**Java程序设计**

**课程设计说明书**

**（2020～2021学年第一学期）**

**题目名称 移动微博系统**

**姓 名 谢晓艳**

**学 号 190611243**

**专 业 物联网工程**

**班 级 物联19102班**

**指导教师 李剑波、丁德红**

**提交日期 2020.12.20**

**湖南应用技术学院课程设计任务书**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 课程名称 | java程序设计课程设计 | | | | 课程代码 | | 190809050406 |
| 设计题目 | 移动微博系统 | | | | 分组人数 | | 1人 |
| 设计时间 | 2020年12月14日—12月18日 | | | | | | |
| 学院 | 信息工程学院 | 专业 | 物联网工程 | 班级 | | 物联19102 | |
| 一、课程设计任务（条件）、开发工具及运行环境  1、设计任务   1. 进行数据库设计，并创建相关的表 2. 实现登录注册界面 3. 实现打开音乐文件、打开音乐文件夹、打开文本、另存为文本等菜单栏功能 4. 实现音乐播放功能 5. 实现用户编辑信息功能 6. 实现发布微博内容功能 7. 实现搜索用户功能 8. 实现添加关注用户功能 9. 实现取消关注用户功能 10. 实现获取关注的人的微博内容功能 11. 实现注销用户功能 12. 测试   2、开发工具及运行环境   1. 开发工具：Eclipse，JDK1.8.0，MySQL 2. 运行环境：Windows10 | | | | | | | |
| 二、对课程设计成果的要求（包括课程设计说明书、图纸、图表、实物等软硬件要求）  1、设计说明书要求  内容完整，概念清楚，方案合理，数据可靠，图文（表）规范、排版工整。  2、设计系统的要求   1. 实现全部功能 2. 执行效率高 3. 界面友好、交互性强   3、编码的要求   1. 声明   成员声明变量：在类的第一个方法之前声明变量  变量声明：代码块开始处声明变量  类和接口声明：在类定义、接口定义或方法定义中，“{”在声明部分的最右侧，“}”独占一行，需要与相应的声明语句对齐。若为空语句，“}”应紧跟在“{”之后   1. 命名规范   类和接口命名：采用驼峰式命名规则，要求第一个单词首字母大写  方法命名：采用驼峰式命名规则，要求第一个单词的首字母小写  变量命名：每个字母全部小写，可用下划线隔开  常量命名：每个字母全部大写，可用下划线隔开   1. 缩进   代码以1个Tab键或4个空格缩进 | | | | | | | |
| 三、课程设计工作进度计划：  2020年12月12日-13日：完成选题。  2020年12月14日：完成《课程设计说明书》的封面、表格内容填写；完成需求分析、概要设计。填写《课程设计说明书》的“一、需求分析”、“二、概要设计”内容的填写。  2020年12月15日：详细设计。绘制模块功能表；确定每个模块的算法、流程图、和模块中各数据说明表；确定各模块的GUI界面。  2020年12月15-16日：代码编写。  2020年12月17日：系统测试；完成《课程设计说明书》中“三、详细设计”内容。  2020年12月17日：完善程序和课程设计说明书、答辩。  2020年12月18日：根据答辩过程指导的意见，对系统和《课程设计说明书》进行修改，提交项目源代码和《课程设计说明书》（.doc文件） | | | | | | | |
| 四、主要参考资料  [1]杨潇,陈秀真,马进,梁浩喆,李生红.基于用户兴趣的微博溯源算法[J].网络与信息安全学报,2020,6(06):164-173.  [2]张文丰.大学生初学Java多线程思路的反思与优化[J].中国新通信,2020,22(19):223-224.  [3]杨洋. 新浪微博测试平台的设计与实现[D].北京交通大学,2018.  [4]王志文,陈珂,陈晓升,郑远飞,陈坚旋.基于MVC设计模式的微博数据采集框架[J].广东石油化工学院学报,2017,27(01):31-36.  [5]叶婷. 基于关键词的微博爬虫系统的设计与实现[D].浙江工业大学,2016.  [6]Gunawan Gunawan,Armin Lawi,Adnan Adnan. Analisis Arsitektur Aplikasi Web Menggunakan Model View Controller (MVC) pada Framework Java Server Faces[J]. Scientific Journal of Informatics,2016,3(1). | | | | | | | |

指导教师（签名）： 教研室主任（签名）：

年 月 日 年 月 日

**目录**

[一、 需求分析 1](#_Toc31108)

[1.1 目标 1](#_Toc16906)

[1.2 用例图 1](#_Toc22093)

[二、 总体设计 2](#_Toc5555)

[2.1 功能模块描述 2](#_Toc20963)

[2.2 系统流程描述 3](#_Toc8537)

[2.3 系统开发目录 3](#_Toc3410)

[三、 详细设计 4](#_Toc18181)

[3.1 模块分析 4](#_Toc15107)

[3.2 数据库设计 4](#_Toc28148)

[3.3 模型层设计 6](#_Toc4558)

[3.4 控制层设计 9](#_Toc29673)

[3.5 视图层设计 13](#_Toc6544)

[四、 程序运行结果测试与分析 16](#_Toc26088)

[4.1 登录注册界面测试 16](#_Toc9579)

[4.2 管理员界面测试 17](#_Toc9201)

[4.3 访客界面测试 18](#_Toc12577)

[五、 结论与心得 22](#_Toc28536)

[六、 附：代码 23](#_Toc16937)

**课程设计说明书**

# 需求分析

## 目标

随着物质水平的提高，人们开始更加注重精神的享受。博客需要组织语言陈述事实或者采取修辞手法来表达心情，而微博只言片语“语录体”的即时表述更加符合现代人的生活节奏和习惯。微博系统要求操作界面美观大方、功能齐全、具有前台和后台管理。

微博系统是一套功能齐全而又简便使用的软件。用户可分为管理员和访客，访客通过注册建立账户，而管理员可对访客进行修改、删除、查看等操作。经分析，可认为微博系统的具体要求为：

能管理微博系统相关的各类主体，如个人、访客、用户关系、微博内容等。

1. 通过使用计算机能方便的维护（增、删、改、查）各信息表。
2. 微博内容的浏览，数据库表设计
3. 用户社交体现：关注用户、取关用户
4. 拉取关注的人的微博内容
5. 系统具有操作方便、简便等特点

## 用例图

访客用例图，如图1.1所示；管理员用例图，如图1.2所示

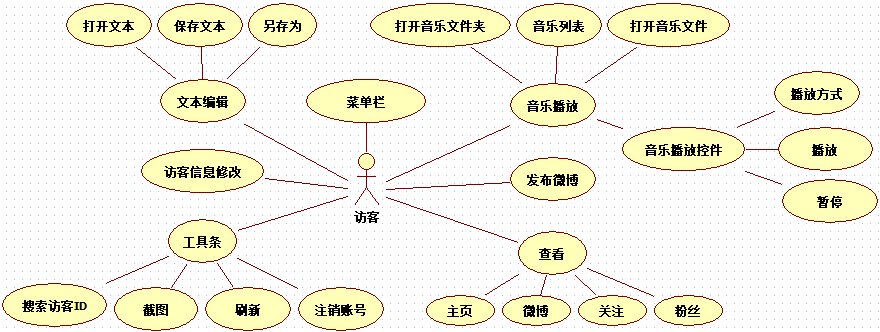


图 1.1访客用例图

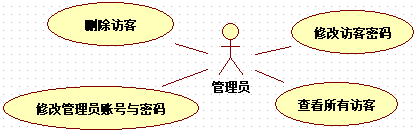


图 1.2管理员用例图

# 总体设计

## 功能模块描述

“桌面微博系统”设置有两级用户：管理员、访客。用户级别不同，其管理权限就不同，显示的界面也不同。

登录模块：可进行访客注册与登录。输入正确的账号和密码后，系统验证输入信息，验证正确后启动与用户角色相对应的主界面。如果验证不正确，则根据不同的错误给出错误信息。

访客模块：包括工具条、菜单条、个人信息面板、编辑发布微博面板、主功能面板、音乐播放控件面板六个子模块。

管理员模块：包括删除访客、修改访客、查看所有访客、修改管理员账号和密码四个子模块。

系统功能结构图，如图2.1所示

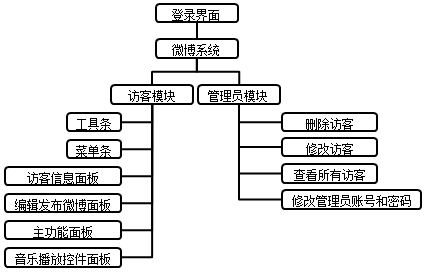


图 2.1系统功能结构图

## 系统流程描述

用户启动系统，将进入系统“登录界面”。在“登录界面”输入账号和密码，单击“登录”按钮后，将进行用户账号和密码的验证工作。通过验证后，根据用户的启动权限启动相应的系统主界面。

