



**（3243700）**

Java程序设计

课程设计报告

**题目： 大学课堂点名系统**

**学院： 信息科学与技术学院**

**教师： 何太军**

**姓名： 江胤佐**

**学号： 2018112664**

**班级： 软件2018-01班**

**2019-2020（1）**

目录

[一、实验内容 5](#_Toc27639653)

[1.1 任务要求 5](#_Toc27639654)

[1.2 完成程度 5](#_Toc27639655)

[二、实验过程 5](#_Toc27639656)

[2.1 需求分析 5](#_Toc27639657)

[2.1.1功能需求 5](#_Toc27639658)

[2.1.2非功能需求 6](#_Toc27639659)

[2.2 业务流程分析 6](#_Toc27639660)

[2.3 开发环境与项目依赖 7](#_Toc27639661)

[2.4 数据设计 8](#_Toc27639662)

[2.4.1 实体关系设计 8](#_Toc27639663)

[2.4.2 文件存储设计 8](#_Toc27639664)

[2.4.3 数据库设计 9](#_Toc27639665)

[2.5 系统概要设计 11](#_Toc27639666)

[2.5.1 软件总体架构 11](#_Toc27639667)

[2.5.2 Java包规划 12](#_Toc27639668)

[2.5.3 静态资源文件规划 15](#_Toc27639669)

[2.6 系统架构设计与实现 16](#_Toc27639670)

[2.6.1 持久化存储辅助类 16](#_Toc27639671)

[2.6.1.1文件读写辅助类 17](#_Toc27639672)

[2.6.1.2 MySQL数据库存储辅助类及相关注解实现 17](#_Toc27639673)

[2.6.2 实体类和数据传输对象（DTO） 21](#_Toc27639674)

[2.6.2.1整体设计 21](#_Toc27639675)

[2.6.2.2实体类属性说明 22](#_Toc27639676)

[2.6.2.3数据传输对象说明 24](#_Toc27639677)

[2.6.3 DAO层 24](#_Toc27639678)

[2.6.3.1接口定义 24](#_Toc27639679)

[2.6.3.2 DAO工厂 24](#_Toc27639680)

[2.6.3.3 DAO文件实现类 25](#_Toc27639681)

[2.6.3.4 DAO数据库实现类 25](#_Toc27639682)

[2.6.4 Service层 26](#_Toc27639683)

[2.6.5 UI层全局状态管理类 27](#_Toc27639684)

[2.6.6 控制台UI 27](#_Toc27639685)

[2.6.6.1 抽象UI类和UI容器类 27](#_Toc27639686)

[2.6.7 JavaFX UI 29](#_Toc27639687)

[2.6.7.1 舞台管理类 29](#_Toc27639688)

[2.6.7.2 面板加载工具类 29](#_Toc27639689)

[2.6.7.3 控制器代理类 30](#_Toc27639690)

[2.7 系统功能详细设计 31](#_Toc27639691)

[2.7.1 登录 31](#_Toc27639692)

[2.7.2 自定义消息弹框 32](#_Toc27639693)

[2.7.3 查看课表 32](#_Toc27639694)

[2.7.4 学生、教学班信息的增删改查 33](#_Toc27639695)

[2.7.5 点名与提问 33](#_Toc27639696)

[2.7.6 点名结果汇总 35](#_Toc27639697)

[2.7.6 点名结果汇总 36](#_Toc27639698)

**目 录**

分页（如果有目录，则目录与正文分页）

**正文**

主要包括任务内容及要求、分析、设计（概要设计、详细设计、关键算法设计等）、实现（程序结构、各类的设计、实现效果、关键代码段）、部署及使用说明等相关内容。

实验报告主要包括：实验内容指实验要完成的任务及程度；实验过程指实验分析、实验步骤、重要的类或数据结构、算法的设计、界面设计、程序结构、实验中间结果（运行结果截图）等；实验结果（最终结果要截图）包括结果介绍、评价；实验体会主要写收获与存在的问题及解决思路等，要体现出条理，不要胡乱拼凑。如果有其他建议或意见请补充到报告后边。附录附上关键代码。

正文字体：建议宋体五号，注意插图、插表要排版整齐，插图不要太大，以图上的文字与正文差不多大、幅面以能够体现要说明的问题即可（实验报告的规范格式也是考核项）。

提交要求为word文档。

实验报告的详细程度应以参照实验报告步骤及主要说明能再次重现实验为准。

# 一、实验内容

## 1.1 任务要求

设计一个大学课堂点名系统，用于课堂点名、分析、课堂提问记录，汇总点名结果。主要功能有：教学班设置、教学班名单、课程表设置、课堂点名操作、课堂提问。涉及技术有：Java面向对象程序设计、GUI、IO、数据库、多线程、异常、集合...

实验需每周在码云的个人仓库上提交代码。

## 1.2 完成程度

分别使用JavaFX和控制台、MySQL数据库和文件来完成上述任务要求中的功能。

# 二、实验过程

## 2.1 需求分析

### 2.1.1功能需求

程序包括学生端和教师端：学生端需具备查看课表、查看点名记录等功能；教师端需具备查看课表、教学班设置、编辑教学班名单、点名（顺序点名、异常点名、二次点名、随机点名）、提问、编辑学生等功能。详细的功能需求如下图所示。



### 2.1.2非功能需求

界面简介美观、交互友好、程序运行流畅。

## 2.2 业务流程分析

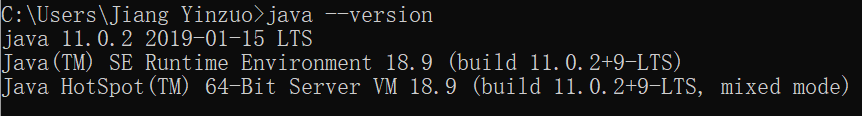


系统业务流程图如上所示。

## 2.3 开发环境与项目依赖

操作系统：Windows 10

Java版本：java 11.0.2 2019-01-15 LTS



MySQL版本：mysql Ver 8.0.16 for Win64 on x86\_64 (MySQL Community Server - GPL)



jar包依赖：

Java操作Excel：jxl-2.6.12

Java操作MySQL：mysql-connector-java-8.0.18

JavaFX：openjfx-13.0.1\_windows-x64\_bin-sdk

开发工具：

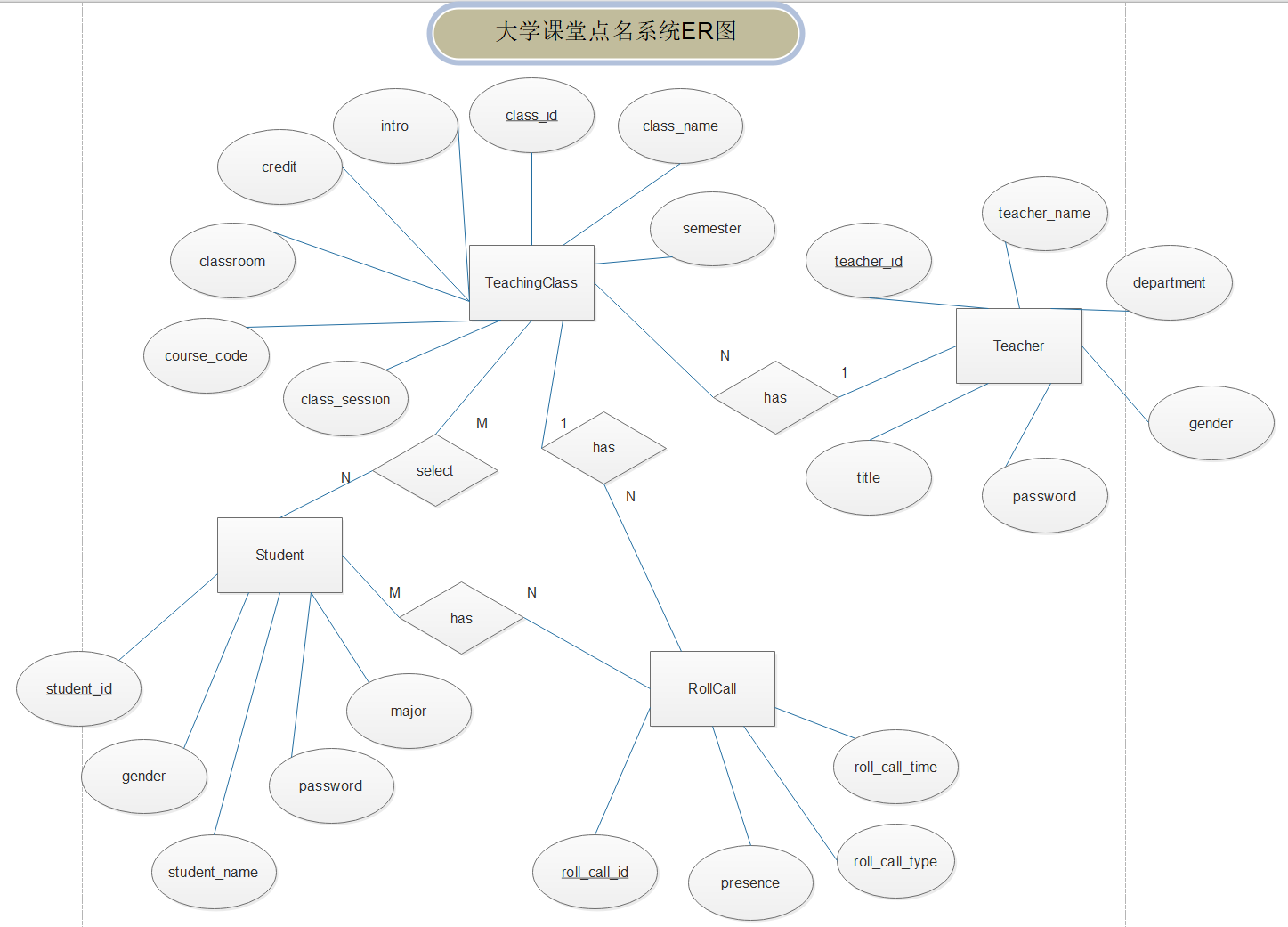
IDE：Eclipse Jee 2019-09、IntelliJ IDEA 2019.2.2 x64

