context:component-scan>
```

配置属于自己的controller

- b. Service 扫描路径配置, spring-mvc-hibernate.xml

```
<!-- 加载service, 此时要排除controller, 因为controller已在spring-mvc中加载过了 -->  
<context:component-scan base-package="jeecg.*">  
    <context:exclude-filter type="annotation" expression="org.springframework.st  
</context:component-scan>  
    <context:component-scan base-package="com.chenhb.*">  
        <context:exclude-filter type="annotation" expression="org.springframework.st  
</context:component-scan>
```

加载自己的service

- c. 实体 Entity 扫描路径配置, spring-mvc-hibernate.xml

```
<!-- 注解方式配置 -->  
<property name="packagesToScan">  
    <list>  
        <value>com.chenhb.entity.*</value>  
        <value>jeecg.system.pojo.*</value>  
        <value>jeecg.demo.entity.*</value>  
        <value>jeecg.test.entity.*</value>  
        <value>jeecg.cgform.entity.*</value>  
    </list>  
</property>
```

配置自己的PO。数据操作实体

4.4.4. 功能测试

①. 添加菜单并授权

重新启动 Tomcat，进入系统主界面->系统管理->菜单管理，点击菜单录入，添加员工管理菜单，如图 4-4 所示。

菜单地址内容为:类映射名.do?方法名，如 personController.do?person



图 4-4 员工管理的菜单添加

菜单添加完成之后，需要将该菜单分配给管理员角色，重新登录系统后，可以在系统管理模块下看到子菜单【员工管理】，如图 4-5 所示。



图 4-5 新增的员工管理菜单项

②. 功能测试

点击菜单项【员工管理】，打开员工管理的主界面如图 4-6 所示。



图 4-6 员工管理主界面

点击【录入】按钮，在弹出的对话框中录入员工基本信息，如图 4-7 所示。

图 4-7 员工信息录入

点击确定按钮，对信息进行保存，此时可以在用户列表中看到新录入的信息，同时在数据库中也可以看到数据被保存入库，如图 4-8 所示。

	ID	NAME	AGE	SALARY	createDt
1	4028ba813dd31cc8013dd328f0400020	Dylan	28	1000.00	2013-04-04 11:47:24

图 4-8 信息被正确保存入库

4.4.5. 代码生成器使用规则

①. 建表规范

- 表必须有唯一主键：ID（字符类型 32 位）
备注：主键采用 UUID 方式生成
主键支持自定义，修改 jeecg_config.properties 的参数 [jeecg_generate_table_id] 即可；
- 如需使用框架自动生成表创建人，创建时间等，必须字段参见“表 4.2 建表模板”
- 表字段必须有注释
备注：JSP 页面字段文本，是根据表字段注释来生成

注：请按照建表模板表 4-1 来创建新表，模板表中原有的字段，生成器会过滤不在页面生成。

表 4-9 建表模板

字段名	类型	长度	备注	主键
ID	varchar	36	主键	TURE
CREATE_BY	varchar	36	创建人	
CREATE_NAME	varchar	32	创建人名字	
CREATE_DATE	datetime	0	创建时间	
UPDATE_BY	varchar	36	修改人	

UPDATE_NAME	varchar	32	修改人名字
UPDATE_DATE	datetime	0	修改时间
DELFLAG	int	2	删除标记
DEL_DATE	datetime	0	删除时间

②. 页面生成规则

说明：JSP 页面字段的文本内容，取表字段的注释前 6 位字符(如果建表字段注释为空，则页面字段文本会为空)

- 默认生成的 JSP 页面前五个字段为必须项，其他字段为非必须输入（需要自己手工加）
- 数据库字段类型为：datetime -->对应页面字段，会自动追加[年月日-时分秒]时间控件
- 数据库字段类型为：date -->对应页面会字段，自动追加[年月日]时间控件
- 数据库字段类型为：Int/Number-->对应页面字段，会自动追加数字校验（不允许输入小数）
- 数据库字段类型为：float/double/decimal-->对应页面页面字段，会自动追加数字校验（允许输入小数）
- 如果表字段为字符类型，并且设置了长度，页面输入框会自动设置 maxlength 对应表字段长度

4.4.6. 一对多的代码生成

①. 一对多代码生成器使用

单表的代码生成器入口类是 test.JeecgOneGUI;

一对多的代码生成器入口类是 test.JeecgOneToMainUtil;

一对多的代码生成器使用示例：

```
//第一步：设置主表
CodeParamEntity codeParamEntityIn = new CodeParamEntity();
codeParamEntityIn.setTableName("jeecg_order_main");//主表[表名]
codeParamEntityIn.setEntityName("Demo4ManyKey"); //主表[实体名]
codeParamEntityIn.setEntityPackage("jeecg"); //主表[包名]
codeParamEntityIn.setFtlDescription("订单主数据"); //主表[描述]

//第二步：设置子表集合
List<SubTableEntity> subTabParamIn = new ArrayList<SubTableEntity>();
```

```
//[1]. 子表一
SubTableEntity po = new SubTableEntity();
po.setTableName("jeecg_order_custom");//子表[表名]
po.setEntityName("DemoMany4CustomKey");//子表[实体名]
po.setEntityPackage("jeecg");           //子表[包]
po.setFtlDescription("订单客户明细"); //子表[描述]
po.setForeignKeys(new String[]{"GORDER_ID", "GO_ORDER_CODE"});//子表[外键:与主表关联外键]
subTabParamIn.add(po);
//[2]. 子表二
SubTableEntity po2 = new SubTableEntity();
po2.setTableName("jeecg_order_product");           //子表[表名]
po2.setEntityName("DemoMany4ProductKey");         //子表[实体名]
po2.setEntityPackage("jeecg");                     //子表[包]
po2.setFtlDescription("订单产品明细");             //子表[描述]
po2.setForeignKeys(new String[]{"GORDER_ID", "GO_ORDER_CODE"});//子表[外键:与主表关联外键]
subTabParamIn.add(po2);
codeParamEntityIn.setSubTabParam(subTabParamIn);