系统流程图，如图2.2所示

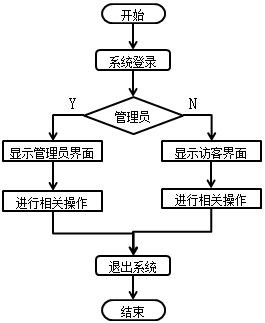


图 2.2系统流程图

## 系统项目目录系统开发目录

1. 项目开发目录

src：源代码文件

images：图片文件

libs：外部类库文件

database：创建数据表的.sql文件、测试文件夹、运行截图的exe文件

1. 包

control：控制包

data：数据包

model：模型包

test：测试包

tools：工具包

图 2.3系统开发目录图

view：通用界面（登录界面、注册界面）

view\_administrator：管理员界面

view\_visitor：访客界面

1. 系统开发目录，如图2.3所示

# 详细设计

## 模块分析

登录模块：根据不同用户的权限，进入不同的界面。

注册模块：通过在访客表中增加一条记录进行访客注册。

管理员模块：对访客进行删除、修改访客密码、查看所有访客、修改管理员账号和密码。

访客界面的菜单栏模块：包括打开音乐文件、打开音乐文件夹、打开文本、保存文本、另存为等功能。

访客界面的工具栏模块：包括搜索访客、运行屏幕截图exe文件、刷新界面、注销访客账号等功能。

访客界面的个人信息面板模块：包括个人信息编辑功能、显示音乐列表功能，、示访客的部分信息等功能。

访客界面的编辑发布微博面板模块：包括清空文本域、发布微博、在文本域右键点击等功能。

访客界面的查看面板模块：使用选项卡面板，主页显示所有关注的访客的微博，微博显示自己写的微博，关注显示所有关注的访客信息，粉丝显示所有粉丝的访客信息。

访客界面的音乐控件面板模块：包括一个显示信息的文本框、暂停音乐的按钮、播放音乐的按钮、循环播放的按钮。

## 数据库设计

1. 数据库实体

数据的实体有个人personal、访客visitor、微博weibo、关注粉丝attentionfans。

数据库实体的E-R图，如图3.1所示

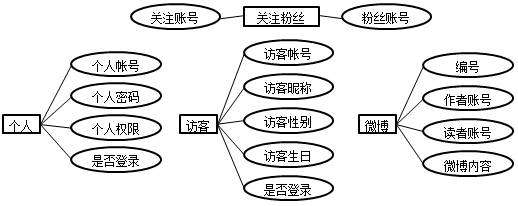


图 3.1数据库实体的E-R图

1. 数据表的设计与实现

“微博系统”涉及四个数据表，分别为：

personal表：个人信息表，如表3-1所示

visitor表：访客信息表，如表3-2所示

weibo表： 微博信息表，如表3-3所示

attentionfans表：关注粉丝信息表，如表1-4所示

表 3-1个人信息表（personal）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **说明** |
| **id（主键）** | varchar | 50 | 个人账号 |
| **password** | varchar | 50 | 个人密码 |
| **power** | varchar | 5 | 个人权限（访客、管理员） |
| **login\_success** | varchar | 5 | 是否登录 |

表 3-2访客信息表（visitor）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **说明** |
| **id（主键）** | varchar | 50 | 访客账号 |
| **visitor\_name** | varchar | 50 | 访客昵称 |
| **visitor\_sex** | varchar | 50 | 访客性别 |
| **visitor\_birthday** | varchar | 50 | 访客生日 |
| **visitor\_yes\_no** | varchar | 20 | 是否登录 |

表 3-3微博信息表（weibo）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **说明** |
| **weibo\_id（主键）** | varchar | 50 | 微博编号 |
| **writer\_id** | varchar | 50 | 作者账号 |
| **reader\_id** | varchar | 50 | 读者账号 |
| **weibo\_content** | varchar | 1000 | 微博内容 |
| **weibo\_id** | varchar | 50 | 微博编号 |

表 3-4关注粉丝信息表（attentionfans）

|  |  |  |  |
| --- | --- | --- | --- |
| **字段名** | **数据类型** | **长度** | **说明** |
| **attention\_id（主键）** | varchar | 50 | 关注者账号 |
| **fans\_id（主键）** | varchar | 50 | 粉丝账号 |

## 模型层设计

1. 模型层设计概述

控制层包括一个包，即model包。

model包：共5个类，包括Personal类、Visitor类、Weibo类、AttentionFans类、Music类。在这些类中为了数据的安全性，将属性的访问权限设计为私有的（private）；为了方便属性值的设置和获取，将setter和getter方法的访问权限设计为公有的（public）。

1. Personal类的设计

Personal类用于保存、设置、获取个人的属性。

Personal类由4个成员变量、8个成员方法、3个构造方法构成。Personal类的UML图，如图3.2所示

成员变量：共4个，用于描述个人表的属性，即个人信息。

成员方法：共8个，其中4个Setter方法和4个Getter方法，Setter方法用于设置个人信息，Getter方法用于获取个人信息。

构造方法：共2个，其中有一个为默认构造方法，另一个是通过字段id和字段password生成的构造方法。

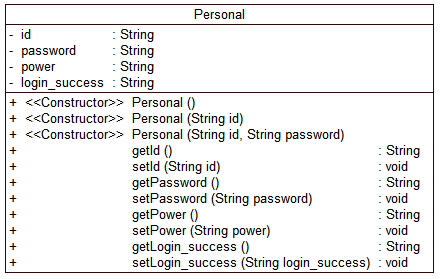


图3.2 Personal类的UML图

1. Visitor类的设计

Visitor类用于保存、设置、获取访客的属性。

Visitor类由5个成员变量、10个成员方法、2个构造方法构成。Visitor类的UML图，如图3.3所示

成员变量：共5个，用于描述访客表的属性，即访客信息。

成员方法：共10个，其中5个Setter方法和5个Getter方法，Setter方法用于设置访客信息，Getter方法用于获取访客信息。

构造方法：共2个，其中有一个为默认构造方法，另一个是通过字段id生成的构造方法。

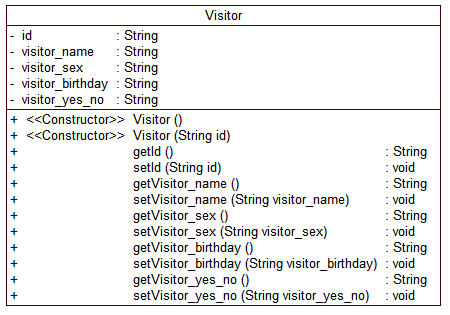


图3.3 Visitor类的UML图

1. Weibo类的设计

Weibo类用于保存、设置、获取微博的属性。

Weibo类由5个成员变量、10个成员方法构成。Weibo类的UML图，如图3.4所示

成员变量：共5个，用于描述微博内容表的属性，即微博信息。

成员方法：共10个，其中5个Setter方法和5个Getter方法，Setter方法用于设置微博信息，Getter方法用于获取微博信息。

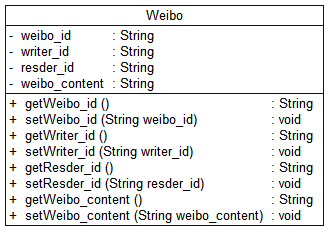


图3.4 Weibo类的UML图

1. AttentionFans类的设计

AttentionFans类用于保存、设置、获取关注粉丝的属性。

AttentionFans类由2个成员变量、4个成员方法构成。AttentionFans类的UML图，如图3.5所示

成员变量：共2个，用于描述关注粉丝表的属性，即关注粉丝信息。

成员方法：共4个，其中2个Setter方法和2个Getter方法，Setter方法用于设置关注粉丝信息，Getter方法用于获取关注粉丝信息。

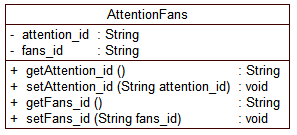


图3.5 AttentionFans类的UML图

1. Music类的设计

Music类用于保存、设置、获取音乐文件的属性。

Music类由5个成员变量、10个成员方法构成。Music类的UML图，如图3.6所示

成员变量：共5个，用于描述音乐文件的属性，即音乐文件信息。

成员方法：共10个，其中5个Setter方法和5个Getter方法，Setter方法用于设置音乐文件信息，Getter方法用于获取音乐文件信息。

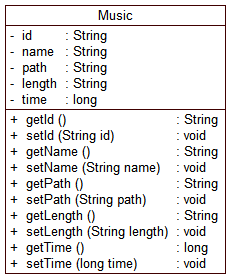


图3.6 Music类的UML图

## 控制层设计

1. 控制层设计概述

控制层包括两个包，即data包、control包。

data包：共4个类，包括MyGetConnection类、Constant类、GlobalVar类、Refresh类。MyGetConnection类的构造方法是进行加载驱动和建立数据库的连接，Constant类存放常量，GlobalVar类存放全局变量，Refresh类进行界面刷新操作。

control包：共6个类，包括Insert类、Delete类、Updata类、Seletc类、ResultSetNum类、LoginRegister类，这些类的构造方法都加载了数据库驱动，连接了数据库。Insert类中主要是对数据库中相关表中的记录进行增加操作的方法，Delete中主要是对数据库中相关表中的记录进行删除操作的方法，Updata类中主要是对数据库中相关表中的记录进行修改操作的方法，Seletc类中主要是对数据库中相关表中的记录进行查找操作的方法，ResultSetNum类中主要是对数据库中相关表中的记录进行统计记录数操作的方法，LoginRegister类主要是进行登录注册操作，即连接数据库查询是否存在相关记录和连接数据库增加相关记录的方法。

