JEHC开源开发文档

## 简介

平台版本开发版本分为Eclipse和STS两个版本，采用技术为：spring，springMVC，Mybatis，Activiti5，（Activiti可视化设计器基于IE，火狐，谷歌，360等浏览器），Solr4.10，Mysql，Netty，Redis，Ehcache，服务器监控模块，tk压缩，Extjs6.2，bootstrap，RPC服务 ，Junit单元测试，Logback与Log4j共用，同时融入了Hessian，session共享，数据库读写分离，MQ消息中间件等技术

## 交流及其他说明：

由于空余时间不多，只能写一部分开发文档模块 主要还是交流为目的

本平台采用开源方式，交流QQ群：330370132

项目地址一（maven多工程依赖版本）：<http://git.oschina.net/jehc/jehc>

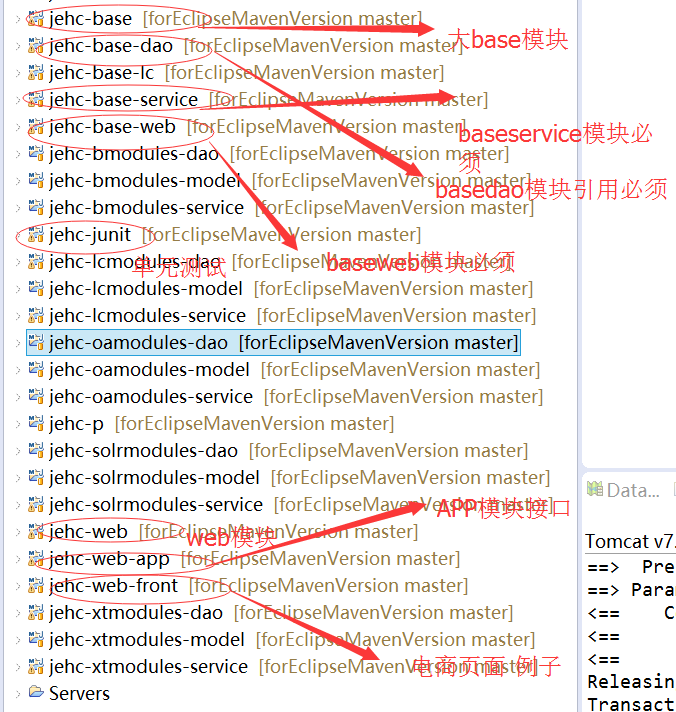
项目地址二（非maven版本）<https://gitee.com/jehc/jehc-none-maven>

项目地址三（maven单工程版本）<https://gitee.com/jehc/jehc-admin>

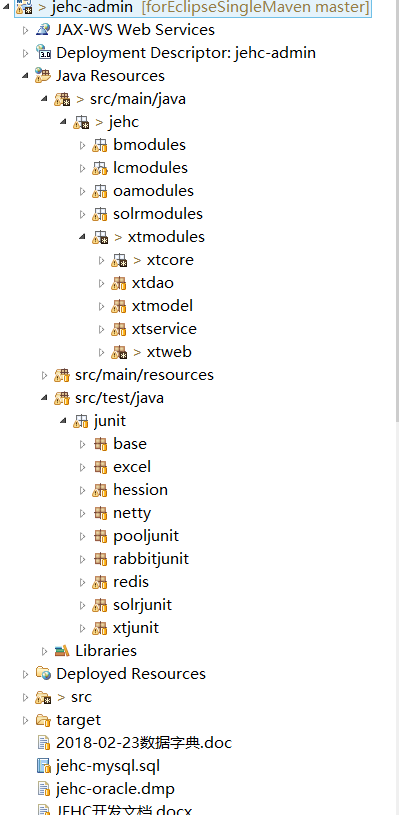
**JEHC平台目前支持Mysql5.6++及Oracle11g数据库版本**

## 特地说明：maven单工程版本、maven多工程依赖版本与、非maven版本功能一致

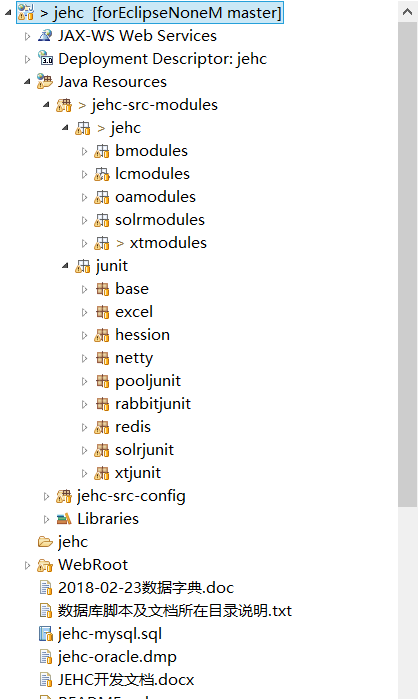
1.Maven多工程依赖版本目录说明如下：



2.Maven单版本目录说明如下：



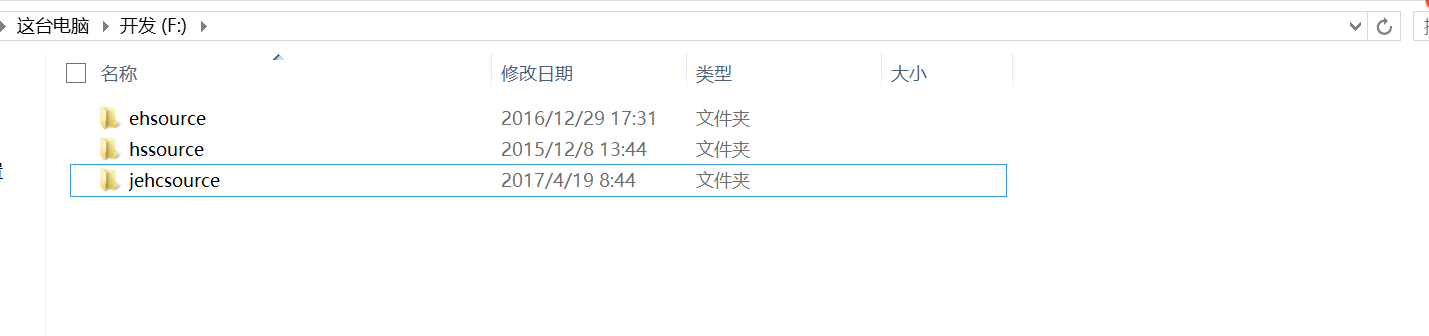
3非maven功能目录

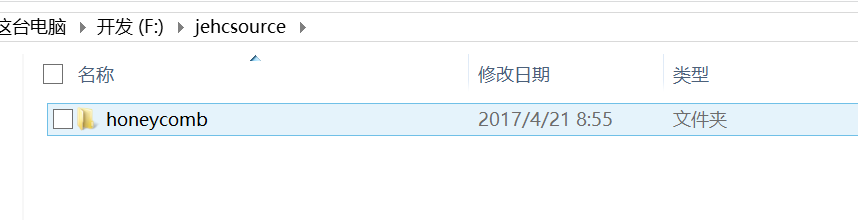


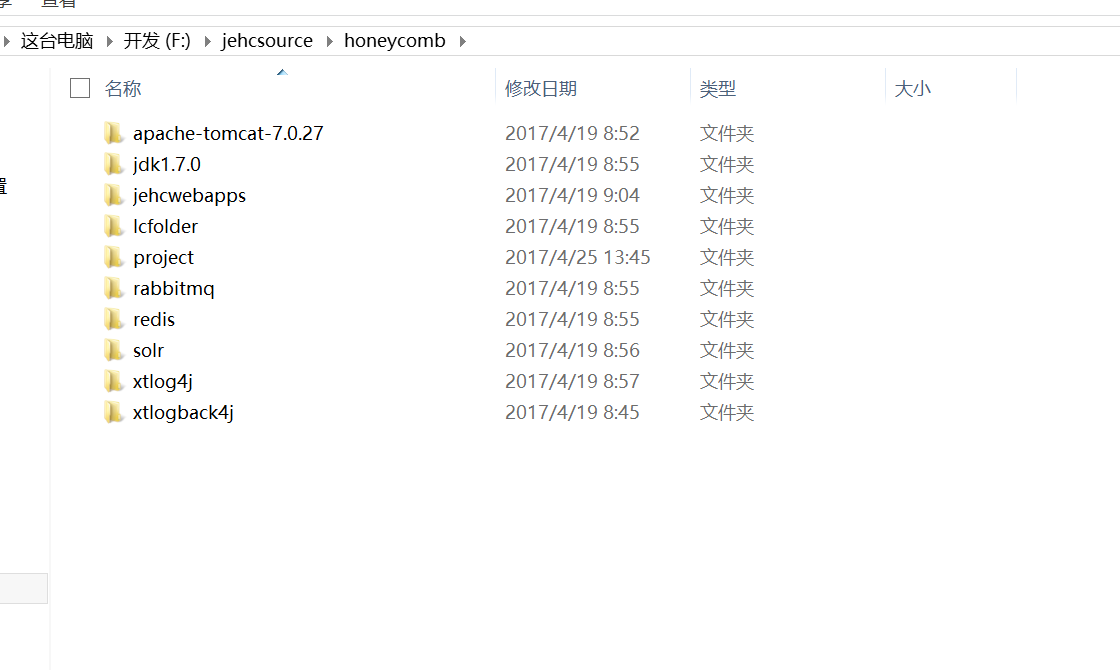
下面以非maven为例子同maven版本功能一致

## 开发环境

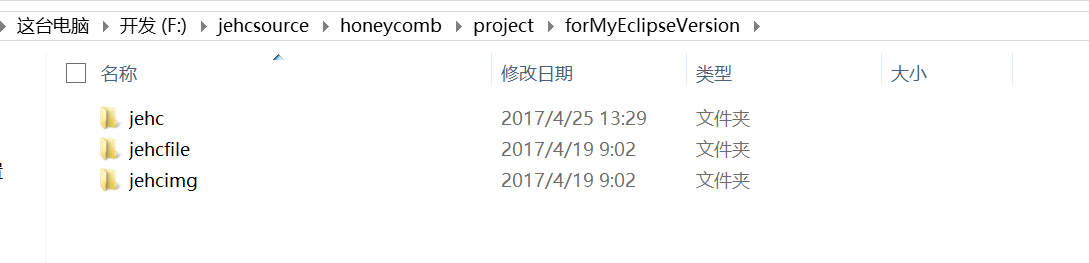
建议开发人员开发环境以如下结构：











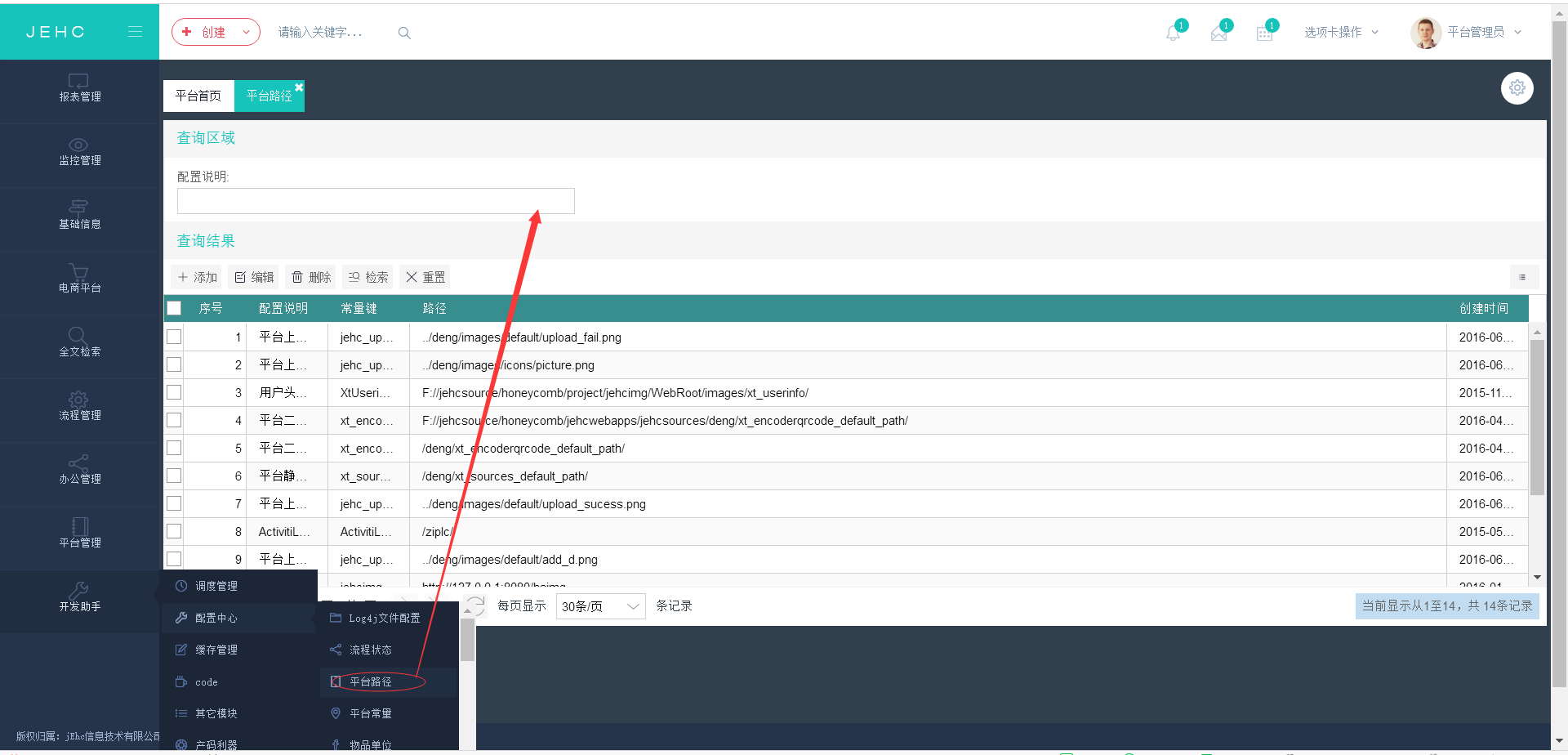
### 开发条件

**强烈建议采用eclipse-mars版本开发**

### 2．Solr如果被使用，则服务必须开启，本平台开发环境包括了SOlR4.10服务即独立应用

### 3．Mysql5.6++版本或Oracle11g版本

### 4．平台采用统一上传文件管理（即必须使用建立文件工程如jehc-file工程专门用来存放文件或图片等，当然也可在JEHC工程中建立文件存储目录或你自定义文件工程。建立结束之后必须在JEHC平台中菜单-开发助手->配置中心->配置->平台路径模块进行配置如图：）



### 5．JDK版本在1.7++如果不采用SOLR则1.6可使用，Tomcat版本为7++

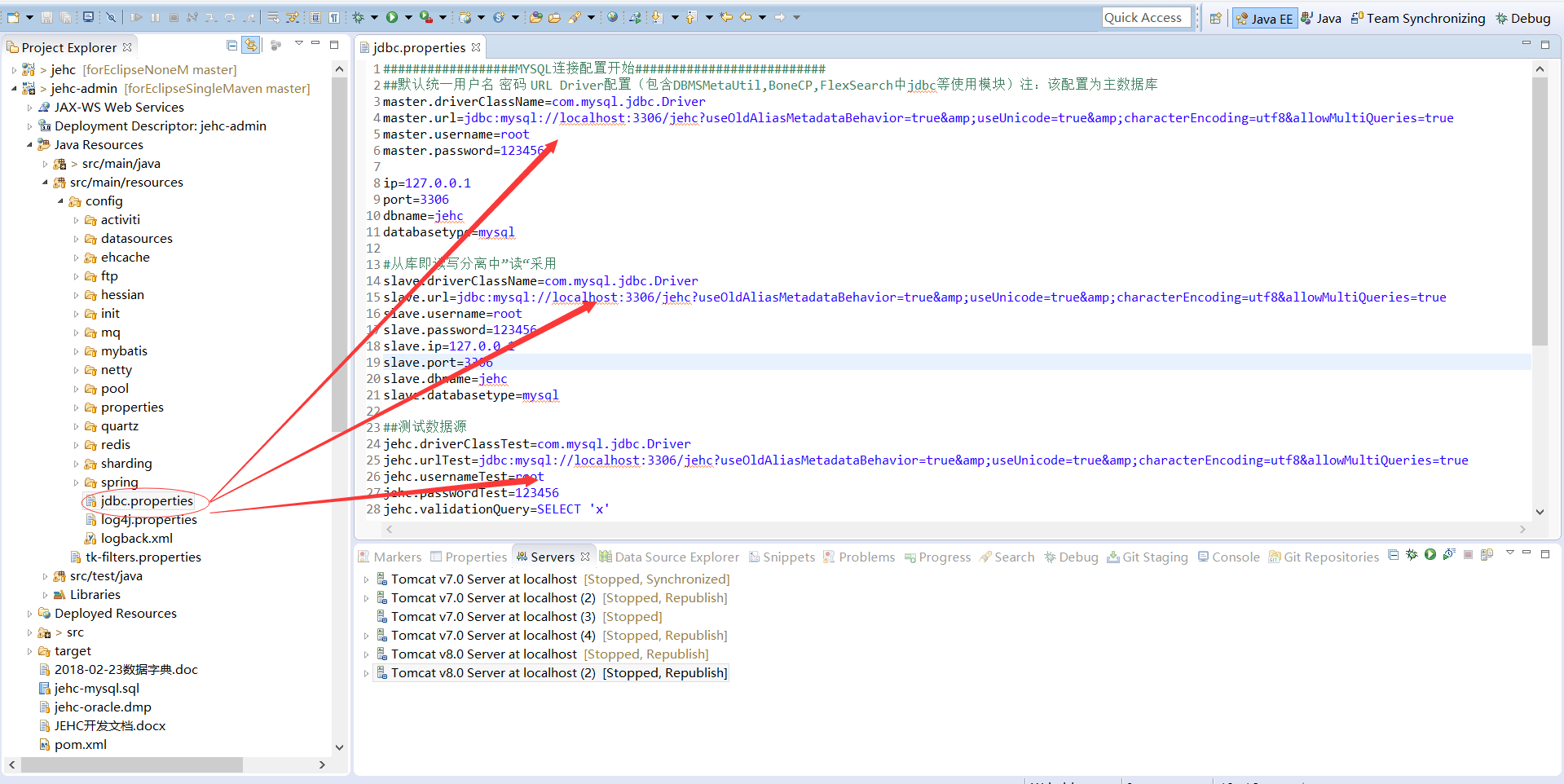
### 6. 注意（如果代码生成器过来的 表一定是存在主键并且一个 并且为UUID类型32位）

