创建lib目录

复制所有的Jar，后期复制项目配置所需要的jar包。

Model类

有的NameQuery引号丢丢失了一个额

@Table(catalog = "test", name = "book)

类名没有大写

多对多的映射

**丢失的**

**import** javax.persistence.Column;

DAO接口

**import** org.skyway.spring.util.dao.JpaDao;

Question:

已经存在的类可以import，生成的类不能被import

# 201708078

Question 1:

.project文档生成后不能正常起作用

.classpath文档同样的额情况

不能作为web项目运行

解决过程：缺乏配置文件，利用项目的facet解决，

Org.eclipse.jdt.core.prefs

eclipse.preferences.version=1

org.eclipse.jdt.core.compiler.codegen.inlineJsrBytecode=enabled

org.eclipse.jdt.core.compiler.codegen.targetPlatform=1.7

org.eclipse.jdt.core.compiler.compliance=1.7

org.eclipse.jdt.core.compiler.problem.assertIdentifier=error

org.eclipse.jdt.core.compiler.problem.enumIdentifier=error

org.eclipse.jdt.core.compiler.source=1.7

Org.eclipse.wst.common.component

<?xml version=*"1.0"* encoding=*"UTF-8"*?><project-modules id=*"moduleCoreId"* project-version=*"1.5.0"*>

<wb-module deploy-name=*"test"*>

<wb-resource deploy-path=*"/"* source-path=*"/WebRoot"* tag=*"defaultRootSource"*/>

<wb-resource deploy-path=*"/WEB-INF/classes"* source-path=*"/src"*/>

<wb-resource deploy-path=*"/WEB-INF/classes"* source-path=*"/resources"*/>

<wb-resource deploy-path=*"/WEB-INF/classes"* source-path=*"/generated"*/>

<property name=*"context-root"* value=*"test"*/>

<property name=*"java-output-path"* value=*"/test/bin"*/>

</wb-module>

</project-modules>

Org.eclpse.wst.common.project.facet.core.xml

<?xml version=*"1.0"* encoding=*"UTF-8"*?>

<faceted-project>

<fixed facet=*"wst.jsdt.web"*/>

<installed facet=*"java"* version=*"1.7"*/>

<installed facet=*"jst.web"* version=*"3.1"*/>

<installed facet=*"wst.jsdt.web"* version=*"1.0"*/>

</faceted-project>

classpath多了2个类路径

<?xml version="1.0" encoding="UTF-8"?>

<classpath>

<classpathentry kind="src" path="src"/>

<classpathentry kind="src" path="resources"/>

<classpathentry kind="src" path="generated"/>

<classpathentry kind="con" path="org.eclipse.jdt.launching.JRE\_CONTAINER/org.eclipse.jdt.internal.debug.ui.launcher.StandardVMType/JavaSE-1.7"/>

<classpathentry kind="con" path="org.eclipse.jst.j2ee.internal.web.container"/>

<classpathentry kind="con" path="org.eclipse.jst.j2ee.internal.module.container"/>

<classpathentry kind="output" path="bin"/>

</classpath>

Xml配置文件首行多了一空行，编译出错

Jar包，css文件，图片文件利用gjet复制功能，文件损坏。

Classpath

Pure relation table donot generated

Set

DAO等饱的导入

Service实现类的方法重复生成了，

控制器文件名正确，类名全部是TeacherController，不正确

20170811

页面获取实体属性报错：

解决：页面获取实体属性的首两个字母不能大写，也不能一大一小，一小一大，不能带有下划线。

20170814

页面传送到后台的中文内容出项乱码：

解决：在web.xml里配置中文过滤器，jsp页面的字符集设置为UTF-8.

