摘　　要

本论文介绍了一个基于 JavaFX 的学生管理系统的设计与实现。JavaFX 作为一个现代化的 GUI 框架，为该系统提供了良好的用户界面和丰富的功能。

系统主要包括学生信息管理、成绩管理等模块。通过采用 JavaFX 的丰富组件和事件驱动编程模型，系统在功能性、可维护性和用户体验上都有较好的表现。本文详细阐述了系统的设计思路、关键技术及实现过程。

关键词：JavaFX；学生管理系统；GUI；信息管理；事件驱动编程

Electrical Design and Application (2)

Abstract

This paper introduces the design and implementation of a student management system based on JavaFX. JavaFX, as a modern GUI framework, provides a good user interface and rich features for the system.

The system mainly includes student information management, performance management and other modules. By adopting JavaFX's rich component and event-driven programming model, the system has a better performance in terms of functionality, maintainability, and user experience. In this paper, the design idea, key technology and implementation process of the system are described in detail.

Key Words： JavaFX; Student management system; GUI; Information management; Event-driven programming

目　　录

引　言

在现代教育环境中，学生管理系统作为教育机构的重要组成部分，扮演着至关重要的角色。随着信息技术的不断进步，传统的纸质管理方式逐渐被数字化、信息化的管理系统所取代。这不仅提高了管理效率，还极大地减少了人为错误，提高了数据的准确性和安全性。

JavaFX 作为一个现代化的图形用户界面（GUI）框架，因其高性能、易于使用和丰富的功能而受到广泛关注和应用。与传统的 Swing 相比，JavaFX 提供了更为灵活和强大的 UI 组件，支持更复杂的图形效果和多媒体功能，使得开发人员能够构建出用户体验更佳的应用程序。

本论文旨在介绍基于 JavaFX 的学生管理系统的设计与实现。该系统旨在通过简化学生信息的录入、查询和维护过程，提高学校管理效率。系统主要功能模块包括学生信息管理、课程管理和成绩管理等。通过使用 JavaFX，我们能够创建一个具有现代化用户界面的高效系统，并为用户提供良好的操作体验。

在本文的后续章节中，我们将详细描述系统的需求分析、设计架构、关键技术、实现过程及其功能测试和性能评估。通过对该系统的全面探讨，期望为同类系统的开发提供有价值的参考和借鉴。

JavaFX 不仅为开发人员提供了丰富的工具和组件，还支持事件驱动的编程模型，使得应用程序的响应更加灵敏和自然。在开发过程中，我们充分利用了 JavaFX 的这些优势，实现了一个功能完善、界面友好的学生管理系统。

1　项目结构介绍

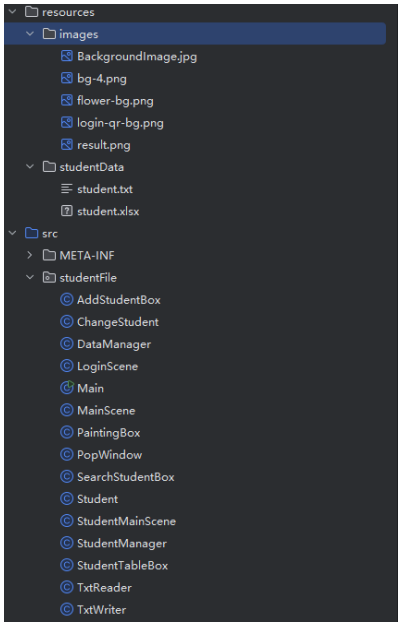


图 1.1　项目结构

本项目的文件结构如图 1.1 所示，下文将按照本管理系统的逻辑循序，依次介绍部分重要的哇类。

2 Student 学生类

2.1 学生类的属性

学生类有 String 类型的 name(姓名)，username(学号)，password(密码)，classes(班级)属性；Integer 类型的 Chinese(语文成绩)，Math(数学成绩)，English(成绩)，Physics(物理成绩)，Chemistry(化学成绩)，Biology(生物成绩) ，sumCount(总成绩)属性以及 Boolean 类型的 isManager(是否为管理员)。

2.2 学生类的方法

学生类中的构造方法为 Student(String name,String username,String password,String classes,int Chinese,int Math,int English,int Physics,int Chemistry,int Biology,int isManager)。总成绩通过各科成绩加和获得，isManager 为 1 表示为 ture，0 表示为 false。

除了 Get 和 Set 方法外，学生类还重写（Override）了 toString 方法，方便把学生导出到外部文件中。以及提供了一个 PrintStudent 方法，以便在测试时，将学生格式化打印在终端中。