### 7．初始化用户名及密码admin/123456

## 项目部署（以jehc-admin单工程maven为例，jehc多工程同其）

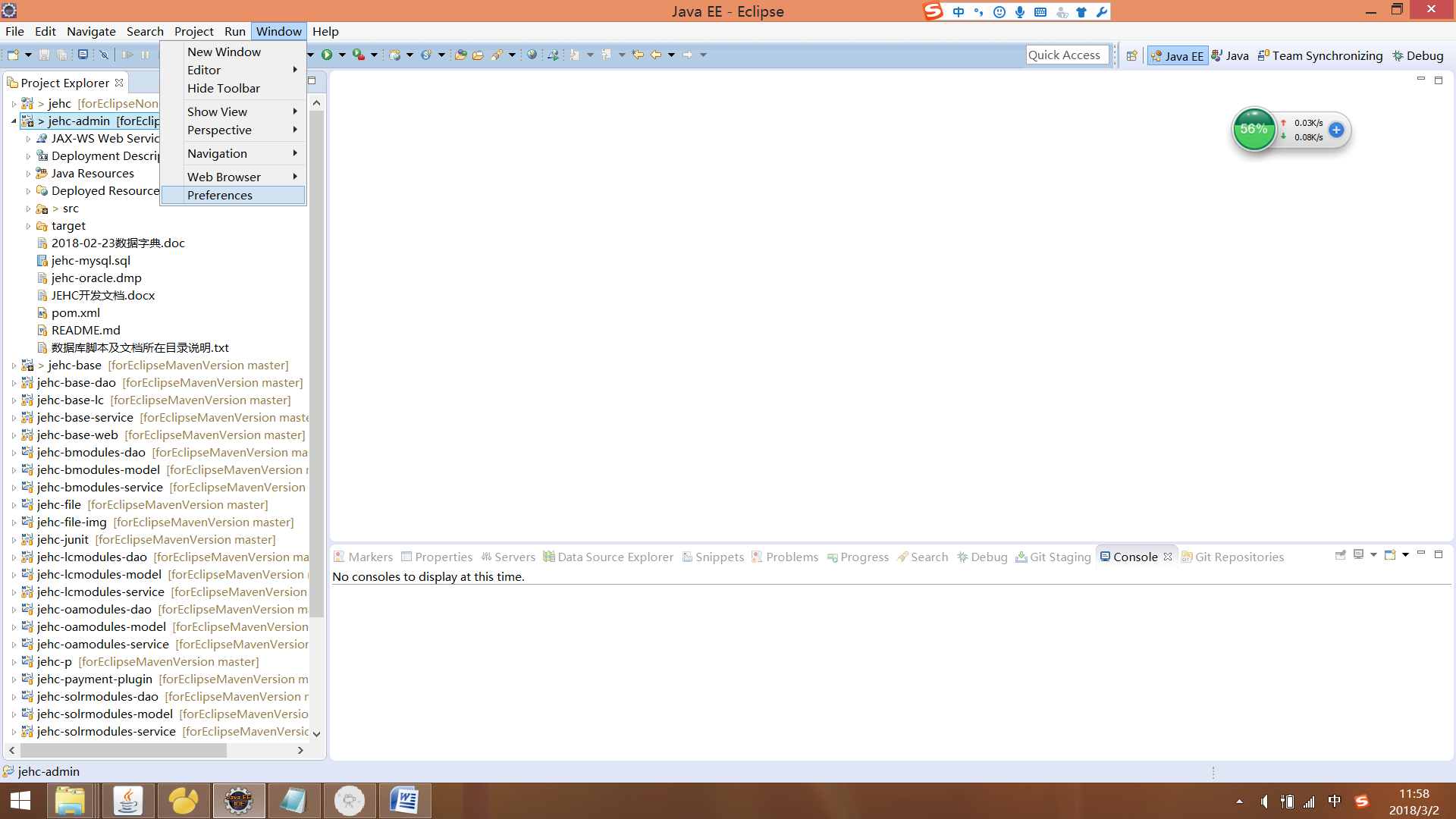
### 1.jdbc配置

当前项目支持读写分离操作 如果你的环境 没有配置主从数据库层级 则可以设置为一样的即可，JEHC平台目前支持Mysql5.6++及Oracle11g数据库版本

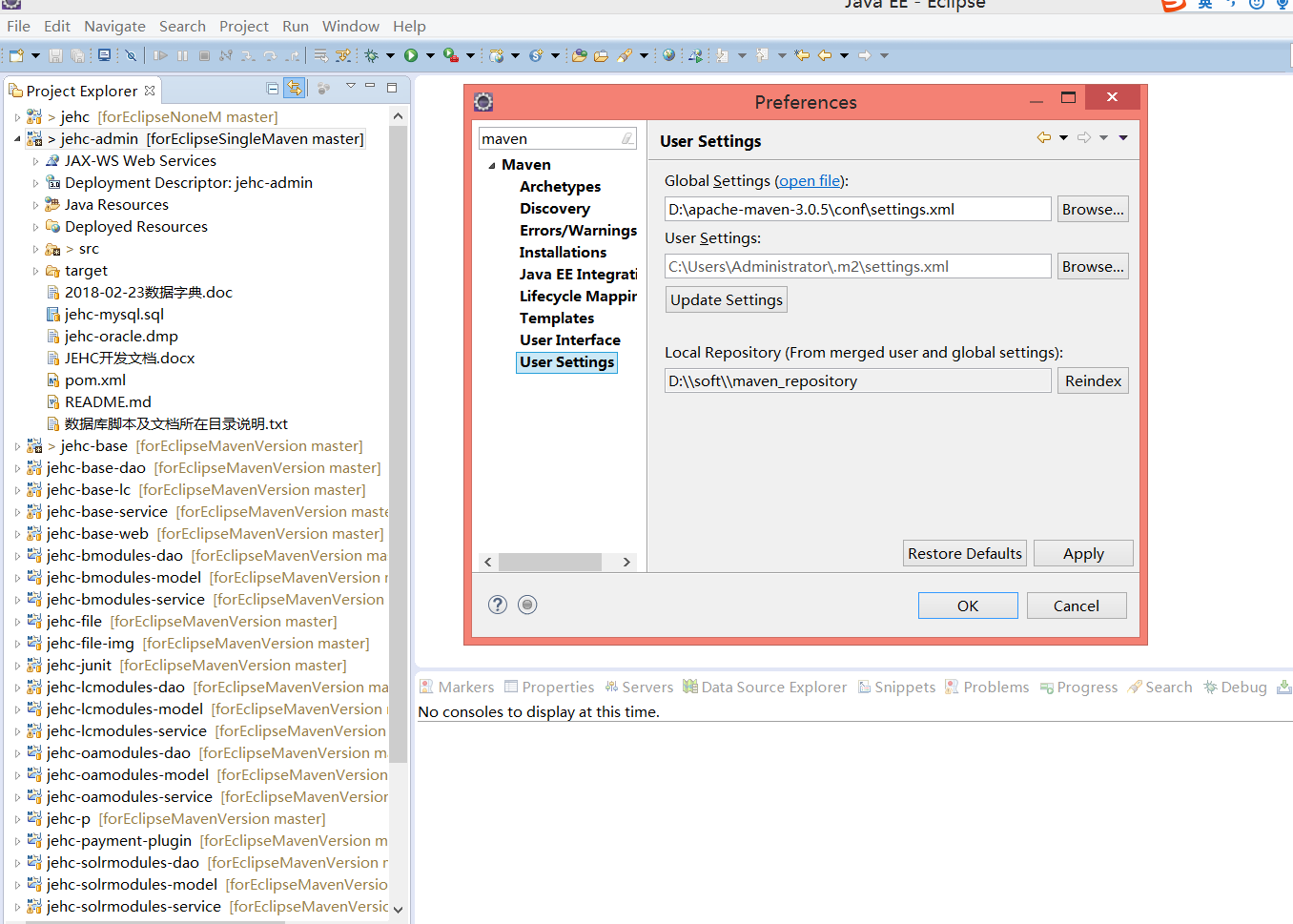


### 2.maven配置

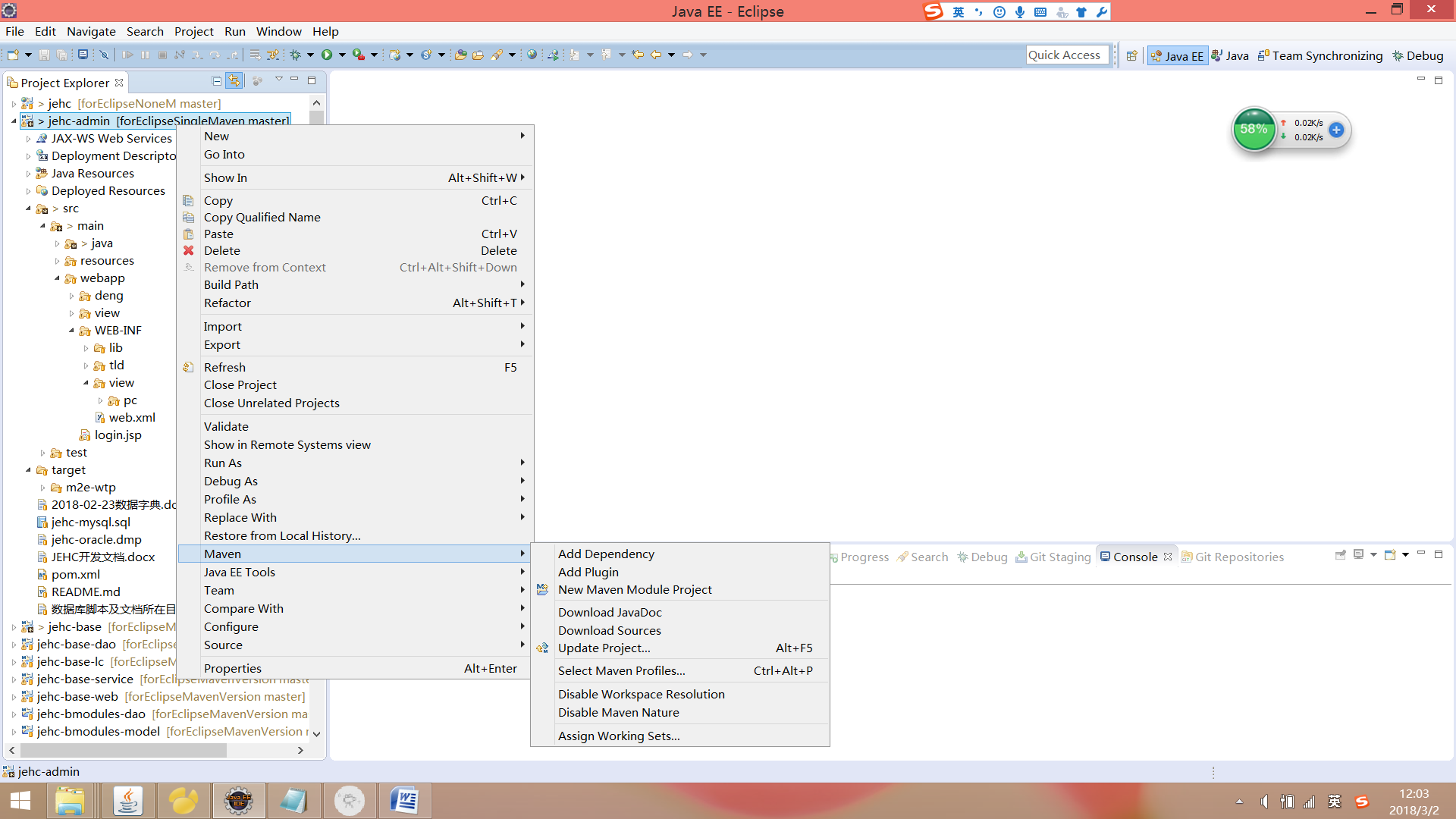
1）window属性



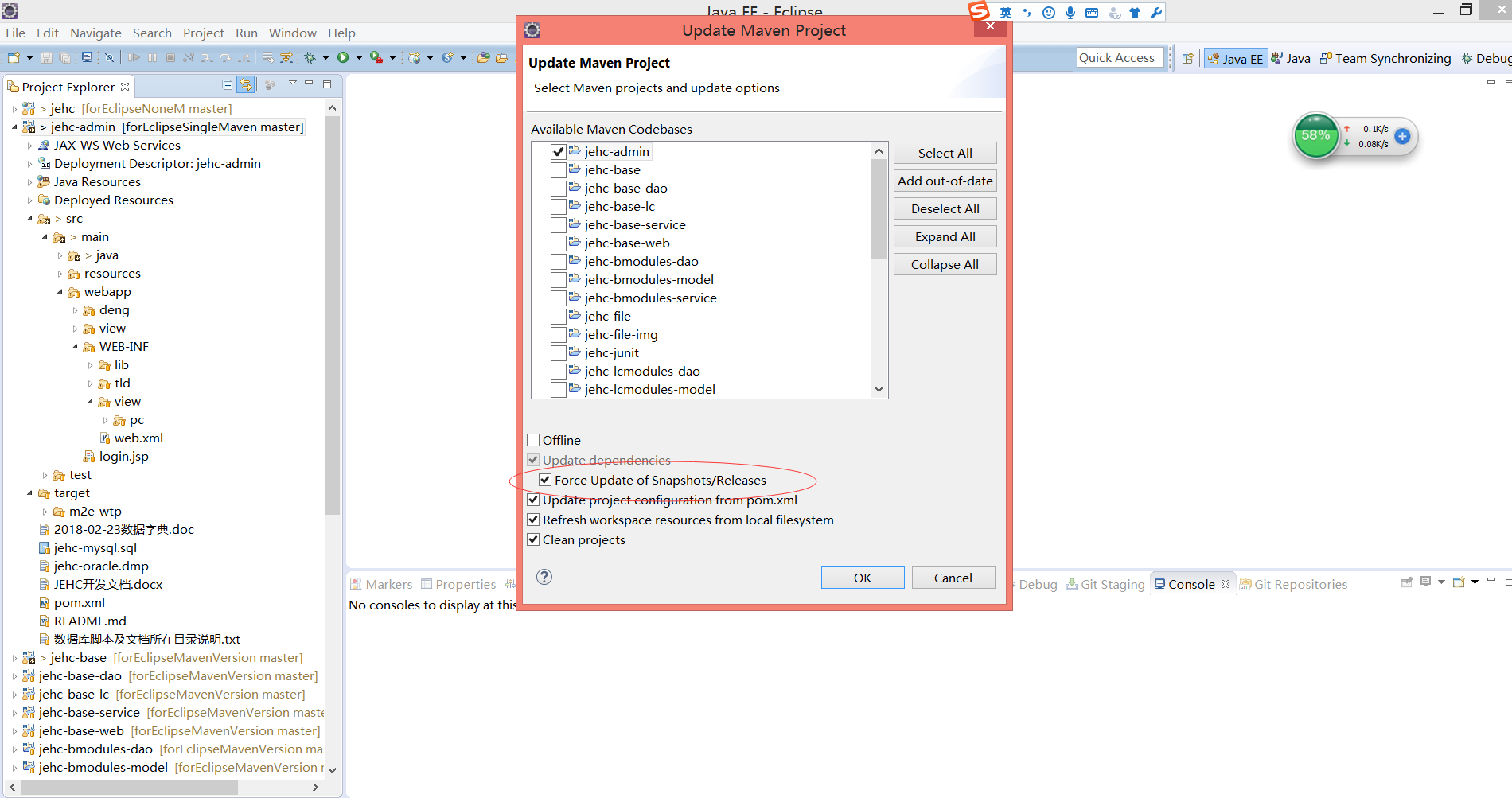
2）搜索maven并点击user Settings 修改maven配置



3)右键jehc-admin工程maven-----点击update-project…按钮

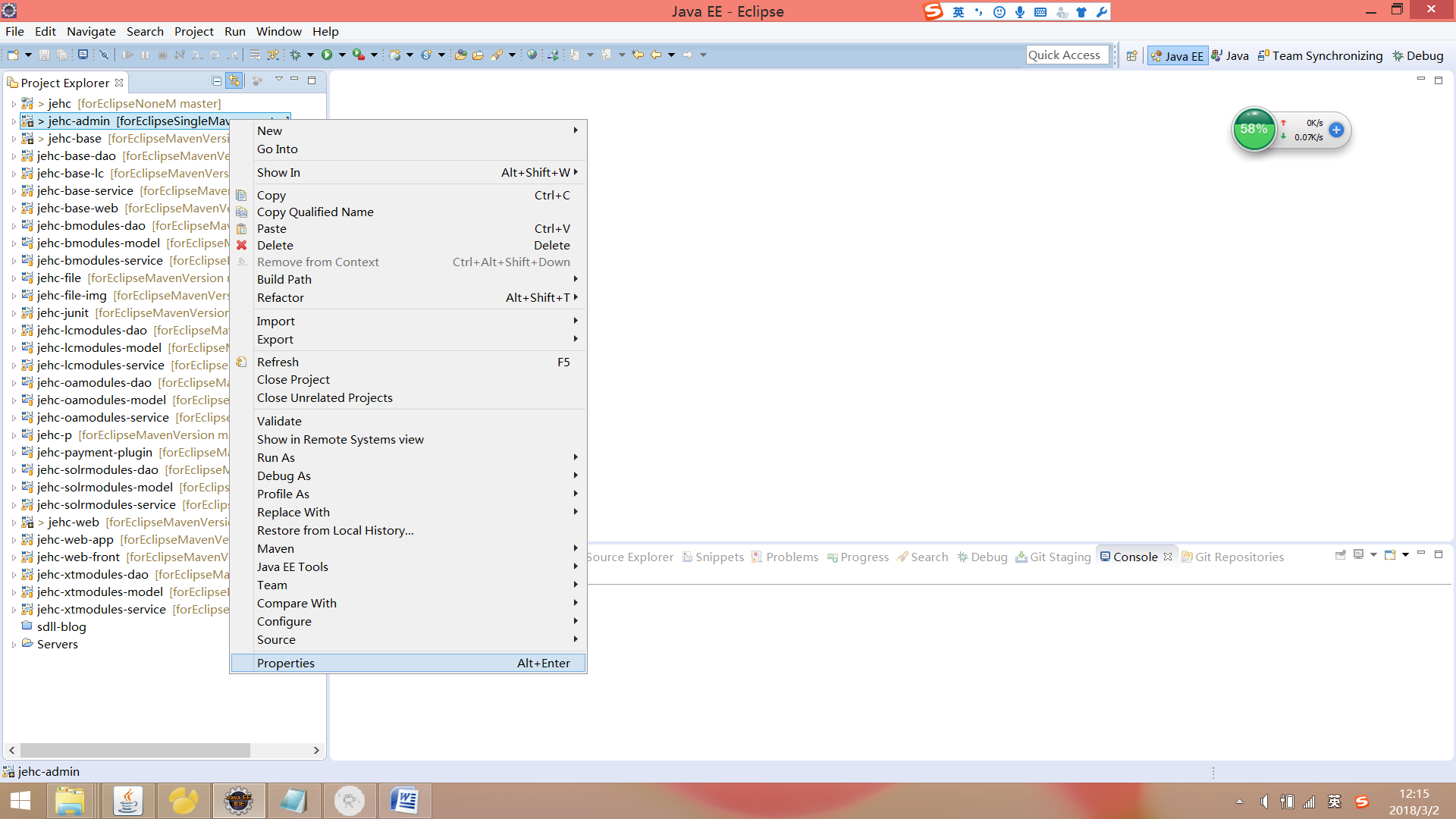


4）勾选force update of shapshots/Releases并点击OK 此时jehc-admin工程 更新中

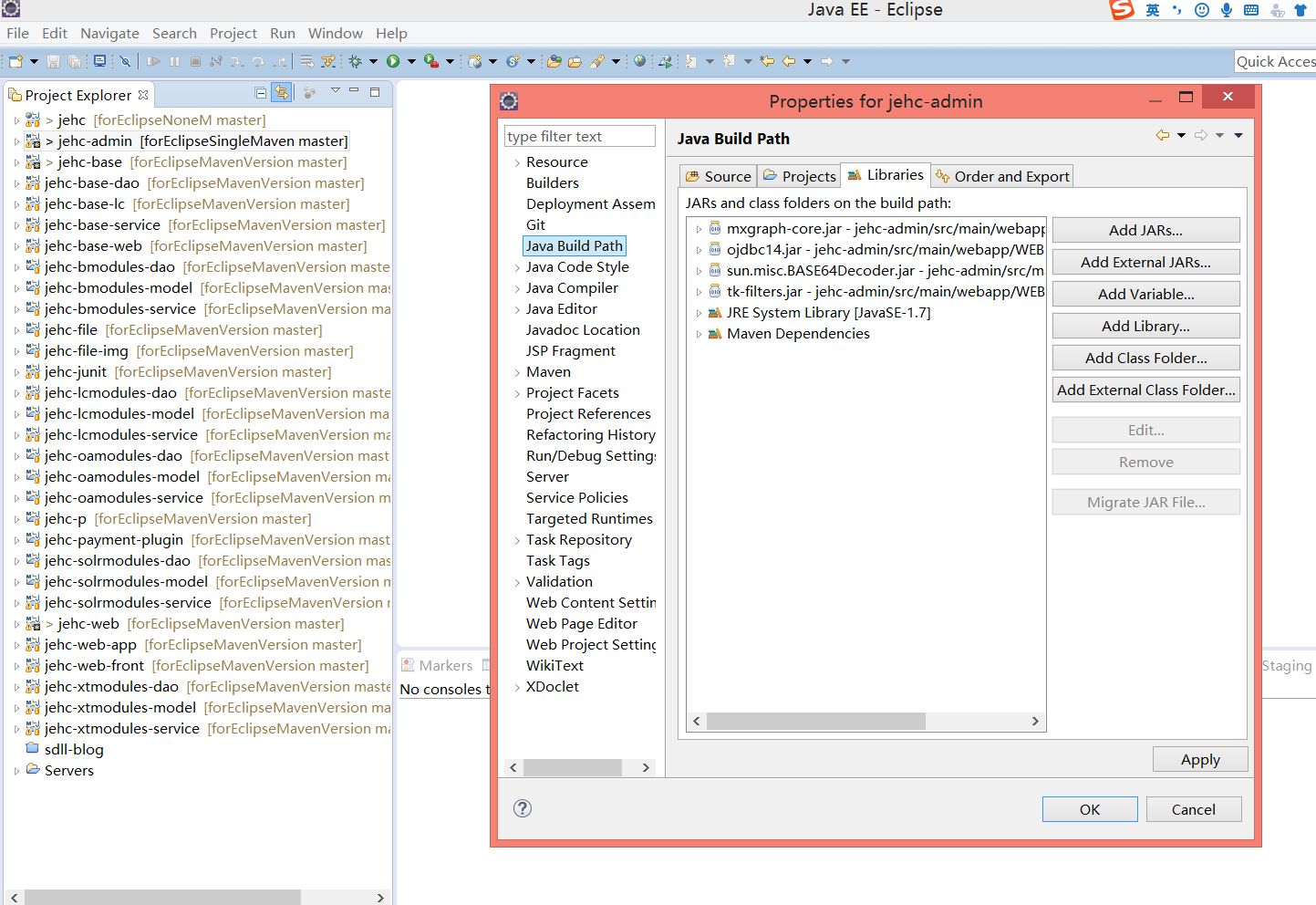


### 配置buildPath

1）右键jehc-admin工程并点击Propertis



2）修改其jre路径等及其他相关jar是否正确并点击OK

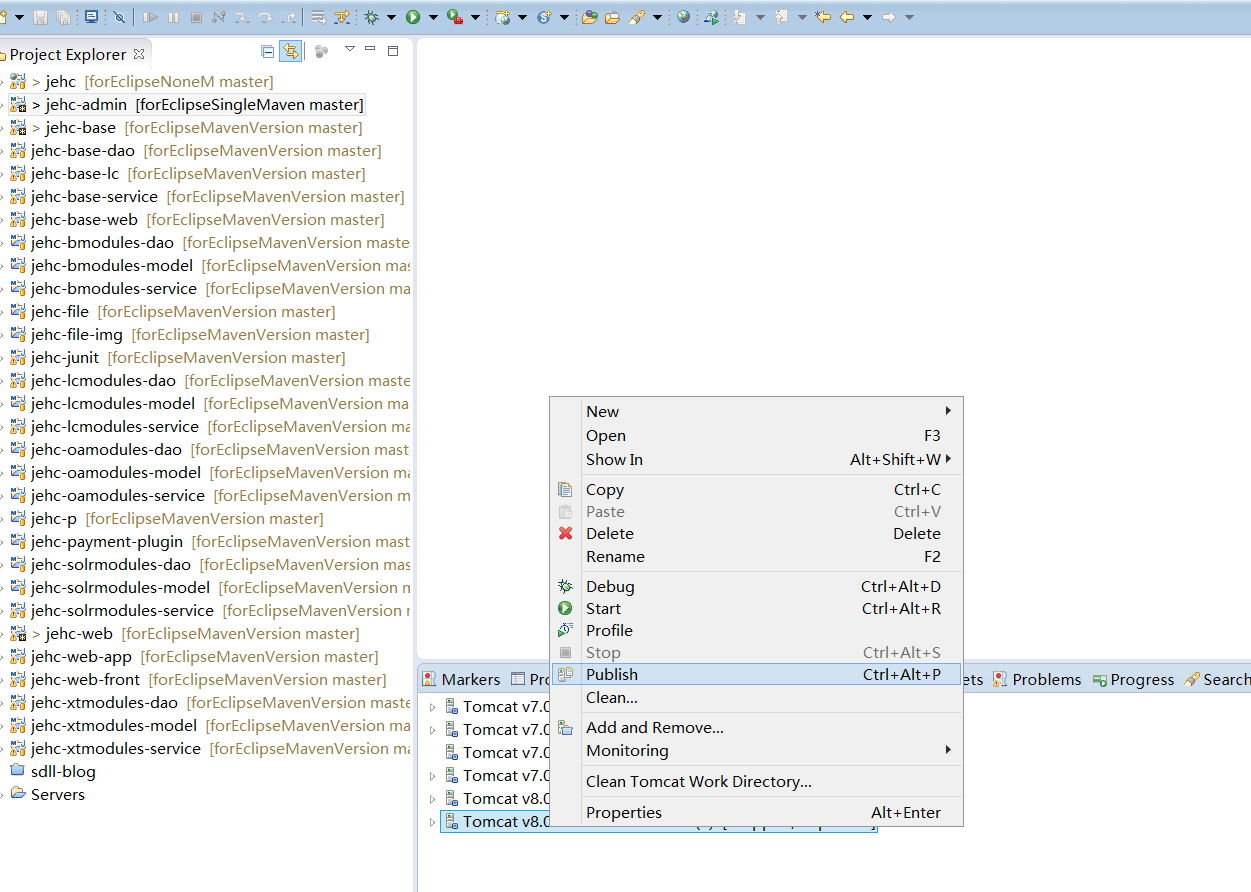


### 部署工程至Tomcat中

1）将jehc-admin部署添加至Tomcat容器中点击finish



2）右键publish操作

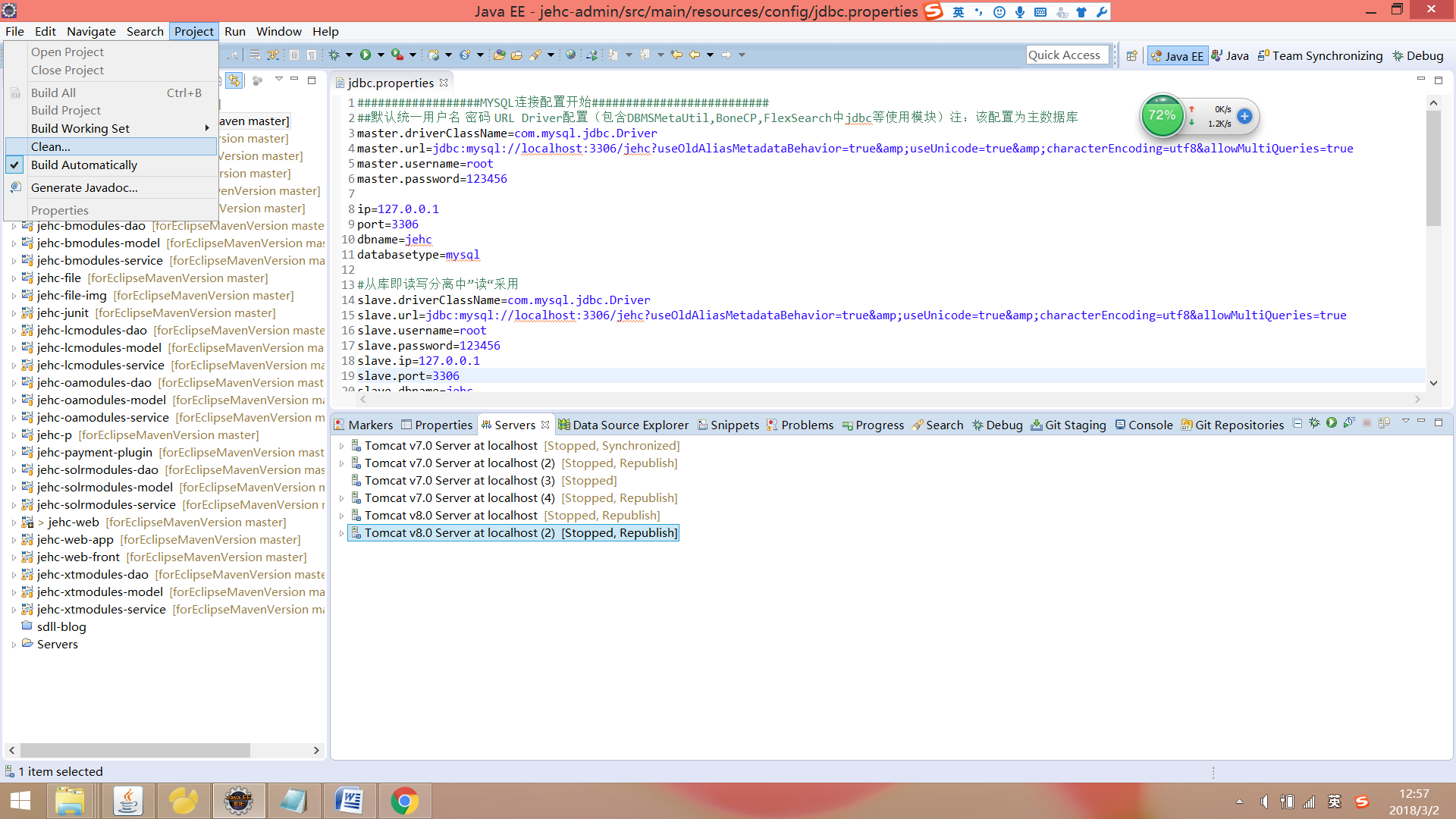


### 启动工程后出现常见异常及解决方案

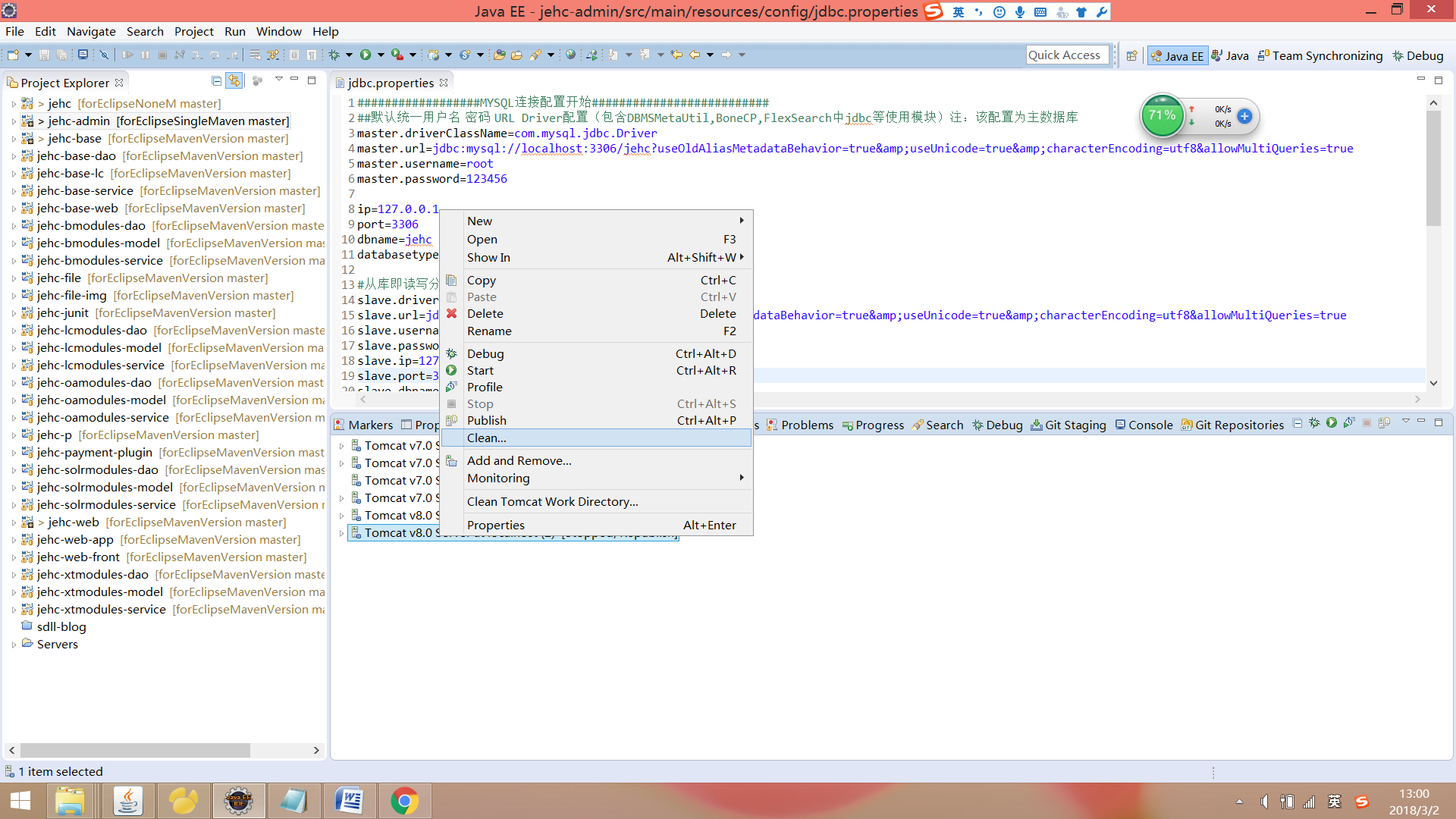
错误信息如：

找不到类（如InitBean,InitSetUp等等）说明出现这种情况 你的工程根本没有编译成功

1）clean工程



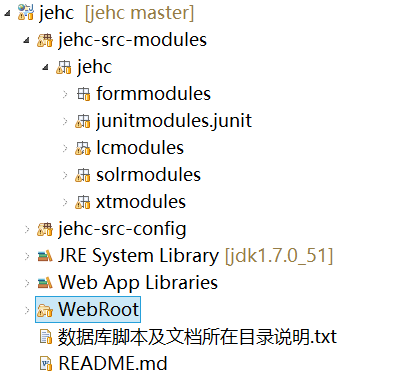
2）clean Tomcat



## 工程目录

### 以Eclipse目录为主如图：

无论maven单工程也好 ，maven多工程依赖也好，还是非maven版本 功能结构功能包名 等都是一致

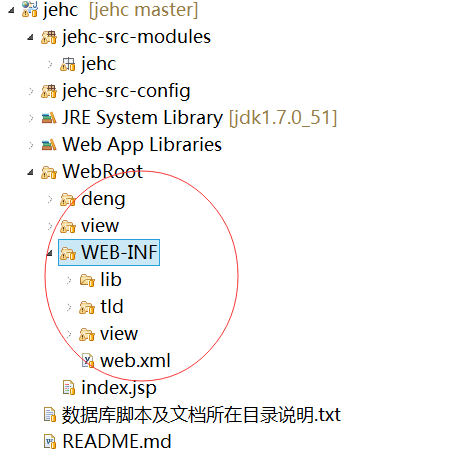


目录包括了xt模块，solr模块，流程模块，junit模块



系统模块中分为core，dao，model，service，web等模块，其中core中为核心工具类，配置文件，代码生成器，动态调度器 可打开进去详阅

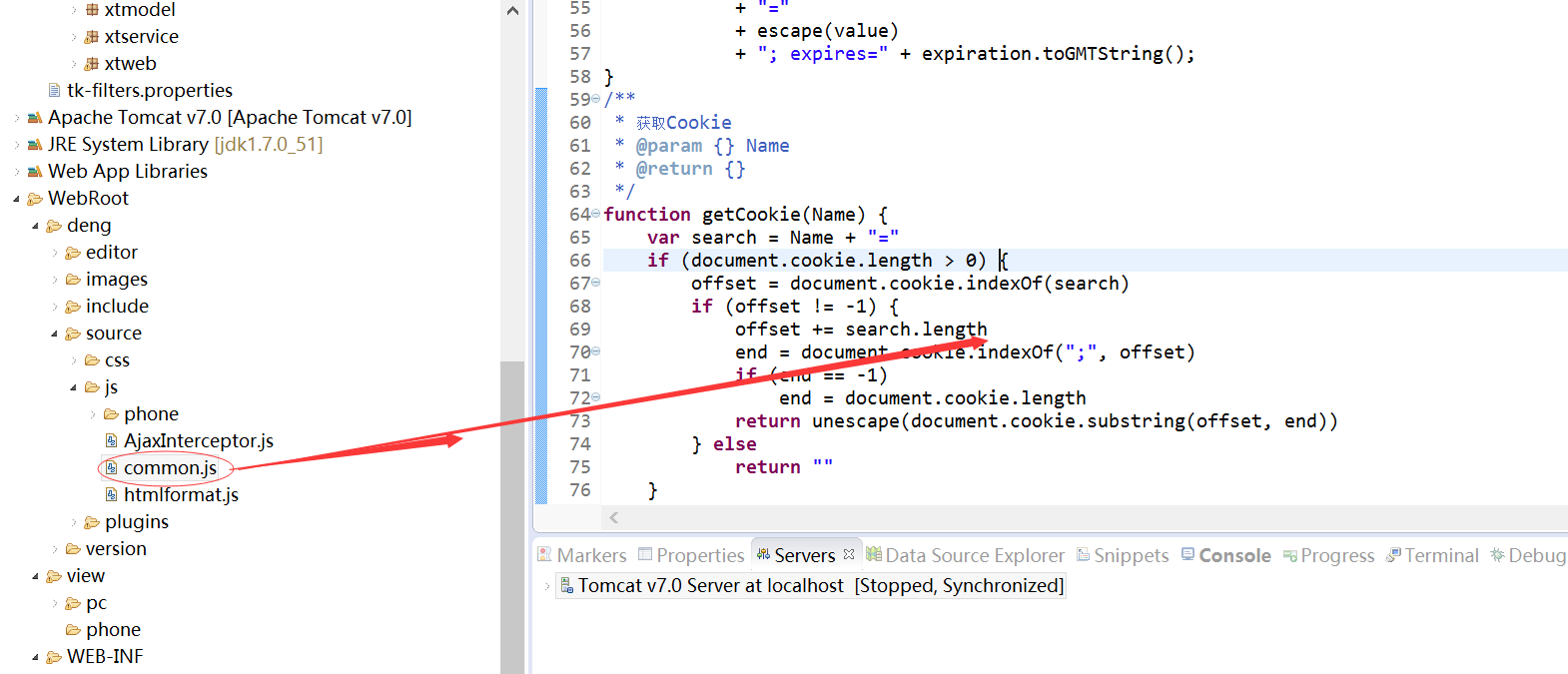
（其他模块风格同上）



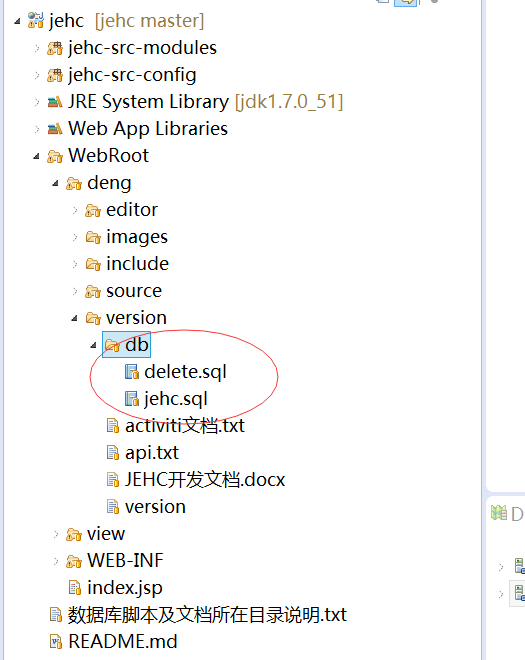
说明：webroot下deng为静态资源，view为相关模块JS

Web-inf下view为相关模块jsp 打开看即可 一目了然

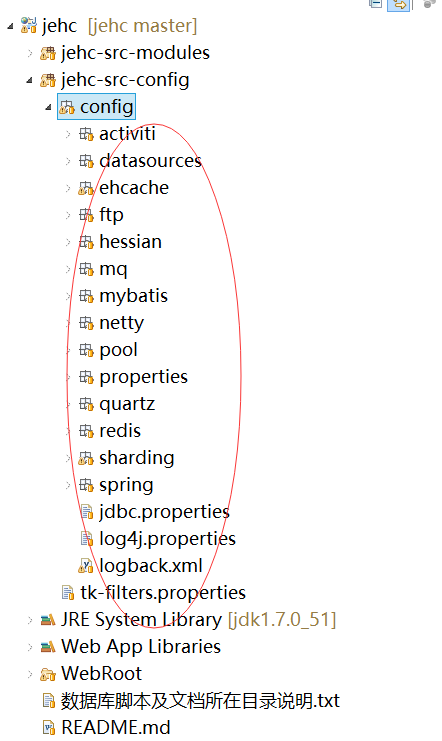
核心JS模块如下：



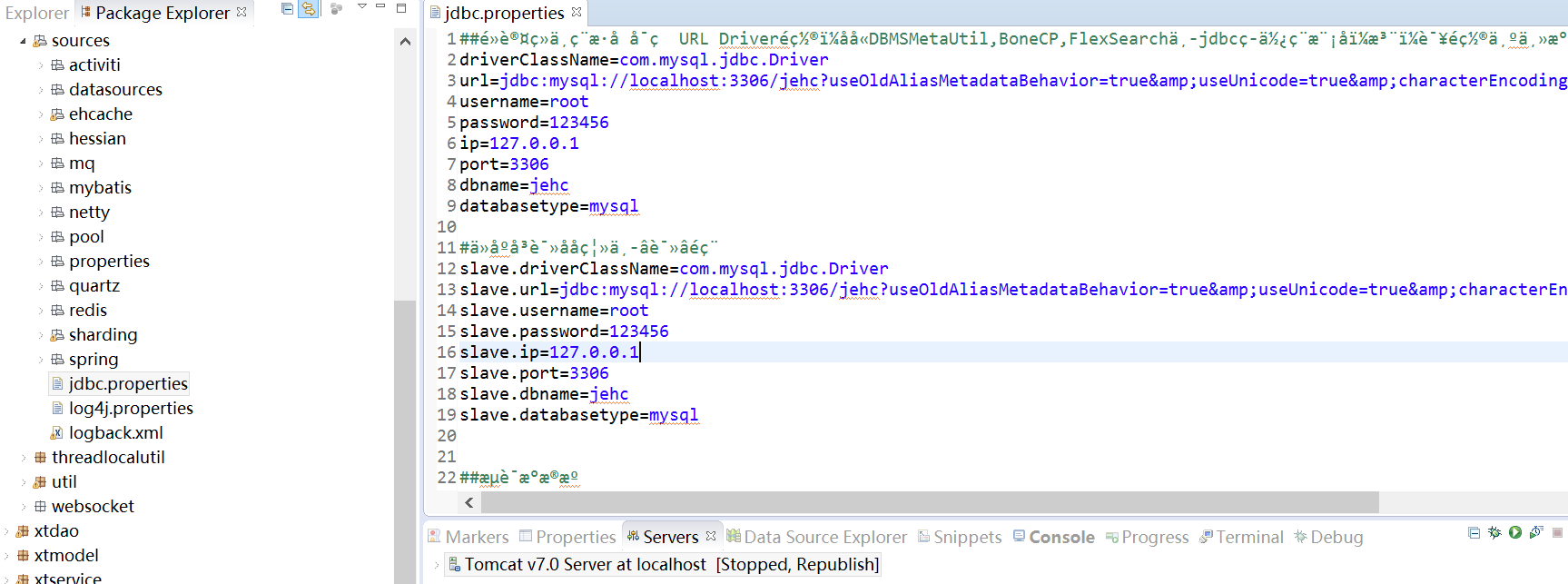
如果你真正上线的时候可以将该js进行压缩或混淆



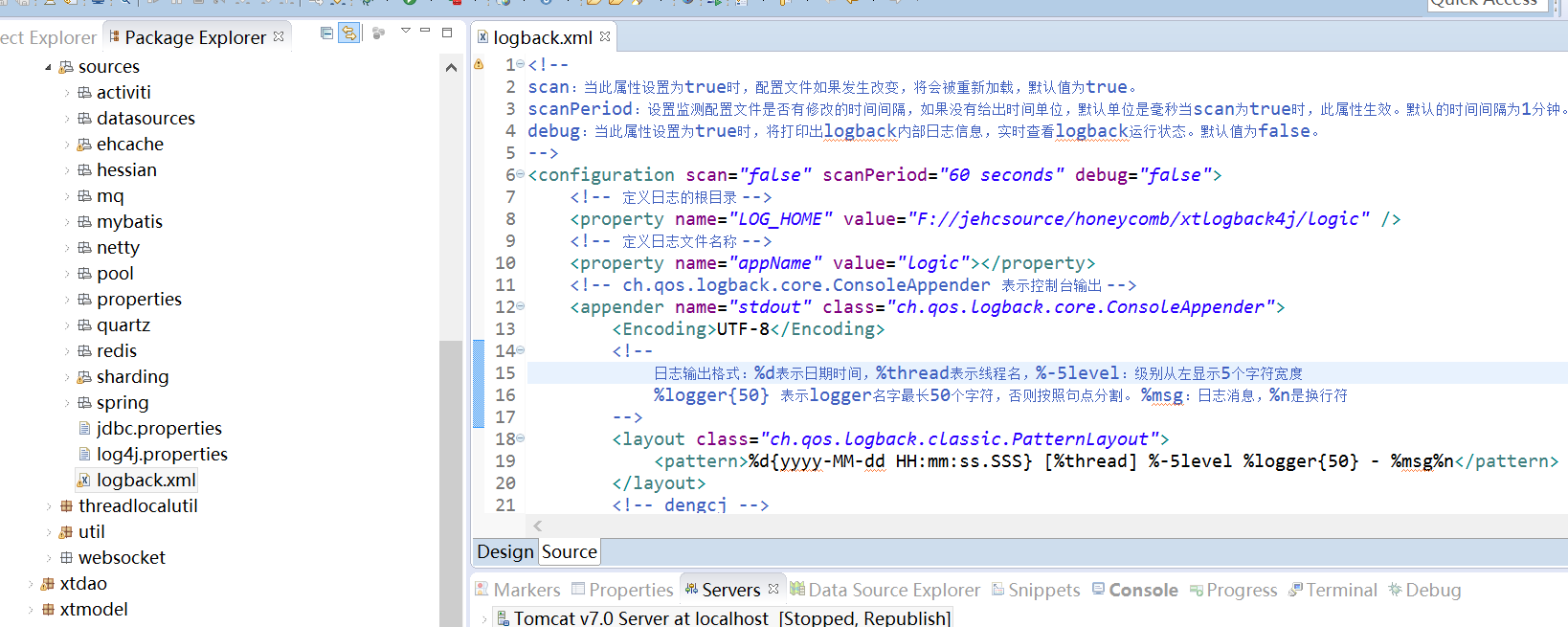
数据库脚本存储目录



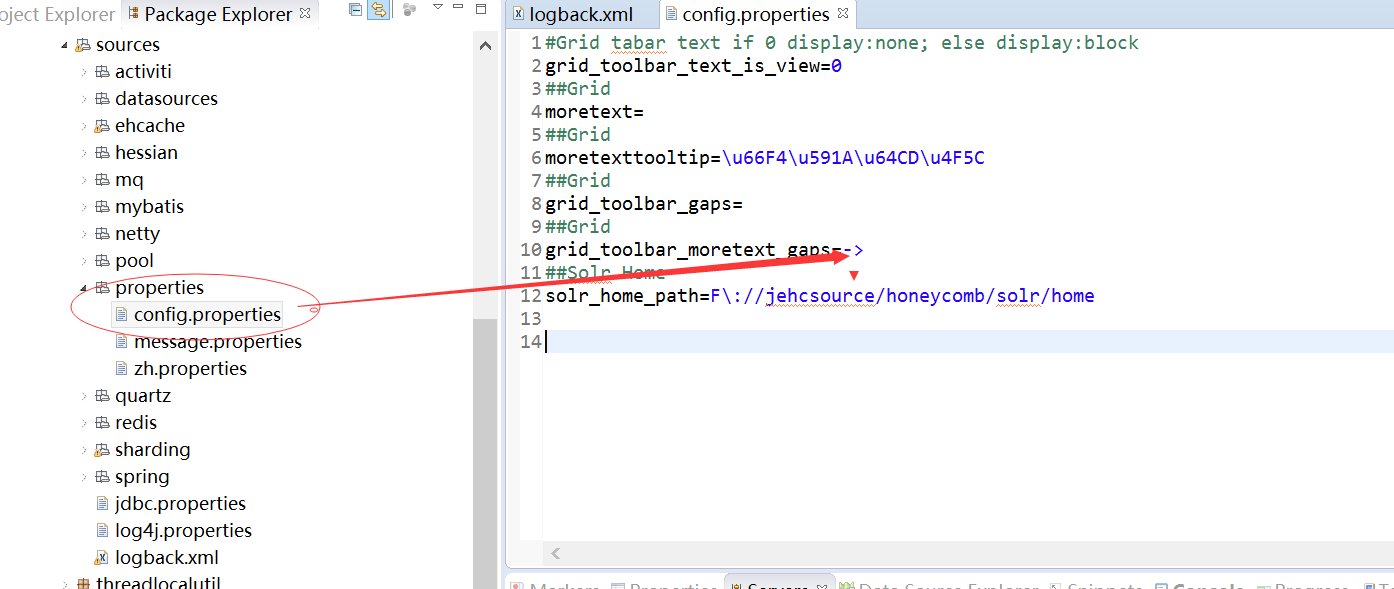
配置文件模块



JDBC连接配置



Logback配置文件



Solr及页面配置文件

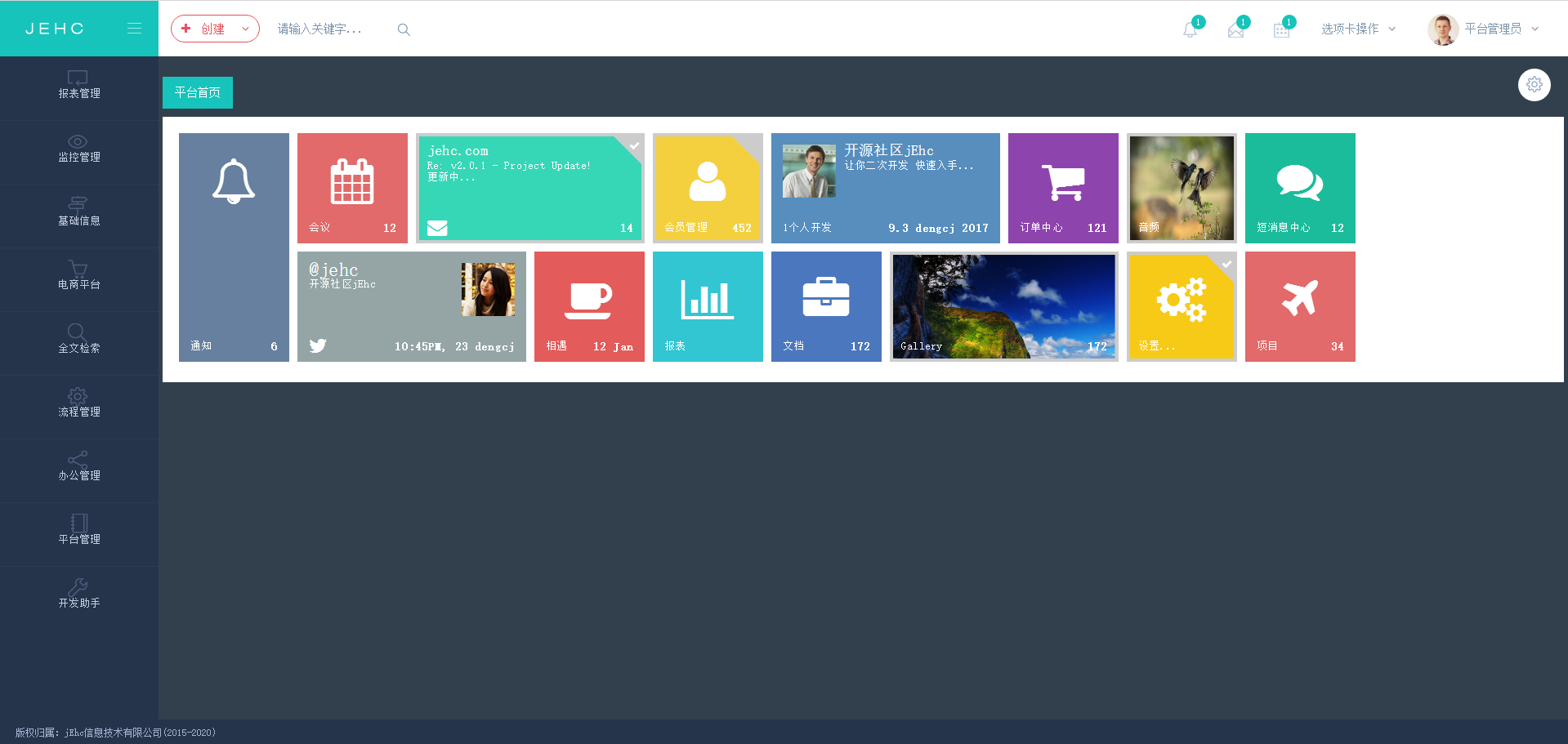
## 功能模块

### 主要模块有

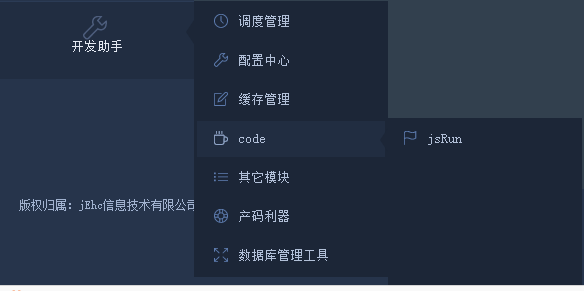
#### 1.登录



#### 2.平台主页面



#### 3.开发助手模块（不再列举）



#### 4.平台管理



#### 5.流程事项



#### 6.代码生成器模块



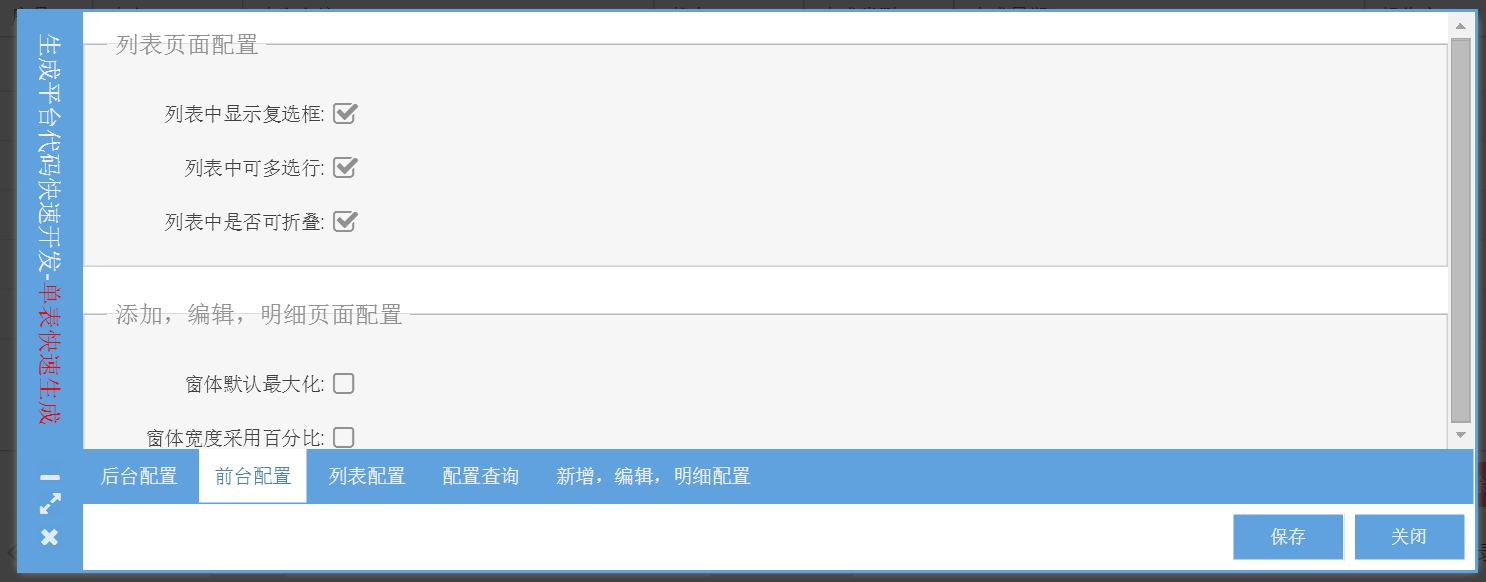
一对多



生成之后例子如：









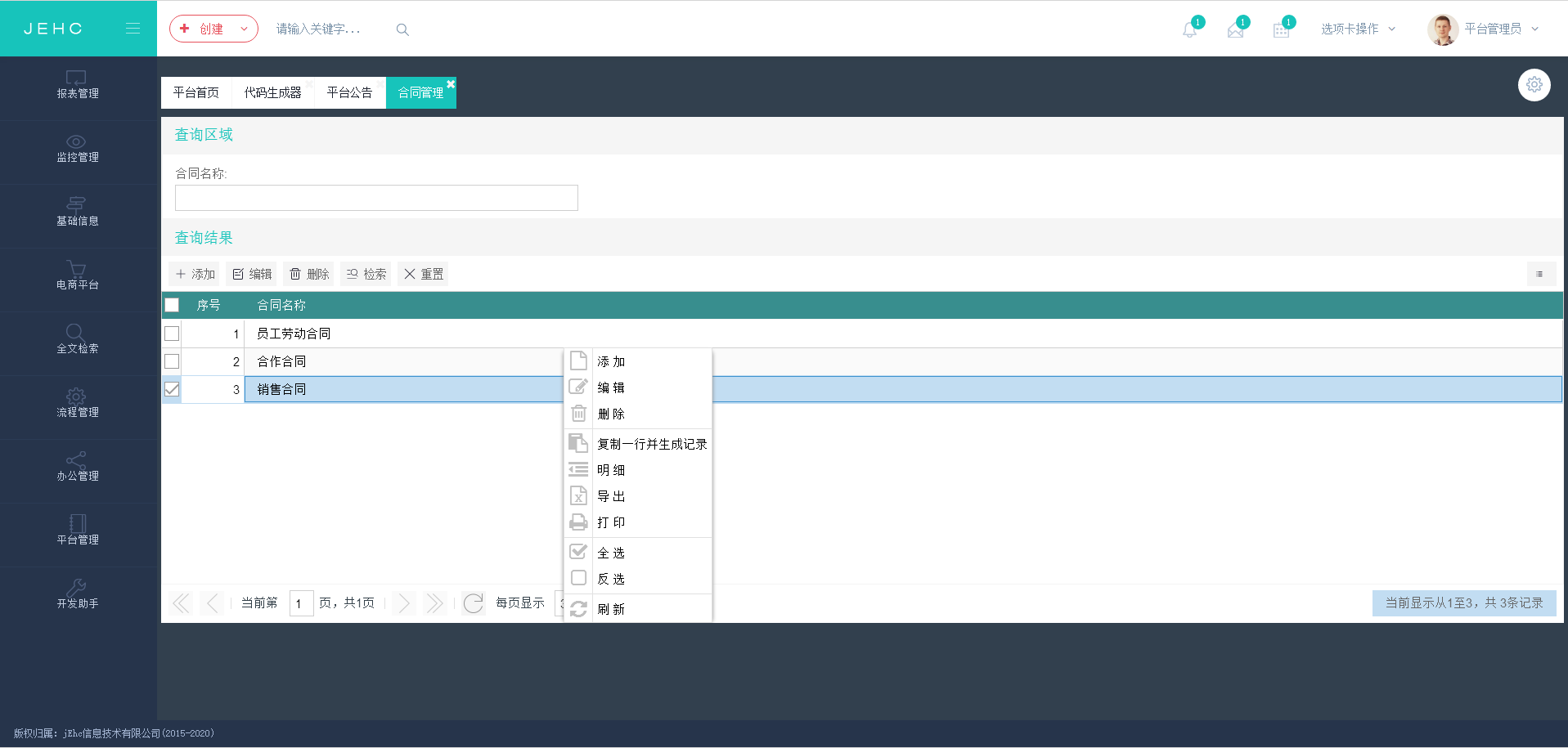




生成之后代码如下：



将这些代码放到工程里重启效果如下：



当然你也可以看看代码风格怎么样是不是很清晰

所生成控件可以是附件 可以是下拉框 可以是数字框，可以是文本域，可以是日期框 同时也可以生成查询信息

一对多同理

注意（如果代码生成器过来的 表一定是存在主键并且一个 并且为UUID类型32位）

## 功能开发说明

### js模块

#### 1.1Extjs方式开发

说明：页面中需要导入include.jsp（相关方法如下：（或查看common.js））