20170816

问题遗留

1. 纯关联表的修改保存
2. 数据库时间类型的生成
3. 数据库一对一和多对多生成的区分

20170821

问题：

1. 保存一对多关系表时，关联的多的一方如是组合主键，不知如何获取组合的主键。

例：

这里如何获取room2device的组合主键？

String[] room2deviceIds = req.getParameterValues("room2devices");

**if** (room2deviceIds != **null**) {

Room thisRoom = roomDAO.findRoomById(room.getId());

Set<Room2device> room2deviceSet = thisRoom

.getRelativeRoom2devices();

**for** (Room2device relate : room2deviceSet) {

relate.setRelativeRoom(**null**);

}

thisRoom.getRelativeRoom2devices().clear();

**for** (String room2deviceId : room2deviceIds) {

Room2device room2devices = **new** Room2device();

这里通过主键来查询

**room2devices = room2deviceDAO.findRoom2deviceById(Integer**

**.*parseInt*(room2deviceId));**

roomService.saveRoomRoom2devices(room.getId(), room2devices);

}

20170822

问题：

八月 22, 2017 9:46:16 上午 org.apache.catalina.core.StandardContext listenerStart

严重: Exception sending context initialized event to listener instance of class org.springframework.web.context.ContextLoaderListener

org.springframework.beans.factory.BeanCreationException: Error creating bean with name 'test' defined in class path resource [test-dao-context.xml]: Invocation of init method failed; nested exception is java.lang.NullPointerException

问题解决：因为组合主键的实体类中映射到id列的属性变量为id，把id换成其他变量。

@Column(name = "id")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

Integer id;

改成：

@Column(name = "id")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

Integer id;

@Column(name = "binary", columnDefinition = "BLOB")

@Basic(fetch = FetchType.*EAGER*)

@Lob

@XmlElement

**byte**[] binaryField;

@Column(name = "varchar")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

String varchar;

@Column(name = "double", precision = 22)

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

BigDecimal double\_;

@Column(name = "smallint")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

Integer smallint;

@Temporal(TemporalType.*TIMESTAMP*)

@Column(name = "timestamp")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

Calendar timestamp;

@Column(name = "bit")

@Basic(fetch = FetchType.*EAGER*)

@XmlElement

Boolean bit;