3　文件读写类

为了将用户的数据保存下来，管理系统在每次运行时，必须从外部文件中读入数据；在每次结束运行时，必须将程序内存中的数据再保存回外部文件中。所以，我编写了 TxtReader 和 TxtWriter 两个类，来进行文件的读取。下文将依次进行介绍。

3.1　文件读入——TxtReader

图 3.1.1　Txt 文件内容

如图 2.1.1 所示，Txt 中保存的数据第一行为每类内容的标题，从第二行开始为每个学生的具体数据。所以在使用 BufferReader 进行读取时，要先读取第一行，将第一行的数据保存在 Java 的 Map(即字典)中，Map 采取 LinkedHashMap<>的方式，以保证存储时的线性顺序。键为 String，存储每一个标题；值为 Integer，储存第几个标题，以便后续数据的字典存储。

在之后的读入时，通过 while ((line = reader.readLine()) != null)不断读取后续行，在读取到一行后，利用 String 的 split 方法，把这一行以空格分隔开（假设为 value[]）。创建一个 Map<String,String>来储存一个学生，遍历循环第一行所创建的 Map（假设为 headerMap）的键，通过 get 获得键的值，然后就能把这个学生保存在字典中，即键对应 headerMap 的键，值对应 value[index]，index 为 get 所获得的键的值。

创建一个静态的 List<Map<String,String>>，用来存储所有的学生，每一次上述的遍历循环 headerMap，便把新建的学生 add 进 ArrayList。

3.2 字典 List 转化成 Student 的 Lsit

字典 List 还不能供系统直接使用，必须将每一个 Map 都变成一个 Student 的实例。再将每个实例添加到新的 List 当中。本系统采取 StudentManager 类来进行这步操作，并管理 StudentList。

遍历循环 MapList，将所有的值通过 Student 的构造方法创建对象。在类中定义一个静态的 List<Student>，把每次循环所创建的对象，add 进 StudentList 中。

StudentManager 中有 SearchStudent 方法，来通过学号精确查找学生，也有 FuzzySearch 方法，通过姓名模糊查找学生，以及一个 PrintStudentList 方法来格式化打印所有学生，以供测试使用。

3.3 文件写出——TxtWriter

文件首行的表头可以随便读取一个上述 ArrayList 中的一个学生字典，遍历字典中的所有键，并把它添加到 title 中，并且每 append 一个键，还要再 append 一个\t 来做分隔符。利用 BufferedWriter 写入 title，再利用 newLine 方法，将这一个分隔开。

遍历 StudentList，利用 Student 的 toString 方法，生成数据行，便可以储存到 Txt 中。

4 登录界面类

登录界面类有静态的登录按钮，账号输入框，密码输入框和这次登录的 thisStudent。以供其他地方使用。

getLoginScene 方法返回注册的 Scene，以供主函数使用。OnClickButton 方法返回 Boolean 类型的数据，判断登录是否成功，以供主函数使用。

在主函数中，当登录按钮被点击时，将调用 OnClickButton 方法，成功的话，将根据对应的 thisStudent 的权限，跳转至不同的 MainScene。

4.1 登录界面

5 主页面类

管理员的主页面类中有静态的当前学生列表 OnStudentList，各种作用的选择框和退出系统的按钮，以供其他地方使用。

GetMainScene 方法返回主页面的 Scene，以供主页面使用。SortByMethod 方法根据选择的排序科目和排序方法进行对 OnStudentList 进行排序。FlashStudentTable 来根据 OnStudentList，对主页面的列表进行刷新。

5.1 学生管理选择框

学生管理选择框中学生列表，学生增删查改以及添加删除管理员的选项。当选择学生列表和学生查询的选项时，会将 OnStudentList 更新为所选择的列表，以供 SortByMethod 和 FlashStudentList 使用。

当使用到列表时，会调用 StudentTableBox 类中的 getStudentTable 静态方法，该方法返回 TableView<Student>类型的表格，该方法需要传入 List<Student>。

用 TableColumn<Student, String> name = new TableColumn<>(key)来表格创建列，该类的 setCellValueFactory 方法会利用 Java 的反射机制，自动的将 Student 类中的属性，与之对应起来，只需要在方法中传入 new PropertyValueFactory<>(该属性在类中的名字)。

AddStudentBox 为增加学生时使用到的类，类中的 GetAddBox 方法返回增加学生的视图。ChangeStudentBox，SearchStudentBox 都与之类似。