#### (1)gTopValue(id)获取TOP层元素

#### (2)sValue(id,val)赋值元素对象

#### (3)sTopValue(id,val)赋值TOP层元素

#### (4)hiddenTopBtn(id)隐藏TOP层按钮

#### (5)hiddenBtn(id)隐藏层按钮

#### (6)showTopBtn(id)显示TOP层按钮

#### (7)showBtn(id)显示层按钮

#### (8)gfiValue(form,itemId)获取form元素对象(条件根据form和itemId)

#### (9)sfiValue(form,itemId,val)赋值form元素对象(条件根据form和itemId)

#### (10)sfValue(form)通过form中id或name进行赋值

#### (11)sfValue(form,id\_name,val)通过form中id或name进行取值

#### (12)sfReadOnly(form,id\_name,isReadOnly)通过form中id或name进行设置只读或可读

#### (13)addTabContent(url,id,text,icon)将内容加到中间面板中

#### (14)showWaitMsg(msg)显示请求等待进度条窗口

#### (15)hideWaitMsg()隐藏请求等待进度条窗口

#### (16)hideTopWaitMsg()隐藏请求等待进度条窗口

#### (17)loadFormData(forms,url)载入form表单数据

实例:

loadFormData(lcApplyFormDetail,'../lcApplyController/getLcApplyById?lc\_apply\_id='+ record.items[0].data.lc\_apply\_id);

#### (18)loadFormDataCallBack(forms,url,fn)载入form表单数据并回调

实例:

loadFormDataCallBack(xtServiceCenterFormDetail,'../xtServiceCenterController/getXtServiceCenterById?xt\_service\_center\_id=1',callFnDetail);

**function** callFnDetail(form, action){

**var** xt\_Service\_Center\_Parameter = action.result.data.xt\_Service\_Center\_Parameter;

**for**(**var** i = 0; i < xt\_Service\_Center\_Parameter.length; i++){

addDetailXtServiceCenterParameterFormDetail(xt\_Service\_Center\_Parameter[i]);

}

}

#### (19)提交form表单数据

**param** {} form:表单名称,url:地址,grids:列表,win:窗体,closeActionType:true时为hide false时为close,isReset:是否清空表单

submitForm(subForm,url,grids,win,isHide,isReset)

#### (20)提交form表单数据并回调函数

**@param** {} form:表单名称,url:地址,grids:列表,win:窗体,closeActionType:true时为hide false时为close,isReset:是否清空表单,fn:回调事件

submitFormCallFn(subForm,url,grids,win,isHide,isReset,fn)

实例：

submitFormCallFn(xtChangePwdFormAdd,'../xtChangePwdController/addXtChangePwd',**null**,**null**,**false**,**true**,backIndex)

**function** backIndex(form, action){

Ext.Msg.confirm('提示',action.result.msg,**function**(btn){

window.location.href=basePath;

});

}

#### (21)提交form表单数据 并且返回form信息

**@param** {} form:表单名称,url:地址,grids:列表,win:窗体,closeActionType:true时为hide false时为close,isReset:是否清空表单,isCallForm:是否回调加载表单,callUrl:加载表单URL地址,callForm:回调表单名称

submitFormCallBack(subForm,url,grids,win,isHide,isReset,isCallForm,callUrl,callForm)