数据库可视化软件：Navicat Premium 12

FXML可视化编辑：SceneBuilder

## 2.4 数据设计

### 2.4.1 实体关系设计



实体类说明：

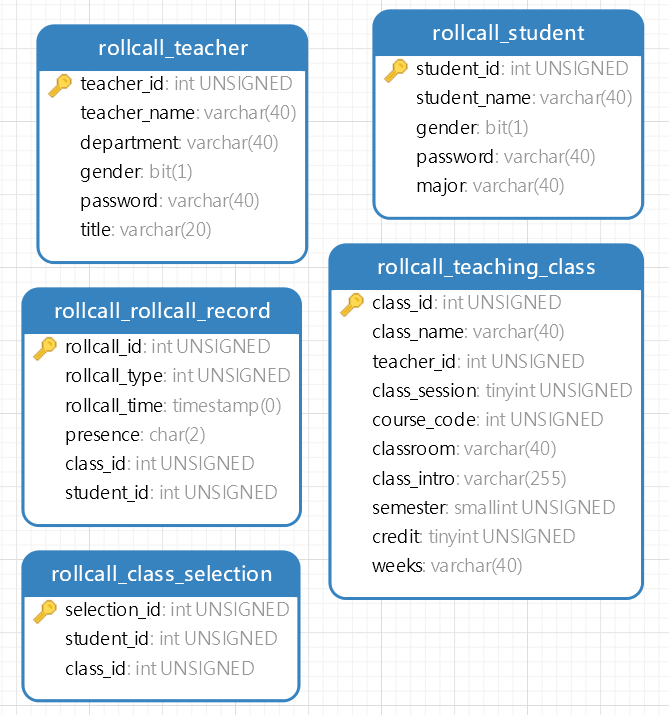
实体包括学生（Student）、点名记录（RollCall）、教师（Teacher）、教学班（TeachingClass）。实体关系及实体属性如上图所示。

### 2.4.2 文件存储设计

通过Java反序列化将实体类存入.txt文件；通过序列化从.txt文件中读取实体类。文件存放在运行程序的当前目录下的files中。包括rollcalls.txt、student.txt、teachers.txt、teachingClasses.txt、classSelection.txt。

### 2.4.3 数据库设计

数据库包含教师表（rollcall\_teacher）、学生表(rollcall\_student)、点名记录表(rollcall\_rollcall\_record)、教学班表（rollcall\_teaching\_class）、选课记录表（rollcall\_class\_selection，同时也是学生—教学班关系表）。数据库模型如下图所示。



各表的DDL如下：

教学班表

CREATE TABLE `rollcall\_teaching\_class` (

`class\_id` int(9) unsigned zerofill NOT NULL AUTO\_INCREMENT COMMENT '教学班id',

`class\_name` varchar(40) NOT NULL COMMENT '课程名称',

`teacher\_id` int(9) unsigned zerofill NOT NULL COMMENT '任课教师id',

`class\_session` tinyint(2) unsigned NOT NULL COMMENT '上课讲次，第一个数代表周几，第二个数代表第几讲',

`course\_code` int(6) unsigned zerofill NOT NULL COMMENT '课程代码',

`classroom` varchar(40) NOT NULL DEFAULT '' COMMENT '教室',

`class\_intro` varchar(255) NOT NULL DEFAULT '' COMMENT '课程简介',

`semester` smallint(5) unsigned zerofill NOT NULL COMMENT '上课学期',

`credit` tinyint(3) unsigned NOT NULL COMMENT '学分',

`weeks` varchar(40) NOT NULL DEFAULT '' COMMENT '周次',

PRIMARY KEY (`class\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=670 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='教学班表';

点名记录表

CREATE TABLE `rollcall\_rollcall\_record` (

`rollcall\_id` int(9) unsigned zerofill NOT NULL AUTO\_INCREMENT COMMENT '点名id',

`rollcall\_type` int(4) unsigned NOT NULL COMMENT '点名类型 1: 点名; 2: 提问',

`rollcall\_time` timestamp NOT NULL DEFAULT CURRENT\_TIMESTAMP COMMENT '点名时间',

`presence` char(2) NOT NULL COMMENT '点名情况',

`class\_id` int(9) unsigned NOT NULL COMMENT '教学班id',

`student\_id` int(9) unsigned zerofill NOT NULL COMMENT '学号',

PRIMARY KEY (`rollcall\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=541 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='点名表';

教师表

CREATE TABLE `rollcall\_teacher` (

`teacher\_id` int(9) unsigned NOT NULL COMMENT '教师工号',

`teacher\_name` varchar(40) NOT NULL COMMENT '教师姓名',

`department` varchar(40) NOT NULL COMMENT '教师所属部门',

`gender` bit(1) NOT NULL COMMENT '1: 男; 0: 女',

`password` varchar(40) NOT NULL COMMENT '密码',

`title` varchar(20) NOT NULL COMMENT '职称',

PRIMARY KEY (`teacher\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='教师表';

学生表

CREATE TABLE `rollcall\_student` (

`student\_id` int(9) unsigned NOT NULL COMMENT '学号',

`student\_name` varchar(40) NOT NULL COMMENT '学生姓名',

`gender` bit(1) NOT NULL COMMENT '1: 男; 0: 女',

`password` varchar(40) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT '123456' COMMENT '密码',

`major` varchar(40) NOT NULL COMMENT '专业',

PRIMARY KEY (`student\_id`)

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='学生表';

选课记录表（学生—教学班关系表）

CREATE TABLE `rollcall\_class\_selection` (

`selection\_id` int(9) unsigned zerofill NOT NULL AUTO\_INCREMENT COMMENT '选课操作id',

`student\_id` int(9) unsigned NOT NULL COMMENT '学号',

`class\_id` int(9) unsigned zerofill NOT NULL COMMENT '课程id',

PRIMARY KEY (`selection\_id`)

) ENGINE=InnoDB AUTO\_INCREMENT=19 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='选课表（学生-教学班关系表）';

## 2.5 系统概要设计

### 2.5.1 软件总体架构



系统采用分层架构，总体可分为DAO层、Service层和UI层。

持久化存储辅助类FileHelper和MySQLHelper分别对Java文件IO、JDBC进行了封装，实现了实体类（Entity）和数据传输对象（DTO）与持久化存储（文件、数据库表的字段）之间的映射。