# Classpath类路径的问题

纯粹的联系表不需要生成Domain类

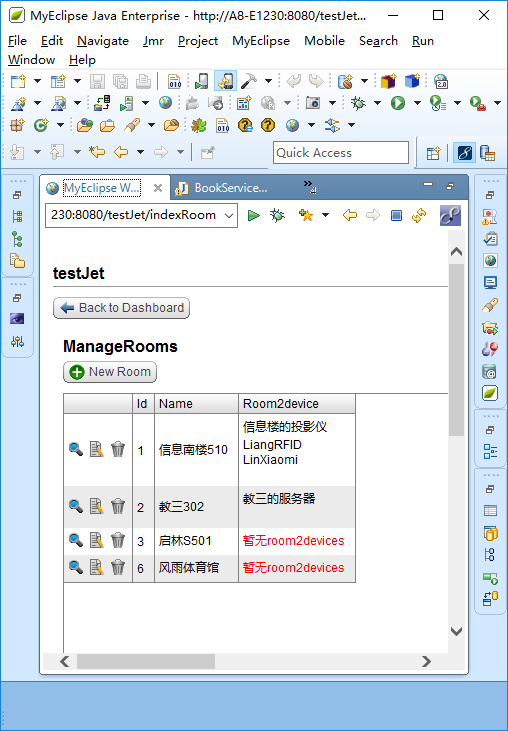
Rest web服务control

最新Jar包问题（调用）

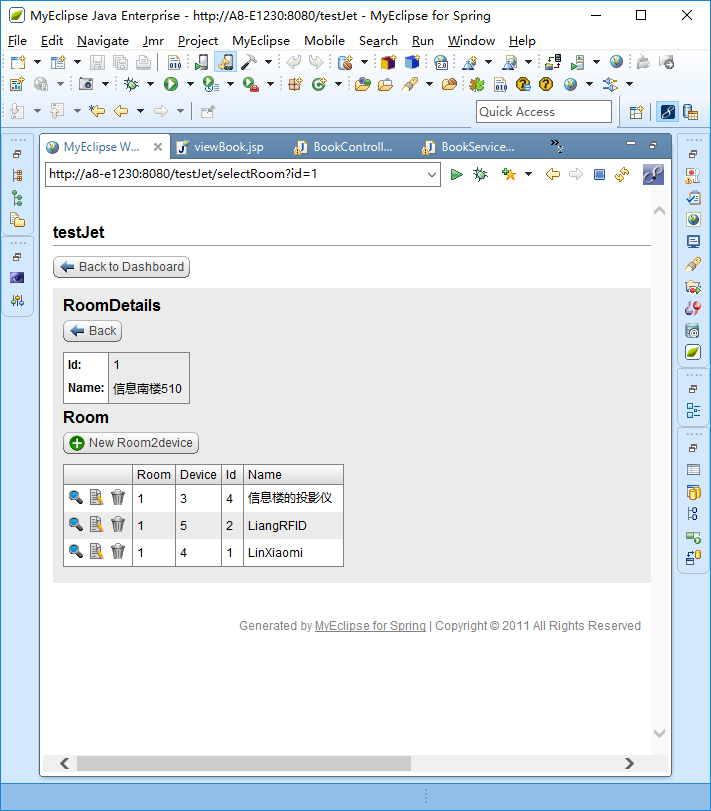
## 20170905缺陷

删除一条记录时，联系表的记录先删除，然后在删除该记录

非联系表(引用)，如果约束是删除设置为空，就设置非联系表中的引用为null，如果是级联删除，就应该删除非联系表中的记录，然后再删除该记录



详细页面：



# 20170908

数据库设计：外键名字要与被引用的数据库表名一致

# 20170913

问题:

1. Controller生成的静态变量被多个方法调用改变其值使其他方法的调用返回有误！

例如：

**static** **int** *cacdevicePageNumber* = 0;

**static** **int** *cacdevicePageSize* = 10;

**static** **int** *cacsensorPageNumber* = 0;

**static** **int** *cacsensorPageSize* = 10;

这些静态变量在selectCac（）方法中用到又在selectCacCacsensors()方法中用到。

1. 自动生成的对象变量有重复生成！

例如：@Autowired

**private** CompanyDAO companyDAO;

@Autowired

**private** CompanyService companyService;

@Autowired

**private** CompanyDAO companyDAO;

@Autowired

**private** CompanyService companyService;

1. Controller中一个实体关联了多个一对多的实体时，数据的请求返回处理方法不知如何去执行！

@RequestMapping("/selectCacrecordtime")

**public** ModelAndView selectCacrecordtime(@RequestParam Integer id, HttpServletRequest request) {

ModelAndView mav = **new** ModelAndView();

**int** cacdevicedataTotalPage = cacdevicedataService

.countRelativeCacrecordtimeCacdevicedatas(id)

/ *cacdevicedataPageSize*;

**String pageType = request.getParameter("page");**

**if** (pageType.equals("homePage"))

*cacdevicedataPageNumber* = 0;

**else** **if** (pageType.equals("previousPage")) {

**if** (*cacdevicedataPageNumber* >= 1)

*cacdevicedataPageNumber* = *cacdevicedataPageNumber* - 1;

**else**

*cacdevicedataPageNumber* = 0;

}

**else** **if** (pageType.equals("nextPage")) {

**if** (*cacdevicedataPageNumber* < cacdevicedataTotalPage)

*cacdevicedataPageNumber* = *cacdevicedataPageNumber* + 1;

**else**

*cacdevicedataPageNumber* = cacdevicedataTotalPage;

}

**else** **if** (pageType.equals("lastPage"))

*cacdevicedataPageNumber* = cacdevicedataTotalPage;

**else** **if** (pageType.equals("jumpPage")) {

*cacdevicedataPageNumber* = Integer.*parseInt*(request

.getParameter("WantToPage")) - 1;

}

**else** **if** (pageType.equals("eachPageShow")) {

*cacdevicedataPageNumber* = 0;