//第三步: 一对多(父子表)数据模型, 代码生成
CodeGenerateOneToMany.oneToManyCreate(subTabParamIn, codeParamEntityIn);
```

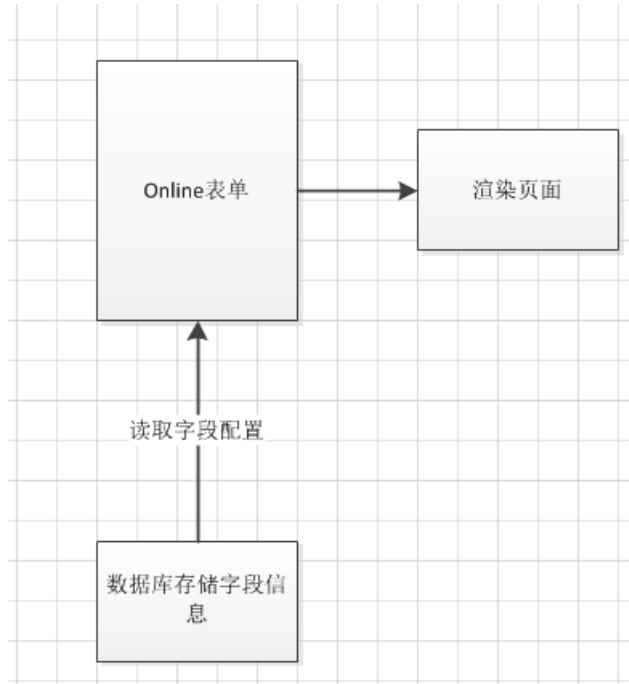
②. 使用规范

1. 目前代码生成器默认的主键生成策略为 UUID
2. 主表和子表的目录最好保持一致
3. 子表和主表的外键规则如下:
 - a) 主表和子表的外键字段名字, 必须相同 (除主键 ID 外)
 - b) 子表引用主表主键 ID 作为外键, 外键字段必须以_ID 结尾

5. Online表单开发

5.1. 原理

系统提供 online 表单开发, 将表单配置信息存入数据库, 即可通过 freemaker 引擎在线渲染, 也可生成代码进行二次开发, 方便灵活。



- ▼ cgform
 - ▶ common
 - ▶ controller
 - ▶ dao
 - ▶ engine
 - ▶ enhance
 - ▶ entity
 - ▶ exception
 - ▶ interceptors
 - ▶ pojo.config
 - ▶ service
 - ▶ sql
 - ▶ util

5.2. 使用

管理表单配置

表类型: 请选择... 表名: 同步数据库: 请选择...

新建表单
 编辑表单
 自定义按钮
 新增
 删除
 表单导出
 表单写入
 代码生成
 数据库写入表单
 Java.enhance

表类型	表名	表单分类	表描述	版本	是否树	是否分页	同步数据库	显示复选框	查询模式	创建人	创建时间	修改人	修改时间	操作
附表	auto_form_db_field	普通表单	表单数据源字段	5	否	是	已同步	否	single	admin	2015/06/15	admin	2015/06/15	删除 刷新
主表	auto_form_db	普通表单	表单数据源	23	否	是	已同步	否	group	admin	2015/06/15	admin	2015/07/14	删除 刷新 模板配置 功能测试 配置地址
单表	t_s_depart	普通表单	t_s_depart	3	是	是	已同步	是	group	admin	2015/06/14	admin	2015/07/17	删除 刷新 模板配置 功能测试 配置地址
单表	t_s_sms	普通表单	t_s_sms	2	否	是	已同步	是	group	admin	2015/06/11	admin	2015/06/27	删除 刷新 同步数据库
附表	t_b_code_detail	普通表单	基础标准代码类数值	14	否	是	已同步	是	group	admin	2015/06/11	admin	2015/06/30	删除 刷新
主表	t_b_code_type	普通表单	基础标准代码类列表	40	否	是	已同步	是	group	admin	2015/06/11	admin	2015/06/30	删除 刷新 模板配置 功能测试 配置地址
单表	test_onetable	测试单表		33	否	是	已同步	否	single	admin	2015/06/05	admin	2015/07/17	删除 刷新 模板配置 功能测试 配置地址
单表	wexin_template	普通表单	微信模板	6	否	是	已同步	否	group	admin	2015/06/02	admin	2015/06/11	删除 刷新 模板配置 功能测试 配置地址
单表	online_tree	普通表单	第一个树	4	是	是	已同步	否	single	admin	2015/05/30	admin	2015/06/01	删除 刷新 模板配置 功能测试 配置地址
单表	t_s_function	普通表单	t_s_function	9	是	是	已同步	是	group	admin	2015/05/30	admin	2015/05/31	删除 刷新 模板配置 功能测试 配置地址

1-10共 17条

© JEECG 版权所有 JEECG Framework -3.5.2 浏览器: JEECG Framework -3.5.2

创建表单

序号	操作	字段名称	字段备注	字段长度	小数点	默认值	字段类型	主键	允许空值
5	<input type="checkbox"/>	update_name	更新人名称	50	0		String	<input type="checkbox"/>	<input checked="" type="checkbox"/>
6	<input type="checkbox"/>	update_by	更新人登录名称	50	0		String	<input type="checkbox"/>	<input checked="" type="checkbox"/>
7	<input type="checkbox"/>	update_date	更新日期	20	0		Date	<input type="checkbox"/>	<input checked="" type="checkbox"/>
8	<input type="checkbox"/>	sys_org_code	所属部门	50	0		String	<input type="checkbox"/>	<input checked="" type="checkbox"/>
9	<input type="checkbox"/>	sys_company_code	所属公司	50	0		String	<input type="checkbox"/>	<input checked="" type="checkbox"/>
10	<input type="checkbox"/>	name	员工姓名	200	0		String	<input type="checkbox"/>	<input checked="" type="checkbox"/>

online 表单开发中 各属性配置与代码生成器使用规则类似。其中，【表单风格】可使用自己的模板（online 表单风格功能）。

<input checked="" type="checkbox"/>	单表	person	普通表单	员工管理	1	否	是	未同步	否	single	admin	2015/07/21	[删除][移除][同步数据库]
-------------------------------------	----	--------	------	------	---	---	---	-----	---	--------	-------	------------	-----------------

新创建的表单配置信息存在数据库里，但数据库中无 person 表。使用【同步数据库】功能，将执行建表 sql 语句，可在数据库中生成配置的 person 表，同步后

person	
<input type="checkbox"/>	单表 person 普通表单 员工管理 1 否 是 已同步 否 single admin 2015/07/21 admin 2015/07/21 [删除][移除][模板配置][功能测试][配置地址]

此时，数据库中已存在相应的表，点击【功能测试】可进行增删改查操作，此功能通过对 freemaker 模板进行渲染实现，如需添加其他复杂业务，可生成代码进行二次开发。

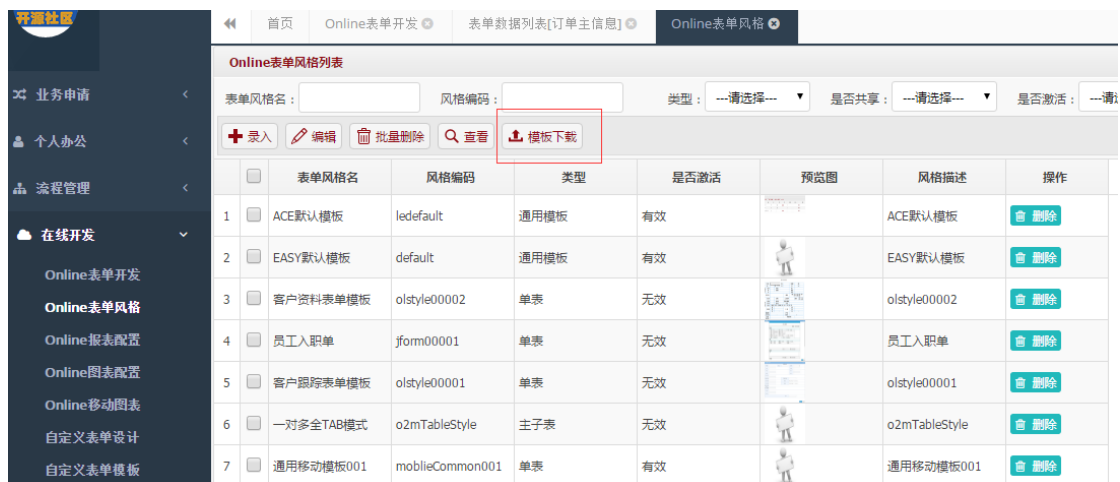
5.3. Online 表单风格

5.3.1. 介绍

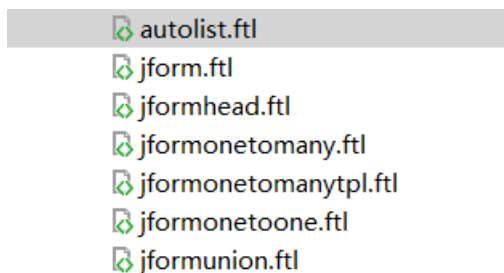
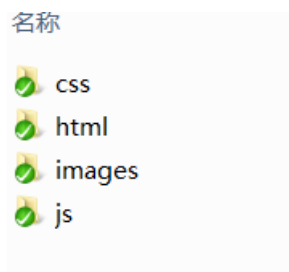
Online 表单通过风格切换，实现一个表单多种展现风格；Online 表单风格支持用户自定义。

5.3.2. 自定义风格方法

下载默认风格，修改html页面和样式。



下载模板包，目录和文件格式如下：

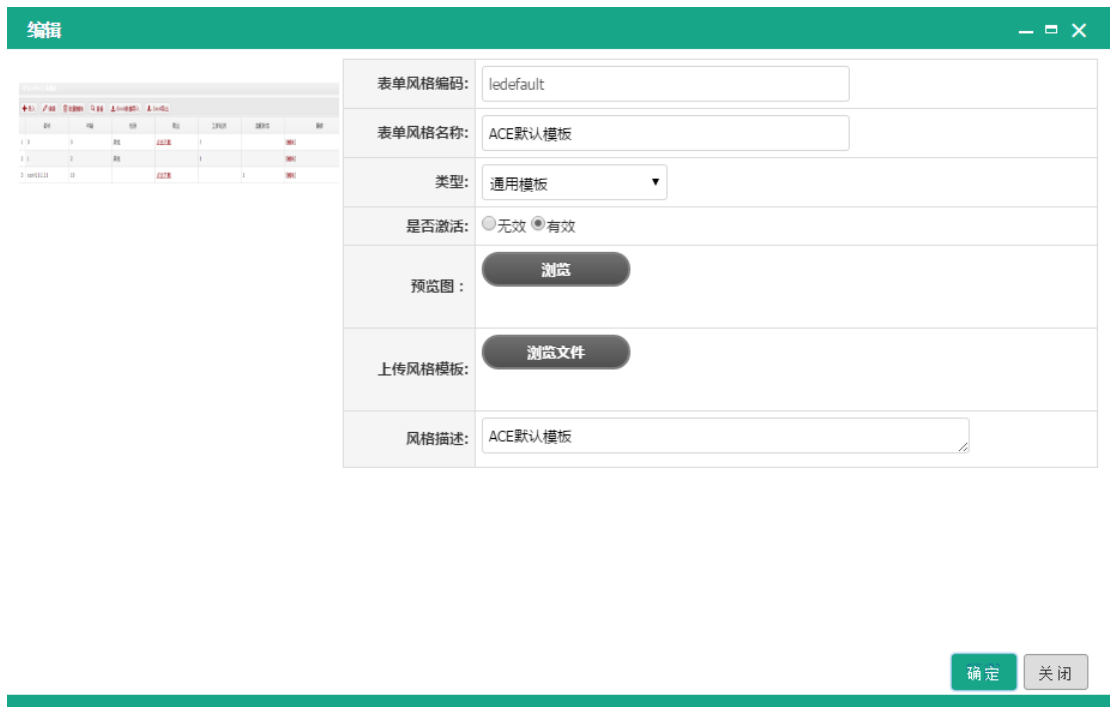


5.3.3. 风格模板命名

注意：文件名是固定的，模板内容可以自行修改

1	单表模型，表单模板名字	jform.ftl
2	一对多模型、一对一模型，表单模板名字	jformunion.ftl
3	列表模板名字	autolist.ftl

5.3.4. 风格上传



模板上传后请在【Online 表单开发】功能编辑页面选择



点击【Online 表单开发】列表页面【功能测试】可查看效果

6. Online报表配置

6.1. 原理

通过对 sql 进行解析配置需要显示的字段，通过 sql 查询直接得出数据

6.2. 使用



【sql 解析】功能可对输入的 sql 进行分析得出字段信息，【数据源】功能可查询不同的数据库，实现一个程序在多数据库的自由切换。【sql 解析】代码如下：

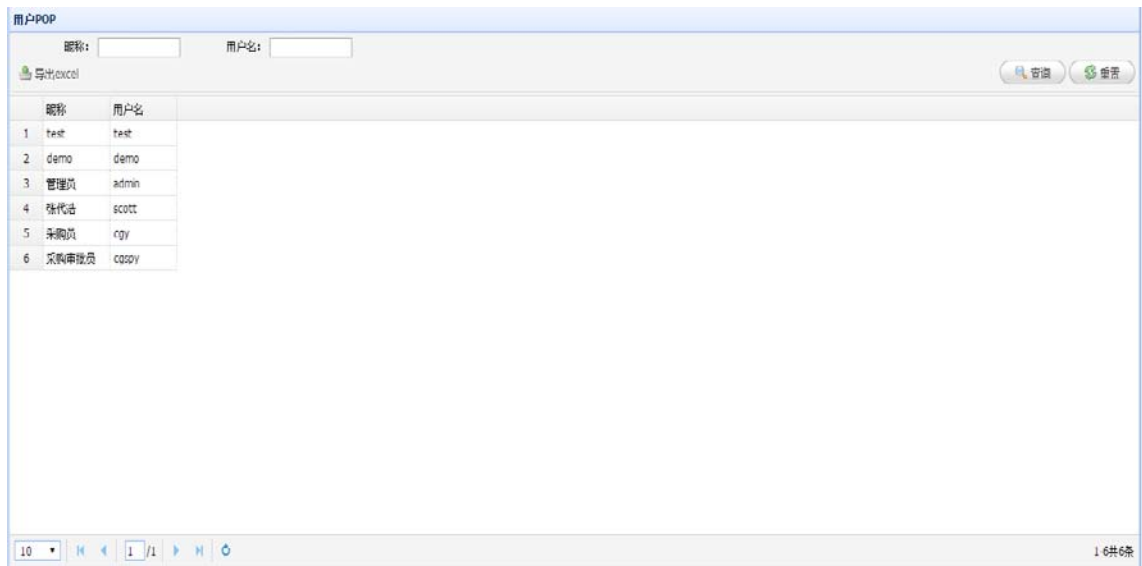
```
@SuppressWarnings("unchecked")
@RequestMapping(params = "getFields", method = RequestMethod.POST)
@ResponseBody
public Object getSqlFields(String sql, String dbKey) {
    List<String> result = null;
    Map reJson = new HashMap<String, Object>();
    try{
        //update-begin--Author:张忠亮 Date:20150619 for: sql解析多数据源
        if(StringUtils.isNotBlank(dbKey)){
            List<Map<String, Object>> dataList=DynamicDBUtil.findList(dbKey, SqlUtil.jeecgCreatePageSql(sql, null));
            if(dataList.size()<1){
                throw new BusinessException("该报表sql没有数据");
            }
            Set fieldsSet= dataList.get(0).keySet();
            result = new ArrayList<String>(fieldsSet);
        }else{
            result = cgReportService.getSqlFields(sql);
        }
        //update-end--Author:张忠亮 Date:20150619 for: sql解析多数据源
    }catch (Exception e) {
        e.printStackTrace();
    }
}
```

页面展示采用 freemaker 渲染:

```
</script>
<table width="100%" id="${config_id}List" toolbar="#${config_id}Listtb"></table>
<div id="${config_id}Listtb" style="...">
<div name="searchColumns">
    <#list config_queryList as x>
        <span style="...">
        <span style="..." title="${x['field_txt']}">${x['field_txt']}: </span>
        <#if x['search_mode']=="group">
            <input type="text" name="${x['field_name']}_begin" style="..." <#if x['field_type']=="Date">
            <span style="...">^</span>
            <input type="text" name="${x['field_name']}_end" style="..." <#if x['field_type']=="Date">clas
        </#if>
        <#if x['search_mode']=="single">
            <#if (x['field_dictlist']?size >0)>
            <select name = "${x['field_name']}" WIDTH="100" style="...">
            <option value = "">---请选择---</option>
            <#list x['field_dictlist'] as xd>
                <option value = "${xd['typecode']}">${xd['typename']}</option>
            </#list>
            </select>
        </#if>
    </#list>
```

配置完成后可通过【配置地址】功能配置菜单查看。

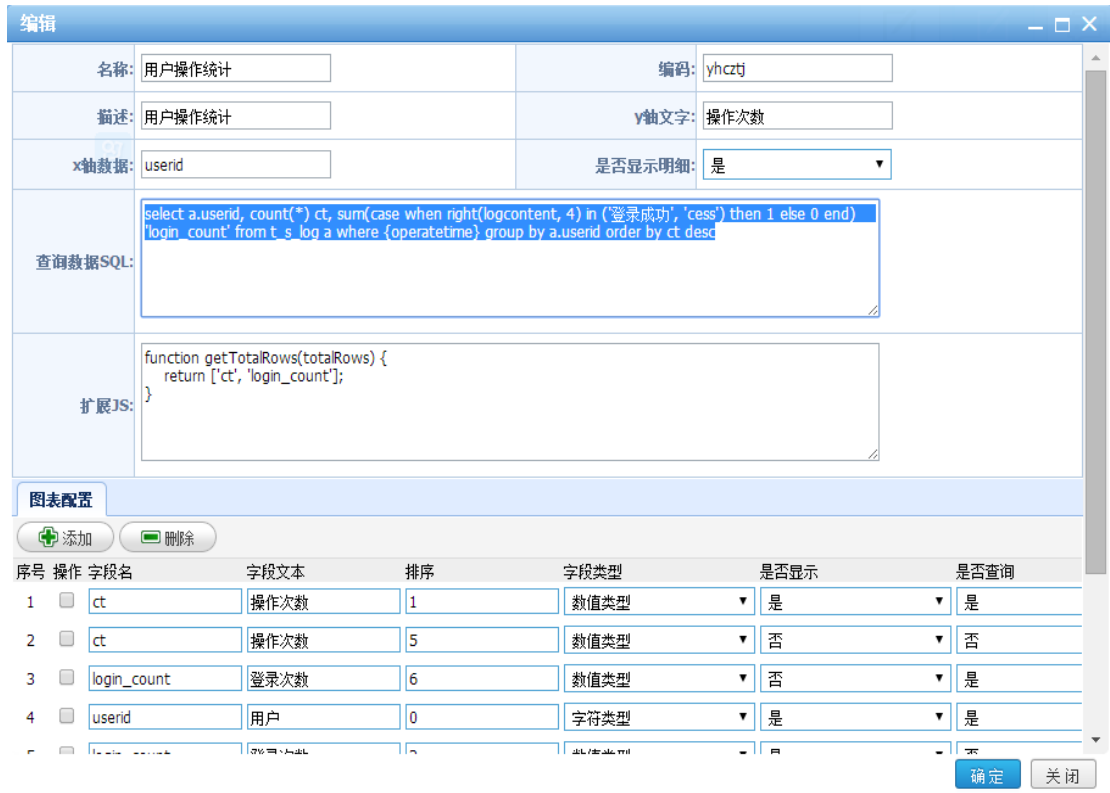
注意：菜单需配置权限。



6.3. Online图表配置

原理

实现原理与 online 报表类似，通过对 sql 进行分析得出相应字段，通过图表进行展示。



模板部分代码如下:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-tran
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<!-- context path -->
${config_iframe}
</head>
<body>
<div class="operations" style="...">
  <div class="bd3">
    <#list config_queryList as x>
      <span style="...">
      <span style="..." title="${x['field_txt']}">${x['field_txt']}: </span>
      <#if x['search_mode']=="group">
        <input type="text" name="${x['field_name']}_begin" style="..." <#if x['field_type']=="Date">
        <span style="...">^</span>
        <input type="text" name="${x['field_name']}_end" style="..." <#if x['field_type']=="Date">
      </#if>
      <#if x['search_mode']=="single">
        <#if (x['field_dictlist']?size >0)>
          <#if x['field_type']=="ComboGrid">
            <input type="text" name="${x['field_name']}" id="${x['field_name']}" style="...">
```

效果展示



7. 查询过滤器

7.1. 功能描述

查询过滤器可以帮助快速生成查询条件，不需要通过编码方式来实现，支持模糊查询、匹配查询、范围查询、不匹配查询等等。

7.2. 查询条件如何实现

第一步：页面采用 UI 标签实现

对需要实现查询的字段，增加属性 `query="true"`，页面自动生成查询框，如图 7-1 所示。

```
<t:datagrid name="jeecgDemoList" checkbox="true" sortName="birthday,name" pagination="true" fitColumns="false" t
<t:dgCol title="id" field="id" hidden="true" queryMode="group" width="120"></t:dgCol>
<t:dgCol title="名称" field="name" query="true" autocomplete="true" width="120"></t:dgCol>
<t:dgCol title="年龄" extend="{style:'width:50px'}" editor="numberbox" field="age" query="true" width="120">
<t:dgCol title="生日" hidden="true" field="birthday" formatter="yyyy-MM-dd" queryMode="group" width="120":
<t:dgCol title="部门" field="depId" query="true" queryMode="single" dictionary="t_s_depart,id,departname" wi
<t:dgCol title="部门code" field="extField"></t:dgCol>
<t:dgCol title="性别" field="sex" query="true" dictionary="sex" width="120"></t:dgCol>
<t:dgCol title="电话" field="phone" queryMode="group" width="120"></t:dgCol>
<t:dgCol title="工资" field="salary" queryMode="group" width="120"></t:dgCol>
<t:dgCol title="创建日期" field="createDate" formatter="yyyy-MM-dd" query="true" queryMode="group" editor="d:
<t:dgCol title="邮箱" field="email" hidden="true" queryMode="group" width="120"></t:dgCol>
<t:dgCol title="入职状态" field="status" query="true" extend="{style:{width:'300px',color:'red'}};datatype:'*'
<t:dgCol title="个人介绍" field="content" hidden="true" queryMode="group" width="500"></t:dgCol>
```

图 7-1 JSP 代码实现



第二步：controller 层处理

Controller 中对应的处理逻辑如图 7-2 所示。

```
CriteriaQuery cq = new CriteriaQuery(TSUser.class, dataGrid);  
//查询条件组装器  
org.jeecgframework.core.extend.hqlsearch.HqlGenerateUtil.installHql(cq, user);
```

图 7-2 Controller 代码

7.3. 查询过滤器高级特性

datagrid 中的查询过滤器默认是单条件查询，即在设置多个 dgCol 的 query="true" 之后，查询条件中同时只能有一个条件被使用，生成的页面效果如图 7-3 所示。

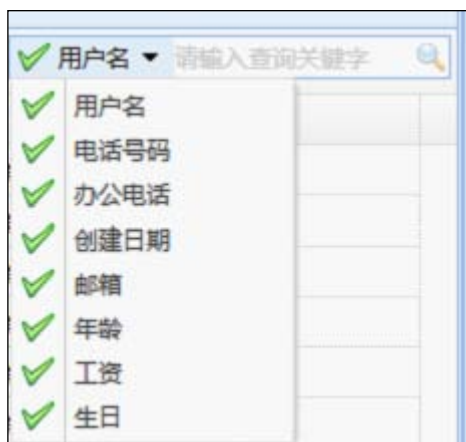


图 7-3 默认查询过滤器效果

当然，可以通过 datagrid 和 dgCol 的参数设置来达到更高级的查询过滤功能，如组合查询条件和值范围查询。

7.3.1. 组合条件查询

设置<t:datagrid>标签的 queryMode="group"（参数默认为"single"，即单条件查询），在页面生成时，会生成一个组合查询条件输入面板。生成的页面效果如图 7-4 所示。



图 7-4 组合查询过滤器效果

7.3.2. 字段范围查询

设置<t:dgCol>标签的 queryMode="group"，在页面生成时，会生成一个范围输入框。生成的页面效果如图 7-4 所示。

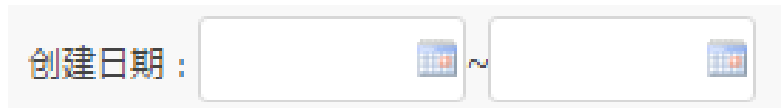


图 7-5 字段范围查询效果

字段范围查询会为该字段生成两个输入框，name 分别为“字段名_begin”和“字段名_end”，具体的查询逻辑查询过滤器自己已经动组装实现，不需要再编码。

7.3.3. 日期字段的数据格式化

在 dategrid 中，对于日期字段，可以通过设置<d:dgCol>的 formatter 属性配置格式化方式，实现对日期数据的格式化，如：

```
<t:dgCol title="创建日期" field="createTime" formatter="yyyy-MM-dd hh:mm:ss"
query="true" queryMode="group"></t:dgCol>
```

对于日期的格式化常用表达式：yyyy-MM-dd hh:mm:ss / yyyy-MM-dd

7.4. 查询规则

要求：页面查询字段，需跟后台 Action(或 Controller)中 Page 的字段对应一致，后台不需写代码自动生成 HQL，追加查询条件；默认生成的查询条件是全匹配；

查询匹配方式分类：

- [1]. 全匹配查询：查询数据没有特殊格式，默认为全匹配查询
- [2]. 模糊查询：查询数据格式需加星号[*] 例如： {MD*/MD*/M*D*}
- [3]. 包含查询：查询数据格式采用逗号分隔[,] 例如： {01,03} (含义：in('01','03'))
- [4]. 不匹配查询：查询数据格式需要加叹号前缀[!] 例如： {!123} (含义：不等于 123)

特殊说明：查询不为 Null 的方法=!null(大小写没关系)；查询不为空字符串的方法 =!(只有一个叹号)。

- [5]. 时间范围范围查询

jsp 页面中使用的 name: 需要查询的日期类型字段名本身（什么都不加），表示查询时查询等于该字段时间的数据

begin: 需要查询的日期类型字段名（首字母大写），表示查询开始时间 查询时查询大于等于开始时间的数据

end: 需要查询的日期类型字段名（首字母大写），表示查询结束时间查询时查询小于等于结束时间的数据

使用举例:

字段名称 private Date birthday

查询开始时间 beginBirthday

查询结束时间 endBirthday

8. 基础用户权限

8.1. 权限设计原理

● 基本概念

权限管理模块涉及到的实体有：用户、角色和系统资源(包括系统菜单、页面控件、数据资源等)。用户可以拥有多个角色，角色可以被分配给多个用户。而权限的意思就是对某个资源的某个操作。一般通用的权限管理模块规定：所谓资源即应用系统中提供的要进行鉴权才能访问的资源(比如各类数据, 系统菜单)；所谓操作即增加、修改、删除、查询等操作。

● 权限模型

用户权限模型，指的是用来表达用户信息及用户权限信息的数据模型。即能证明“你是谁？”、“你能访问哪些受保护资源？”。

用户与角色之间构成多对多关系。表示同一个用户可以拥有多个角色，一个角色可以被多个用户所拥有。

角色与资源之间构成多对多关系。表示同一个资源可以被多个角色访问，一个角色可以访问多个资源。

权限设计模型如图 8-1 所示。

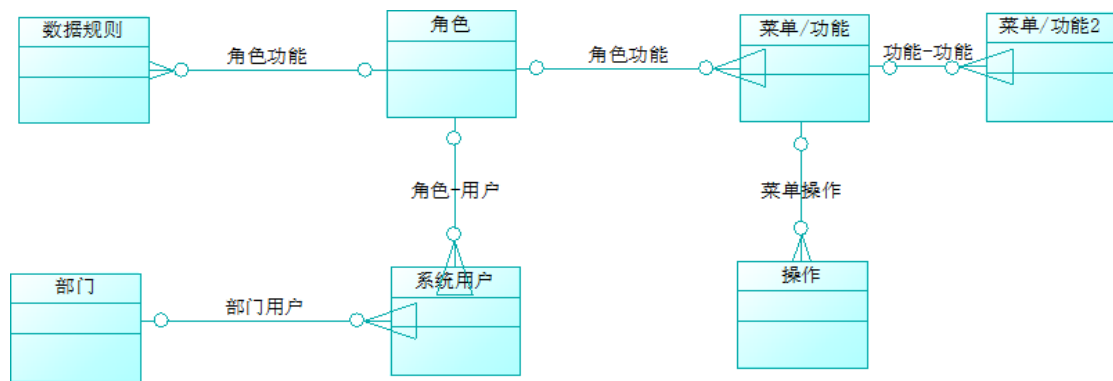


图 8-1 权限设计模型

8.2. 权限设计目标

权限设计及权限管理的目标包括：

- 1) 对用户授予相应的角色
- 2) 对角色授予不同的菜单
- 3) 对角色授予不同的操作页面控件操作权限
- 4) 进行数据级别的权限控制（行级别、列级别）

8.3. 权限数据表

数据表	实体类	说明
t_s_user	org.jeecgframework.web.system.pojo.base.TSUser	[用户权限]系统用户表
t_s_base_user	org.jeecgframework.web.system.pojo.base.TSBaseUser	[用户权限]系统用户父类表
t_s_role	org.jeecgframework.web.system.pojo.base.TSRole	[用户权限]角色
t_s_role_user	org.jeecgframework.web.system.pojo.base.TSRoleUser	[用户权限]用户角色
t_s_depart	org.jeecgframework.web.system.pojo.base.TSDepart	[用户权限]部门机构表
t_s_role_function	org.jeecgframework.web.system.pojo.base.TSRoleFunction	[用户权限]角色权限表
t_s_operation	org.jeecgframework.web.system.pojo.base.TSOperation	[用户权限]操作权限表
t_s_function	org.jeecgframework.web.system.pojo.base.TSFunction	[用户权限]菜单权限表
t_s_data_rule	org.jeecgframework.web.system.pojo.base.TSDataRule	[用户权限]数据权限表