DAO层借助持久化存储的辅助类分别对文件和MySQL数据库进行持久化操作。DAO层定义了一系列接口，由文件实现类和MySQL实现类分别操作文件和数据库，对Service层接口一致，体现了面向接口编程的思想。

Service层是对DAO层进一步的封装，向UI层提供业务逻辑方法接口。Service层通过工厂方法和反射来读取配置文件，并动态创建DAO的实现类。

UI层包括控制台UI和JavaFX UI。

控制台UI借助控制台IO辅助类来完成用户交互。UI容器类负责创建UI对象，并加载界面。

JavaFX UI采用MVC模式。Controller实现用户交互的业务逻辑，调用Service层的持久化操作接口，并借助FXML加载工具类和舞台跳转管理类来加载FXML、CSS、图片等静态资源文件。

### 2.5.2 Java包规划

pers

└─jiangyinzuo

└─rollcall

├─common

│ CustomException.java // 自定义异常处理类

│

├─config

│ Config.java // 全局配置类

│

├─dao // DAO层接口

│ │ RollCallDao.java

│ │ StudentDao.java

│ │ TeacherDao.java

│ │ TeachingClassDao.java

│ │

│ ├─fileimpl // DAO层文件实现类

│ │ RollCallDaoFileImpl.java

│ │ StudentDaoFileImpl.java

│ │ TeacherDaoFileImpl.java

│ │ TeachingClassDaoFileImpl.java

│ │

│ └─mysqlimpl // DAO层数据库实现类

│ RollCallDaoMysqlImpl.java

│ StudentDaoMysqlImpl.java

│ TeacherDaoMysqlImpl.java

│ TeachingClassDaoMysqlImpl.java

│

├─domain // 领域对象模型

│ ├─dto // 数据传输对象，在实体类无法满足需求时使用

│ │ ClassSelectionRecordDTO.java

│ │ StudentRollCallResultDTO.java

│ │

│ ├─entity // 实体类

│ │ RollCall.java

│ │ Student.java

│ │ Teacher.java

│ │ TeachingClass.java

│ │

│ ├─mapper // 实体类或DTO映射到数据库表字段的注解

│ │ FieldMapper.java

│ │ TableMapper.java

│ │

│ └─repository // 同实体类密切相关的持久化操作

│ RollCallRepo.java

│ StudentRepo.java

│ TeachingClassRepo.java

│

├─factory // 创建DAO对象的工厂

│ DaoFactory.java

│

├─helper // 持久化存储和控制台IO的辅助类

│ ConsoleIoHelper.java

│ FileHelper.java

│ MySqlHelper.java

│

├─service // Service层接口

│ │ LoginService.java

│ │ RollCallService.java

│ │ StudentService.java

│ │ TeachingClassService.java

│ │

│ └─impl // Service层实现类

│ LoginServiceImpl.java

│ RollCallServiceImpl.java

│ StudentServiceImpl.java

│ TeachingClassServiceImpl.java

│

├─ui // UI层

│ ├─common

│ │ RollCallManager.java // 控制台UI和JavaFX UI

// 共同的点名管理类

│ │

│ ├─console // 控制台UI

│ │ │ AbstractUi.java // UI的抽象基类

│ │ │ LoginUi.java

│ │ │ Main.java

│ │ │ UiContainer.java // 创建UI的容器类

│ │ │

│ │ ├─student // 学生界面UI

│ │ │ StudentMainUi.java

│ │ │

│ │ └─teacher // 教师界面UI

│ │ RollCallUi.java

│ │ TeacherMainUi.java

│ │ TeachingClassDetailUi.java

│ │

│ ├─javafx // JavaFX UI

│ │ │ Main.java // JavaFX应用的入口类

│ │ │

│ │ ├─common

│ │ │ CustomAlertBoard.java // 自定义消息弹框

│ │ │

│ │ ├─controller //作为场景加载的 JavaFX 控制器

│ │ │ │ AddTeachingClassController.java

│ │ │ │ EditClassController.java

│ │ │ │ FxController.java

│ │ │ │ LoginController.java

│ │ │ │ RollCallBoardController.java

│ │ │ │ RollCallDetailController.java

│ │ │ │ RollCallRecordController.java

│ │ │ │ RollCallStatisticController.java

│ │ │ │ ScheduleController.java

│ │ │ │ StudentListController.java

│ │ │ │ StudentMainBoardController.java

│ │ │ │ StudentManagementController.java

│ │ │ │ TeacherMainBoardController.java

│ │ │ │

│ │ │ ├─components // 作为面板加载的JavaFX 控制器

│ │ │ │ RollCallCmpController.java

│ │ │ │ RollCallRecordCmpController.java

│ │ │ │ RollCallStatisticCmpController.java

│ │ │ │ StudentCmpController.java

│ │ │ │ TeachingClassCmpController.java

│ │ │ │

│ │ │ └─proxy // 控制器访问代理类

│ │ │ ControllerProxy.java

│ │ │

│ │ ├─router // 舞台跳转管理类

│ │ │ StageManager.java

│ │ │

│ │ └─utils // FXML面板加载工具类

│ │ FxmlCmpLoaderUtil.java

│ │

│ └─state // 全局状态管理类，存储用户信息、当前教学班等

│ SelectedTeachingClassState.java

│ UserState.java

│

└─util

DeepCopy.java // 对象深拷贝工具类

### 2.5.3 静态资源文件规划

resources1

└─ // 作为场景加载的fxml文件

│ AddTeachingClass.fxml

│ EditClass.fxml

│ Login.fxml

│ RollCallBoard.fxml

│ RollCallDetail.fxml

│ RollCallRecord.fxml

│ RollCallStatistic.fxml

│ Schedule.fxml

│ StudentList.fxml

│ StudentMainBoard.fxml

│ StudentManagement.fxml

│ TeacherMainBoard.fxml

│ tree.txt

│

├─components // 作为面板加载的fxml文件

│ RollCallCmp.fxml

│ RollCallRecordCmp.fxml

│ RollCallStatisticCmp.fxml

│ StudentCmp.fxml

│ TeachingClassCmp.fxml

│

└─static // 图片、CSS等静态资源文件

classManage.png

Login.css

loginBackground.jpg

newClass.png

schedule.png

search.png

submit.png

teacher-main-board.css

## 2.6 系统架构设计与实现

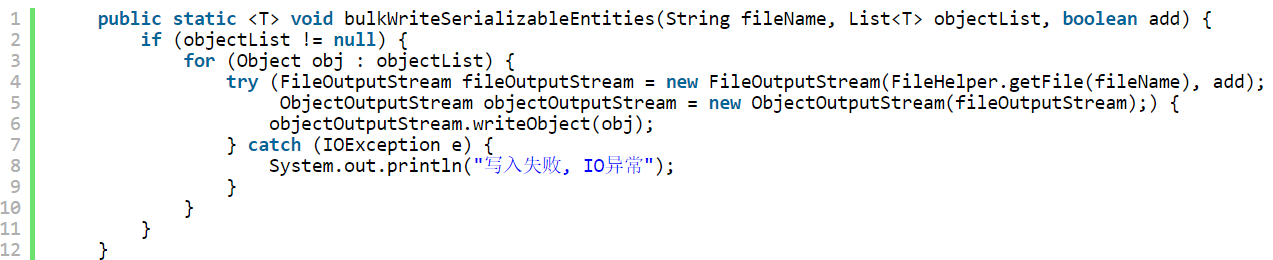
### 2.6.1 持久化存储辅助类

#### 2.6.1.1文件读写辅助类

pers.jiangyinzuo.rollcall.helper.FileHelper辅助类是对Java文件IO的简单封装，主要包括读和写操作。需要注意的是，实体类需要实现Serializable接口。