实例：

submitFormCallBack(xtCompanyFormEdit,'../xtCompanyController/delXtCompany',**null**,**null**,**false**,**true**,**true**,'../xtCompanyController/getXtCompany',xtCompanyFormEdit);

#### (22)提交form表单数据并追加参数

**@param** {} form:表单名称,url:地址,grids:列表,win:窗体,closeActionType:true时为hide false时为close,isReset:是否清空表单,params:追加参数提交

submitFormIncludeParams(subForm,url,grids,win,isHide,isReset,params)

实例：

**var** params = {solrIndexSqlJSON:encodeURI(getSolrIndexSqlJSON()),solrIndexSqlFiledJSON:encodeURI(getSolrIndexSqlFiledJSON())}

submitFormIncludeParams(solrEntityFormAdd,'../solrEntityController/addSolrEntity',grid,solrEntityWinAdd,**false**,**true**,params);

#### (23)提交form表单数据 刷新TreeGrid

**@param** {} form,url,treeGrid,win,closeActionType,isExpandAll:展开TreeGrid

submitForm(subForm,url,grids,win,isHide,isReset,isExpandAll)

#### (24)ajaxRequest(url,grids,params,msg)提交AjaxRequest表单数据

实例：

**var** params = {solr\_core\_id:solr\_core\_id};

ajaxRequest('../solrCoreController/delSolrCore',grid,params,'正在执行删除操作中！请稍后...');

#### (25)ajaxRequestCallFn(url,grids,params,msg,fn)提交AjaxRequest表单数据并回调方法

实例:

**var** params = {xt\_loadinfo\_id:xt\_loadinfo\_id};

ajaxRequestCallFn('../xtLoadinfoController/delXtLoadinfo',grid,params,'正在执行删除操作中！请稍后...',fnRefresh);

**function** fnRefresh(response, opts){

**var** obj=Ext.decode(response.responseText);

xt\_column\_store.reload();

}

#### (26)ajaxReq(url,gridArray,params,msg)提交AjaxRequest表单数据 多grid提交

实例：

**var** params = {xt\_role\_id:xt\_role\_id};

**var** gridArray = [];

gridArray.push(grid);

gridArray.push(grid\_deleted);

ajaxReq('../xtRoleinfoController/recoverXtRoleinfo',gridArray,params,'正在执行恢复操作中！请稍后...');

#### (27)reGetWidthAndHeight()获取高度和宽度

实例:

先调用reGetWidthAndHeight() 然后clientWidth及clientHeight都有值了

#### (28)reGetTopWidthAndHeight同上

#### (29)clearCookie(name)从缓存中清除Cookie

#### (30)getnavigator()获取浏览器版本

#### (31)selectAll(grid)全选grid

#### (32)unSelectAll(grid)反选grid

#### (33)refreshGrid(grid)刷新grid

#### (34)printerGrid(grid)打印grid

#### (35)exportExcel(grid,url)导出grid数据至excel

#### (36)exportExcelByCondition(url,searchForm)根据条件筛选导出

#### (37)resetTitle(panel,isSetBackGround)设置panel标题并判断是否设置背景样式

#### (38)panelPosition上部(north) ,中部(center), 左边(west) ,右部(east) ,底部(south)

items 表单中元素