1. data包的设计

data包中类的UML图，如图3.7所示

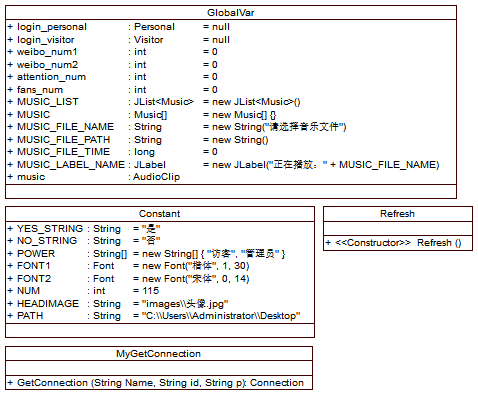


图3.7 data包中类的UML图

1. Insert类的设计与实现

在Inset类中void insert(Personal personal)、void insert(Visitor visitor)、void insert(Weibo weibo)、boolean insert(AttentionFans attentionFans)方法都是需要连接数据库，且设计思路和算法相似，流程图相似，这里不再重复绘制。流程图，如图3.8所示。

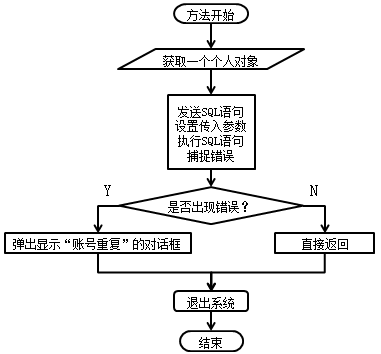


图3.8 void insert(Personal personal)方法的流程图

在Inset类中String[] insert(String[] arr, String... str)、Music[] insert(Music[] musicArr, Music... musics)、Weibo[] insert(Weibo[] weiboArr, Weibo... weibos)、Visitor[] insert(Visitor[] visitorArr, Visitor... visitors)方法都是静态方法，都是用于在不同的数组中追加数据，且设计思路和算法相似。

Insert类的UML图，如图3.9所示

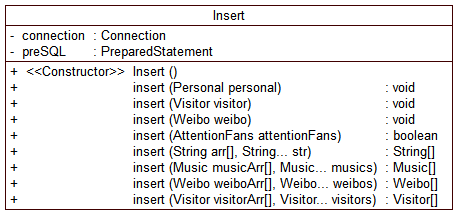


图3.9 Insert类的UML图

1. Delete类的设计与实现

在Delete类中的方法都是需要连接数据库，且与insert类中的方法设计思路和算法相似，流程图相似，这里不再重复绘制。

Delete类的UML图，如图3.10所示

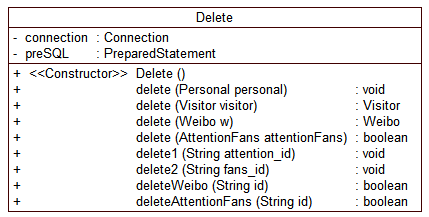


图3.10 Delete类的UML图

1. Update类的设计与实现

在Update类中的方法都是需要连接数据库，且与insert类中的方法设计思路和算法相似，流程图相似，这里不再重复绘制。

Update类的UML图，如图3.11所示

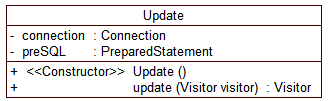


图3.11 Update类的UML图

1. Seletc类的设计与实现

在Seletc类中的方法都是需要连接数据库，且与insert类中的方法设计思路和算法相似，流程图相似，这里不再重复绘制。

Select类的UML图，如图3.12所示

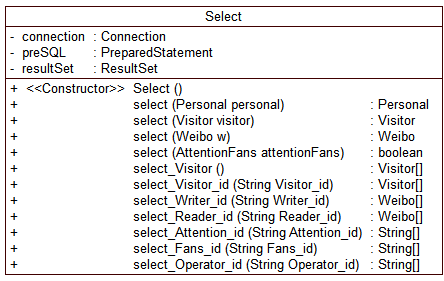


图3.12 Select类的UML图

1. ResultSetNum类的设计与实现

在ResultSetNum类中的方法都是需要连接数据库，且与insert类中的方法设计思路和算法相似，流程图相似，这里不再重复绘制。

ResultSetNum类的UML图，如图3.13所示

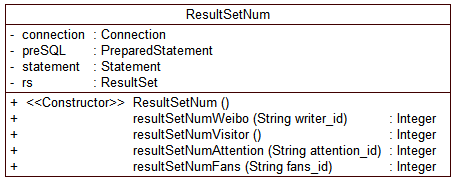


图3.13 ResultSetNum类的UML图

1. LoginRegister类的设计与实现

Personal register(String id, String password1)方法的说明：

通过Select类的select(Personal personal)和select(Visitor visitor)方法在数据库查找是否存在相同的个人对象和访客对象，如果存在则弹出显示“账号重复”的对话框；如果不存在则通过Insert类的void insert(Personal personal)和insert(Visitor visitor)在数据库增加相关记录。同时为了在主页面板能够显示自己写的微博，可以在注册访客账号时，让自己关注自己，即通过Insert类的boolean insert(AttentionFans attentionFans)方法在数据库增加一条记录。

Personal login(String id, String password)方法的说明：

首先在数据库查找是否存在相同的个人对象，如果不存在则弹出显示“登录失败，请重新登录”的对话框；如果存在则判断个人对象的权限，是管理员则进入管理员界面，是访客则进入访客界面，并将对一些全局变量进行赋值或创建对象。

LoginRegister类的UML图，如图3.14所示

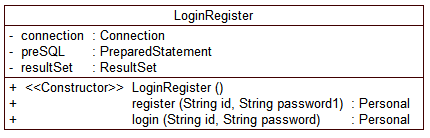


图3.14 LoginRegister类的UML图

## 视图层设计

1. view包的设计

view包：共2个类，包括Interface\_Login类、Interface\_Register类。Interface\_Login类是一个登录界面，Interface\_Register类是一个注册界面。

Interface\_Login类的UML图，如图3.15所示

Interface\_Register类的UML图，如图3.16所示

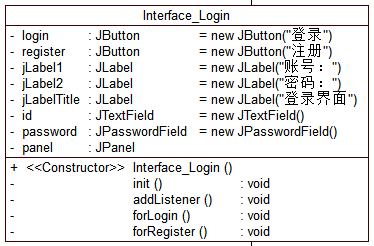


图3.15 Interface\_Login类的UML图

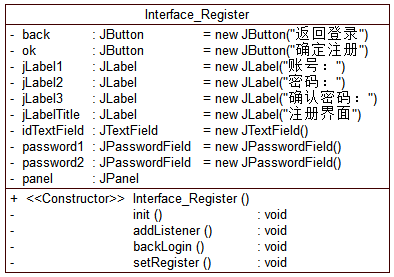


图3.16 Interface\_Register类的UML图

1. view\_administrator包的设计

view\_administrator包：共5个类，包括Interface\_MainFrame类、PanelAdministrator类、PanelDelete类、PanelShow类、PanelUpdate类。Interface\_MainFrame类为管理员主界面，继承了JFarm，设置了一个分割面板，在分割面板的左面板上存在四个按钮，分别是删除访客、修改访客、查看访客、修改账号密码，这四个按钮可对右面板进行切换。PanelAdministrator类是修改管理员账号和密码面板，PanelDelete类是通过搜索访客id删除访客面板，PanelShow类是显示所有访客信息米面板，PanelUpdate类是通过搜索访客id删除访客信息面板。

管理员界面设计图，如图3.17所示

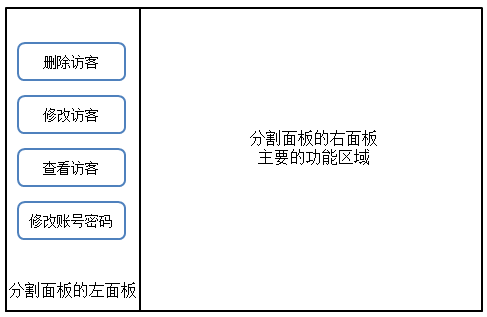


图3.17管理员界面设计图

1. view\_visitor包的设计

view\_visitor包：共8个类，包括Interface\_MainFrame类、Panel1\_PersonalInformation类、Panel2\_EditorialPublished类、Panel3\_MainFunction类、Panel4\_PlaybackControls类、PanelTXT类、SetupJMenuBar类、ShowFram类。Interface\_MainFrame类为访客主界面，继承了JFarm，设置了三个分割面板。第一个分割面板设置为垂直方向，上面是个人信息显示面板通过Panel1\_PersonalInformation类实现，下面是编辑发布微博面板通过Panel2\_EditorialPublished类实现。第一个分割面板设置为垂直方向，上面是个人信息显示面板通过Panel1\_PersonalInformation类实现，下面是编辑发布微博面板通过Panel2\_EditorialPublished类实现。第二个分割面板设置为垂直方向，上面是主功能面板通过Panel3\_MainFunction类实现，下面是音乐控件面板通过Panel4\_PlaybackControls类实现。第三个分割面板设置为水平方向，上面是主功能面板通过Panel3\_MainFunction类实现，下面是音乐控件面板通过Panel4\_PlaybackControls类实现。PanelTXT类的功能是在主功能面板显示一条条微博，SetupJMenuBar类的功能是设置菜单条，ShowFram类的功能是搜索访客id后打开查看该访客的信息窗口。