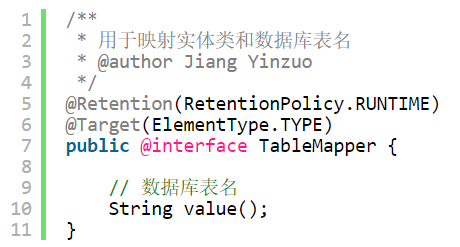
批量读取序列化实体类

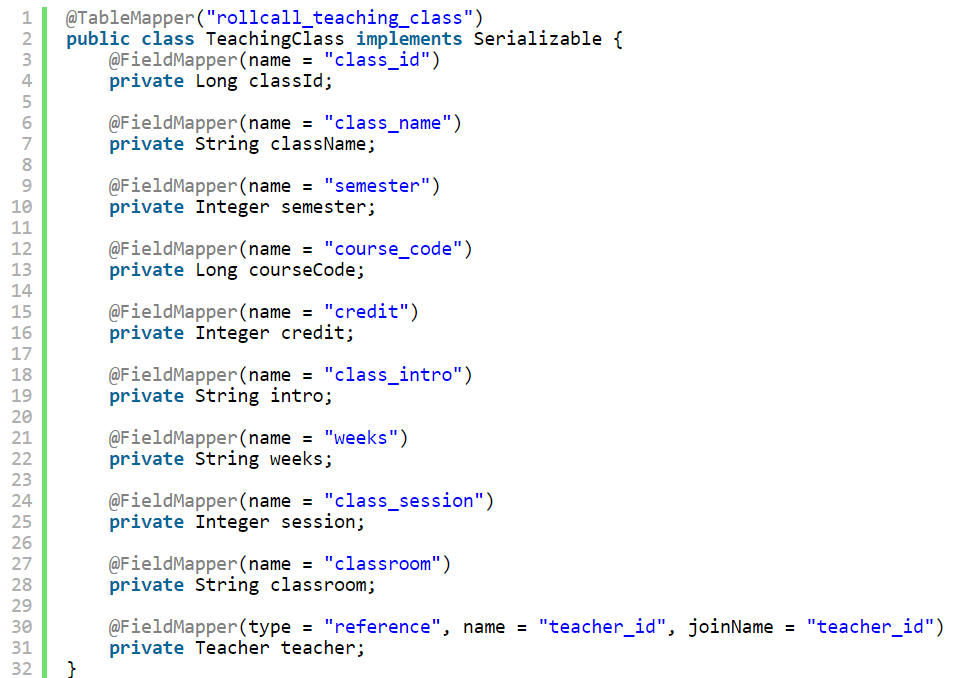


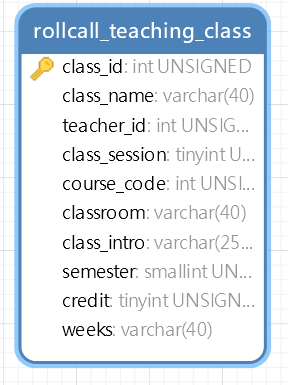
批量写入序列化实体类

#### 2.6.1.2 MySQL数据库存储辅助类及相关注解实现

pers.jiangyinzuo.rollcall.helper.MySqlHelper是对JDBC的进一步封装。为实现实体类、DTO到数据库表记录的映射，需配合pers.jiangyinzuo.rollcall.domain.mapper. FieldMapper和pers.jiangyinzuo.rollcall.domain.mapper.TableMapper注解实现。

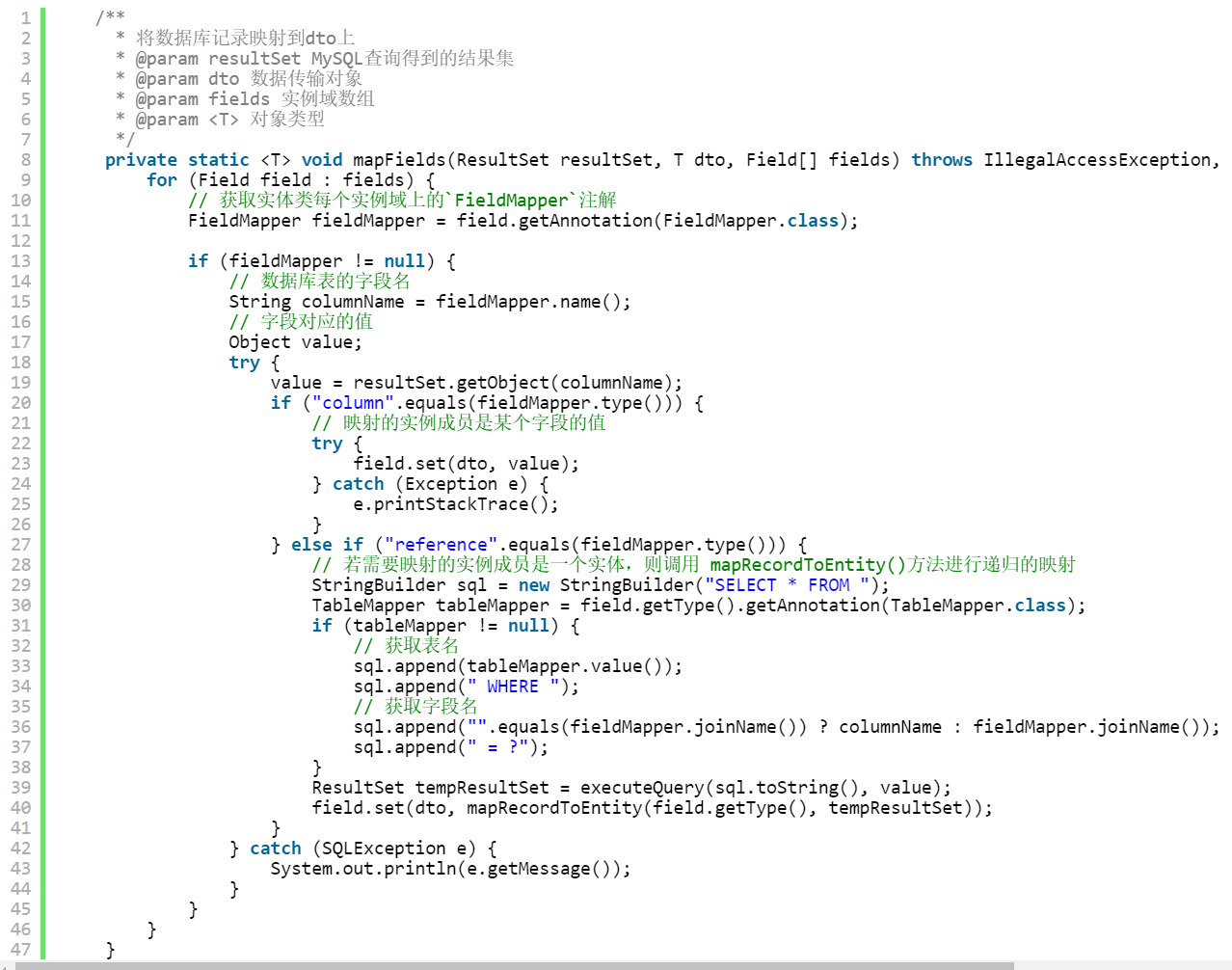


以教学班实体类和教学班表为例，下图展现了注解在实体类或DTO中的使用方法



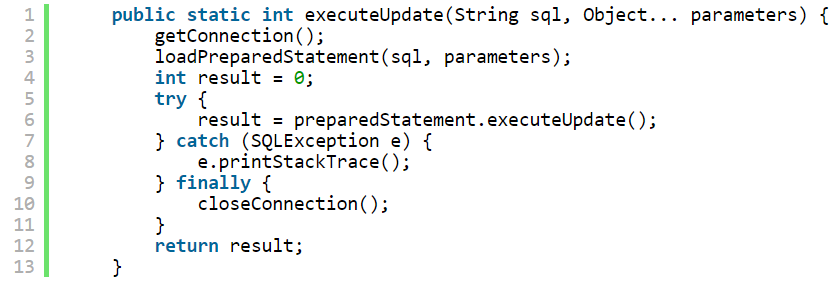
表字段名和@FieldMapper注解中的name属性一一对应。

在MySQLHelper类中，实体类到数据库表的映射主要由mapRecordsToEntities、mapRecordToEntity、mapRecordsSystemType配合mapFields实现。下图以mapFields和mapRecordToEntity为例说明。