isTop 表单所属位置

labelPosition标签位置'top'上面 ，'left'左边 ，'right'右边

initSearchForm(panelPosition,items,isTop,labelPosition)

实例:

/\*\*查询区域可扩展\*\*/

**var** items = Ext.create('Ext.FormPanel',{

xtype:'form',

maxHeight:150,

waitMsgTarget:**true**,

defaultType:'textfield',

autoScroll:**true**,

fieldDefaults:{

labelWidth:70,

labelAlign:'left',

flex:1,

margin:'2 5 4 5'

},

items:[

{

fieldLabel:'标题',

xtype:'textfield',

labelWidth:70,

id:'xt\_encoderqrcode\_title',

name:'xt\_encoderqrcode\_title',

anchor:'30%',

labelAlign:'top'

}

]

});

initSearchForm('north',items,**false**,'left');

Ext.create('Ext.Viewport', {

layout:'border',

xtype:'viewport',

items:[searchForm,grid]

});

#### (39)initSearchFormByUserdefined(panelPosition,items,isTop,labelPosition)自定义form查询扩展

#### (40)initSearch(store,url,searchForm)调用查询方法

实例：

/\*\*查询操作\*\*/

**function** search(){

initSearch(store,'../xtEncoderqrcodeController/getXtEncoderqrcodeListByCondition',searchForm);

}

#### (41)overLoad()页面加载完成

#### (42)initDataGrid(grid)与initData(dataStore,value) 重新设置Grid中列值与initData配合使用

实例：

{

header:'性别',

width:50,

dataIndex:'xt\_userinfo\_sex',

renderer:**function**(value){

**return** initData(xtUserinfoSexList,value);

}

}

#### (43)通用上传开始

/\*\*

\*初始化附件右键

\*flag:标识符号1正常（即窗体在frame里面）2最外层显示（即窗体显示在最外层）

\*isUpAndDelete:表示是否拥有上传和删除功能1是2否（即明细页面使用） 如明细不需要上传 和删除功能

\*validateparameter:校验非法参数组装字符串

\*validateSize:校验大小

\*xt\_path\_absolutek:平台路径配置中心键（自定义上传绝对路径使用）

\*xt\_path\_relativek:平台路径配置中心键（自定义上传相对路径使用）

\*xt\_path\_urlk:平台路径配置中心键（自定义上传路径 自定义URL地址）

\*\*/

initFileRight(fieldid,picid,flag,isUpAndDelete,validateparameter,validateSize,xt\_path\_absolutek,xt\_path\_relativek,xt\_path\_urlk)

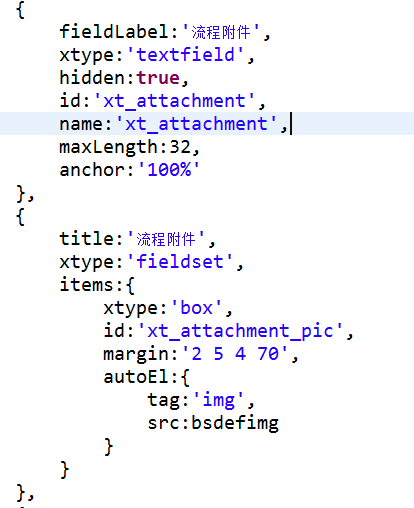
实例：

1. 普通（在新增页面）

/\*\*初始化附件右键菜单开始 参数4为1表示不拥有上传和删除功能 即明细页面使用\*\*/

initFileRight('xt\_attachment','xt\_attachment\_pic',1,2);

/\*\*初始化附件右键菜单结束\*\*/

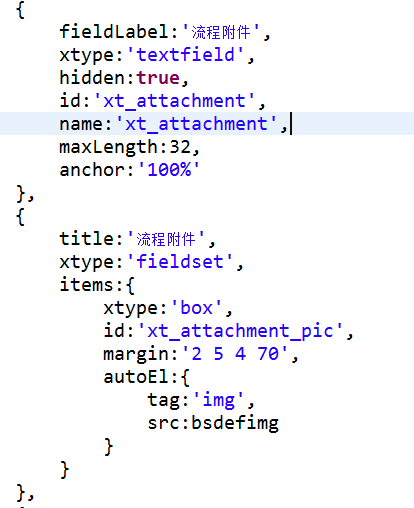


（2）高级自定义（在新增页面）

/\*\*初始化附件右键菜单开始 参数4为1表示拥有上传和删除功能 即新增和编辑页面使用\*\*/

initFileRight('xt\_attachment','xt\_attachment\_pic',1,1,'zip,rar','1000000','ActivitiLc','ActivitiLcRelative');