*cacdevicedataPageSize* = Integer.*parseInt*(request

.getParameter("pageSize"));

cacdevicedataTotalPage = cacdevicedataService

.countRelativeCacrecordtimeCacdevicedatas(id)

/ *cacdevicedataPageSize*;

}

**if** (*cacdevicedataPageNumber* >= 0

&& *cacdevicedataPageNumber* <= cacdevicedataTotalPage)

mav.addObject("cacdevicedatas", cacdevicedataDAO

.findCacrecordtimeCacdevicedatas(id,

*cacdevicedataPageNumber* \* *cacdevicedataPageSize*,

*cacdevicedataPageSize*));

mav.addObject("cacdevicedataPageNumber", *cacdevicedataPageNumber*);

mav.addObject("cacdevicedataPageSize", *cacdevicedataPageSize*);

mav.addObject("cacdevicedataTotalPage", cacdevicedataTotalPage);

**int** cacmalfunctionTotalPage = cacmalfunctionService

.countRelativeCacrecordtimeCacmalfunctions(id)

/ *cacmalfunctionPageSize*;

**String pageType = request.getParameter("page");**

**if** (pageType.equals("homePage"))

*cacmalfunctionPageNumber* = 0;

**else** **if** (pageType.equals("previousPage")) {

**if** (*cacmalfunctionPageNumber* >= 1)

*cacmalfunctionPageNumber* = *cacmalfunctionPageNumber* - 1;

**else**

*cacmalfunctionPageNumber* = 0;

}

**else** **if** (pageType.equals("nextPage")) {

**if** (*cacmalfunctionPageNumber* < cacmalfunctionTotalPage)

*cacmalfunctionPageNumber* = *cacmalfunctionPageNumber* + 1;

**else**

*cacmalfunctionPageNumber* = cacmalfunctionTotalPage;

}

**else** **if** (pageType.equals("lastPage"))

*cacmalfunctionPageNumber* = cacmalfunctionTotalPage;

**else** **if** (pageType.equals("jumpPage")) {

*cacmalfunctionPageNumber* = Integer.*parseInt*(request

.getParameter("WantToPage")) - 1;

}

**else** **if** (pageType.equals("eachPageShow")) {

*cacmalfunctionPageNumber* = 0;

*cacmalfunctionPageSize* = Integer.*parseInt*(request

.getParameter("pageSize"));

cacmalfunctionTotalPage = cacmalfunctionService

.countRelativeCacrecordtimeCacmalfunctions(id)

/ *cacmalfunctionPageSize*;

}

**if** (*cacmalfunctionPageNumber* >= 0

&& *cacmalfunctionPageNumber* <= cacmalfunctionTotalPage)

mav.addObject("cacmalfunctions", cacmalfunctionDAO

.findCacrecordtimeCacmalfunctions(id,

*cacmalfunctionPageNumber* \* *cacmalfunctionPageSize*,

*cacmalfunctionPageSize*));

mav.addObject("cacmalfunctionPageNumber", *cacmalfunctionPageNumber*);

mav.addObject("cacmalfunctionPageSize", *cacmalfunctionPageSize*);

mav.addObject("cacmalfunctionTotalPage", cacmalfunctionTotalPage);

**int** cacsensordataTotalPage = cacsensordataService

.countRelativeCacrecordtimeCacsensordatas(id)

/ *cacsensordataPageSize*;

**String pageType = request.getParameter("page");**

**if** (pageType.equals("homePage"))

*cacsensordataPageNumber* = 0;

**else** **if** (pageType.equals("previousPage")) {

**if** (*cacsensordataPageNumber* >= 1)

*cacsensordataPageNumber* = *cacsensordataPageNumber* - 1;

**else**

*cacsensordataPageNumber* = 0;

}

**else** **if** (pageType.equals("nextPage")) {

**if** (*cacsensordataPageNumber* < cacsensordataTotalPage)

*cacsensordataPageNumber* = *cacsensordataPageNumber* + 1;