访客界面设计图，如图3.18所示

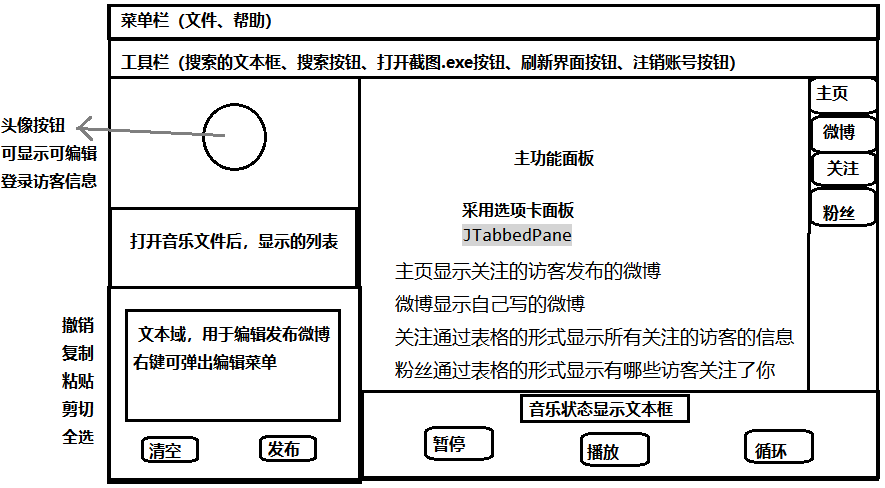


图3.18访客界面设计图

# 程序运行结果测试与分析

## 登录注册界面测试

在登录界面输入账号和密码后，点击登录后会根据账号和密码在数据库的personal表查找个人信息，用从数据库中查找到的个人信息对全局变量类（GlobalVar）中的当前登录个人对象（login\_personal）。点击注册按钮将进入注册界面。注册界面输入账号和密码后，点击注册后会根据账号和密码在数据库的personal表添加个人信息，在visitor表添加访客信息，在attentionfans表添加关注粉丝信息（自己关注自己）。

难点：数据库连接

登录界面，如图4.1所示

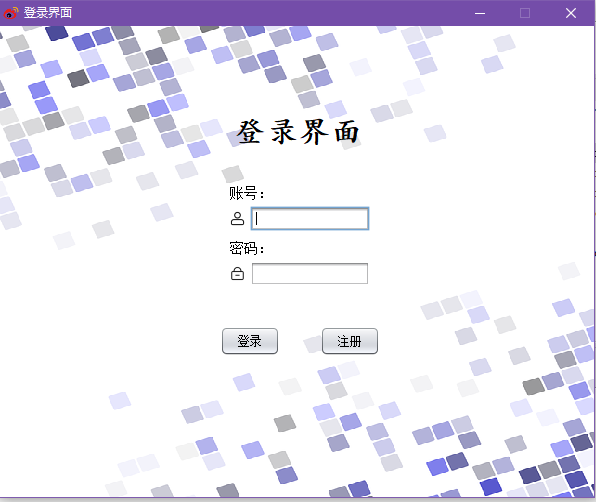


图4.1登录界面图

注册界面，如图4.2所示

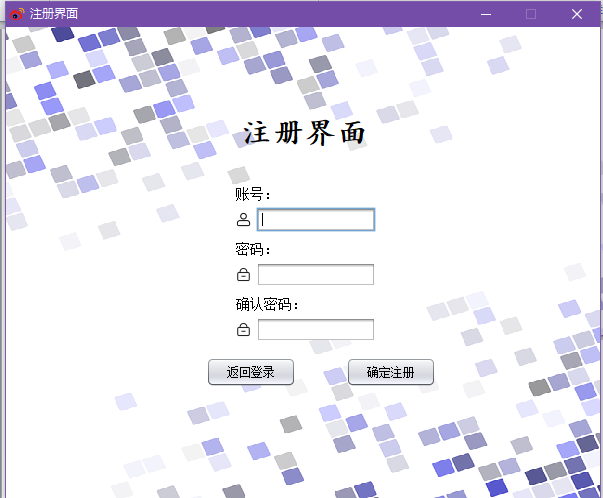


图4.2注册界面图

## 管理员界面测试

难点：切换面板、查看访客的表格，查看所有访客时需连接数据库对返回的结果集进行处理、自定义对话框

管理员初始界面，如图4.3所示



图4.3管理员初始界面图

删除访客界面与修改访客界面类似，如图4.4所示，是在删除访客界面点击查看按钮后的情况

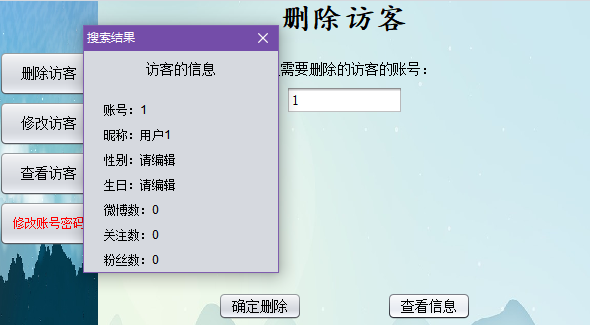


图4.4删除访客界面图

查看访客界面，如图4.5所示



图4.5查看访客界面图

修改管理员账号密码界面，如图4.6所示

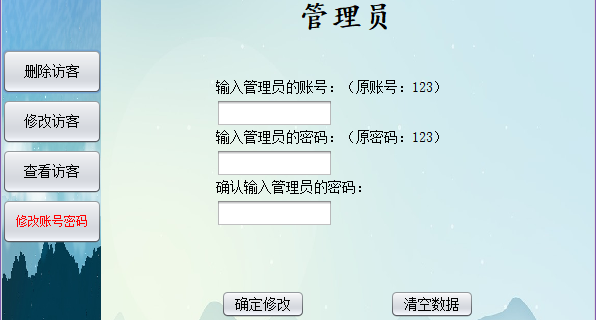


图4.6修改管理员账号密码界面图

## 访客界面测试

难点：数据库操作（增、删、改、查）、头像按钮点击、生日选择、音乐播放、音乐播放列表、文件选择器中文件图标的改变、刷新功能、发布微博并显示、微博编辑的弹出菜单栏、搜索访客