8.4. 权限设计实现

8.4.1. 访问菜单权限

菜单管理如图 8-2 所示。

菜单名称	图标	菜单类型	菜单地址	菜单顺序	菜单图标样式	操作
1 > 插件模块		菜单类型		0	icon-download-alt	删除 页面控件权限 数据规则
2 > 统计报表		菜单类型		2	fa-pie-chart	删除 页面控件权限 数据规则
3 > 在线演示		菜单类型		3	fa-bar-chart-o	删除 页面控件权限 数据规则
4 > 在线开发		菜单类型		4	fa-play-circle	删除 页面控件权限 数据规则
5 > 在线开发		菜单类型		5	fa-cloud	删除 页面控件权限 数据规则
6 Online表单开发		菜单类型	cgFormHeadController.do?cgFormHeadList	1		删除 页面控件权限 数据规则
7 Online表单样式		菜单类型	cgFormTemplateController.do?cgFormTemplate	2		删除 页面控件权限 数据规则
8 Online报表配置		菜单类型	cgReportConfigHeadController.do?cgReportConfigHead	3		删除 页面控件权限 数据规则
9 Online图表配置		菜单类型	cgFormGraphReportHeadController.do?cgFormGraphReportHead	4		删除 页面控件权限 数据规则
10 Online移动图表		菜单类型	cgDynamGraphConfigHeadController.do?cgDynamGraphConfigHead	5		删除 页面控件权限 数据规则
11 Online移动图表_vm		菜单类型	cgDynamGraphConfigHeadController.do?cgDynamGraphConfigHeadVM	6		删除 页面控件权限 数据规则
12 Online报表配置_vm		菜单类型	cgReportConfigHeadController.do?cgReportConfigHeadVM	7		删除 页面控件权限 数据规则
13 online图表配置_VM		菜单类型	cgFormGraphReportHeadVMController.do?cgFormGraphReportHeadVM	8		删除 页面控件权限 数据规则
14 自定义表单列表		菜单类型	autoFormController.do?autoForm	9		删除 页面控件权限 数据规则
15 自定义表单模板		菜单类型	autoFormStyleController.do?autoFormStyle	10		删除 页面控件权限 数据规则

图 8-2 权限菜单

8.4.2. 列表按钮权限

使用说明

页面控件级别的权限依赖于菜单权限，也就是说，需要先为角色分配菜单，在已分配的菜单中，可以选择可以操作的页面控件。

页面控件权限的添加在菜单管理页面，点击对应菜单的【页面控件权限】，设置该菜单页面相关控件的操作权限，如图 8-3 所示。

操作录入	页面控件	页面控件编码	类型	状态	权限名称	操作
1	检索列表	person	检索	有效	团队人员检索	删除
2	name	name	检索	有效	团队人员检索	删除

图 8-3 操作按钮设置

页面控件操作权限的分配在角色管理页面，在权限设置时，先为角色分配菜单，点击相应的菜单，在右侧的“操作按钮列表”面板中显示该菜单可分配的操作按钮，如图 8-4 所示。

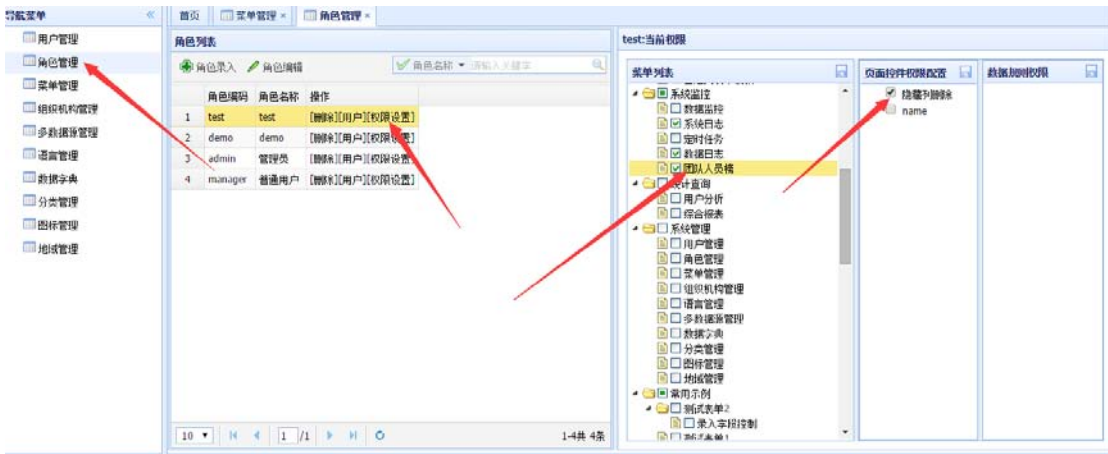
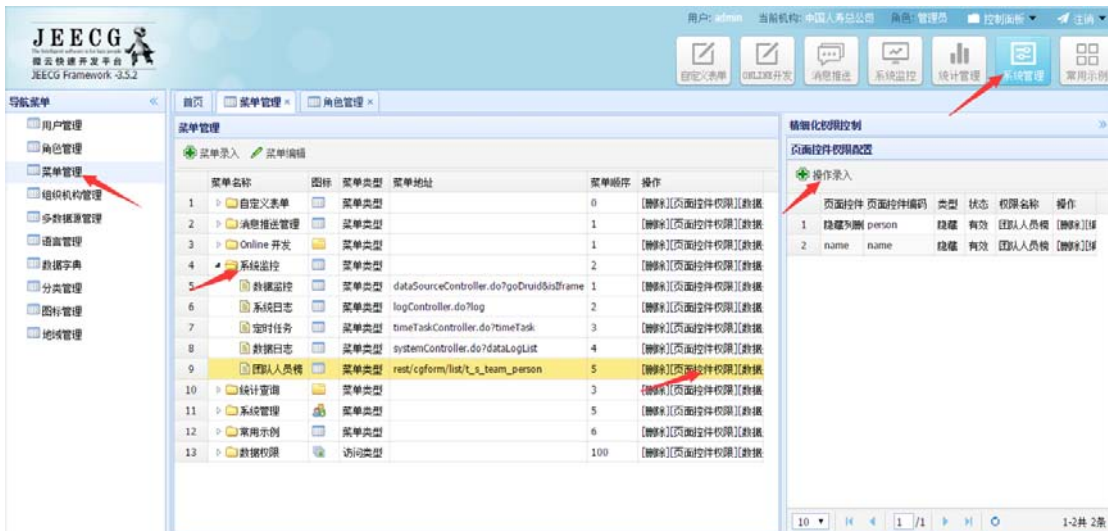


图 8-4 按钮权限分配

1) admin 用户默认拥有全部权限，不受页面控件权限控制。

操作步骤

① 页面控件权限设置：进入【系统管理】→【菜单管理】→【系统监控】，点击【团队人员榜】的【页面控件权限】



操作录入
— □ ×

页面控件名称: 操作名称范围2~20位字符

页面控件编码:

规则类型: 隐藏 ▼

状态: 有效 ▼ 必须为数字

确定
关闭

输入名称,建议以[隐藏]开头
如: 隐藏删除按钮

页面控件权限配置

+ 操作录入						
	页面控件名称	页面控件编码	类型	状态	权限名称	操作
1	隐藏列删除	person	隐藏	有效	团队人员榜	[删除][编辑]
2	name	name	隐藏	有效	团队人员榜	[删除][编辑]

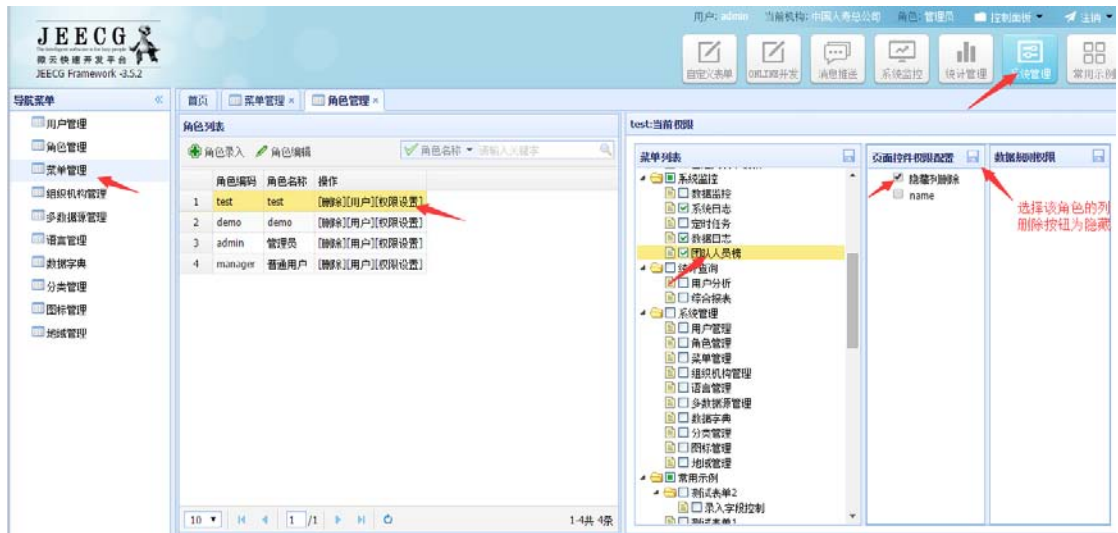
录入完成

② 代码中对按钮加入操作代码。

```

<t:datagrid name="tSTeamPersonList" checkbox="true" fitColumns="false" title="团队人员榜" actionUrl="tSTeamPersonController.do?datagrid" >
<t:dgCol title="主键" field="id" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="创建人名称" field="createName" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="创建人登录名称" field="createBy" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="创建日期" field="createDate" formatter="yyyy-MM-dd" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="更新人名称" field="updateName" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="更新日期" field="updateBy" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="更新日期" field="updateDate" formatter="yyyy-MM-dd" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="所属部门" field="sysOrgCode" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="所属公司" field="sysCompanyCode" hidden="true" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="名称" field="name" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="头像" field="imgSrc" image="true" queryMode="single" imageSize="imageSize(200,50)" width="120"></t:dgCol>
<t:dgCol title="简介" field="introduction" queryMode="single" width="500"></t:dgCol>
<t:dgCol title="加入时间" field="jionDate" formatter="yyyy-MM-dd" queryMode="single" width="120"></t:dgCol>
<t:dgCol title="操作" field="opt" width="100"></t:dgCol>
<t:dgDelOpt title="删除" url="tSTeamPersonController.do?doDel&id={id}" operationCode="person"/>
<t:dgToolBar title="录入" icon="icon-add" url="tSTeamPersonController.do?goAdd" funname="add"></t:dgToolBar>
<t:dgToolBar title="编辑" icon="icon-edit" url="tSTeamPersonController.do?goUpdate" funname="update"></t:dgToolBar>
<t:dgToolBar title="批量删除" icon="icon-remove" url="tSTeamPersonController.do?doBatchDel" funname="deleteALLSelect"></t:dgToolBar>
<t:dgToolBar title="查看" icon="icon-search" url="tSTeamPersonController.do?goUpdate" funname="detail"></t:dgToolBar>
<t:dgToolBar title="导入" icon="icon-put" funname="ImportXls"></t:dgToolBar>
<t:dgToolBar title="导出" icon="icon-putout" funname="ExportXls"></t:dgToolBar>
</t:datagrid>
    
```

③ 角色管理中对菜单设置按钮权限



④ 以角色为【test 用户】的账户登录系统。



8.4.3. 页面表单权限

Jeecg 中，目前按表单权限设置，是通过平台自己封装的标签（<t:authFilter />等）进行设置。而在开发的过程中，有一些按钮标签是普通的<a href>或<button>形式的。对于这种开发者自定义表单控件的权限设置，目前 jeecg 也可以支持了。具体设置方法如下：