DAO主要调用MysqlHelper的queryMany、queryOne、executeUpdate方法，分别用于多条记录查询、单条记录查询和更新操作。

查询以queryOne为例，如图所示，queryOne方法根据传入的Class<T>，将ResultSet转换成实体类，并返回给DAO。

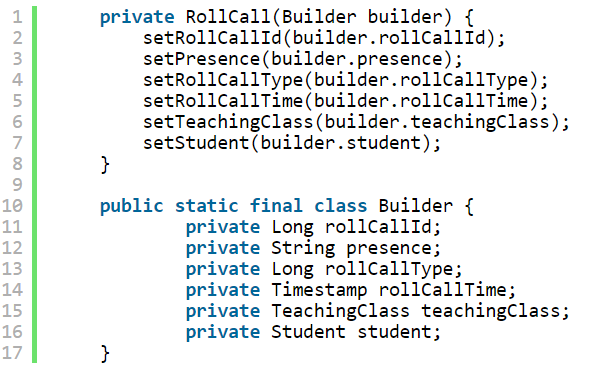
executeUpdate方法首先调用loadPreparedStatement方法，对静态preparedStatement成员变量进行初始化后调用preparedStatement.executeUpdate()方法。

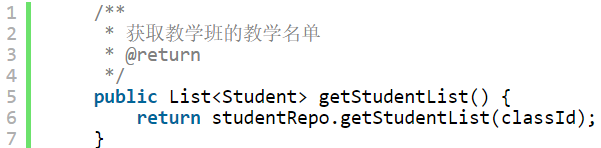
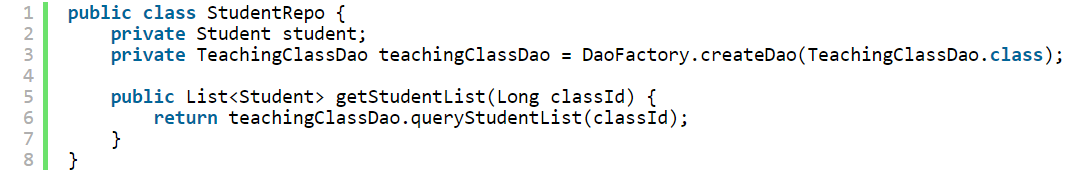
### 2.6.2 实体类和数据传输对象（DTO）

#### 2.6.2.1整体设计

大多数实体类和DTO实现了Serializable接口并带有TableMapper注解，用于持久化存储操作。同时实体类实现了建造者模式，用于方便地实例化对象。

下图为RollCall实体类的省略了getter和setter的建造者模式的部分代码。

部分实体类中有Repo对象，例如TeachingClass实体中包含了StudentRepo对象。这些对象用于丰富实体类的功能：从面向对象的角度来说，TeachingClass应该有获取其教学

名单的行为。Repo赋予了实体类调用DAO接口方法的能力。

#### 2.6.2.2实体类属性说明

RollCall 点名实体类

|  |  |
| --- | --- |
| presence | 点名情况 |
| rollCallTime | 点名时间戳 |
| teachingClassId | 教学班ID |
| student | 学生实体类 |
| rollcallId | 点名ID |
| rollcallType | 点名类型 |

Student学生实体类

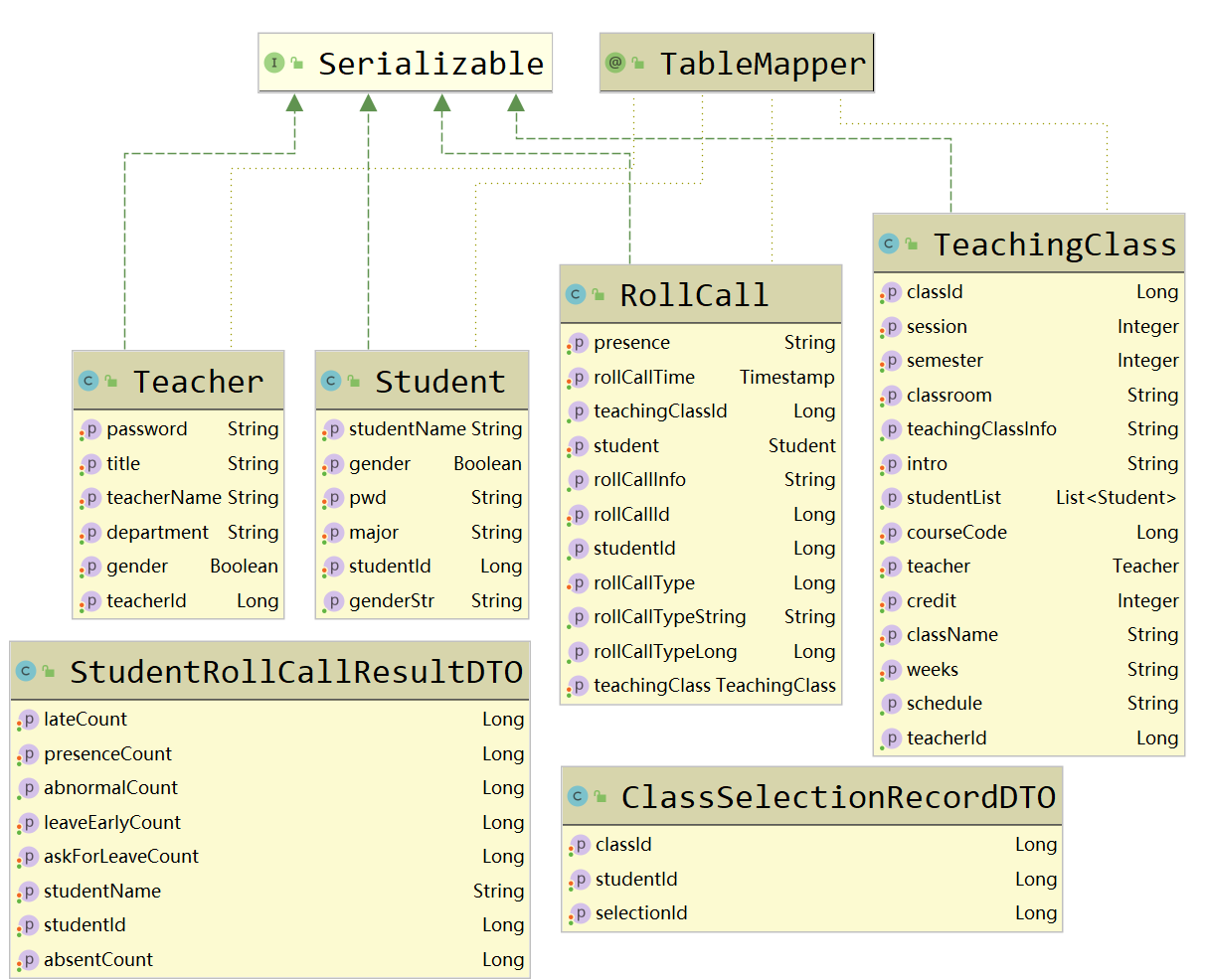
|  |  |
| --- | --- |
| studentId | 学生ID |
| studentName | 学生姓名 |
| gender | 性别 |
| password | 密码 |
| major | 专业 |

Teacher 教师实体类

|  |  |
| --- | --- |
| teacherId | 教师ID |
| teacherName | 教师姓名 |
| department | 部门 |
| gender | 性别 |
| title | 职称 |
| password | 密码 |

TeachingClass 教学班实体类

|  |  |
| --- | --- |
| classId | 教学班ID |
| session | 周几第几讲 |
| semester | 第几学期 |
| classroom | 上课教室 |
| intro | 简介 |
| weeks | 周次 |
| courseCode | 课程代码 |
| teacher | 任课教师 |
| Credit | 学分 |
| ClassName | 班级名称 |



#### 2.6.2.3数据传输对象说明

由于部分业务逻辑的实现需要多个实体类属性的组合表示，为了代码复用，定义数据传输对象。其中StudentRollCallResultDTO用于存储点名结果，ClassSelectionRecordDTO用于存储选课结果。

### 2.6.3 DAO层

#### 2.6.3.1接口定义

DAO层总共有TeachingClassDao、RollCallDao、TeacherDao、StudentDao四个接口，方法如上图所示。四个接口分别负责教学班实体类、点名实体类、教师实体类、学生实体类的增删改查。

#### 2.6.3.2 DAO工厂

程序运行时，DaoFactory类首先加载资源文件dao.properties中的dao属性。若为MySql，则创建MySql实现类；若为File，则创建File实现类。工厂创建DAO的用法如下：