访客登录初始界面，如图4.7所示

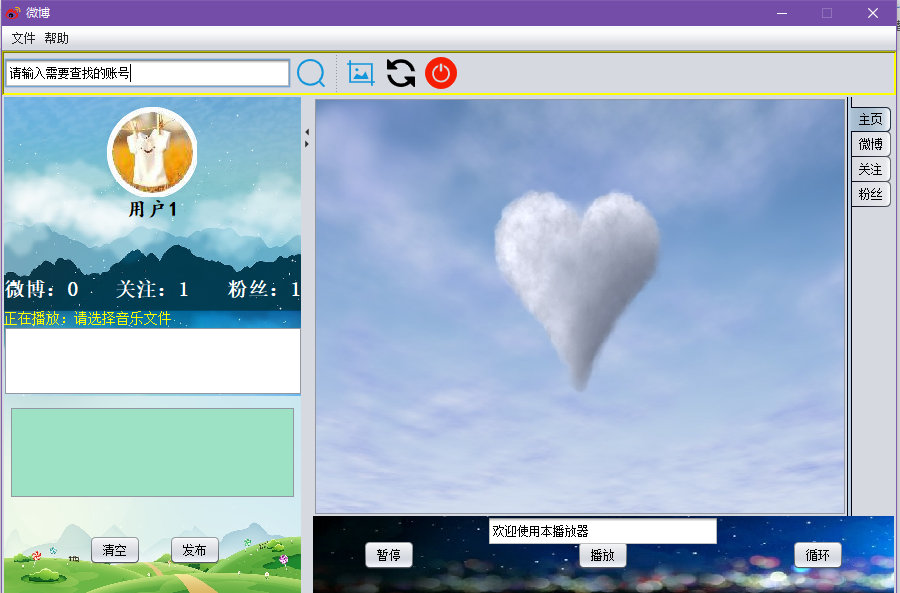


图4.7访客登录初始界面图

点击头像按钮可进入当前登录访客信息的显示或编辑对话框，如图4.8所示



图4.8编辑显示个人信息界面图

打开文本，会将txt文件中的文字输出到文本域中，如图4.9所示

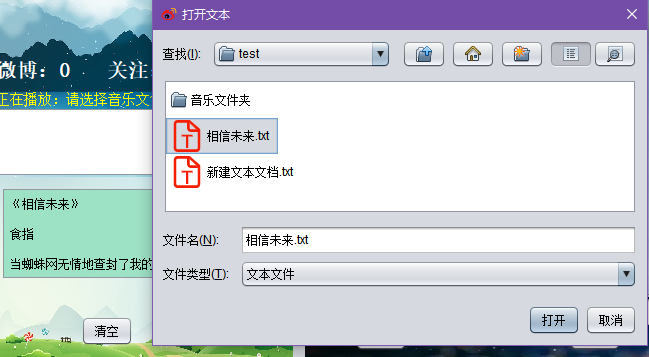


图4.9打开文本图

打开音乐文件夹，会将文件夹中的音乐信息显示到音乐列表中。

点击发布按钮，会将文本域中的文字发布到主页，如图4.10所示

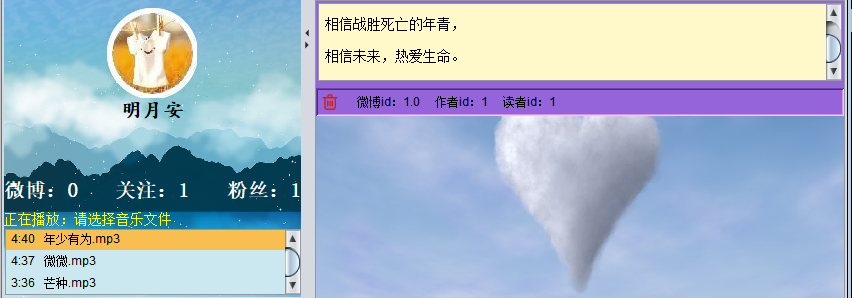


图4.10音乐列表显示和发布微博图

搜索当前登录访客，如图4.11所示

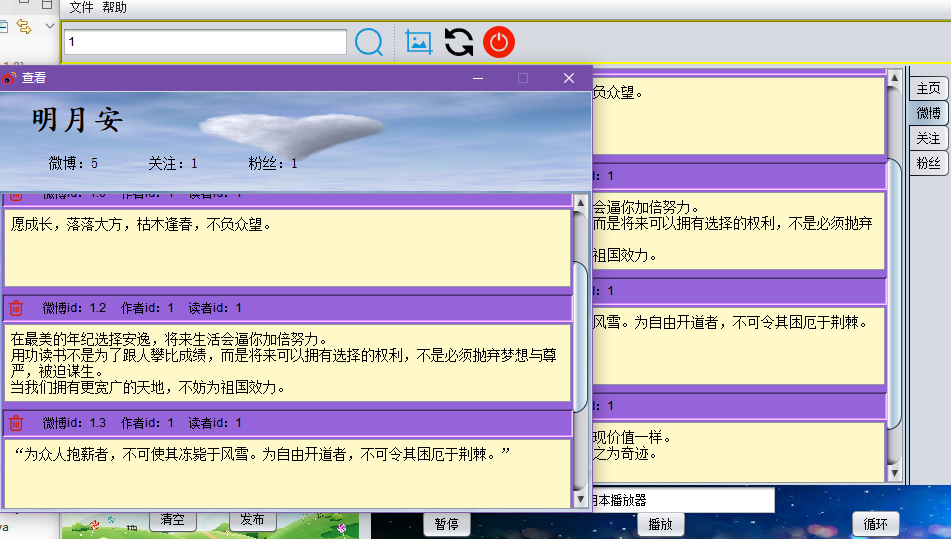


图4.11搜索当前登录访客界面图

搜索其他访客，如图4.12所示



图4.12搜索其他用户界面图

选项卡中的关注面板，如图4.13所示

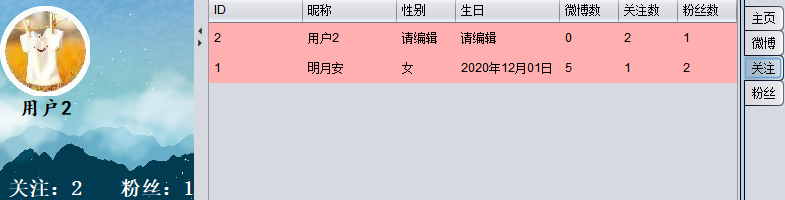


图4.13选项卡中的关注面板界面图

# 结论与心得

通过这次课程设计，发现其中有很多知识没有接触过，上网查资料的时候才发现自己学到的仅仅只是皮毛，还有很多需要我们掌握的东西不太清楚。也有许多已经学过的知识点没有理解到位，不能灵活的运用于实际，不能很好的用来解决问题，例如：接口、多线程、文件流等。在本次课程设计中接口、多线程和文件流只用到了一点点，感觉自己并没有深入地了解这些知识点，只学到了一点点皮毛。

课程设计正式完成之前，我有计划使用Java网络编程技术，使得这个微博系统能够在其他人的电脑上运行，也有计划使用新浪微博提供的开发工具SDK，使得在自己写的微博系统中发布微博的同时能同步发布到自己的新浪微博账号中，但这些功能因为自身的技术水平不够，最终取消了这些计划。本来完成之前，是设计的三重用户权限，即管理员、操作云员、访客。访客可发布微博、关注其他访客、举报其他访客、举报微博内容、为微博点赞；操作员可审核被多人举报的微博或访客；管理员可以增加、删除、修改、查看操作员信息。但在写课程设计的过程中因点赞数、举报数的数据有错误，也因时间不够的关系，最终删去了这些功能。希望在这次课程设计过后能够完善这些功能。

之前都是讲的自己在课程设计时的不足，下面就是在本次课程设计做的好的地方了。整体的界面框架很好，功能丰富而又不混乱。实现了音乐播放功能，可打开音乐文件夹，导入音乐文件夹中所有音乐。对于文件选择器中的文件图标进行了修改，使得界面美观。头像按钮设计成圆形，并将头像按钮的图片裁剪成圆形，这个功能花了很多时间去查资料，最终还是自己去学了一些知识，自己写出来的。编辑微博内容区域的右键点击弹出菜单功能的实现，发布微博能在主功能区显示。关注、粉丝以表格的形式显示。

回顾此次课程设计，从理论到实践，这些日子学到了很多知识。只有理论知识是远远不够的，只有把所学的理论知识与实践结合起来，从而提高自己的实际动手能力和独立思考能力。在课程设计的过程中也有遇到许多问题，可以说是困难重重，但可喜的是最终都解决了。

# 附：代码

**管理员主界面代码：**

package view\_administrator;

import java.awt.Color;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JSplitPane;

import data.Constant;

import tools.BackgroundPanel;

/\*\*

\* @Author 作者

\* @Description 说明：管理员窗口

\* @Date 时间：2020-12-2

\*/

@SuppressWarnings("serial")

public class Interface\_MainFrame extends JFrame {

// 按钮（增、删、改、查）

private JButton button1, button2, button3, button4;

// 创建中间容器[设置布局]

private BackgroundPanel panel1, panel2;

// 分割面板VERTICAL\HORIZONTAL

private JSplitPane splitPane = new JSplitPane(JSplitPane.HORIZONTAL\_SPLIT, true);

/\*\*

\* 构造方法

\*/

public Interface\_MainFrame() {

this.setVisible(false);

// 设置窗口标题

this.setTitle("管理员界面");

// 设置窗口大小

this.setSize(600, 500);

// 居中显示

this.setLocationRelativeTo(null);

// 设置图标

this.setIconImage(new ImageIcon("images\\图标\_微博.png").getImage());

// 设置点击关闭窗口后做出的处理

// JFrame.DO\_NOTHING\_ON\_CLOSE 什么也不做

// JFrame.HIDE\_ON\_CLOSE 隐藏当前窗口

// JFrame.DISPOSE\_ON\_CLOSE 隐藏当前窗口，并释放窗体占有的其他资源

// JFrrame.EXIT\_ON\_CLOSE 结束窗口所在应用程序

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

/\* 面板设置 \*/

this.panelSettings();

/\* 事件监听 \*/

this.addListener();

// 将分割面板添加到窗口

this.add(splitPane);

// 设置窗口是否可缩放

this.setResizable(false);

// 设置窗口是否可见

this.setVisible(true);

}

/\*\*

\* @Description 说明：面板设置

\*/

private void panelSettings() {

/\* 面板1设置 \*/

panel1 = new BackgroundPanel(new ImageIcon("images\\背景\_1.jpg").getImage());

panel1.setLayout(null);

// 按钮

button1 = new JButton("删除访客");

button1.setFont(Constant.FONT2);

button1.setBounds(0, 50, 100, 45);

button2 = new JButton("修改访客");

button2.setFont(Constant.FONT2);

button2.setBounds(0, 100, 100, 45);

button3 = new JButton("查看访客");

button3.setFont(Constant.FONT2);

button3.setBounds(0, 150, 100, 45);

button4 = new JButton("修改账号密码");

button4.setForeground(Color.red);

button4.setBounds(0, 200, 100, 45);

// 添加组件

panel1.add(button1);

panel1.add(button2);

panel1.add(button3);

panel1.add(button4);

/\* 面板2设置 \*/

setupPanel();

/\* 分割面板设置 \*/

splitPane.setLeftComponent(panel1);

splitPane.setRightComponent(panel2);

// 设置分割线大小

splitPane.setDividerSize(0);

// 设置分割线位置

splitPane.setDividerLocation(100);

// 设置分割线拖动

splitPane.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane.setOneTouchExpandable(false);

}

/\*\*

\* 面板panel2初始界面

\*/

private void setupPanel() {

// TODO 自动生成的方法存根

panel2 = new BackgroundPanel(new ImageIcon("images\\背景\_2.jpg").getImage());

JLabel Title = new JLabel("欢迎来到管理员界面！");

Title.setFont(Constant.FONT1);

panel2.add(Title);

}

/\*\*

\* @Description 说明：事件监听

\*/

private void addListener() {

// TODO 自动生成的方法存根

button1.addActionListener((e) -> setup\_PanelDelete(panel2, new PanelDelete()));

button2.addActionListener((e) -> setup\_PanelUpdate(panel2, new PanelUpdate()));

button3.addActionListener((e) -> setup\_PanelSelect(panel2, new PanelShow()));

button4.addActionListener((e) -> update(panel2, new PanelAdministrator()));

}

private Object update(BackgroundPanel panel, PanelAdministrator panelAdministrator) {

// TODO 自动生成的方法存根

panel.removeAll();// 移除面板中的所有组件

panel.add(panelAdministrator);// 添加要切换的面板

panel.repaint();// 刷新页面，重绘面板

panel.validate();// 使重绘的面板确认生效

// 初始化

panelAdministrator.init(panel.getWidth(), panel.getHeight());