/\*\*初始化附件右键菜单结束\*\*/



#### (44)ajaxFilePathBackRequest(url,params,flag)表单中单个或多个附件路径回显 @param {} url,params,flag（标识符号1正常（即窗体在frame里面）2最外层显示（即窗体显示在最外层））

实例：

/\*\*配置附件回显方法开始\*\*/

**var** params = {xt\_attachment\_id:record.items[0].data.xt\_attachment,field\_name:'xt\_attachment'};

ajaxFilePathBackRequest('../xtCommonController/getAttachmentPathPP',params,1);

/\*\*配置附件回显方法结束\*\*/

#### (45)初始化右键 在TOP层

/\*\*

\*初始化附件右键

\*isUpAndDelete表示是否拥有上传和删除功能1是2否（即明细页面使用） 如明细不需要上传 和删除功能

\*validateparameter:校验非法参数组装字符串

\*validateSize:校验大小 格式：1024-10240

\*xt\_path\_absolutek:平台路径配置中心键（自定义上传绝对路径使用）

\*xt\_path\_relativek:平台路径配置中心键（自定义上传相对路径使用）

\*xt\_path\_urlk:平台路径配置中心键（自定义上传路径 自定义URL地址）

\*\*/

initTopFileRight(fieldid,picid,flag,isUpAndDelete,validateparameter,validateSize,xt\_path\_absolutek,xt\_path\_relativek,xt\_path\_urlk);

实例：

同上

#### (46)downOrExport(url)通过iFrame实现类ajax文件下载

#### (47)initrightgridcontextmenu(gd,contextmenu,items)初始化右键

实例：

/\*\*调用右键\*\*/

initRight();

**function** initRight(){

**var** contextmenu = **new** Ext.menu.Menu({

id:'theContextMenu',

items:[{

text:'添 加',

glyph:0xf016,

id:'addLcProcessItem',

handler:**function**(){addLcProcess();}

},{

text:'编 辑',

glyph:0xf044,

id:'updateLcProcessItem',

handler:**function**(){updateLcProcess();}

},{

text:'删 除',

glyph:0xf014,

id:'delLcProcessItem',

handler:**function**(){delLcProcess();}

},'-',{

text:'复制一行并生成记录',

glyph:0xf0ea,

id:'copyLcProcessItem',

handler:**function**(){copyLcProcess();}

},{

text:'明 细',

glyph:0xf03b,

id:'detailLcProcessItem',

handler:**function**(){detailLcProcess();}

},{

text:'导 出',

glyph:0xf1c3,

handler:**function**(){

exportLcProcess(grid,'../lcProcessController/exportLcProcess');

}

},{

text:'打 印',

glyph:0xf02f,

handler:**function**(){printerGrid(grid);}

},'-',{

text:'全 选',

glyph:0xf046,

handler:**function**(){selectAll(grid);}

},{

text:'反 选',

glyph:0xf096,

handler:**function**(){unSelectAll(grid);}

},'-',{

text:'刷 新',

glyph:0xf021,

handler:**function**(){refreshGrid(grid);}

}]

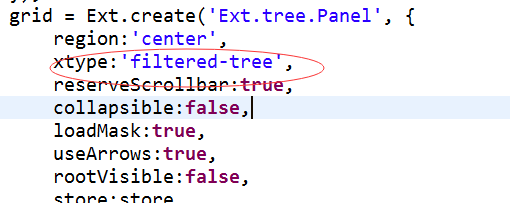
});

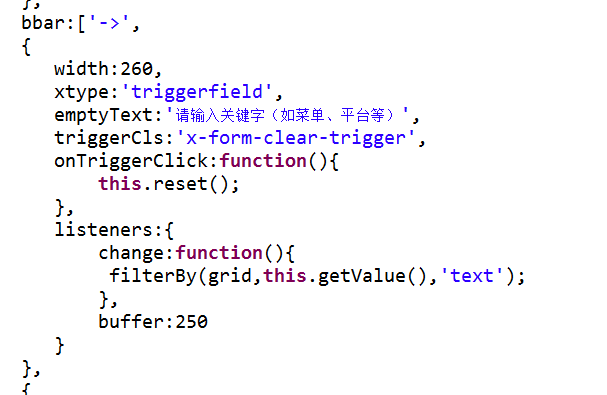
initrightgridcontextmenu(grid,contextmenu,['updateLcProcessItem','delLcProcessItem','copyLcProcessItem','detailLcProcessItem']);

}

#### (48)filterBy(grid,text, by)根据字段筛选

实例:





#### 1.2Bootstrap方式开发

说明：页面需要导入includeboot.jsp即可（用法参考其boot.js）

1.2.1

### 数据权限

#### 2.1mybatis层

<select id=*"getXtNoticeListByCondition"* resultType=*"jehc.xtmodules.xtmodel.Xt\_Notice"* parameterType=*"map"*>

SELECT

xt\_notice.`xt\_notice\_id`,

xt\_notice.`xt\_title`,

xt\_userinfo.`xt\_userinfo\_realName`

FROM

xt\_notice xt\_notice LEFT JOIN xt\_userinfo AS xt\_userinfo ON xt\_notice.xt\_userinfo\_id = xt\_userinfo.xt\_userinfo\_id

WHERE 1=1

<if test=*"null != xt\_userinfo\_id"*>

AND xt\_notice.`xt\_userinfo\_id` IN

<foreach item=*"item"* index=*"index"* collection=*"xt\_userinfo\_id"* open=*"("* separator=*","* close=*")"*>

#{item}

</foreach>

</if>

</select>

描述：xt\_userinfo\_id 采用操作人作为数据权限例子 可自定义 如创建人 修改人等 controller会有传值描述

#### 2.2controller层

@ResponseBody

@RequestMapping(value="/getXtNoticeListByCondition",method={RequestMethod.***POST***,RequestMethod.***GET***})

**public** String getXtNoticeListByCondition(BaseSearch baseSearch,HttpServletRequest request){

Map<String, Object> condition = baseSearch.convert();

commonHPager(condition,request);

dataAuthForXtUID(request,"xt\_userinfo\_id", condition);

List<Xt\_Notice> xt\_NoticeList = xt\_NoticeService.getXtNoticeListByCondition(condition);

PageInfo<Xt\_Notice> page = **new** PageInfo<Xt\_Notice>(xt\_NoticeList);

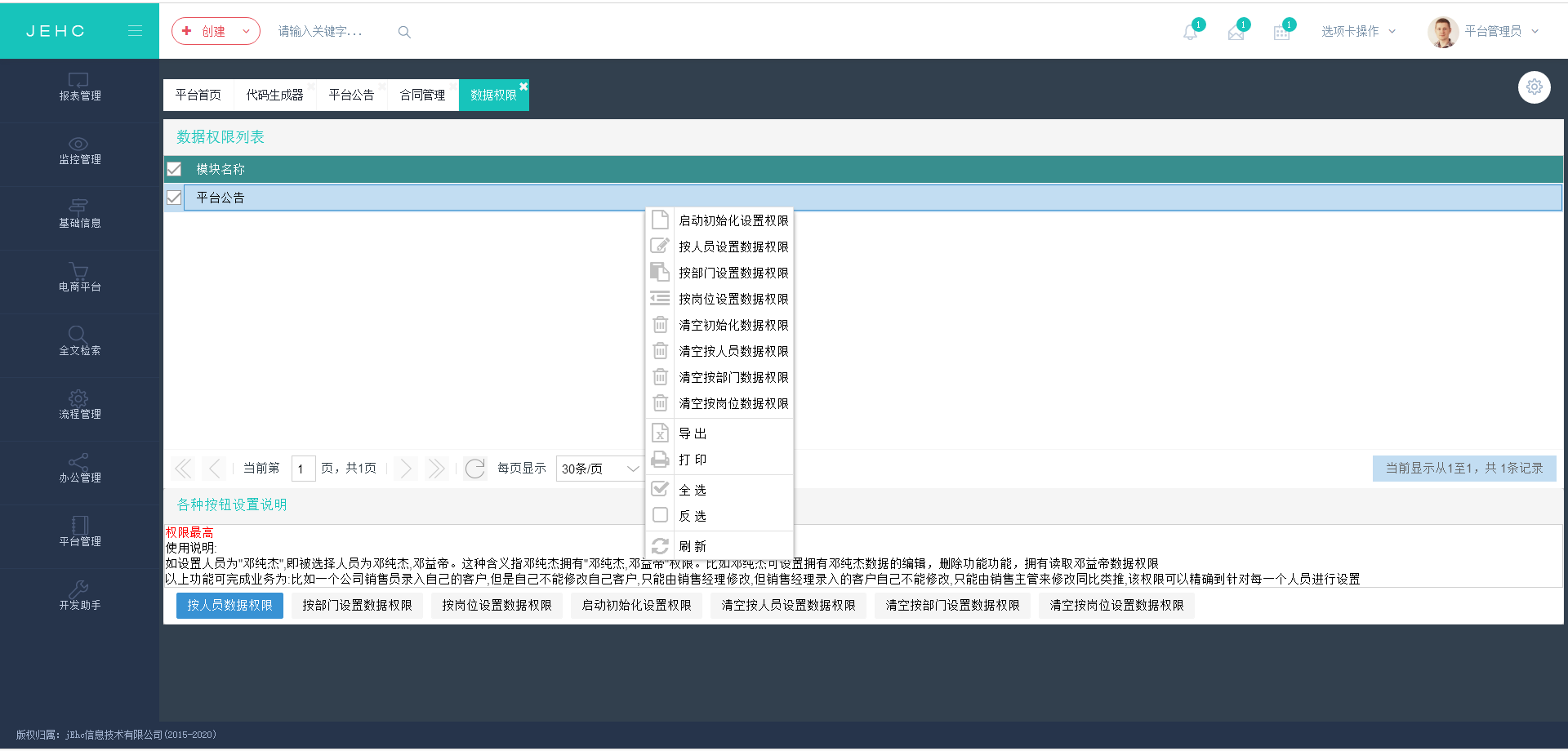
**return** outPageStr(page,request);

}

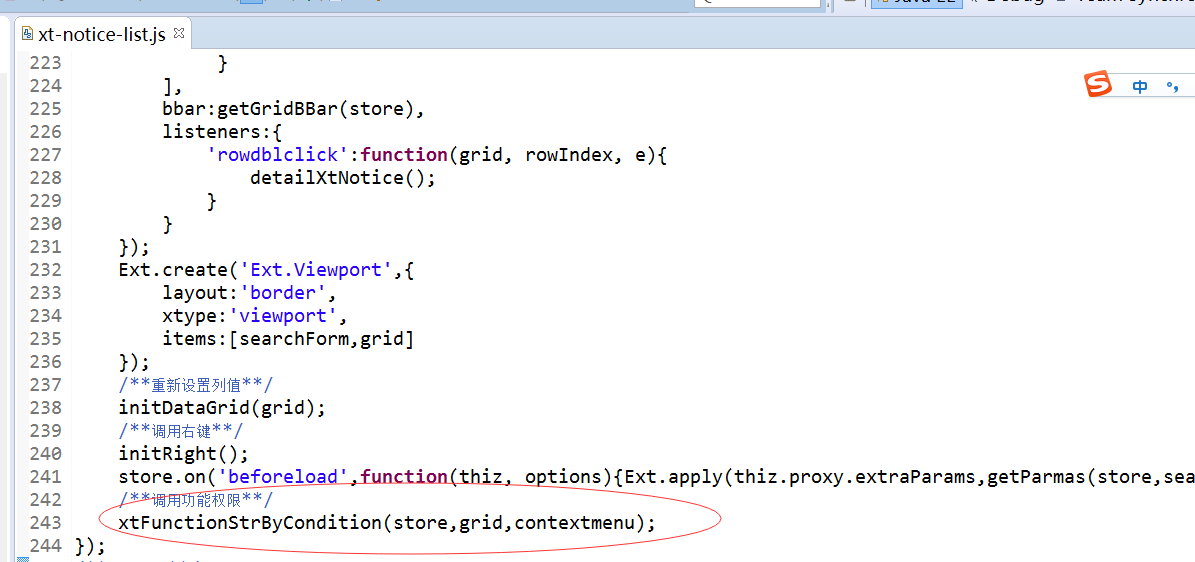
描述：xt\_userinfo\_id作为数据权限传入参数 可以自定义 如createId等