5.2 排序选择框

排序选择框有排序科目和排序方法两个选择框，当两个选择框都有值时，才会调用 SortByMethod 方法。在使用查询学生或查看所有学生等会更新 OnStudentList 的功能时，也会根据排序选择框的值，进行排序操作。

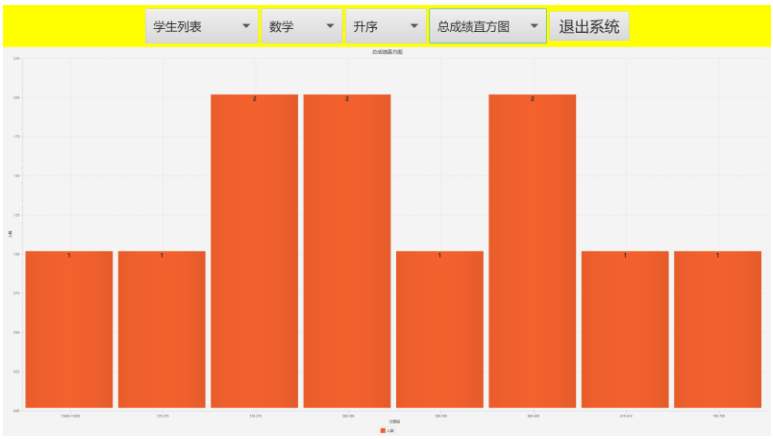
5.3 图标选择框

图标选择框在选择之后，会调用 PaintingBox 类中的 createHistogram 方法，该方法会返回 BarChart<String, Number>类型的图表，供 MainScene 使用。该方法需要 List<Integer>类型的参数，即画直方图所依赖的数据，String 类型的参数，即该直方图的名称。

5.4 非管理员的主页面

非管理员用户只有修改密码和成绩查询两个功能，实现方法都可以在管理员的 MainScene 中找到参考。

5.1 主页面列表



5.2 总成绩直方图

5.3 非管理员用户界面

结　　论

通过本文的研究与实现，我们成功地设计并开发了一个基于 JavaFX 的学生管理系统。该系统包括学生信息管理、课程管理和成绩管理等核心模块，提供了友好的用户界面和丰富的功能，有效提高了学校的管理效率和信息处理能力。

在系统开发过程中，JavaFX 的现代化 GUI 框架发挥了重要作用。其丰富的组件库和灵活的布局机制使得我们能够快速构建美观且功能强大的用户界面。事件驱动的编程模型提高了系统的响应速度和用户体验，使得系统在实际应用中表现出色。

通过系统测试和用户反馈，我们验证了该系统的稳定性和实用性。用户对系统的界面设计和功能实现给予了高度评价，认为系统操作简单直观，极大地方便了日常的学生信息管理工作。此外，系统的扩展性和可维护性也得到了验证，为后续功能的增加和系统的持续改进奠定了基础。

总的来说，本系统在功能性、用户体验和技术实现方面都达到了预期目标。基于 JavaFX 的学生管理系统不仅提高了教育管理的效率，还为未来类似系统的开发提供了宝贵经验。未来的工作中，可以进一步优化系统性能，增加更多智能化功能，如数据分析和预测，以更好地满足现代教育管理的需求。

本研究展示了 JavaFX 在开发现代化学生管理系统中的强大潜力和优势，期望本文的成果能够为相关领域的研究人员和开发者提供有益的参考和启示。

附录 A　部分代码展示

```
1. package studentFile;
2.
3. import javafx.application.Application;
4. import javafx.application.Platform;
5. import javafx.scene.Scene;
6. import javafx.stage.Stage;
7.
8. import java.io.IOException;
9.
10.
11. public class Main extends Application {
12.     @Override
13.     public void start(Stage stage) throws Exception {
14.         //舞台
15.         stage.setTitle(DataManager.StageTitle);
16.         stage.setFullScreen(true);
17.
18.         //注册界面
19.         Scene loginScene = LoginScene.getLoginScene();
20.
21.         stage.setScene(loginScene);
22.
23.         //跳转主界面
24.         if(LoginScene.loginButton != null) {
25.             LoginScene.loginButton.setOnAction(actionEvent -> {
26.                 if(LoginScene.OnClickButton()) {
27.                     Scene mainScene;
28.                     if(LoginScene.thisStudent.isManager) {
29.                         mainScene = MainScene.GetMainScene();
30.                     }
31.                     else {
32.                         mainScene = StudentMainScene.GetMainScene();
33.                     }
34.                     stage.setScene(mainScene);
35.                     stage.setFullScreen(true);
36.                 }
37.             });
38.         }
39.
40.         //退出系统
41.         MainScene.exitButton.setOnAction(actionEvent -> {
42.             try {
43.                 TxtWriter.WriterTxt(DataManager.txtPath);
44.                 Platform.exit();
45.             } catch (IOException e) {
46.                 throw new RuntimeException(e);
47.             }
48.         });
49.
50.         //退出系统
51.         StudentMainScene.exitButton.setOnAction(actionEvent -> {
52.             try {
53.                 TxtWriter.WriterTxt(DataManager.txtPath);
```

```
                Platform.exit();
            } catch (IOException e) {
                throw new RuntimeException(e);
            }
        });

        stage.show();
    }

    public static void main(String[] args) {
        Application.launch();
    }
}
```