// 更换分割面板

splitPane.setLeftComponent(panel1);

splitPane.setRightComponent(panelAdministrator);

// 设置分割线大小

splitPane.setDividerSize(0);

// 设置分割线位置

splitPane.setDividerLocation(100);

// 设置分割线拖动

splitPane.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane.setOneTouchExpandable(false);

return null;

}

private Object setup\_PanelSelect(BackgroundPanel panel, PanelShow panelSelect) {

// TODO 自动生成的方法存根

panel.removeAll();// 移除面板中的所有组件

panel.add(panelSelect);// 添加要切换的面板

panel.repaint();// 刷新页面，重绘面板

panel.validate();// 使重绘的面板确认生效

// 初始化

panelSelect.init(panel.getWidth(), panel.getHeight());

// 更换分割面板

splitPane.setLeftComponent(panel1);

splitPane.setRightComponent(panelSelect);

// 设置分割线大小

splitPane.setDividerSize(0);

// 设置分割线位置

splitPane.setDividerLocation(100);

// 设置分割线拖动

splitPane.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane.setOneTouchExpandable(false);

return null;

}

private Object setup\_PanelUpdate(BackgroundPanel panel, PanelUpdate panelUpdate) {

// TODO 自动生成的方法存根

panel.removeAll();// 移除面板中的所有组件

panel.add(panelUpdate);// 添加要切换的面板

panel.repaint();// 刷新页面，重绘面板

panel.validate();// 使重绘的面板确认生效

// 初始化

panelUpdate.init(panel.getWidth(), panel.getHeight());

// 更换分割面板

splitPane.setLeftComponent(panel1);

splitPane.setRightComponent(panelUpdate);

// 设置分割线大小

splitPane.setDividerSize(0);

// 设置分割线位置

splitPane.setDividerLocation(100);

// 设置分割线拖动

splitPane.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane.setOneTouchExpandable(false);

return null;

}

private Object setup\_PanelDelete(BackgroundPanel panel, PanelDelete panelDelete) {

// TODO 自动生成的方法存根

panel.removeAll();// 移除面板中的所有组件

panel.add(panelDelete);// 添加要切换的面板

panel.repaint();// 刷新页面，重绘面板

panel.validate();// 使重绘的面板确认生效

// 初始化

panelDelete.init(panel.getWidth(), panel.getHeight());

// 更换分割面板

splitPane.setLeftComponent(panel1);

splitPane.setRightComponent(panelDelete);

// 设置分割线大小

splitPane.setDividerSize(0);

// 设置分割线位置

splitPane.setDividerLocation(100);

// 设置分割线拖动

splitPane.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane.setOneTouchExpandable(false);

return null;

}

}

**访客主界面代码：**

package view\_visitor;

import java.awt.BorderLayout;

import java.awt.Color;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import java.io.File;

import java.io.IOException;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JOptionPane;

import javax.swing.JSplitPane;

import javax.swing.JTextField;

import javax.swing.JToolBar;

import javax.swing.border.BevelBorder;

import control.Delete;

import control.Select;

import data.Constant;

import data.GlobalVar;

import data.Refresh;

import model.Visitor;

/\*\*

\* @Author 作者

\* @Description 说明：访客主界面

\* @Date 时间：2020-11-20

\*/

@SuppressWarnings("serial")

public class Interface\_MainFrame extends JFrame {

// 工具栏

private JToolBar jToolBar = new JToolBar();

private JButton button1, button2, button3, button;

private JTextField idTextField;

// 创建中间容器[设置布局]

private Panel1\_PersonalInformation panel1;

private Panel2\_EditorialPublished panel2;

private Panel3\_MainFunction panel3;

private Panel4\_PlaybackControls panel4;

// 分割面板VERTICAL\HORIZONTAL

private JSplitPane splitPane1 = new JSplitPane(JSplitPane.VERTICAL\_SPLIT, true);

private JSplitPane splitPane2 = new JSplitPane(JSplitPane.VERTICAL\_SPLIT, true);

private JSplitPane splitPane3 = new JSplitPane(JSplitPane.HORIZONTAL\_SPLIT, true);

/\*\*

\* 构造方法

\*/

public Interface\_MainFrame() {

System.out.println("-----登录访客对象的部分信息");

System.out.println("访客ID：" + GlobalVar.login\_visitor.getId());

System.out.println("访客昵称：" + GlobalVar.login\_visitor.getVisitor\_name());

System.out.println("可见微博总个数：" + GlobalVar.weibo\_num2);

System.out.println("微博个数：" + GlobalVar.weibo\_num1);

System.out.println("关注个数：" + GlobalVar.attention\_num);

System.out.println("粉丝个数：" + GlobalVar.fans\_num);

// 顶层容器的设置

// 设置窗口标题

this.setTitle("微博");

// 设置窗口大小

this.setSize(900, 600);

// 居中显示

this.setLocationRelativeTo(null);

// 设置图标

this.setIconImage(new ImageIcon("images\\图标\_微博.png").getImage());

// 设置点击关闭窗口后做出的处理

// JFrame.DO\_NOTHING\_ON\_CLOSE 什么也不做

// JFrame.HIDE\_ON\_CLOSE 隐藏当前窗口

// JFrame.DISPOSE\_ON\_CLOSE 隐藏当前窗口，并释放窗体占有的其他资源

// JFrrame.EXIT\_ON\_CLOSE 结束窗口所在应用程序

// this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

/\* 面板设置与数据处理 \*/

this.panelSettings();

/\* 事件监听 \*/

this.addListener();

// 添加工具栏

this.add(jToolBar, BorderLayout.NORTH);

// 将分割面板添加到窗口

this.add(splitPane3);

// 设置窗口是否可缩放

this.setResizable(false);

// 设置窗口是否可见

this.setVisible(true);

// 添加窗口监听器

addWindowListener(new WindowAdapter() {

public void windowClosing(WindowEvent e) {

Panel2\_EditorialPublished.editArea.requestFocus();

String currentValue = Panel2\_EditorialPublished.editArea.getText();

if (currentValue.equals(SetupJMenuBar.oldValue) == true || Panel2\_EditorialPublished.editArea.getText().equals("")) {

System.exit(0);

} else {

int exitChoose = JOptionPane.showConfirmDialog(panel2, "您的文件尚未保存，是否保存？", "退出提示", JOptionPane.YES\_NO\_CANCEL\_OPTION);

if (exitChoose == JOptionPane.YES\_OPTION) {

new SetupJMenuBar().SaveTextFileAs();

} else if (exitChoose == JOptionPane.NO\_OPTION) {

System.exit(0);

}

}

}

});

}

/\*\*

\* @Description 说明：面板设置与数据处理

\*/

private void panelSettings() {

/\* 设置菜单条 \*/

this.setJMenuBar(new SetupJMenuBar());

/\* 设置工具栏 \*/

this.setJToolBar();

/\* 面板panel1设置——个人信息 \*/

panel1 = new Panel1\_PersonalInformation();

/\* 面板panel2设置——编辑、发布微博 \*/

panel2 = new Panel2\_EditorialPublished();

/\* 面板panel3设置——浏览微博 \*/

panel3 = new Panel3\_MainFunction();

/\* 面板panel4设置——音乐播放控件 \*/

panel4 = new Panel4\_PlaybackControls();

/\* 设置分割面板 \*/

this.setupSplitPane();

}

/\*\*

\* @Description 说明：设置工具栏

\*/

private void setJToolBar() {

// 设置工具栏不可浮动

jToolBar.setFloatable(false);

// 设置工具栏边框导角方式

jToolBar.setBorder(new BevelBorder(BevelBorder.LOWERED));

jToolBar.setBackground(Color.yellow);

button = new JButton(new ImageIcon("images/1\_截图.png"));

button.setToolTipText("截图");

idTextField = new JTextField("请输入需要查找的账号", 25);

button1 = new JButton(new ImageIcon("images/1\_搜索.png"));

button1.setToolTipText("搜索");

button2 = new JButton(new ImageIcon("images/1\_刷新.png"));

button2.setToolTipText("刷新");

button3 = new JButton(new ImageIcon("images/1\_注销.png"));

button3.setToolTipText("注销");

jToolBar.add(idTextField);

jToolBar.add(button1);

jToolBar.addSeparator();

jToolBar.add(button);

jToolBar.add(button2);

jToolBar.add(button3);

}

/\*\*

\* 设置分割面板

\*/

private void setupSplitPane() {

splitPane1.setLeftComponent(panel1);

splitPane1.setRightComponent(panel2);

// 设置分割线大小

splitPane1.setDividerSize(0);

// 设置分割线位置

splitPane1.setDividerLocation(300);

// 设置分割线拖动

splitPane1.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane1.setOneTouchExpandable(false);

splitPane2.setLeftComponent(panel3);

splitPane2.setRightComponent(panel4);

// 设置分割线大小

splitPane2.setDividerSize(0);

// 设置分割线位置

splitPane2.setDividerLocation(420);

// 设置分割线拖动

splitPane2.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane2.setOneTouchExpandable(false);

splitPane3.setLeftComponent(splitPane1);

splitPane3.setRightComponent(splitPane2);

// 设置分割线大小

splitPane3.setDividerSize(10);

// 设置分割线位置

splitPane3.setDividerLocation(300);

// 设置分割线拖动

splitPane3.setEnabled(false);

// 在分隔符上提供UI小部件以快速扩展/折叠分隔符

splitPane3.setOneTouchExpandable(true);