**else**

*cacsensordataPageNumber* = cacsensordataTotalPage;

}

**else** **if** (pageType.equals("lastPage"))

*cacsensordataPageNumber* = cacsensordataTotalPage;

**else** **if** (pageType.equals("jumpPage")) {

*cacsensordataPageNumber* = Integer.*parseInt*(request

.getParameter("WantToPage")) - 1;

}

**else** **if** (pageType.equals("eachPageShow")) {

*cacsensordataPageNumber* = 0;

*cacsensordataPageSize* = Integer.*parseInt*(request

.getParameter("pageSize"));

cacsensordataTotalPage = cacsensordataService

.countRelativeCacrecordtimeCacsensordatas(id)

/ *cacsensordataPageSize*;

}

**if** (*cacsensordataPageNumber* >= 0

&& *cacsensordataPageNumber* <= cacsensordataTotalPage)

mav.addObject("cacsensordatas", cacsensordataDAO

.findCacrecordtimeCacsensordatas(id,

*cacsensordataPageNumber* \* *cacsensordataPageSize*,

*cacsensordataPageSize*));

mav.addObject("cacsensordataPageNumber", *cacsensordataPageNumber*);

mav.addObject("cacsensordataPageSize", *cacsensordataPageSize*);

mav.addObject("cacsensordataTotalPage", cacsensordataTotalPage);

mav.addObject("cacrecordtime",

cacrecordtimeDAO.findCacrecordtimeByPrimaryKey(id));

mav.setViewName("cacrecordtime/viewCacrecordtime.jsp");

**return** mav;

}

1. Controller中的重定位如何带上参数？（**return** "forward:/indexTeacher";）

## 在Eclipse中测试pig项目

### 

控制台输出error信息：

templates/content/java/daoImpl.jet:

Error: The Java content could not be formatted. Check that it is well-formed and that the kind attribute (if specified) is compatible with the actual snippet.

templates/content/java/controller.jet:

Error: The Java content could not be formatted. Check that it is well-formed and that the kind attribute (if specified) is compatible with the actual snippet.

templates/process/image.jet(24,1): <ws:copyFile src='templates/content/images/icons/view.gif' binary='true' target='{$projectName}/WebRoot/images/icons/view.gif'>

Writing file: pigMedicine/WebRoot/images/icons/view.gif

Errors occurred during execution

PigDAO、

Fodder\_takeDAO和Fodder\_takeDAOImpl、Medicine\_takeDAO、存在2个相同的方法，不是重复，应该有所区分

**public** Set<Medicine\_take> findEmployeeMedicine\_takes(Integer Employee, **int** start, **int** max);

**public** Set<Fodder\_take> findEmployeeFodder\_takes(Integer Employee, **int** start, **int** max);

**public** Set<Fodder\_take> findEmployeeFodder\_takes(Integer Employee, **int** start, **int** max);

FoddertakenviewDAOImpl

缺少**import** javax.persistence.Query和**import** javax.persistence.NoResultException的导入

@Transactional

**public** Foddertakenview findFoddertakenviewByPrimaryKey( **int** startResult, **int** maxRows) **throws** DataAccessException{

**try** {

Query query = createNamedQuery("findFoddertakenviewByPrimaryKey", startResult, maxRows, );

**return** (cn.edu.scau.cmi.liangzaoqing.domain.Foddertakenview) query.getSingleResult();

} **catch** (NoResultException nre) {

**return** **null**;

}

}

result = (**int**) (prime \* result + ((Id == **null**) ? 0 : Id.hashCode()));

**if** ((Id == **null** && equalCheck.Id != **null**) || (Id != **null** && equalCheck.Id == **null**))

**return** **false**;

**if** (Id != **null** && !Id.equals(equalCheck.Id))

**return** **false**;

**return** **true**;

RS中有很多重复的属性

# 2017124

把外关联表的属性的命名规范由@referenceTable改为@name

把被关联表的属性的命名规范由@name改为@name+/column/@name组合。