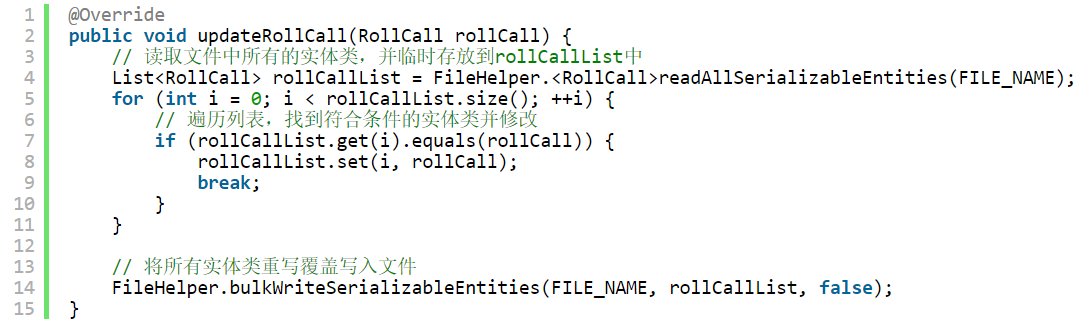


下图为工厂类的实现



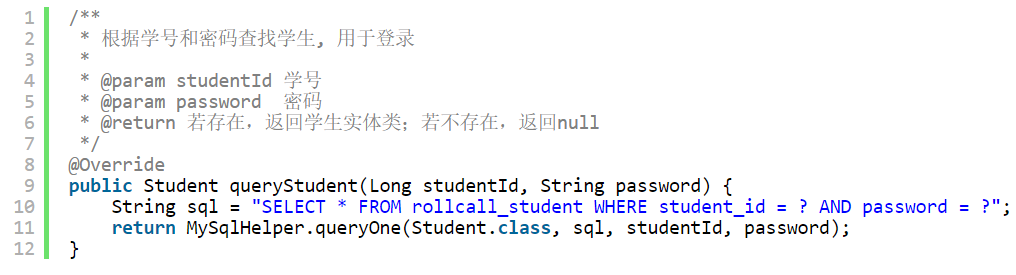
#### 2.6.3.3 DAO文件实现类

文件实现类调用FileHelper读写文件的方法。由于文件读写难以直接修改、删除序列化后的实体对象。故修改和删除操作需要先读取文件中所有的实体类，存放到List<T>中，再遍历列表对实体对象进行查找、修改或删除。下例是修改点名记录的方法。



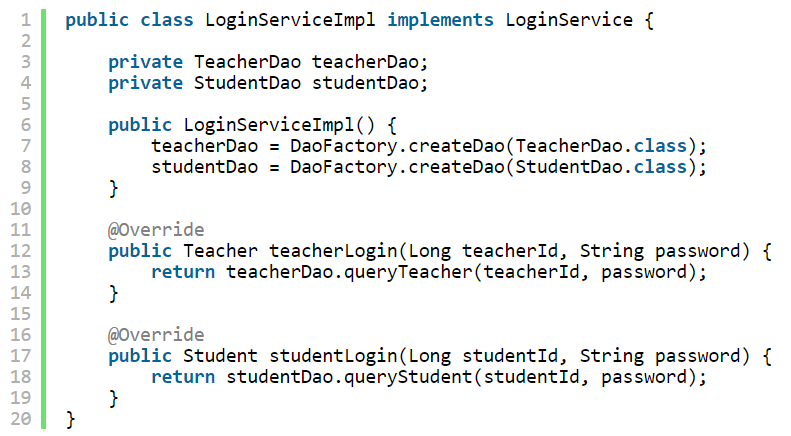
#### 2.6.3.4 DAO数据库实现类

数据库实现类将sql语句传入MySqlHelper中直接调用MySqlHelper的查询或更新方法即可。下例为根据账号和密码查询学生。



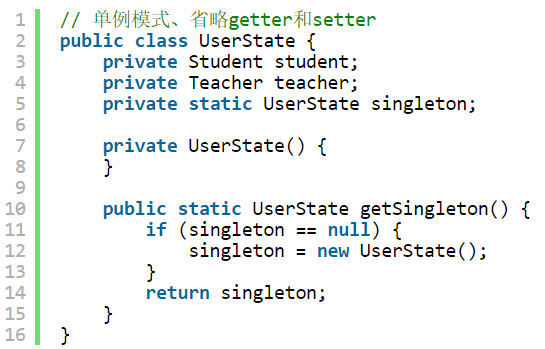
### 2.6.4 Service层

Service是对DAO的进一步封装。Service实现类中通常有使用静态工厂方法创建的DAO对象，并调用DAO的方法。RollCallService、StudentService、LoginService、TeachingClassService分别负责处理点名、学生管理、登录、教学班管理相关的业务逻辑。下例为LoginService实现。



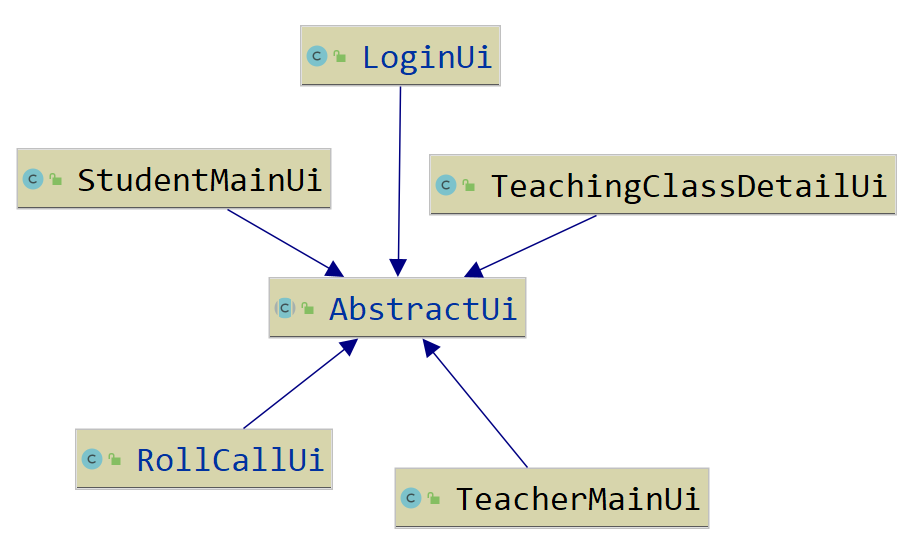
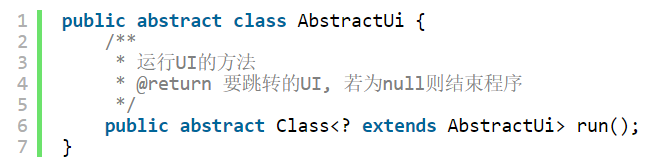
### 2.6.5 UI层全局状态管理类

用户信息、选中教学班等数据需要在全局中使用。故采用单例模式实现UserState类和SelectedTeachingClassState类，用来存储整个系统都要用到的用户信息与教学班信息。下图为省略getter和setter的用户全局状态管理类