}

/\*\*

\* 事件监听

\*/

private void addListener() {

/\* 截图 \*/

button.addActionListener((e) -> {

// 相对地址

File file = new File("database", "屏幕截图.exe");

Runtime runtime = Runtime.getRuntime();

System.out.println("屏幕截图.exe的绝对地址：" + file.getAbsolutePath());

try {

runtime.exec(file.getAbsolutePath());

} catch (IOException e1) {

System.out.println(e1);

}

});

/\* 搜索 \*/

button1.addActionListener((e) -> {

/\* 访客 \*/

Visitor visitor = new Visitor(idTextField.getText());

new Select().select(visitor);

if (visitor.getVisitor\_yes\_no().equals(Constant.YES\_STRING) == true) {

new ShowFram(visitor, this);

} else {

System.out.println("不可查看");

JOptionPane.showMessageDialog(null, "账号不存在", "警告", JOptionPane.INFORMATION\_MESSAGE);

}

});

/\* 刷新 \*/

button2.addActionListener((e) -> {

this.setVisible(false);

new Refresh();

});

/\* 注销 \*/

button3.addActionListener((e) -> {

// 对话框（询问是否需要注销）

int buttonChoose = JOptionPane.showConfirmDialog(this, "您即将注销该账号，请确认注销", "提示", JOptionPane.YES\_NO\_OPTION);

if (buttonChoose == JOptionPane.NO\_OPTION) {

return;

} else if (buttonChoose == JOptionPane.YES\_OPTION) {

new Delete().delete(GlobalVar.login\_personal);

new Delete().delete(GlobalVar.login\_visitor);

new Delete().deleteWeibo(GlobalVar.login\_visitor.getId());

new Delete().delete1(GlobalVar.login\_visitor.getId());

new Delete().delete2(GlobalVar.login\_visitor.getId());

System.exit(0);

}

});

}

}

**访客界面菜单栏设置代码：**

package view\_visitor;

import java.io.BufferedReader;

import java.io.BufferedWriter;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.io.InputStreamReader;

import java.io.OutputStreamWriter;

import java.util.Map;

import javax.sound.sampled.AudioFileFormat;

import javax.sound.sampled.AudioSystem;

import javax.sound.sampled.SourceDataLine;

import javax.swing.ImageIcon;

import javax.swing.JFileChooser;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.JMenuItem;

import javax.swing.JOptionPane;

import javax.swing.filechooser.FileNameExtensionFilter;

import control.Insert;

import data.Constant;

import data.GlobalVar;

import model.Music;

import tools.FileIconView;

/\*\*

\* @Author 作者

\* @Description 说明：设置菜单栏

\* @Date 时间：2020-12-4

\*/

@SuppressWarnings("serial")

public class SetupJMenuBar extends JMenuBar {

public SourceDataLine sourceDataLine;// 输出设备

// 其他变量

public static String oldValue;// 存放编辑区原来的内容，用于比较文本是否有改动

private boolean isNewFile = true; // 是否新文件(未保存过的)

private File currentFile;// 当前文件名

private String string = Constant.PATH;

// 菜单+菜单项

private JMenu fileMenu = new JMenu("文件");

private JMenuItem fileMenu\_OpenMP3 = new JMenuItem("打开音乐");

private JMenuItem fileMenu\_OpenMP3List = new JMenuItem("打开文件夹");

private JMenuItem fileMenu\_OpenTXT = new JMenuItem("打开文本");

private JMenuItem fileMenu\_SaveTXT = new JMenuItem("保存文本");

private JMenuItem fileMenu\_SaveTXTAs = new JMenuItem("另存为...");

private JMenu helpMenu = new JMenu("帮助");

private JMenuItem helpMenu\_About = new JMenuItem("关于");

private JMenuItem helpMenu\_Con = new JMenuItem("帮助内容");

SetupJMenuBar() {

// 文件（打开音乐、打开文件夹 打开文本、保存文本、另存为）

fileMenu.add(fileMenu\_OpenMP3);

fileMenu.add(fileMenu\_OpenMP3List);

fileMenu.addSeparator();

fileMenu.add(fileMenu\_OpenTXT);

fileMenu.add(fileMenu\_SaveTXT);

fileMenu.add(fileMenu\_SaveTXTAs);

// 帮助（关于、帮助内容）

helpMenu.add(helpMenu\_About);

helpMenu.add(helpMenu\_Con);

this.add(fileMenu);// 添加菜单

this.add(helpMenu);// 添加帮助

// 事件监听

addListener();

}

/\*\*

\* 事件监听

\*/

public void addListener() {

fileMenu\_OpenMP3.addActionListener((e) -> OpenMusicFile()); // 打开音乐

fileMenu\_OpenMP3List.addActionListener((e) -> OpenFolder()); // 打开文件夹

fileMenu\_OpenTXT.addActionListener((e) -> OpenTextFile()); // 打开文本

fileMenu\_SaveTXT.addActionListener((e) -> SaveTextFile()); // 保存文本

fileMenu\_SaveTXTAs.addActionListener((e) -> SaveTextFileAs()); // 另存为

helpMenu\_About.addActionListener((e) -> About()); // 关于

helpMenu\_Con.addActionListener((e) -> HelpContent()); // 帮助内容

}

/\*\*

\* @Description 说明：帮助，帮助内容

\*/

private Object HelpContent() {

// TODO 自动生成的方法存根

Panel2\_EditorialPublished.editArea.requestFocus();

JOptionPane.showMessageDialog(this, "路漫漫其修远兮，吾将上下而求索。", "帮助主题", JOptionPane.INFORMATION\_MESSAGE);

return null;

}

/\*\*

\* @Description 说明：帮助，关于

\*/

private Object About() {

// TODO 自动生成的方法存根

Panel2\_EditorialPublished.editArea.requestFocus();

JOptionPane.showMessageDialog(this,

"&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&\n" + " 编写者：谢晓艳\n" + "\n" + "&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&&\n", "桌面微博",

JOptionPane.INFORMATION\_MESSAGE);

return null;

}

/\*\*

\* @Description 说明：文件，另存为

\*/

public Object SaveTextFileAs() {

// TODO 自动生成的方法存根

Panel2\_EditorialPublished.editArea.requestFocus();

// 创建文件选择器

JFileChooser fileChooser = new JFileChooser();

// 定义按钮

fileChooser.setApproveButtonText("另存为");

// 定义文件选择框标题

fileChooser.setDialogTitle("另存为。。");

// 设置默认目录

fileChooser.setCurrentDirectory(new File(string));

// 设置默认文件名

fileChooser.setSelectedFile(new File(".txt"));

// 文件过滤器

fileChooser.setFileFilter(new FileNameExtensionFilter("文本文件", "txt"));

// 禁用all files过滤器

fileChooser.setAcceptAllFileFilterUsed(false);

// 为文件选择器显示的每个文件提供特定的图标和文件描述

fileChooser.setFileView(new FileIconView(fileChooser.getFileFilter(), new ImageIcon("images\\txt.png")));

// 显示打开对话框

int result = fileChooser.showOpenDialog(this.getComponent());

result = fileChooser.showSaveDialog(this.getComponent());// 显示保存对话框

if (result == JFileChooser.APPROVE\_OPTION) {

File saveFileName = fileChooser.getSelectedFile();

if (saveFileName == null || saveFileName.getName().equals(".txt")) {

JOptionPane.showMessageDialog(this, "不合法的文件名", "不合法的文件名", JOptionPane.ERROR\_MESSAGE);

} else {

System.out.println("另存为的文件名称为：" + saveFileName.getName());

try {

OutputStreamWriter writer = new OutputStreamWriter(new FileOutputStream(saveFileName), "UTF-8");

BufferedWriter bufferedWriter = new BufferedWriter(writer);

bufferedWriter.write(Panel2\_EditorialPublished.editArea.getText(), 0, Panel2\_EditorialPublished.editArea.getText().length());

bufferedWriter.flush();

writer.close();

oldValue = Panel2\_EditorialPublished.editArea.getText();

} catch (IOException ioException) {

}

}

} else if (result == JFileChooser.CANCEL\_OPTION) {

System.out.println("没有保存任何文件");

}

return null;

}

/\*\*

\* @Description 说明：文件，保存

\*/

private Object SaveTextFile() {

// TODO 自动生成的方法存根

Panel2\_EditorialPublished.editArea.requestFocus();

if (isNewFile) {

// 创建文件选择器

JFileChooser fileChooser = new JFileChooser();

// 定义按钮

fileChooser.setApproveButtonText("保存");

// 定义文件选择框标题

fileChooser.setDialogTitle("保存文本");

// 设置默认目录

fileChooser.setCurrentDirectory(new File(string));

// 设置默认文件名

fileChooser.setSelectedFile(new File(".txt"));

// 文件过滤器

fileChooser.setFileFilter(new FileNameExtensionFilter("文本文件", "txt"));

// 禁用all files过滤器

fileChooser.setAcceptAllFileFilterUsed(false);

// 为文件选择器显示的每个文件提供特定的图标和文件描述

fileChooser.setFileView(new FileIconView(fileChooser.getFileFilter(), new ImageIcon("images\\txt.png")));

// 显示打开对话框

int result = fileChooser.showOpenDialog(this.getComponent());

if (result == JFileChooser.APPROVE\_OPTION) {

File saveFileName = fileChooser.getSelectedFile();

if (saveFileName == null || saveFileName.getName().equals(".txt")) {

JOptionPane.showMessageDialog(this, "不合法的文件名", "不合法的文件名", JOptionPane.ERROR\_MESSAGE);