Main 类

```
package studentFile;

import javafx.geometry.Insets;
import javafx.geometry.Pos;
import javafx.scene.Scene;
import javafx.scene.control.*;
import javafx.scene.image.Image;
import javafx.scene.layout.*;
import javafx.scene.paint.Color;
import javafx.scene.text.Font;
import javafx.scene.text.FontWeight;
import javafx.scene.text.Text;

import java.io.FileInputStream;
import java.io.FileNotFoundException;

public class LoginScene {

    public static Button loginButton = new Button("登录");
    public static TextField username = new TextField("请输入账号");
    public static PasswordField password = new PasswordField();
    public static Student thisStudent;

    static public Scene getLoginScene() throws FileNotFoundException {
        GridPane loginGridPane = new GridPane();

        String backgroundPath = "resources/images/bg-4.png";
        Background background = getBackground(backgroundPath,DataManager.StageWidth,DataManager.StageHeight);
        loginGridPane.setBackground(background);
        loginGridPane.setHgap((float)DataManager.StageWidth/5); // 设置行间距
        loginGridPane.setVgap((float)DataManager.StageHeight/20); // 设置列间距


        Text titleText = new Text("统一登陆界面");
        titleText.setFont(Font.font("楷体", FontWeight.BOLD,155));
        titleText.setFill(Color.PINK);
        HBox titleHBox = new HBox();
        titleHBox.setPadding(new Insets(30,0,0,30)); //上右下左
        titleHBox.getChildren().add(titleText);
```

```
40.
41.         //账号
42.         Label usernameLabel = new Label("账号");
43.         HBox usernameHBox = new HBox(20);
44.         usernameHBox.getChildren().addAll(usernameLabel,username);
45.         usernameLabel.setFont(new Font("楷体", 50));
46.         usernameLabel.setTextFill(Color.BLUE);
47.         username.setFont(new Font("楷体", 50));
48.         usernameHBox.setAlignment(Pos.CENTER);
49.
50.         //密码
51.         Label passwordLabel = new Label("密码");
52.         HBox passwordHBox = new HBox(20);
53.         passwordHBox.getChildren().addAll(passwordLabel,password);
54.         passwordLabel.setFont(new Font("楷体", 50));
55.         passwordLabel.setTextFill(Color.BLUE);
56.         password.setFont(new Font("楷体", 50));
57.         passwordHBox.setAlignment(Pos.CENTER);
58.
59.         //按钮
60.         loginButton.setFont(new Font("楷体", 50));
61.
62.         VBox vBox = new VBox(20);
63.         vBox.getChildren().addAll(usernameHBox,passwordHBox,loginButton);
64.         vBox.setAlignment(Pos.CENTER);
65.         vBox.setMinWidth(DataManager.StageWidth/3);
66.         vBox.setMinHeight(DataManager.StageHeight*2/3);
67.         Background vBoxBG = getBackground("resources/images/result.png", vBox
    .getWidth(), vBox.getHeight());
68.         vBox.setBackground(vBoxBG);
69.
70.         loginGridPane.add(titleHBox,0,0);
71.         loginGridPane.add(vBox,     1,1);
72.
73.         return new Scene(loginGridPane);
74.     }
75.
76.     static public boolean OnClickButton(){
77.         String thisUsername = username.getText();
78.         String thisPassword = password.getText();
79.
80.         Student student = DataManager.studentManager.SearchStudent(thisUserna
    me);
81.         if(student==null){
82.             PopWindow.showAlert(DataManager.LoginErrorTitle,DataManager.Login
    ErrorContent);
83.             return false;
84.         }
85.         if(thisPassword.equals(student.password)){
86.             thisStudent = student;
87.             return true;
88.         }
89.         PopWindow.showAlert(DataManager.LoginErrorTitle,DataManager.LoginErro
    rContent);
90.         return false;
91.     }
92.
```

```
93.     private static Background getBackground(String path,double width,double h
    eight) throws FileNotFoundException {
94.         FileInputStream input = new FileInputStream(path);
95.         Image image = new Image(input);
96.         BackgroundSize backgroundSize = new BackgroundSize(width,height,
97.                 false,false,true,true);
98.         BackgroundImage backgroundImage = new BackgroundImage(image, Backgrou
    ndRepeat.NO_REPEAT,
99.                 BackgroundRepeat.NO_REPEAT, BackgroundPosition.DEFAULT, backg
    roundSize);
100.            return new Background(backgroundImage);
101.        }
102.    }
```