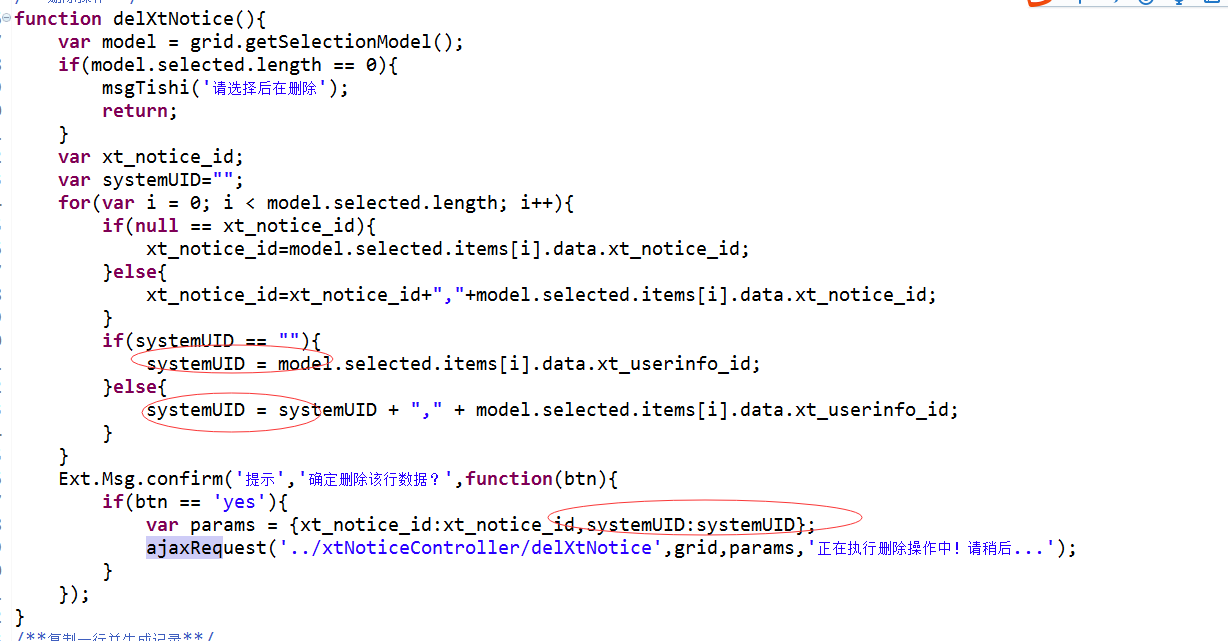
#### 2.3view层

2.3.0配置页面功能

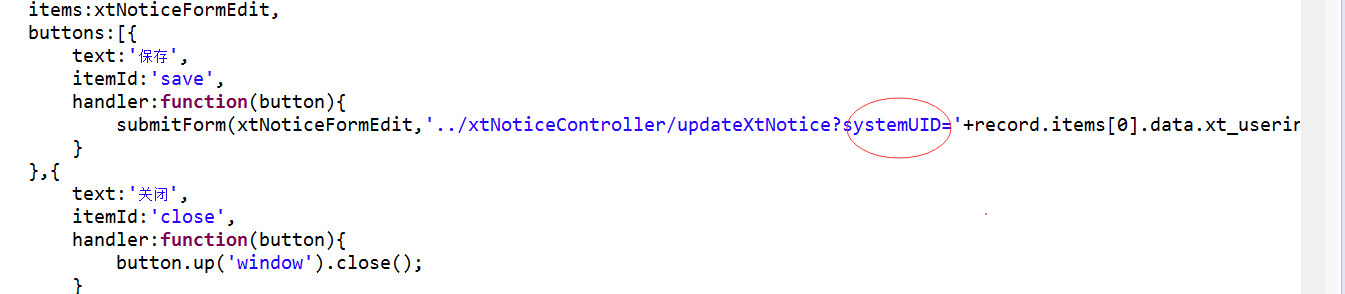


2.3.1list页面层





2.3.2编辑页面

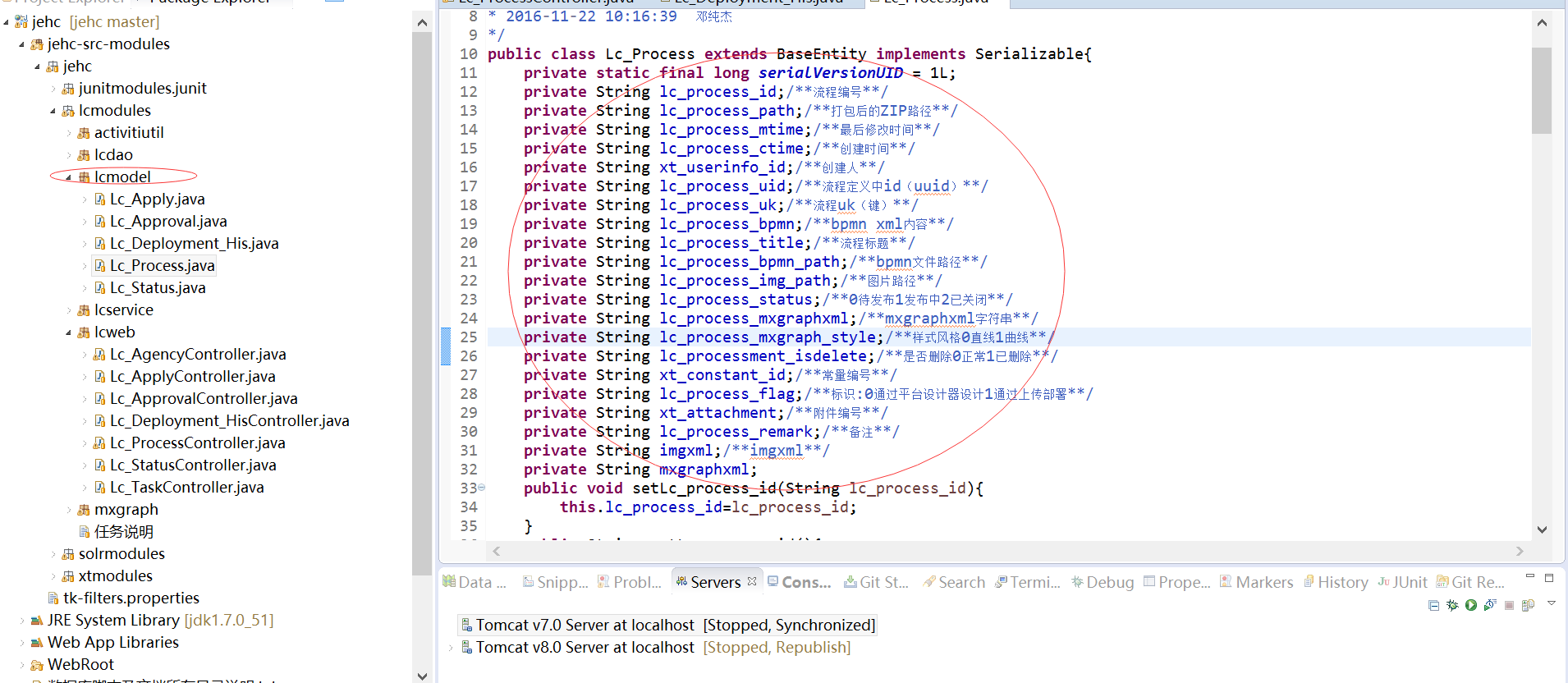


**强烈注意systemUID为数据权限唯一标识**

### 工作流

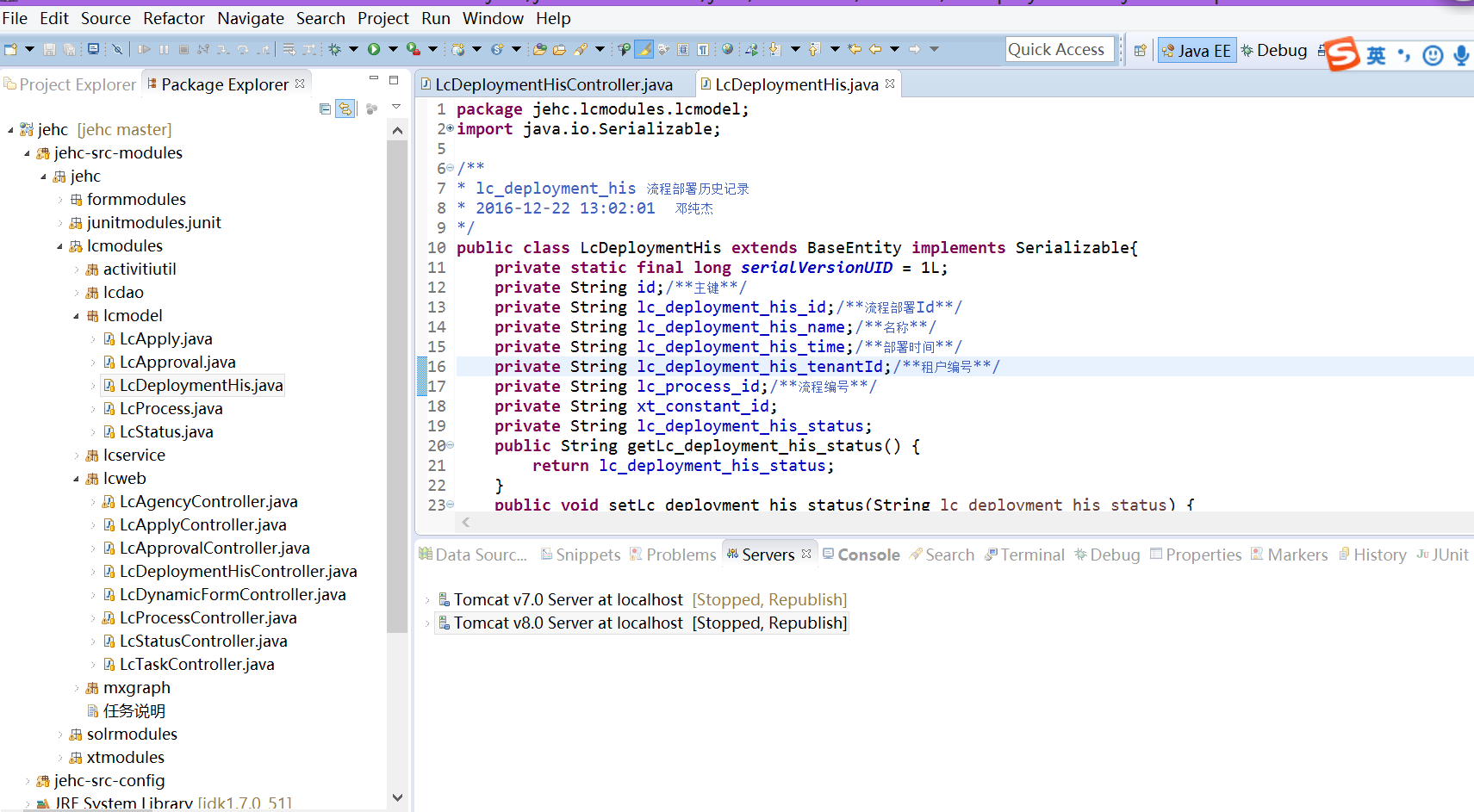
介绍：

#### 业务支撑lc\_process



**说明lc\_process\_uid;流程定义id编号，lc\_process\_uk;流程键 ，lc\_process\_bpmn\_path activiti生成路径 该表保存了 流程定义 用于部署使用 包含生成了bpmn文件，bpmn的png图片 bpmn的zip压缩文件等 很重要**

#### 业务支撑lc\_deployment\_his



**说明：该历史发布版本即存放所有发布过的流程版本即 指定流程 指定发布历史记录 用于 每次业务中发起实例 取最新部署**

#### 相关方法

类型一平台自身提供相关主要方法

所在类Lc\_ProcessController

1.createBPMN(Lc\_Process lc\_Process,MxGraphModel mxGraphModel,HttpServletRequest request,HttpServletResponse response)通过在线设计生成流程信息

2.generateBpmnAndImg(String url,String imgxml,String bpmn,String processName,String w,String h,HttpServletResponse response,Lc\_Process lc\_Process)生成bpmn,图片两个文件及压缩文件

3.createDeployment(String lc\_process\_id,HttpServletRequest request,HttpServletResponse response)部署流程

4.startProcessInstance(String lc\_deployment\_his\_id)发起流程实例

5.suspendProcessInstanceById(String id)挂起流程实例（可作为撤销流程）

6.activateProcessInstanceById(String id)激活流程实例(开启流程实例)

7.completeTask(String taskId)完成任务

8.downFile(String lc\_process\_id,HttpServletRequest request,HttpServletResponse response)下载Bpmn文件

9.downFileImg(String lc\_process\_id,HttpServletRequest request,HttpServletResponse response)下载Img文件

activiti支撑的方法如下：

工具类：activitiUtil

1. createDeployment(File file)部署流程
2. deleteDeploymentById(String deploymentId)删除流程部署
3. getProcessDefinition(String deploymentId)获取流程定义对象
4. getProcessDefinitionByDid(String processDefinitionId)根据流程定义id查询流程定义
5. getProcessDefinitionList()获取所有流程定义集合
6. startProcessInstanceByKey(String key,String businessKey,Map<String, Object> variables)根据流程定义的key启动工作流实例

/\*\*

\* 设置跨节点提交

\* 根据流程定义的key启动工作流实例

\* 发起一个流程实例

\* **@param** key流程部署

\* **@param** businessKey业务键

\* **@param** initialActivityId [设置跨节点提交]

\* **@param** variables参数

\* **@return**

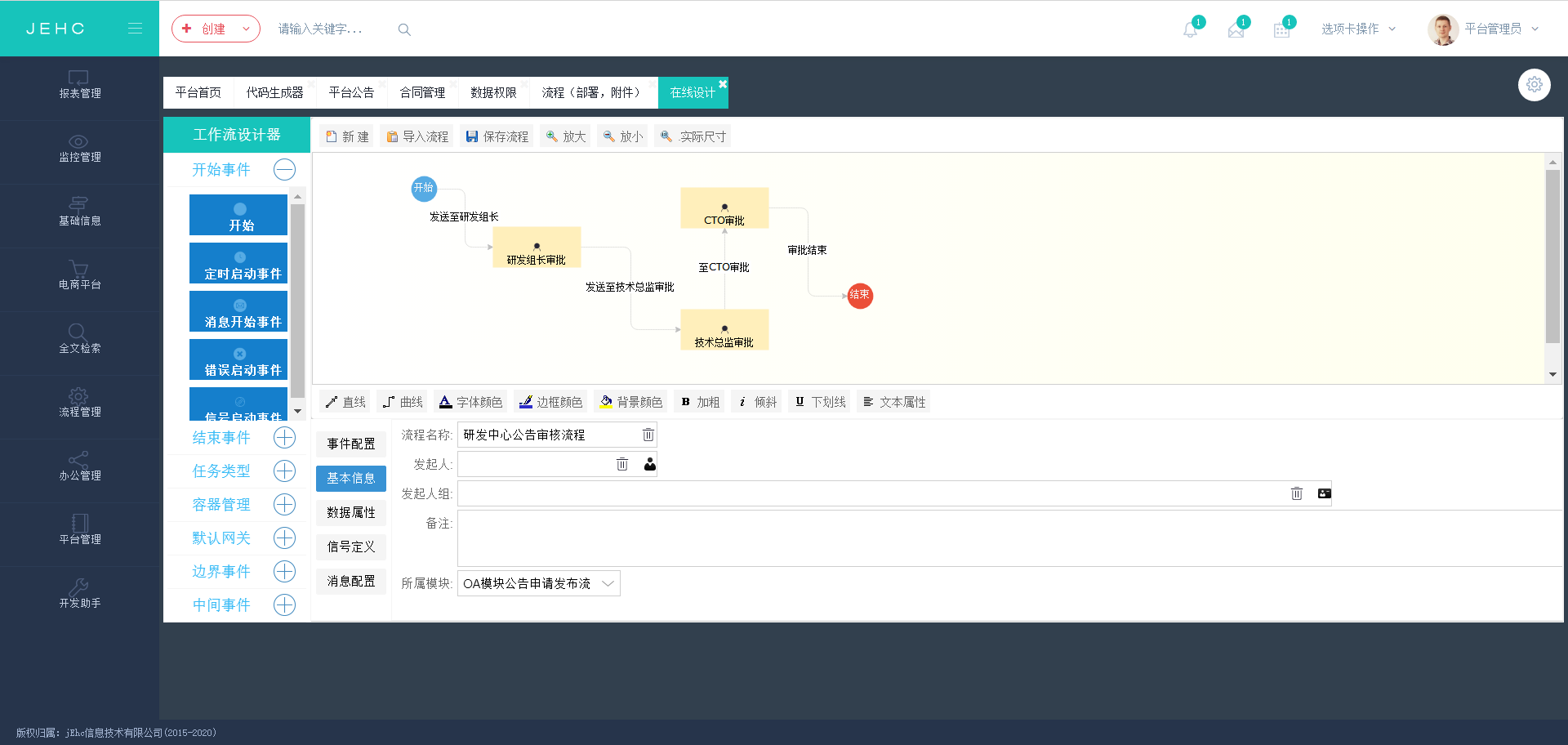
\*/

1. startProcessInstanceByKey(String key,String businessKey,String initialActivityId,Map<String, Object> variables)
2. getProcessInstanceById(String processInstanceId)根据流程实例获取流程对象
3. getProcessInstanceImageById(String processInstanceId,HttpServletResponse response)根据流程实例查找流程图
4. getProcessInstanceImageById(String processInstanceId,HttpServletResponse response)根据流程实例查找流程图
5. getActivityImageInfo(String executionId,String processInstanceId)获取流程图信息
6. getTask(String taskId) 根据任务ID获取activityId,businessKey[业务Key],variables[节点变量]
7. completeTask(String taskId,Variable variable)完成任务
8. claimTask(String taskId,String UID)根据用户ID+taskId签收任务
9. validatePEnd(String processInstanceID)判断流程实例是否结束
10. suspendProcessInstanceById(String processInstanceID)挂起流程实例（可作为撤销流程）
11. activateProcessInstanceById(String processInstanceID)激活流程实例(开启流程实例)
12. getActivityImpl(String taskID)根据任务ID获取ActivityImpl
13. nextTaskDefinition(String processInstanceID,String elString) 根据实例编号查找下一个任务节点
14. getNextNode(String procInstanceId)获取当前流程的下一个节点
15. getNextTaskDefinition(String taskID,String elString)根据任务ID查找下一步节点
16. nextTaskDefinition(ActivityImpl activityImpl,String activityId, String elString)下一个任务节点
17. getNextTaskDefinition(String processDefinitionId,String activityId,String elString,**boolean** flag)根据与流程定义processDefinitionId获取当前节点的下一个任务节点
18. getNextTaskDefinition(String taskID)获取下一步相关内容
19. getNextTaskDefinition(TaskDefinition taskDefinition)API通过下个节点 获取节点中各个属性
20. getProcessInstaceList()查询所有实例
21. getProcessInstaceList(String processDefinitionKey)根据流程定义Key查找所有实例
22. getProcessMap(ProcessInstance pi,ProcessDefinition processDefinition)获取活动任务
23. getProcessInstanceImg(String processInstanceId)根据流程实例获取流程图并高亮
24. getProcessImgEd(String deploymentId,String name)根据流程定义获取流程图
25. addApply(String id,String businessKey,Map<String, Object> variables,Lc\_Apply lc\_Apply)添加审批（发起申请）
26. addApplySetAssignee(String id,String businessKey,Map<String, Object> variables,Lc\_Apply lc\_Apply)添加审批（发起申请）并设置第一个节点发起人并完成第一个节点
27. claim(String taskId,String userId)将组任务指定个人任务（该方法：检查该任务是否已经被认领，如果被认领则会抛出ActivitiTaskAlreadyClaimedException）
28. setAssignee(String taskId,String userId)将个人任务再回退到组任务（前提：之前这个任务是组任务）
29. setOwner(String taskId,String userId)设置任务的归属者
30. addGroupUser(String taskId,String userId)向组任务中添加成员
31. deleteGroupUser(String taskId,String userId)向组任务中删除成员
32. getAssigneeTaskPageList(Map<String, Object> condition)查找个人任务
33. getAssigneeTaskList(Map<String, Object> condition)查找个人任务(不分页)
34. getCandidateTaskPageList(Map<String, Object> condition)查找候选人任务
35. getCandidateGroupTaskPageList(Map<String, Object> condition)查找处理组任务
36. completeTask(String taskId)完成任务
37. isEnded(String processInstanceId)判断流程实例是否已经结束
38. getTaskList(Map<String, Object> condition)获取所有任务
39. getTaskListByInstanceId(Map<String, Object> condition)获取实例下所有任务
40. jump(String executionId,String activityId)调用跳转节点
41. jointProcess(String taskId, List<String> userCodes)会签操作
42. backProcess(String taskId, String activityId,Map<String, Object> variables)驳回流程
43. callBackProcess(String taskId, String activityId)
44. clearTransition(ActivityImpl activityImpl)清空指定活动节点流向
45. commitProcess(String taskId, Map<String, Object> variables,String activityId)提交流程
46. endProcess(String taskId)结束流程
47. filterNewestActivity(ProcessInstance processInstance,List<ActivityImpl> tempList)根据流入任务集合，查询最近一次的流入任务节点
48. findActivitiImpl(String taskId, String activityId)根据任务ID和节点ID获取活动节点
49. findBackAvtivity(String taskId)查询指定任务节点的最新记录
50. findHistoricUserTask(ProcessInstance processInstance, String activityId)
51. findParallelGatewayId(ActivityImpl activityImpl)根据当前节点，查询输出流向是否为并行终点，如果为并行终点，则拼装对应的并行起点ID
52. findProcessDefinitionEntityByTaskId(String taskId)根据任务ID获取流程定义
53. findProcessInstanceByTaskId(String taskId)根据任务ID获取对应的流程实例
54. TaskEntity findTaskById(String taskId) 根据任务ID获得任务实例
55. findTaskListByKey(String processInstanceId, String key) 根据流程实例ID和任务key值查询所有同级任务集合
56. restoreTransition(ActivityImpl activityImpl,List<PvmTransition> oriPvmTransitionList)还原指定活动节点流向
57. reverList(List<ActivityImpl> list) 反向排序list集合，便于驳回节点按顺序显示
58. transferAssignee(String taskId, String userCode)转办流程
59. turnTransition(String taskId, String activityId,Map<String, Object> variables)流程转向操作
60. getImageStream(String taskId)根据当前任务获取图片
61. getUserInfo(String userid)获取用户详细信息
62. getUserOfGroup(String userid)根据用户id查询用户所在的组
63. getGroupInfo(String groupId)根据groupId查询组详细信息
64. memberOfGroup(String groupId)列出组内的所有用户
65. getTaskByProcessInstanceID(String processInstanceID)根据流程实例编号查找当前任务