1) 给页面上的自定义页面控件增加 id 或 class 。

```
<a id='workerinputhref' href="#" class="easyui-linkbutton workerAddButton" p
    onclick="add('基本信息录入','workerBaseController.do

<a href="#" class="easyui-linkbutton workerAddButton workerotherinput" pl
    onclick="update('银行卡录入','workerBankController.d

<a href="#" class="easyui-linkbutton workerAddButton workerotherinput"
    onclick="update('教育经历录入','workerEduController.d
```

小提示：对于具有相同权限的多个按钮，可以设定一个共同的 class，将会更加便捷。

2) 将自定义页面控件的 id 或 class 设置到操作按钮中。

方式一： ID 设置

页面控件名称:	隐藏姓名输入框	✓ 通过信息验证!
页面控件编码:	#name	
规则类型:	隐藏	
状态:	有效	必须为数字

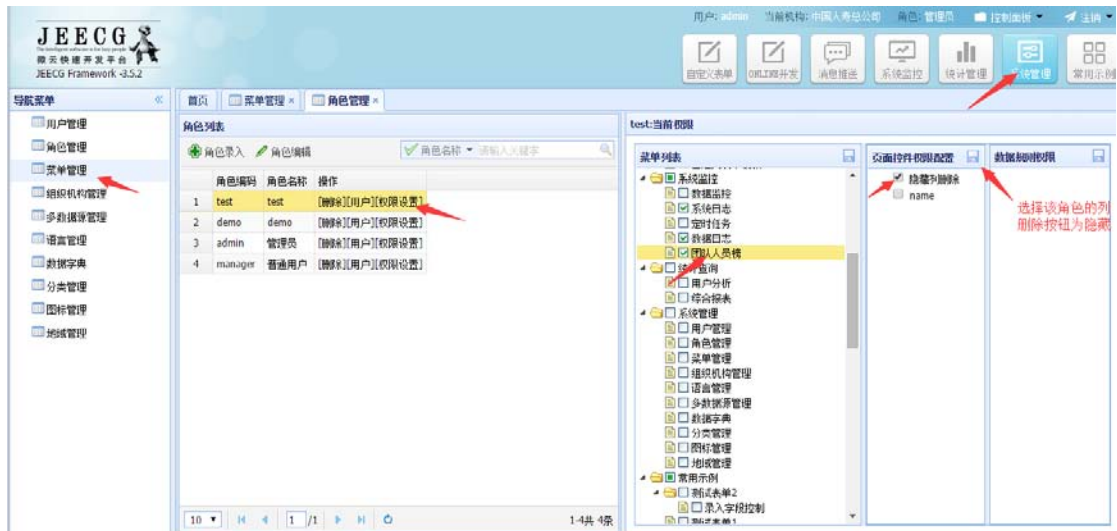
如控件编码内容为ID,则在ID前加入 # 号与jQuery的id选择器相同.

方式二： Class 设置

页面控件名称:	隐藏姓名输入框	✓ 通过信息验证!
页面控件编码:	.name	
规则类型:	隐藏	
状态:	有效	必须为数字

如控件编码的为class,则加 . 号与jQuery的类选择器规则相同

3) 在角色列表中，进行权限授权即可



8.4.4. 页面表单权限标签

表单权限控制，通过权限标签实现，两种标签两种不同使用方式。

<p>表单权限智能标签</p>	<pre><t:authFilter /></pre>
<p>用法:</p>	<p>将该标签放在 JSP 页面最底部即可，不要采用包含写法</p> <p>规则: 采用 JQuery 思路，用 JS 方式对页面控件进行控制</p> <p>页面控件编码，规则如下:</p> <p>#id => 表单控件 ID</p> <p>.class => 表单控件样式</p> <p>控制精度: 可控制表单片段的隐藏和禁用</p>
<p>表单权限包含标签</p>	<pre><t:hasPermission code="add"> <input name="mobile" class="inputtxt" value="{depart.mobile}"/> </t:hasPermission></pre>
<p>用法:</p>	<p>采用包含的方式，</p> <p>code 对应页面控件权限的[页面控件编码]</p> <p>规则: 包含权限标签，包含的表单代码片段，会通过 code 权限匹配，匹配成功，页面片段将不显示;</p> <p>控制精度: 只能控制表单片段的隐藏（不区分隐藏和禁用）</p>

8.4.5. 数据权限控制

数据权限是对数据的请求 URL 进行控制，可设定数据权限规则，即定义数据查询过滤的规则，每个表中都有公司 ID、单位 ID 和用户 ID 等，可自由选择其一作为过滤条件。

新建菜单项，选择菜单类型为权限类型（一般为***Controller.do?datagrid），添加完成后，点击右侧的“数据规则”按钮进入页面规则列表页面。

菜单录入

菜单名称: 菜单名称范围4~15位字符,且不为空

菜单类型: 访问类型 通过信息验证!

菜单等级: 下级菜单

父菜单: 数据权限

菜单地址: ***Controller.do?datagrid

图标: 默认

桌面图标: 用户管理

菜单顺序:

数据权限	访问类型	100	[删除][页面控件权限][数据规则]
user list ctl	访问类型	userController.do?user	1 [删除][页面控件权限][数据规则]
user add ctr	访问类型	userController.do?addorupdate	1 [删除][页面控件权限][数据规则]
事例录入	访问类型	jeecgDemoController.do?addorupdate	1 [删除][页面控件权限][数据规则]
请假单录入	访问类型	cgFormBuildController.do?flForm&tableName=jform_leave	2 [删除][页面控件权限][数据规则]
团队人员 ADD	访问类型	tSTeamPersonController.do?goAdd	4 [删除][页面控件权限][数据规则]
团队人员	访问类型	tSTeamPersonController.do?datagrid	5 [删除][页面控件权限][数据规则]
系统日志	访问类型	logController.do?datagrid	6 [删除][页面控件权限][数据规则]

点击数据规则页面的“操作录入”按钮，弹出页面录入一条规则。

操作编辑

规则名称: 操作名称范围2~20位字符

规则字段: 任意

条件规则:

规则值: 对象属性,可级联

填写说明:

- (1) 规则名称: [字段名称]=[规则值]限制 (可自定义)
- (2) 规则字段: [字段名称]
- (3) 条件规则: 大于/大于等于/小于/小于等于/等于/包含/模糊/不等于
- (4) 规则值: 指定值

例如:

规则名称: 部门=00001

规则字段: department

条件规则: 等于

规则值: 00001

以上规则表示页面查询数据时将在 sql 中添加 department= '00001' 的条件, 即按照部门=00001 进行了数据权限控制。

规则字段如需要使用对象级联的方式, 需要在实体中配置对象映射关系。

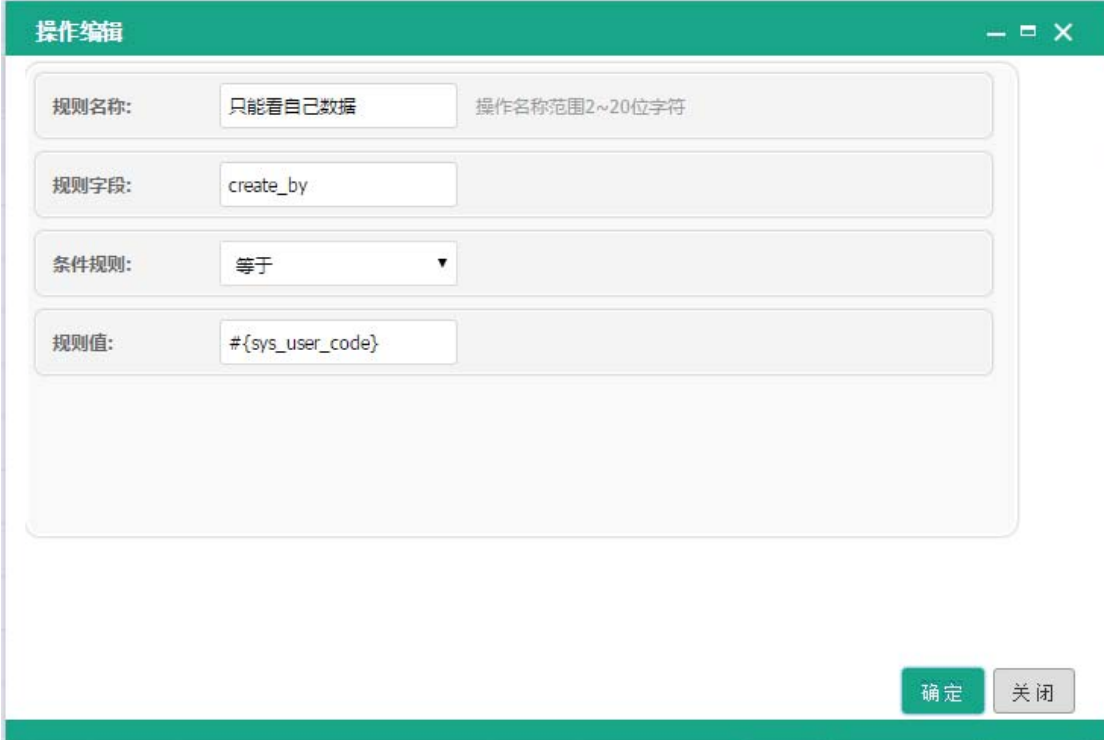
8.4.6. 数据权限当前用户上下文变量

注意: 规则字段处可以填写系统字段, 包括:

sys_company_code	当前登录用户公司编号
sys_org_code	当前登录用户部门编号
sys_user_code	当前登录用户账号
sys_user_name	当前用户真实名称
sys_date	当前日期

sys_time 当前时间

写法如下: #{sys_org_code}



操作编辑

规则名称: 只能看自己数据 操作名称范围2~20位字符

规则字段: create_by

条件规则: 等于

规则值: #{sys_user_code}

确定 关闭

9. 多数据源

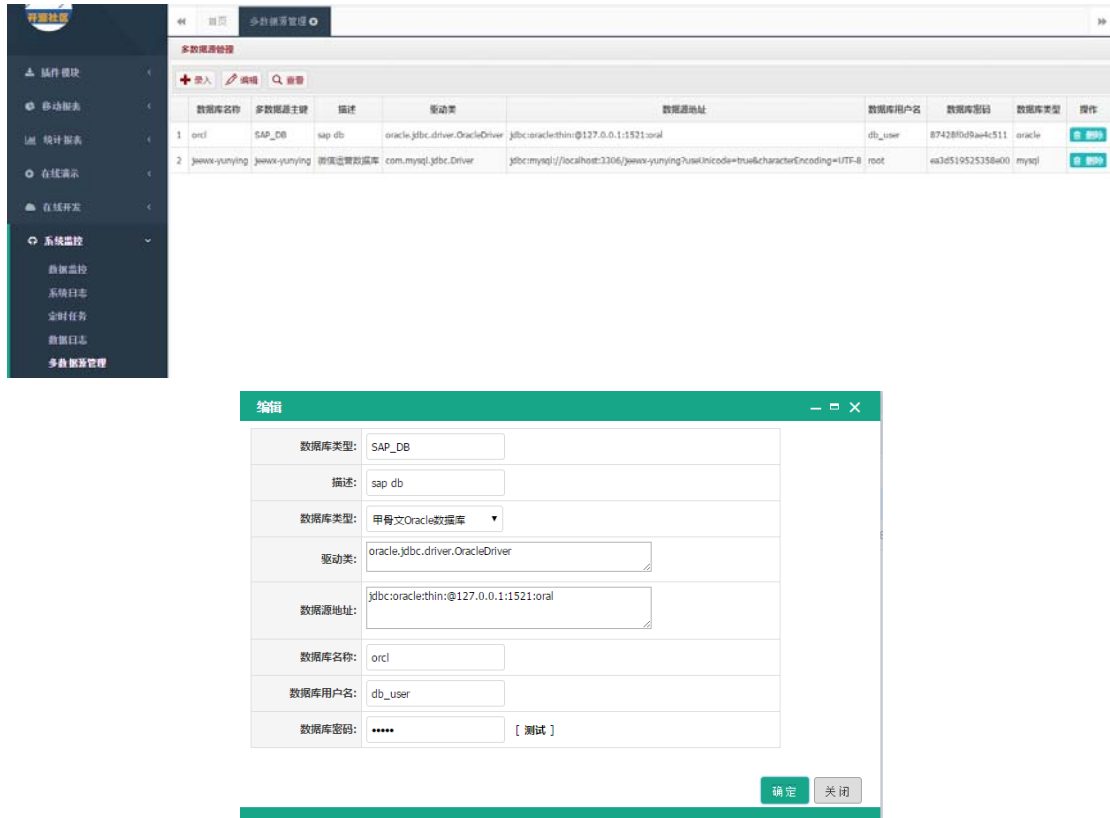
9.1. 多数据源背景

通常一个系统只需要连接一个数据库就可以了。但是在企业应用的开发中往往会和其他子系统交互，特别是对于一些数据实时性要求比较高的数据，我们就需要做实时连接查询，而不是做同步。这个时候就需要用到多数据源。