LoginScene 类

```
1. package studentFile;
2.
3. import java.io.IOException;
4. import java.util.ArrayList;
5. import java.util.List;
6. import java.util.Map;
7.
8. public class StudentManager {
9.     public List<Student> StudentList = new ArrayList<>();
10.
11.     StudentManager() throws IOException {
12.
13.         TxtReader.ReadTxt(DataManager.txtPath);
14.
15.         for(Map<String, String> record:TxtReader.studentData){
16.             String name = record.get("姓名");
17.             String username = record.get("学号");
18.             String password = record.get("密码");
19.             String classes = record.get("班级");
20.             int Chinese = Integer.parseInt(record.get("语文"));
21.             int Math = Integer.parseInt(record.get("数学"));
22.             int English = Integer.parseInt(record.get("英语"));
23.             int Physics = Integer.parseInt(record.get("物理"));
24.             int Chemistry = Integer.parseInt(record.get("化学"));
25.             int Biology = Integer.parseInt(record.get("生物"));
26.             int isManager = Integer.parseInt(record.get("管理员权限"));
27.             Student tempStudent = new Student(name,username,password,classes,
    Chinese,
28.                     Math,English,Physics,Chemistry,Biology,isManager);
29.             StudentList.add(tempStudent);
30.         }
31.     }
32.
33.     public Student SearchStudent(String id){
34.         for(Student student:StudentList){
35.             if(id.equals(student.username)){
36.                 return student;
37.             }
38.         }
39.         return null;
40.     }
41.
```

```
    public List<Student> FuzzySearch(String keyword){
        List<Student> studentList = new ArrayList<>();
        for(Student student:StudentList){
            int targetLength = student.name.length();
            int keywordLength = keyword.length();

            for (int i = 0; i <= targetLength - keywordLength; i++) {
                String substring = student.name.substring(i, i + keywordLengt
h);
                if (substring.equalsIgnoreCase(keyword)) {
                    studentList.add(student);
                }
            }
        }

        return studentList;
    }

    public void PrintStudentList(){
        for(Student student:StudentList){
            student.PrintStudent();;
        }
    }
}
```

StudentManager 类

```
package studentFile;

import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
import java.util.*;

public class TxtReader {
    public static List<Map<String,String>> studentData = new ArrayList<>();

    public static void ReadTxt(String txtPath) throws IOException {
        try {
            FileReader fileReader = new FileReader(txtPath);
            BufferedReader reader = new BufferedReader(fileReader);

            // 读取第一行（标签行）
            String headerLine = reader.readLine();
            if (headerLine == null) {
                System.out.println("文件是空的");
                return;
            }

            // 分割标签行，假设标签之间以空格分隔
            String[] headers = headerLine.split("\\s+");
            Map<String, Integer> headerMap = new LinkedHashMap<>();
            for (int i = 0; i < headers.length; i++) {
                headerMap.put(headers[i], i);
            }
            System.out.println(headerMap.toString());

            // 读取后续行数据
```

```
            String line;
            while ((line = reader.readLine()) != null) {

                // 分割每行的数据
                String[] values = line.split("\\s+");

                Map<String, String> subMap = new LinkedHashMap<>(); //每一个学生
                //for each 遍历字典
                for (String header : headerMap.keySet()) {

                    //通过 get 方法来用 key 获取到 value
                    int index = headerMap.get(header);

                    if (index < values.length) {
                        subMap.put(header, values[index]);
                    }
                    else {
                        subMap.put(header, ""); // 如果该行数据不足，使用空字符串填充
                    }
                }
                studentData.add(subMap);
            }

            reader.close();
            fileReader.close();
        }
        catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
}
```

TxtReader 类