} else {

System.out.println("保存的文件名称为：" + saveFileName.getName());

try {

OutputStreamWriter writer = new OutputStreamWriter(new FileOutputStream(saveFileName), "UTF-8");

BufferedWriter bufferedWriter = new BufferedWriter(writer);

bufferedWriter.write(Panel2\_EditorialPublished.editArea.getText(), 0, Panel2\_EditorialPublished.editArea.getText().length());

bufferedWriter.flush();

bufferedWriter.close();

isNewFile = false;

currentFile = saveFileName;

oldValue = Panel2\_EditorialPublished.editArea.getText();

} catch (Exception e) {

System.out.println("读取文件内容出错");

e.printStackTrace();

}

}

} else if (result == JFileChooser.CANCEL\_OPTION) {

System.out.println("没有保存任何文件");

}

} else {

System.out.println("保存的文件名称为：" + currentFile.getName());

try {

OutputStreamWriter writer = new OutputStreamWriter(new FileOutputStream(currentFile), "UTF-8");

BufferedWriter bufferedWriter = new BufferedWriter(writer);

bufferedWriter.write(Panel2\_EditorialPublished.editArea.getText(), 0, Panel2\_EditorialPublished.editArea.getText().length());

bufferedWriter.flush();

writer.close();

} catch (Exception e) {

System.out.println("读取文件内容出错");

e.printStackTrace();

}

}

return null;

}

/\*\*

\* @Description 说明：打开文本功能实现

\*/

private void open() {

// 创建文件选择器

JFileChooser fileChooser = new JFileChooser();

// 定义文件选择框标题

fileChooser.setDialogTitle("打开文本");

// 设置默认目录

fileChooser.setCurrentDirectory(new File(string));

// 文件过滤器

fileChooser.setFileFilter(new FileNameExtensionFilter("文本文件", "txt"));

// 禁用all files过滤器

fileChooser.setAcceptAllFileFilterUsed(false);

// 为文件选择器显示的每个文件提供特定的图标和文件描述

fileChooser.setFileView(new FileIconView(fileChooser.getFileFilter(), new ImageIcon("images\\txt.png")));

// 显示打开对话框

int result = fileChooser.showOpenDialog(this.getComponent());

if (result == JFileChooser.APPROVE\_OPTION) {

File fileName = fileChooser.getSelectedFile();

if (fileName == null || fileName.getName().equals(".txt")) {

JOptionPane.showMessageDialog(this, "不合法的文件名", "不合法的文件名", JOptionPane.ERROR\_MESSAGE);

} else {

System.out.println("打开的文件名称为：" + fileName.getName());

try {

// 判断文件是否存在

if (fileName.isFile() && fileName.exists()) {

InputStreamReader read = new InputStreamReader(new FileInputStream(fileName), "UTF-8");

BufferedReader bufferedReader = new BufferedReader(read);

String lineTxt = null;

while ((lineTxt = bufferedReader.readLine()) != null) {

Panel2\_EditorialPublished.editArea.append(lineTxt + "\n");

}

read.close();

isNewFile = false;

currentFile = fileName;

oldValue = Panel2\_EditorialPublished.editArea.getText();

} else {

System.out.println("找不到指定的文件");

}

} catch (Exception e) {

System.out.println("读取文件内容出错");

e.printStackTrace();

}

}

} else if (result == JFileChooser.CANCEL\_OPTION) {

System.out.println("没有打开任何文件");

}

}

/\*\*

\* @Description 说明：文件，打开文本

\*/

private Object OpenTextFile() {

// TODO 自动生成的方法存根

Panel2\_EditorialPublished.editArea.requestFocus();

String currentValue = Panel2\_EditorialPublished.editArea.getText();

boolean isTextChange = (currentValue.equals(oldValue)) ? false : true;

// 判断是否改变文本

if (isTextChange && !Panel2\_EditorialPublished.editArea.getText().equals("")) {

// 对话框（询问是否需要保存）

int saveChoose = JOptionPane.showConfirmDialog(this, "您的文件尚未保存，是否保存？", "提示", JOptionPane.YES\_NO\_OPTION);

if (saveChoose == JOptionPane.NO\_OPTION) {

open();// 打开文本功能

} else if (saveChoose == JOptionPane.YES\_OPTION) {

SaveTextFileAs();// 另存为功能

}

} else {

open();// 打开文本功能

}

return null;

}

/\*\*

\* @Description 说明：文件，打开文件夹

\*/

private Object OpenFolder() {

// TODO 自动生成的方法存根

int result = 0;

// 创建文件选择器

JFileChooser fileChooser = new JFileChooser();

// 定义文件选择框标题

fileChooser.setDialogTitle("打开文件夹");

// 设置默认目录

fileChooser.setCurrentDirectory(new File(string));

// 只打开目录

fileChooser.setFileSelectionMode(JFileChooser.DIRECTORIES\_ONLY);

// 文件过滤器

fileChooser.setFileFilter(new FileNameExtensionFilter("音乐文件", "mp3"));

// 禁用all files过滤器

fileChooser.setAcceptAllFileFilterUsed(false);

// 为文件选择器显示的每个文件提供特定的图标和文件描述

fileChooser.setFileView(new FileIconView(fileChooser.getFileFilter(), new ImageIcon("images\\mp3.png")));

// 显示打开对话框

result = fileChooser.showOpenDialog(this.getComponent());

if (result == JFileChooser.APPROVE\_OPTION) {

String path = fileChooser.getSelectedFile().getPath();

File file = new File(path);

String[] filelist = file.list();

for (int i = 0; i < filelist.length; i++) {

// 清空音乐列表

GlobalVar.MUSIC\_LIST.removeAll();

// 判断文件名称尾部

if (filelist[i].endsWith(".mp3") || filelist[i].endsWith(".wav")) {

// 获取音频时长

int s = 0;

long total = 0;

try {

File f = new File(path, filelist[i]);

AudioFileFormat aff = AudioSystem.getAudioFileFormat(f);

@SuppressWarnings("rawtypes")

Map props = aff.properties();

if (props.containsKey("duration")) {

total = (long) Math.round((((Long) props.get("duration")).longValue()) / 1000);

}

s = (int) total / 1000;

// System.out.println("打开的文件名称为："+filelist[i]);

// System.out.println("打开的文件夹路径为：" + fileChooser.getSelectedFile().getPath());

// System.out.println("音频时长：" + s / 60 + ":" + s % 60);

} catch (Exception e) {

e.printStackTrace();

}

// 添加列表

Music music = new Music();

music.setName(filelist[i]);

music.setPath(path + "\\" + filelist[i]);

music.setLength(s / 60 + ":" + s % 60);

GlobalVar.MUSIC = Insert.insert(GlobalVar.MUSIC, music);

GlobalVar.MUSIC\_LIST.setListData(GlobalVar.MUSIC);

}

}

} else if (result == JFileChooser.CANCEL\_OPTION) {

System.out.println("没有打开任何文件夹");

}

return null;

}

/\*\*

\* @Description 说明：文件，打开音乐

\*/

private Object OpenMusicFile() {

// TODO 自动生成的方法存根

File file = null;

int result = 0;

// 创建文件选择器

JFileChooser fileChooser = new JFileChooser();

// 定义文件选择框标题

fileChooser.setDialogTitle("打开音乐");

// 设置默认目录

fileChooser.setCurrentDirectory(new File(string));

// 文件过滤器

fileChooser.setFileFilter(new FileNameExtensionFilter("音乐文件", "mp3"));

// 禁用all files过滤器

fileChooser.setAcceptAllFileFilterUsed(false);

// 为文件选择器显示的每个文件提供特定的图标和文件描述

fileChooser.setFileView(new FileIconView(fileChooser.getFileFilter(), new ImageIcon("images\\mp3.png")));

// 显示打开对话框

result = fileChooser.showOpenDialog(this.getComponent());

if (result == JFileChooser.APPROVE\_OPTION) {

file = fileChooser.getSelectedFile();

// 获取音乐文件名

String filename = file.getName();

// 清空音乐列表

GlobalVar.MUSIC\_LIST.removeAll();

// 判断文件名称尾部

if (filename.endsWith(".mp3") || filename.endsWith(".wav")) {

// 获取音频时长

int s = 0;

long total = 0;

try {

AudioFileFormat aff = AudioSystem.getAudioFileFormat(file);

@SuppressWarnings("rawtypes")

Map props = aff.properties();

if (props.containsKey("duration")) {

total = (long) Math.round((((Long) props.get("duration")).longValue()) / 1000);

}

s = (int) total / 1000;

// System.out.println("打开的文件名称为：" + file.getName());

// System.out.println("打开的文件的绝对路径为：" + file.getAbsolutePath());

// System.out.println("音频时长：" + s / 60 + ":" + s % 60);

} catch (Exception e) {

e.printStackTrace();

}

// 添加列表

Music music = new Music();

music.setName(filename);

music.setPath(file.getAbsolutePath());

music.setLength(s / 60 + ":" + s % 60);

music.setTime(total);

GlobalVar.MUSIC = Insert.insert(GlobalVar.MUSIC, music);

GlobalVar.MUSIC\_LIST.setListData(GlobalVar.MUSIC);

}

} else if (result == JFileChooser.CANCEL\_OPTION) {

System.out.println("没有打开任何文件");

}

return null;

}

}