举个简单的例子某企业要做订单网上订单系统这里面就可以涉及到多个子系统的连接，比如：产品主数据的数据源，项目管理系统的数据库（项目可以产品订单）等多个不同数据库类似的数据源，他们可能是 ORACLE，SQL SERVER，MYSQL 等多种混合数据源。

9.2. 多数据源的配置

通过菜单：系统监控 -> 多数据源管理，对多数据源进行配置管理。



9.3. 多数据源的使用

多数据源使用通过工具类: `DynamicDBUtil` 的通用方法调用, SQL 语法采用原生态 sql 写法, 通过 `springjdbc` 来执行。

举例:

- 插入修改数据: `DynamicDBUtil.update('SAP_DB', 'delete from user ??');`
返回: `int`
- 查询单条数据: `DynamicDBUtil.findOne('SAP_DB', 'delete from user ??');`
返回: `Map<String, Object>`
- 查询数据列表: `DynamicDBUtil.findList('SAP_DB', 'delete from user ??');`
返回: `List<Map<String, Object>>`

说明: 第一个参数 {**SAP_DB**} => 对应数据源 KEY

10. 国际化

10.1. 国际化背景

没有国际化的框架是一个不完整的框架，特别在全球信息化的今天，国际化不再是鸡肋，而是在选择开发平台时必须首要的考试因素，特别在有些公司平台是否国际化具有一票否决要素，所以我们要搞国际化，而不是仅仅是简单的高大尚。

10.2. 国际化语言维护

通过菜单：系统管理->国际化语言，对语言进行维护管理。



10.3. 国际化标签用法 | t:mutiLang

10.3.1. 参数

属性名	类型	描述	是否必须	默认值
langKey	string	语言 key, 如 common.login 输出的中文可能为“登录”	是	null
langArg	string	对 langKey 进行翻译后如包含{0}, {1}这样的字符, 此参数可对其进行翻译, 多个以英文逗号分隔	否	null

10.3.2. 用法

```
<t:mutiLang langKey="common.copyright"/>
```

特殊：带参数，参数多个以逗号隔开

```
<t:mutiLang langKey="common.edit.param" langArg="common.operation"/>
```


10.4. 其他标签国际化用法

针对 jeecg UI 标签库，title 参数统一做了国际化处理，直接配置国际化 KEY 即可。

10.4.1. 列表datagrid

```
<t:datagrid name="userList" title="common.operation"
actionUrl="userController.do?datagrid"
```

10.4.2. 列表字段

```
<t:dgCol title="common.role" field="userKey" ></t:dgCol>
```

10.4.3. 列表按钮

```
<t:dgToolBar title="common.add.param" langArg="common.language"
```

11. 定时任务

11.1. 定时任务配置文件

src/main/resources/spring-mvc-timeTask.xml

配置一：具体定时任务配置

```
<!-- 定时任务 Job 配置 -->
<bean id="taskDemoServiceTaskJob"
class="org.springframework.scheduling.quartz.MethodInvokingJobDetailFactoryBean">
    <property name="targetObject" ref="taskDemoService" />
    <property name="targetMethod" value="work" />
    <property name="concurrent" value="true" />
</bean>
<!-- 定时任务 Trigger 配置 -->
<bean id="taskDemoServiceTaskCronTrigger"
class="org.jeecgframework.core.timer.DataBaseCronTriggerBean">
    <property name="jobDetail" ref="taskDemoServiceTaskJob" />
    <property name="cronExpression" value="0 0/1 * * * ?" />
</bean>
```

Job 参数说明:

targetObject	Spring Service bean 名称
targetMethod	Service 方法名
concurrent	是否支持并发

此处 Trigger 配置的 cronExpression 无用，以定时任务管理模块有效。

配置二： Service 定义

```
@Service("taskDemoService")
public class TaskDemoServiceImpl implements TaskDemoServiceI {

    @Override
    public void work() {
        System.out.println("-----任务测试-----");
    }
}
```

配置三： 定时任务调度器配置

```
<!-- 定时任务调度器 -->
<bean id="schedulerFactory" lazy-init="false" autowire="no"
      class="org.jeecgframework.core.timer.DataBaseSchedulerFactoryBean">
    <property name="triggers">
        <list>
            <ref bean="taskDemoServiceTaskCronTrigger" />
            <ref bean="smsSendTaskCronTrigger" />
        </list>
    </property>
</bean>
```

11.2. 定时任务在线管理

菜单： 系统监控 -> 定时任务管理

在线方式控制定时任务启动，停止、表达式维护

◀ 首页		定时任务 ×								
定时任务管理										
+ 录入							✎ 编辑		🔍 查看	
任务ID	任务描述	cron表达式	是否生效	运行状态						
1	smsSendTaskCronTrigger	消息中间件定时任务	0 0/1 * * * ?	已生效	停止	▶ 启动	🗑 删除			
2	taskDemoServiceTaskCronTrigger	测试Demo	0 0/1 * * * ?	已生效	停止	▶ 启动	🗑 删除			

编辑 _ □ ×

任务ID:	<input type="text" value="taskDemoServiceTaskCronTrigger"/>
任务描述:	<input type="text" value="测试Demo"/>
cron表达式:	<input type="text" value="0 0/1 * * * ?"/> 通过信息验证!

12. 消息中心

12.1. 简介

消息中心主要是为系统提供消息提醒功能，比如：短信,邮件,微信等推送服务.

主要有以下特点:

- 消息模板化
- 消息定时器
- 可在线测试
- 消息记录可追溯

12.2. 使用方式

消息中间件功能是使用模板技术,以实际业务 SQL 作为数据结果集,填充模板的指定域后,生成一条格式化的系统消息,并通过一定的消息发送途径将其发送

开发的基本路径:

编写消息模板 → 编写业务 SQL → 消息业务配置 → 编写推送测试后台代码 →
编写定时器 → 运行推送测试 → 运行定时器测试 → 消息中心查阅消息推送记录

代码调用方法:

```
TuiSongMsgUtil.sendMessage(msgType, code, Map, sentTo);
```

参数说明:

code	
Map	SQL 查询参数
sentTo	zhangsan@qq.com

12.3. 使用详解

1. 编写消息模板

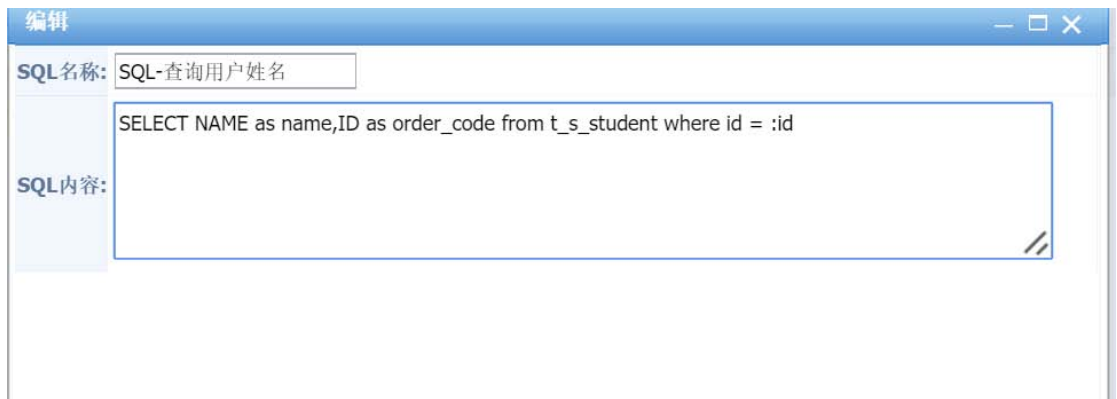


其中模板内容如下:

`${name}`你好, 你的订单`${order_code}`已付款!

模板中的 `name` 和 `order_code` 为可替换代码段,与业务 SQL 相关.

2. 编写业务 SQL



SELECT NAME as name, ID as order_code from t_s_student where id = :id

SQL 中的 as 部分的值与之前的模板中的可替换代码段互相匹配

:id 为可传递参数的 key. 于在 java 方法中填充查询参数使用

3. 消息业务配置



配置 CODE 需要是唯一编码

在业务 SQLID 和消息模板 ID 中下拉选择匹配的业务 SQL 与消息模板

4. 编写后台推送代码



录入业务配置完成后,可以在列表中看到对应的数据,并在操作列中有[推送测试]的功能

此处的推送测试,需要与后台的 java 代码进行联动.

具体代码详见:

```
AjaxJson j = new AjaxJson();
if (StringUtils.isBlank(tSSmsTemplateSql.getCode())){
    j.setSuccess(false);
    j.setMsg("配置CODE不能为空");
}else {
    Map<String,Object> map = new HashMap<String,Object>();
    map.put("id", "4028d881436d514601436d521ae80165");
    String r = TuiSongMsgUtil.sendMessage("消息推送测试333","2", tSSmsTemplateSql.getCode(), map, "411944058@qq.com");
    if ("success".equals(r)){
        j.setSuccess(false);
        j.setMsg(r);
    }
}
}
```

调用方法: TuiSongMsgUtil.sendMessage(msgType, code,Map,sentTo);//发送消息

```
/**
 * sendMessage 统一消息发送接口
 *
 * @param @param msgType 消息类型
 * @param @param code 业务配置CODE
 * @param @param map 数据参数
 * @param @param sentTo 发送给谁
 * @param @return 发送结果
 * @throws
 */
public static String sendMessage(String title,String msgType, String code,
    Map<String, Object> map, String sentTo) {
```

5. 编写定时器

代码路径:

org.jeecgframework.web.sms.util.task.SmsSendTask 也可以编写自己的实体类的方法

XML 配置路径:src\main\resources\spring-mvc-timeTask.xml 进行相关 bean 的配置,

并打开 schedulerFactory 的 list 节点的注释,接入对应的 bean 后即可

6. 测试

任务ID	任务描述	cron表达式	是否生效	运行状态	操作	
1	smsSendTaskCronTrigger	消息中间件定时任务	0 0/1 * * * ?	已生效	运行	[停止][删除]
2	taskDemoServiceTaskCronTrigger	测试Demo	0 0/1 * * * ?	已生效	停止	[启动][删除]