说明紫色背景为不常用

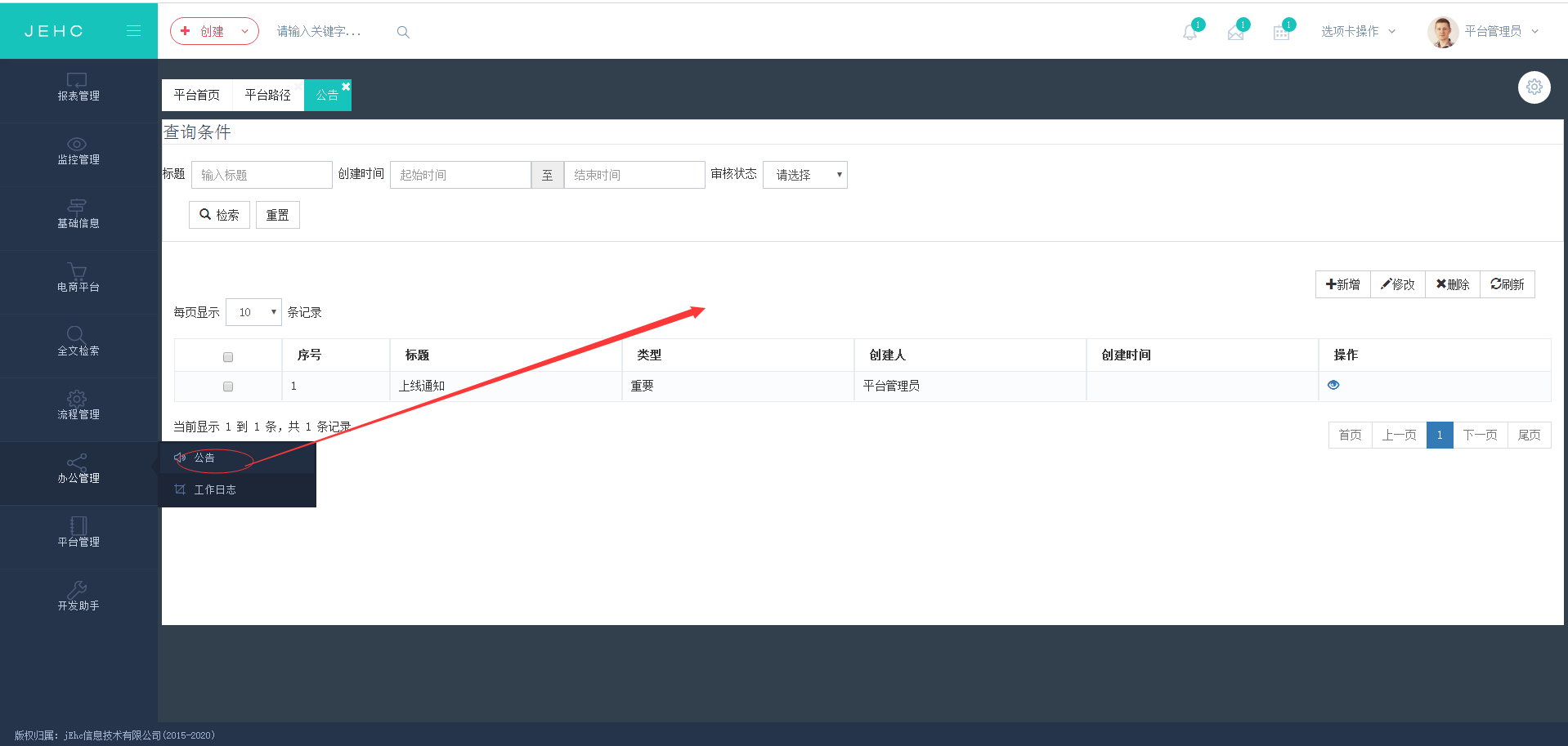
#### 流程设计器





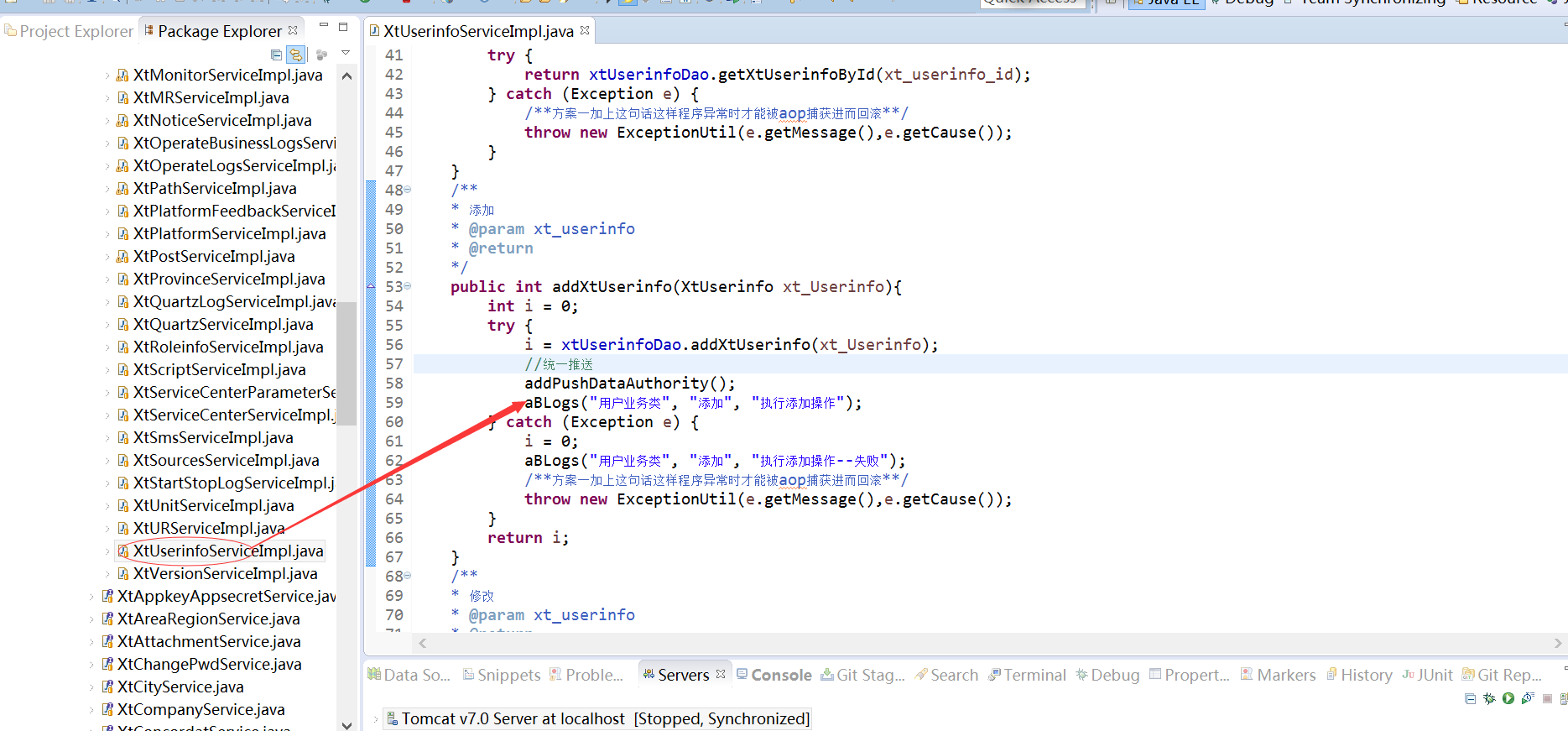
#### 流程demo

已OA模块为实例：



### 业务操作日志使用

Service层调用aBLogs方法如下图所示：



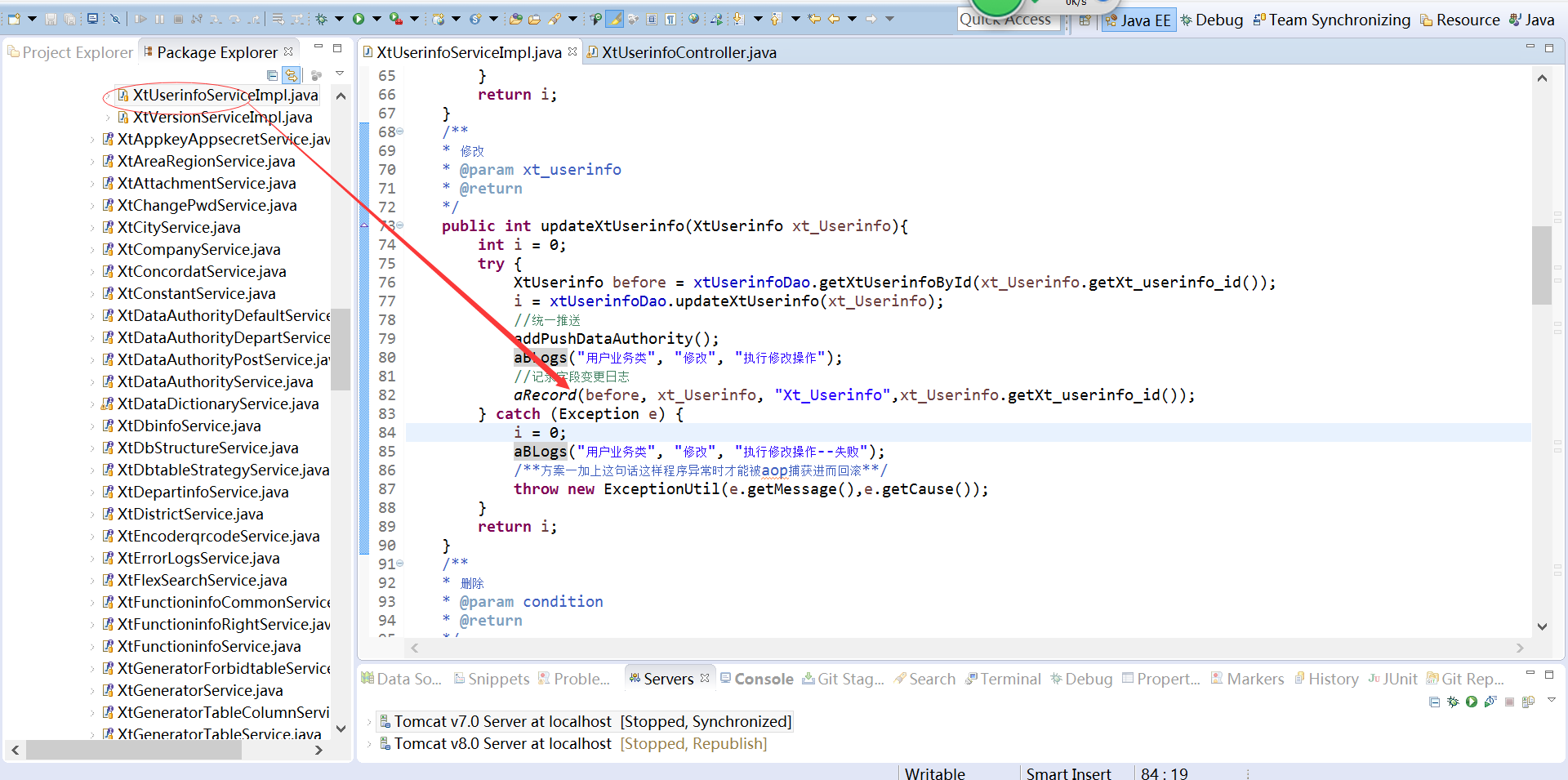
Controller层调用aBLogs方法如下图所示：



以上两种方法调用皆可以完成操作日志

### 字段修改前修改后调用统一方法

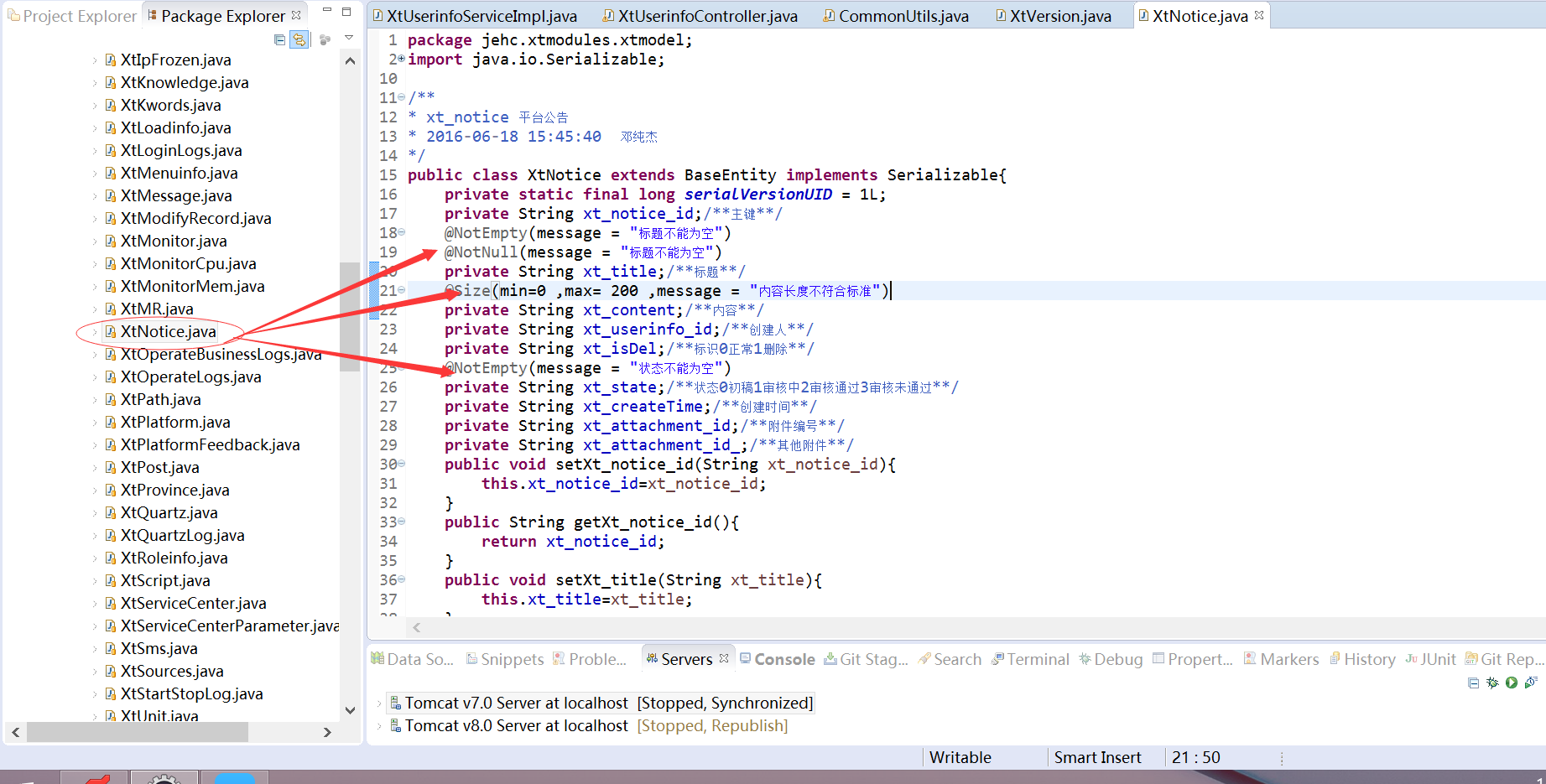
Service层之间调用aRecord方法如下图所示：



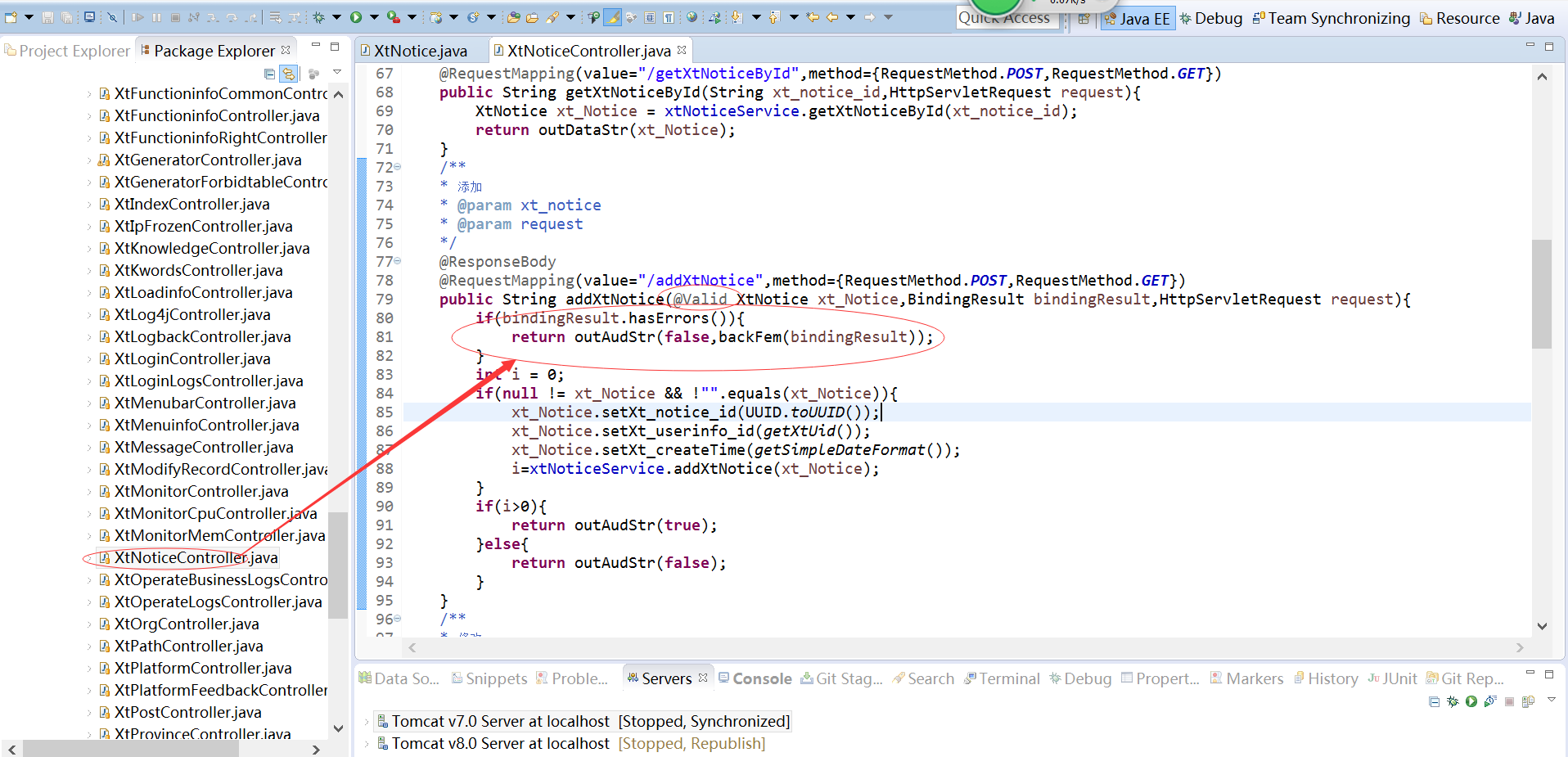
说明：before指操作前对象，参数二指当前对象，参数三指模块即实体类，参数四业务编号，如果需要记录指定字段 则追加参数五即可，参数类型为list<String>

### 通过实体类字段注解验证参数

model层调用如下



Controller调用如下



### 全文检索

#### 介绍：采用solr4.10版本

## 交流及其他说明：

由于空余时间不多，只能写一部分开发文档模块 主要还是交流为目的

本平台采用开源方式，交流QQ群：330370132

项目地址：http://git.oschina.net/jehc/jehc