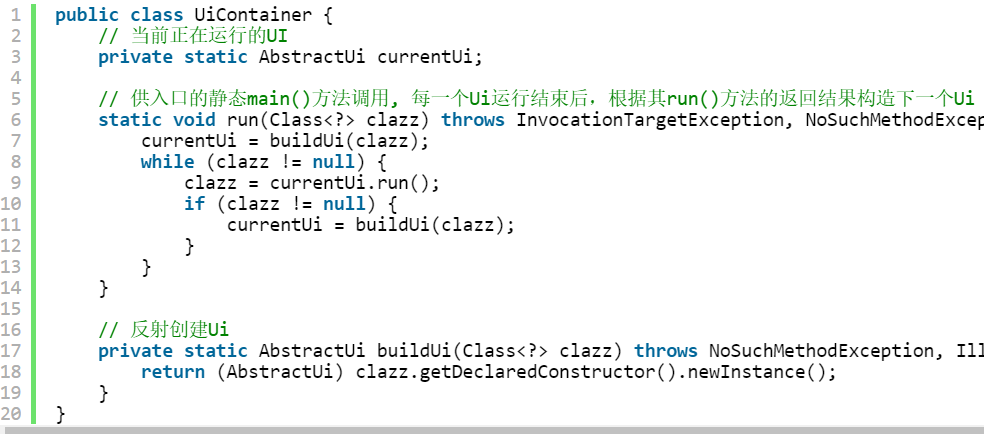
### 2.6.6 控制台UI

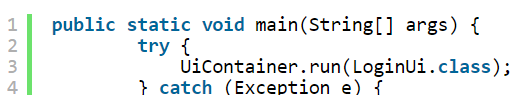
#### 2.6.6.1 抽象UI类和UI容器类

AbstractUi是所有Ui类的抽象基类，其中只有一个返回Class<>泛型的抽象方法。

控制台总共有登录、教学班详情、点名、教师主界面、学生主界面5个Ui。

Ui容器类负责创建Ui，并调用UI的run()方法来显示UI。当前UI运行结束后，将返回给UiContainer一个Class对象，表示下一个需要创建的UI。

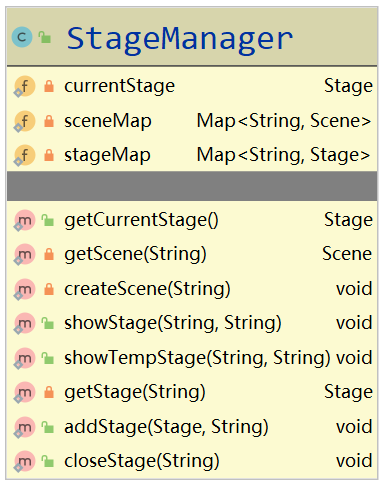




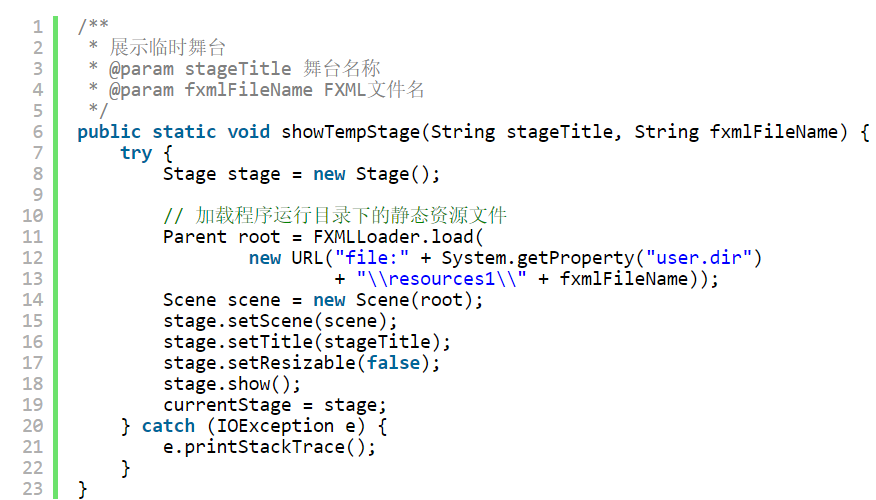
静态main()方法调用UiContainer的run方法，并将第一个登录界面的类对象作为参数传递。

### 2.6.7 JavaFX UI

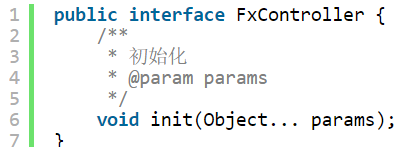
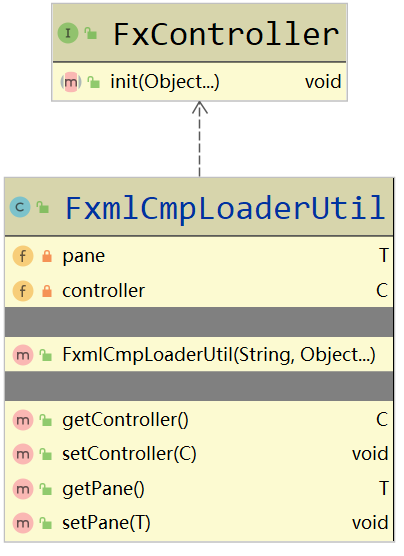
#### 2.6.7.1 舞台管理类



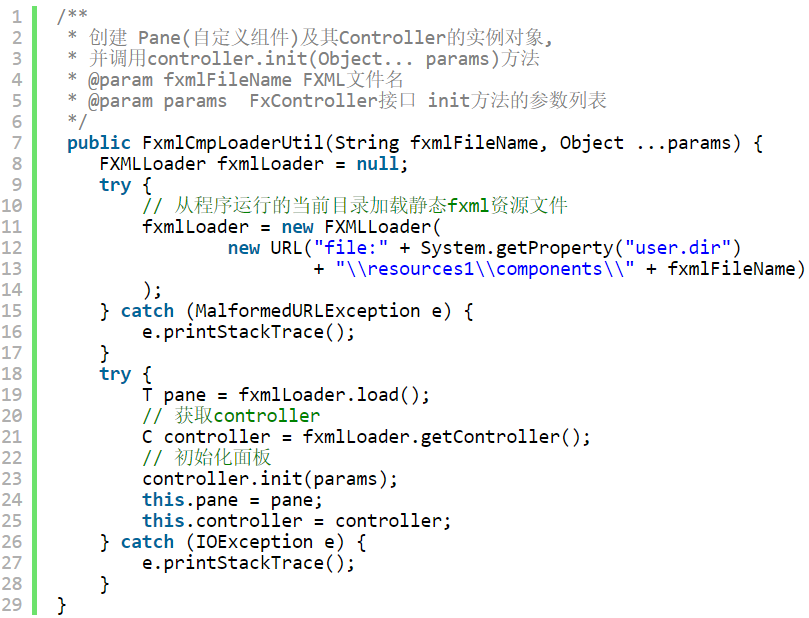
JavaFX程序通过FXMLloader类加载fxml等静态资源。舞台管理类包括获取当前舞台、根据标题获取舞台等方法。下图为较为核心的展现舞台方法。



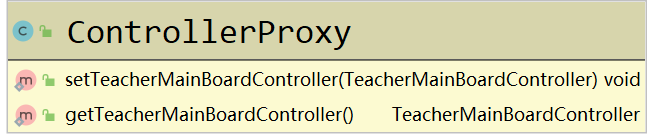
#### 2.6.7.2 面板加载工具类



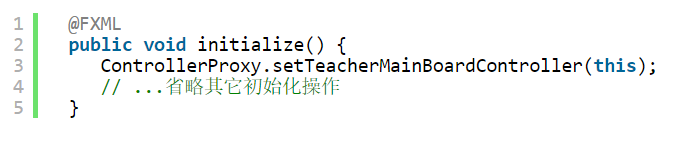
面板加载工具类的构造方法是对FXMLloader的封装，可以通过此工具类获取fxml文件对应的controller和pane，进而实现面板的初始化、动态加载等功能。其中controller必须实现FxController接口。该接口包括一个可以传入可变参数个数的init()方法，FxmlCmpLoaderUtil类的构造方法将调用FxController的init()方法。



#### 2.6.7.3 控制器代理类



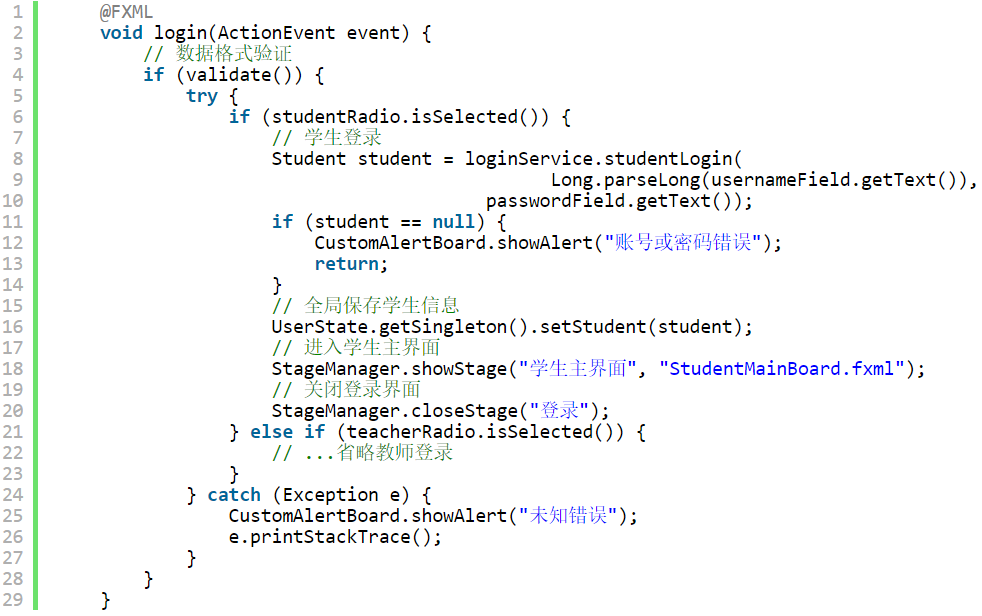
当一个控制器需要调用另一个控制器方法时，可以通过ControllerProxy来获取控制器。在本系统中，教师主面板经常需要被其它控制器调用。因此在教师主面板控制器被初始化时，调用ControllerProxy的set方法就能将该控制器注册到ControllerProxy中。