后台打印

```
-----消息发送开始-----
[org.hibernate.hql.internal.ast.HqlSqlWalker] [DEPRECATION] Encountered positional parameter near line 1, column 73. Positional parameter are considered deprecated; use :
DEBUG: setDebug: javaMail version 1.4.1ea-SNAPSHOT
DEBUG: getProvider() returning javax.mail.Provider[TRANSPORT, smtp, com.sun.mail.smtp.SMTPTransport, Sun Microsystems, Inc]
DEBUG SMTP: useEhlo true, useAuth true
DEBUG SMTP: trying to connect to host "smtp.126.com", port 25, isSSL false
220 126.com Anti-spam GT for Coremail System (126com|20140526)
DEBUG SMTP: connected to host "smtp.126.com", port: 25

EHLO chris-surface
250-mail
250-PIPELINING
250-AUTH LOGIN PLAIN
250-AUTH=LOGIN PLAIN
250-coremail 1Uxr2xKj7kG0xkI17xGrU7I0s8FY2U3Uj8Cz28x1UUUUU7Ic2I0Y2Ur1kS3WUCa0xDrUUUUj
250-STARTTLS
250-8BITMIME
DEBUG SMTP: Found extension "PIPELINING", arg ""
DEBUG SMTP: Found extension "AUTH", arg "LOGIN PLAIN"
DEBUG SMTP: Found extension "AUTH=LOGIN", arg "PLAIN"
DEBUG SMTP: Found extension "coremail", arg "1Uxr2xKj7kG0xkI17xGrU7I0s8FY2U3Uj8Cz28x1UUUUU7Ic2I0Y2Ur1kS3WUCa0xDrUUUUj"
DEBUG SMTP: Found extension "STARTTLS", arg ""
DEBUG SMTP: Found extension "8BITMIME", arg ""
DEBUG SMTP: Attempt to authenticate
AUTH LOGIN
334 dXNlcmlhbnVWU6
Yk5jaGZvZGFyZW4=
334 UGFzc3dvenQ6
K1oqK1oqK1oq
835 Error: authentication failed
[org.jeecgframework.core.aop.HiberAspect]当前session为空,无法获取用户
[org.jeecgframework.core.aop.HiberAspect]当前session为空,无法获取用户
-----消息发送结束-----
[org.jeecgframework.core.util.LogUtil][org.jeecgframework.web.sms.util.task.SmsSendTask:run():34] - -----消息中间件定时任务结束-----
[org.jeecgframework.core.util.LogUtil][org.jeecgframework.web.sms.util.task.SmsSendTask:run():37] - 总耗时1675毫秒
[org.jeecgframework.core.util.LogUtil][org.jeecgframework.web.sms.util.task.SmsSendTask:run():28] - -----消息中间件定时任务开始-----
```

7. 消息中心

消息标题	消息类型	发送人	接收人	内容	创建日期	发送时间	发送状态	备注
消息推送测试333	邮件提醒		411944058@qq.com	张三你好, 你的订单4028d881436d514601436d521ae80165已付款!	2015-12-24 09:43:04.0	2015-12-24 11:41:58.0	发送失败	认证失败错误的用户名或
消息推送测试333	邮件提醒		411944058@qq.com	张三你好, 你的订单4028d881436d514601436d521ae80165已付款!	2015-06-05 17:05:59.0	2015-06-05 17:06:01.0	发送失败	认证失败错误的用户名或

由于本地没有接入短信网关只预制了接口,因此发送是失败状态。

12.4. 系统配置文件

配置文件: src/main/resources/sysConfig.properties

邮件配置:

```
38
39
40 mail.smtpHost=s[REDACTED]
41 mail.sender=[REDACTED]@126.com
42 mail.user=an[REDACTED]
43 mail.pwd=*****
44
```

短信配置:

org.jeecgframework.web.system.sms.util.CMPPSenderUtil.sendMsg

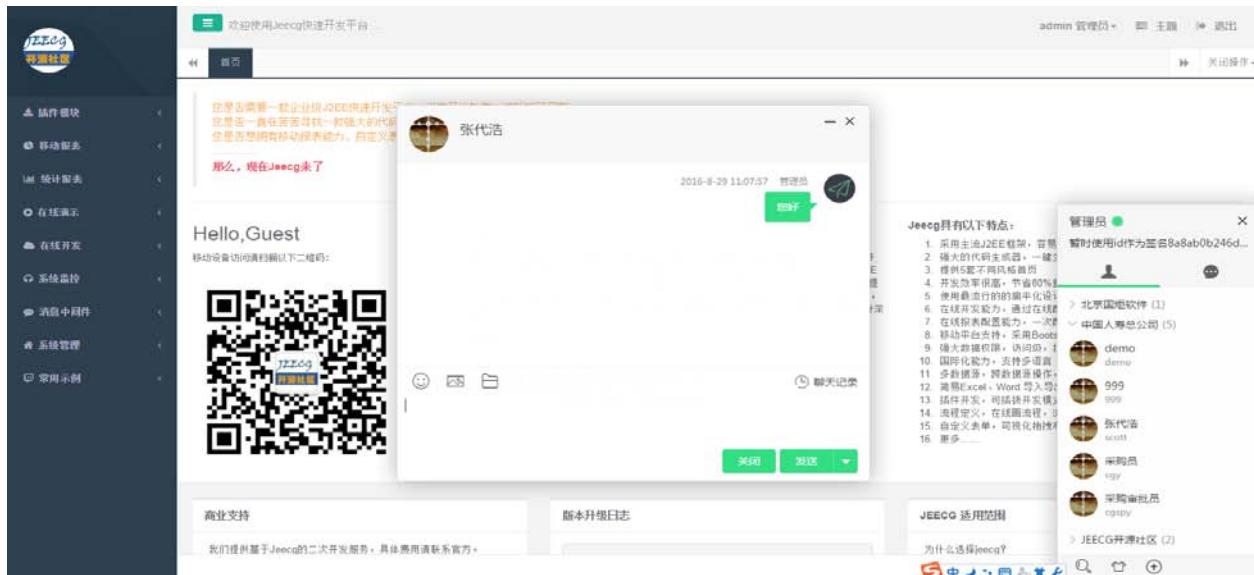
13. 插件模块集成文档

13.1. 在线聊天插件

前提： 采用 jeecg_3.6.3 版本以上（Maven 工程）

插件项目： 在线聊天工具

截图：



集成步骤：

1. pom.xml 引入 {聊天插件} 依赖
 <!-- 在线聊天工具 -->
 <dependency>
 <groupId>org.p3framework</groupId>
 <artifactId>jeecg-p3-biz-chat</artifactId>
 <version>1.0.0</version>
 <type>jar</type>
 <scope>compile</scope>
 </dependency>
2. 增加聊天 websocket 配置文件：config.js
 路径位置：src/main/webapp/plugin/layim/config.js
 修改 websocket ip 配置：
 var chatIp = "192.168.0.113";
 <<config.js>>
3. 增加文件：layui.jsp
 路径：src/main/webapp/context/layui.jsp
 <<layui.jsp>>

4. 首页引入通用 layui.jsp

修改文件：src/main/webapp/webpage/main/hplus_main.jsp

代码修改位置：652 行，新增如下代码：

<!-- 在线聊天 -->

<%@include file="/context/layui.jsp"%>

14. 附录

14.1. UI 标签规则

- 1) 列表页面，datagrid 的 name 属性不允许存在重复的，否则页面显示白板：

```
<t:datagrid name="jeecgDemoList" title="开发 DEMO 列表"  
actionUrl="jeecgDemoController.do?datagrid" idField="id" fit="true">
```

- 2) 表单验证采用 Validform
- 3) 时间控件采用 my97，不要使用 easyui 的时间控件，因为加载效率慢
- 4) 上传文件使用规则

使用 SWFUpload 插件上传，可同时传多个文件，需要安装 Falsh 软件，

```
<div class="form">  
  <t:upload name="fiels" buttonText="上传文件"  
    uploader="systemController.do?saveFiles" extend="office"  
    id="file_upload" formData="documentTitle"></t:upload>  
</div>  
<div class="form" id="filediv" style="height:50px">  
</div>
```

extend 参数说明：

office：表示可上传 offices 格式后缀的文件

pic：表示可上传图片格式后缀的文件

详细文档请看《JEECG UI 标签帮助文档》

- 5) jsp 代码注释规范，采用隐式注释不能用显式注释，不然标签还是能读到：

隐式注释：<%-- --%>

显式注释：<!-- -->

14.2. 列表自定义查询条件

简述

代码生成器默认生成的查询方式为单字段查询，如果想实现字段组合查询，需要采用如下方式。

实现步骤

第一步：设置 dategrid 字段查询属性 query="true"

第二步：对应 query="true" 的 dategrid 字段设置查询字段组件

```
<input type="text" name="userName" id="userName" style="width: 80px"/>
```

第三步：设置查询按钮

```
<a href="#" class="easyui-linkbutton" iconCls="icon-search"
onclick="userListsearch()">查询</a>
```

注意点

1. 这种写法 t:dgToolBar 这个标签不能使用，不然会有冲突，查询 form 显示不出来；
2. 查询函数的名字规则 "[dategrid 组件 name]search()"

[1]. dategrid 组件 name

```
<t:dategrid name="userMe"
```

[2]. 组合查询 DIV

```
<div id="userMetb"
```

[3]. 查询按钮对应的 js 方法

```
<a href="#" class="easyui-linkbutton" iconCls="icon-search"
onclick="userMearch()">查询</a>
```

参考示例： /WebRoot/webpage/system/user/userList.jsp

示例代码如图 14-1 所示：

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@include file="/context/mytags.jsp"%>
<t:dataGrid name="userList" title="用户管理" actionUrl="userController.do?datagrid" idField="id">
  <t:dgCol title="编号" field="id" hidden="false"></t:dgCol>
  <t:dgCol title="用户名" sortable="false" field="userName" query="true" width="20"></t:dgCol>
  <t:dgCol title="部门" field="TSDepart_depar tname"></t:dgCol>
  <t:dgCol title="真实姓名" field="realName" query="true"></t:dgCol>
  <t:dgCol title="状态" sortable="true" field="status" replace="正常_1, 禁用_0, 超级管理员_1"></t:dgCol>
  <t:dgCol title="操作" field="opt" width="100"></t:dgCol>
  <t:dgFunOpt funname="szqm(id)" title="设置签名" />
  <t:dgDelOpt title="删除" url="userController.do?del&id={id}@userName={userName}" />
</t:dataGrid>
<div id="userListtb" style="padding: 3px; height: 25px">
  <div style="float:left;">
    <a href="#" class="easyui-linkbutton" plain="true" icon="icon-add" onclick="add('用户录入', 'userController.do?addorupdate')">用户录入</a>
    <a href="#" class="easyui-linkbutton" plain="true" icon="icon-add" onclick="update('用户编辑', 'userController.do?addorupdate', 'id')">用户编辑</a>
  </div>
  <div align="right">
    用户名: <input type="text" name="userName" id="userName" style="width: 80px"/>
    真实姓名: <input type="text" name="realName" id="realName" style="width: 80px"/>
    <a href="#" class="easyui-linkbutton" iconCls="icon-search" onclick="userListsearch()">查询</a>
  </div>
</div>a
```

图 14-1 组合查询示例代码

14.3. Formvalid新增属性tiptype的使用

Formvalid 中的 tiptype 用来定义提示信息的显示方式，一共有 4 种取值，在其官方的说明中，不同取值的含义如下：

取值	含义
1	自定义弹出框提示；
2	侧边提示(会在当前元素的父级的 next 对象的子级查找显示提示信息的对象，表单以 ajax 提交时会弹出自定义提示框显示表单提交状态)；
3	侧边提示(会在当前元素的 siblings 对象中查找显示提示信息的对象，表单以 ajax 提交时会弹出自定义提示框显示表单提交状态)；
4	侧边提示(会在当前元素的父级的 next 对象下查找显示提示信息的对象，表单以 ajax 提交时不显示表单的提交状态)

在 jeecg 中，tiptype 的属性配置代码如下：