## 2.7 系统功能详细设计

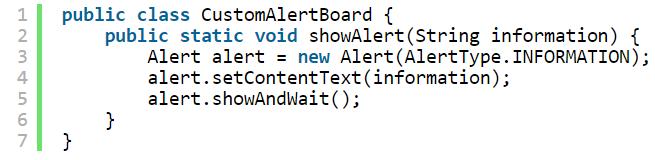
### 2.7.1 登录

先在客户端进行数据格式验证，再将账号密码发送给service。若查询到的实体类为空，则说明账号或密码错误；否则将查询到的实体类存入到全局用户状态管理类中，并进入主界面。





### 2.7.2 自定义消息弹框

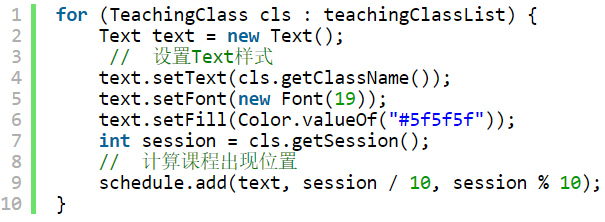


包含一个简单封装展现消息弹框的静态方法。

### 2.7.3 查看课表

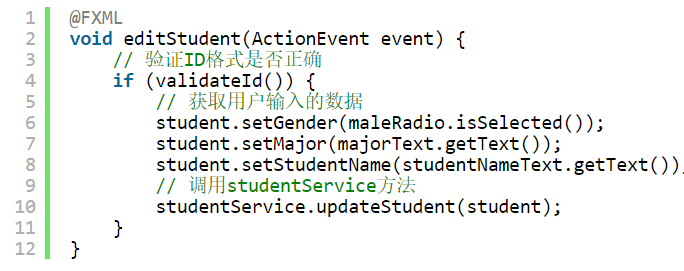


获取学生或教师的教学班列表后。根据session属性计算课程名称应该出现在GridPane的哪一个格子中。计算方法如下。



### 2.7.4 学生、教学班信息的增删改查

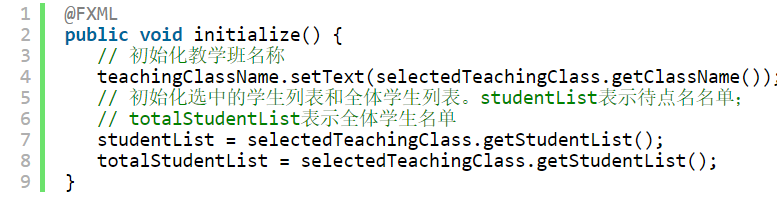
 在相应的按钮触发事件方法中获取TextField中的参数，调用相应的Service方法即可。

上图为“修改”学生按钮触发事件的方法。查询、删除、添加实现方法大致相同。

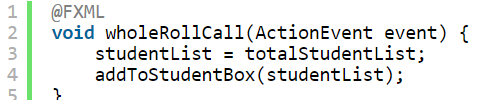
### 2.7.5 点名与提问



点名包括“顺序点名”、“异常点名”、“随机点名”、“二次点名“。点名页面初始化时，将获取选中教学班的全体学生列表。点击四种点名按钮中的一种后，程序将生成一份点名名单。教师可以方便地选择每个学生的下拉菜单，修改点名状态。“提交全部”按钮可以将所有的点名记录提交至数据库。“提交”按钮可以单独提交某个学生的点名状况。



“顺序点名“：直接令studentList = totalStudentList;



“异常点名“：调用service接口方法，查询数据库中记录的有异常点名记录的学生名单。

“二次点名“：点击“提交全部”或“提交”按钮后，程序将保存这份点名名单在内存中。点击“二次点名”按钮将上一次点名操作重新加载到列表中。

“随机点名”：根据设置的人数随机生成一份点名名单，方法如下。



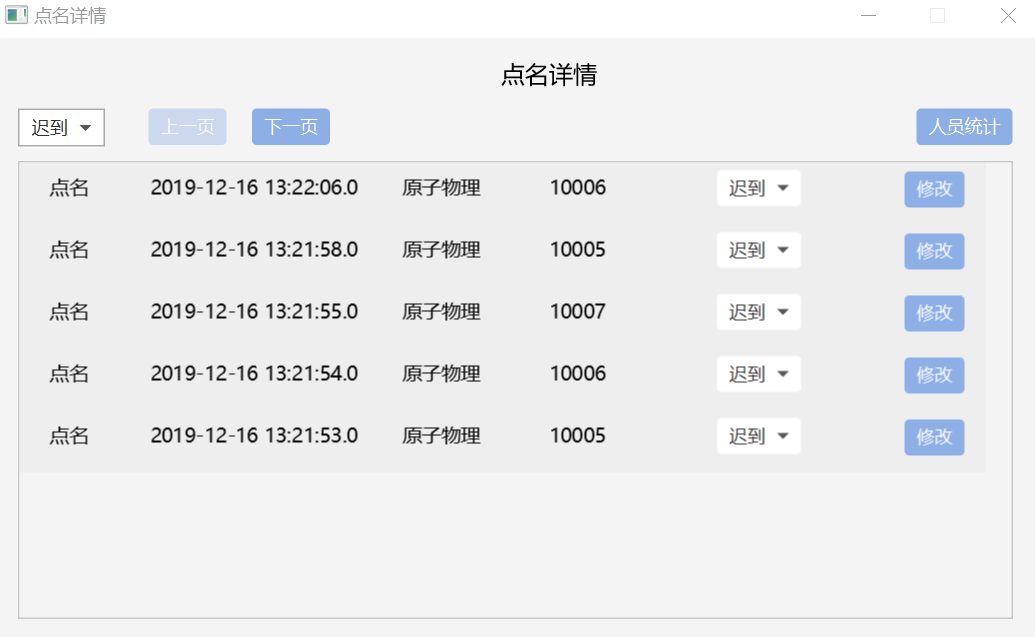
动态将单条学生点名面板添加至界面的方法如下：



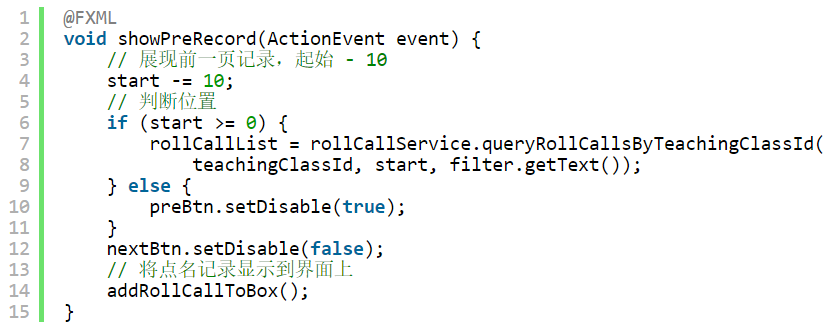
### 2.7.6 点名结果汇总

点击“提交全部”按钮后，程序将调用生成一个statisticMap<>的映射，用于统计点名情况，并显示到屏幕上。关键代码如上图所示。

### 2.7.7 点名记录的查询与修改



程序可以根据点名结果分页查询点名记录，并修改点名。分页查询通过修改起始查询位置实现。下图为展现前一页记录的方法，展现后一页记录同理。



### 2.7.8 点名结果统计及导出Excel