```
<t:formvalid formid="formobj" dialog="true" usePlugin="password" layout="table"
tiptype="1" action="jeecgOrderMainController.do?save">
```

与官方的用法不同的是，JEECG 中对取值为 1 时的样式以及校验方式进行了改造，官方版是在提交时才给出提示，而 JEECG 中是在 onblur 的时候就会提示，当输入正确后，1 秒中后会自动消失。

注：<t:formvalid>标签中不写 tiptype 时默认为 4. 即侧边显示。

使用建议：单表可以不用给定 tiptype 属性，即使用默认的侧边校验，主从表的数据校验给定 tiptype="1"。

单表和主从表的数据校验提示效果分别如图 14-2 和图 14-3 所示。

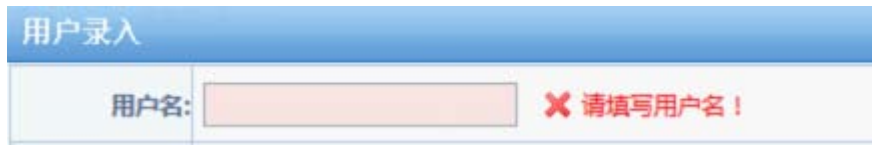


图 14-2 单表使用侧边提示方式

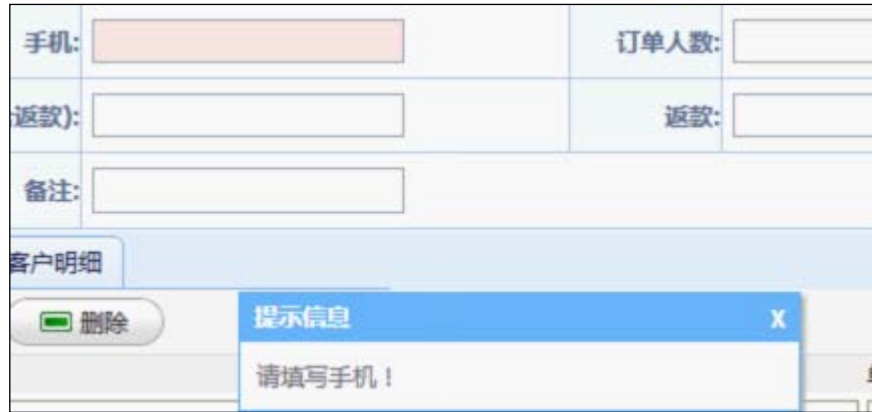


图 14-3 主从表使用弹出提示方式

14.4. 使用toolbar 自定义js参数规则

第一步：定义按钮

```
<t:dgToolBar title="JS增强" icon="icon-edit"
url="cgFormHeadController.do?jsPlugin" funname="jsPlugin"></t:dgToolBar>
```

第二步：定义 js 方法

三个参数说明：

- 三个参数缺一不可
- 三个参数顺序不能变
- 有且只有三个参数
- id 为 datagrid 的 name 属性

```
function jsPlugin(title,url,id){
var rowData = $('#'+id).datagrid('getSelected');
if (!rowData) {
tip('请选择编辑项目');
return;
}
url += '&id='+rowData.id;
$.dialog({
content: "url:"+url,
```

```
lock : true,
title: "JS增强编辑["+rowData.tableName+"-"+rowData.content+"]",
opacity : 0.3,
width:900,
height:500,
cache:false,
ok: function(){
    iframe = this.iframe.contentWindow;
    iframe.goForm();
    return false;
},
cancelVal: '关闭',
cancel: true /*为true等价于function(){}*/
});
}
```

14.5. 表单字段重复校验方法

目的：实现通用表单字段重复校验，

例如：部门管理模块，部门名称重复校验

```
<input name="departname" class="inputxt"
value="{depart.departname }" validType="t_s_depart,departname,id"
datatype="s3-10">
```

1) 代码配置

给input标签，增加validType属性，格式如：t_s_depart,departname,id 即（数据表名称、对应的数据库字段、业务实体的隐藏域主键的Id属性）

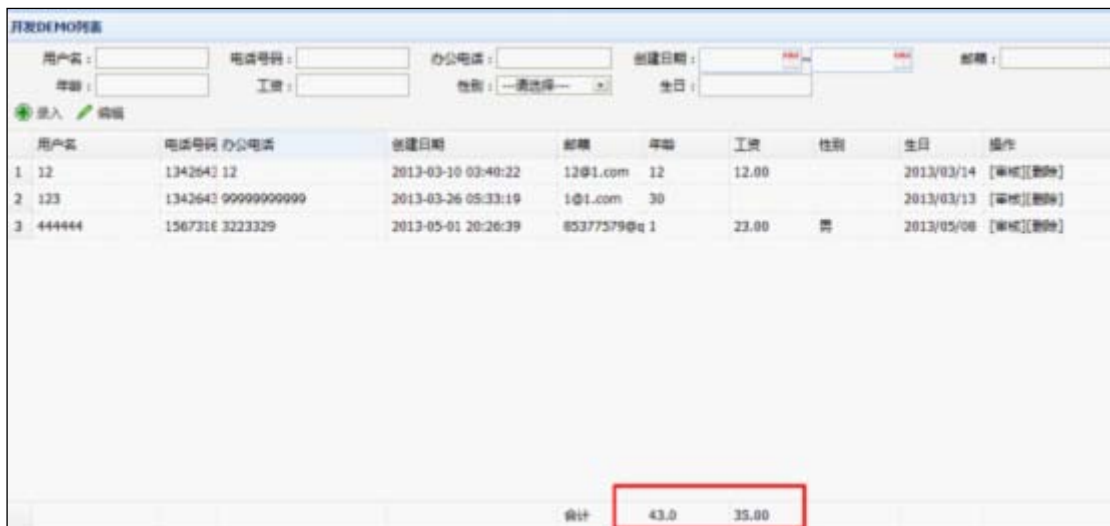
2) 消息提示方式，两种方式

[1]. 提示弹出层：如下所示：给t:formvalid 增加tiptype="1" 属性
<t:formvalid formid="formobj" tiptype="1" layout="table"...

[2]. 提示信息在文本框后面提示
不需要给t:formvalid 增加任何属性。

14.6. 数据列表合计功能

进行数据的列表展示时，为数据显示合计数是一个很有用的功能，在 jeecg 的 datagrid 中实现该功能的效果如图 12-4 所示。



用户名称	电话号码	办公电话	创建日期	邮箱	年龄	工资	性别	生日	操作
1 12	1342642	12	2013-03-30 03:40:22	12@1.com	12	12.00		2013/03/14	[审核][删除]
2 123	1342642	999999999999	2013-03-26 05:33:19	1@1.com	30			2013/03/13	[审核][删除]
3 444444	1567316	3223329	2013-05-01 20:26:39	85377579@qq.1		23.00	男	2013/05/08	[审核][删除]
合计						43.0	33.00		

图 12-4 列表数据合计效果图

该功能的实现，主要是通过加载 datagrid 的数据时，统计出所需的合计值，并放在 datagrid 对象的 footer 中。示例代码如下：

```
1 @RequestMapping(params = "datagrid")
2 public void datagrid(JeecgDemo jeecgDemo, HttpServletRequest request,
3   HttpServletResponse response, DataGrid dataGrid) {
4     CriteriaQuery cq = new CriteriaQuery(JeecgDemo.class, dataGrid);
5     // 查询条件组装器
6     org.jeecgframework.core.extend.hqlsearch.HqlGenerateUtil.installHql(cq,
7   jeecgDemo);
8     String ctBegin = request.getParameter("createTime_begin");
9     String ctEnd = request.getParameter("createTime_end");
10    if (StringUtil.isNotEmpty(ctBegin) && StringUtil.isNotEmpty(ctEnd)) {
11        try {
12            cq.ge("createTime", new
13   SimpleDateFormat("yyyy-MM-dd").parse(ctBegin));
14            cq.le("createTime", new
15   SimpleDateFormat("yyyy-MM-dd").parse(ctEnd));
16        } catch (ParseException e) {
17            e.printStackTrace();
18        }
19        cq.add();
20    }
21    this.jeecgDemoService.getDataGridReturn(cq, true);
22    // update-begin--Author:zhaojunfu Date:20130520 for: TASK #109 datagrid
23    // 标签没有封装合计功能
24    String total_salary =
25    String.valueOf(jeecgDemoService.findOneForJdbc("select sum(salary) as ssum
26    from jeecg_demo").get("ssum"));
```

```
20 /*
21 * 说明: 格式为 字段名:值(可选, 不写该值时为分页数据的合计) 多个合计 以 , 分割
22     */
23 dataGrid.setFooter("salary:"+total_salary+",age,email:合计");
24 //update-end--Author:zhaojunfu Date:20130520 for: TASK #109 datagrid 标
    签没有封装合计功能
25     TagUtil.datagrid(response, dataGrid);
26 }
```

在该示例代码中, 需要重点注意的是这里的第 23 行:

```
dataGrid.setFooter("salary:"+total_salary+",age,email:合计");
```

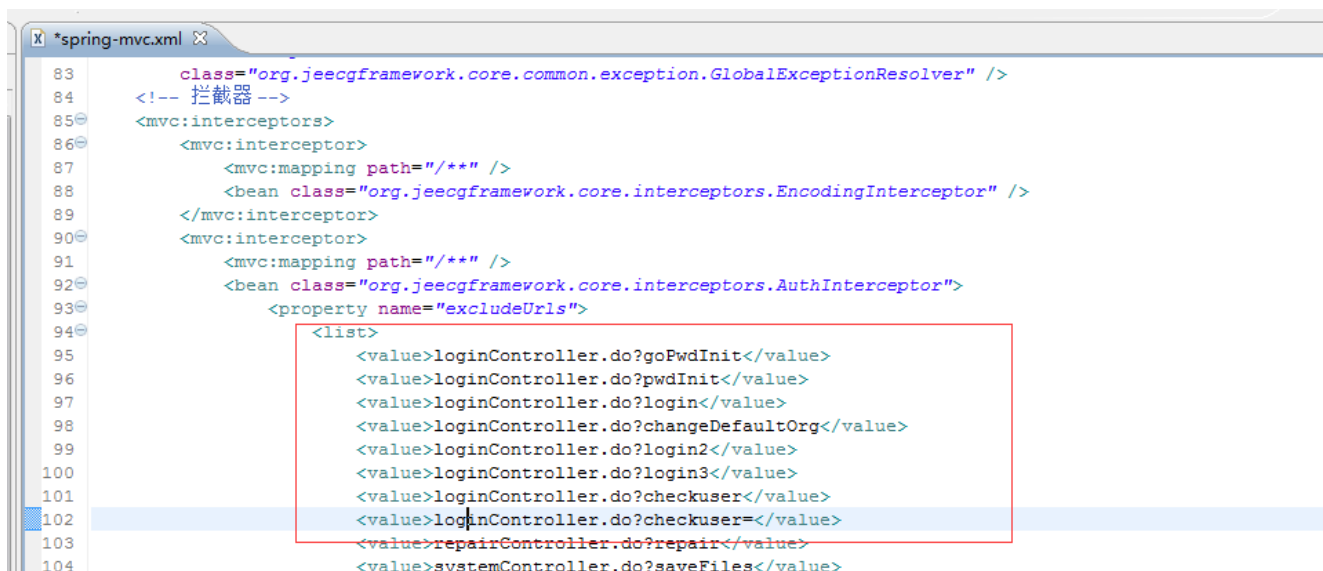
setFooter() 方法接收一个字符串, 其格式为 s: 字段名[:值], 其中值为选填项, 填了则使用给定的值, 没填则自动统计分页合计, 示例:

```
salary:35.00,age,email:合计
```

这里将 salary 的合计值通过查询数据库得出, 而 age 则通过当前分页数据自动合计, email 给定一个值“合计”, 其作用是在 datagrid 对应于 email 列的下方显示一个说明信息。

14.7. 登录权限拦截器排除方法

配置文件: src/main/resources/spring-mvc.xml



14.8. 列表拓展字段展示

Control 层面实现:

```
//-----jeecg 标签列表, 扩展字段显示-----
```

```
List<GzUserInfoYw> gzUserInfoList = dataGrid.getResults();
for(GzUserInfoYw temp:gzUserInfoList){
    Map<String, Map<String, Object>> extMap = new HashMap<String,
    Map<String, Object>>();
    Map<String, Object> m = new HashMap<String, Object>();
    m.put("mobile", "手机号值");
    m.put("nicknameTxt", "昵称");
    extMap.put(temp.getId(), m);
}
TagUtil.datagrid(response, dataGrid, extMap);
```

14.9. JEECG常见问题解决贴

<http://www.jeecg.org/forum.php?mod=viewthread&tid=1830&extra=page%3D1>

<https://my.oschina.net/matt0614/blog/822225>