Java语言与网络

课程设计

题目名称：网络聊天程序、网页爬虫程序、邮件客户端程序

姓 名： 王瑛

学 号： 8208190406

专 业： 计算机科学与技术

班 级： 计科1902班

指导教师： 张德宇

编写日期： 2021年7月16日

目录（单独页）

（自第一章正文起，每章单独起始页）

1. 开发工具及环境说明
2. 网络聊天程序业务分析
3. 网络聊天程序系统设计

2.1 系统功能定义

2.2 关键问题设计

1. 聊天程序源代码清单
2. 聊天程序运行结果与测试分析
3. 网页爬虫程序业务分析
4. 网页爬虫程序系统设计

7.1 系统功能定义

7.2 关键问题设计

1. 爬虫程序源代码清单
2. 爬虫程序运行结果与测试分析
3. 邮件客户端程序业务分析
4. 邮件客户端程序系统设计

11.1 系统功能定义

11.2 关键问题设计

1. 邮件客户端程序源代码清单
2. 邮件客户端程序运行结果与测试分析
3. 结论与心得

目录

[第一章 开发工具及环境说明 4](#_Toc77452713)

[第二章 网络聊天程序业务分析 5](#_Toc77452714)

[第三章 网络聊天程序系统设计 5](#_Toc77452715)

[3.1系统功能定义 5](#_Toc77452716)

[（1）界面模块 5](#_Toc77452717)

[（2）消息包模块 6](#_Toc77452718)

[（3）服务器模块 7](#_Toc77452719)

[（4）客户端模块 7](#_Toc77452720)

[3.2 关键问题设计 8](#_Toc77452721)

[（1）多个客户端连接服务器 8](#_Toc77452722)

[（2）用户之间私聊 8](#_Toc77452723)

[（3）多对多聊天 8](#_Toc77452724)

[（4）登录时同步好友在线情况 9](#_Toc77452725)

[（5）系统公告、群发消息 10](#_Toc77452726)

[（6）下线时在其他客户端上实时更新 10](#_Toc77452727)

[（7）服务器使用户强制下线 10](#_Toc77452728)

[第四章 聊天程序源代码清单 11](#_Toc77452729)

[4.1 package qqclient\_view 11](#_Toc77452730)

[(1) ClientLogin.java: 11](#_Toc77452731)

[(2) FriendList.java: 14](#_Toc77452732)

[(3) ManageQqChat.java: 21](#_Toc77452733)

[(4) ManageQqFriendList.java: 22](#_Toc77452734)

[(5) qqchat.java: 22](#_Toc77452735)

[4.2 package qqserver.view: 24](#_Toc77452736)

[(1) myServer\_view.java: 24](#_Toc77452737)

[(2) ManageView.java: 29](#_Toc77452738)

[4.3 package qqserver.model: 30](#_Toc77452739)

[(1) myQqServer.java: 30](#_Toc77452740)

[(2) SerConClientThread.java: 31](#_Toc77452741)

[(3) ManageClientThread.java: 34](#_Toc77452742)

[4.4 package qqClient\_model: 35](#_Toc77452743)

[(1) ClientConServer.java: 35](#_Toc77452744)

[(2) ClientUser.java: 37](#_Toc77452745)

[(3) ClientConServerThread.java: 37](#_Toc77452746)

[(4) ManageClientConServerThread.java: 40](#_Toc77452747)

[4.5 package common: 40](#_Toc77452748)

[(1) Message.java: 40](#_Toc77452749)

[(2) MessageType.java: 42](#_Toc77452750)

[(3) User.java: 42](#_Toc77452751)

[第五章 聊天程序运行结果与测试分析 43](#_Toc77452752)

[第六章 网页爬虫程序业务分析 48](#_Toc77452753)

[第七章 网页爬虫程序系统设计 49](#_Toc77452754)

[7.1 系统功能定义 49](#_Toc77452755)

[（1）界面模块 49](#_Toc77452756)

[（2）单个网址爬虫模块 49](#_Toc77452757)

[（3）敏感词匹配模块 49](#_Toc77452758)

[（4）文件爬虫模块 49](#_Toc77452759)

[7.2 关键问题设计 50](#_Toc77452760)

[（1）爬取网页源代码 50](#_Toc77452761)

[（2）爬取网页文本 50](#_Toc77452762)

[（3）敏感词匹配 50](#_Toc77452763)

[（4）爬取文件中网址文本与敏感词库匹配并保存统计结果 51](#_Toc77452764)

[第八章 爬虫程序源代码清单 51](#_Toc77452765)

[第九章 爬虫程序运行结果与测试分析 60](#_Toc77452766)

[第十章 邮件客户端程序业务分析 66](#_Toc77452767)

[第十一章 邮件客户端程序系统设计 66](#_Toc77452768)

[11.1 系统功能定义 66](#_Toc77452769)

[（1）登录界面模块 66](#_Toc77452770)

[（2）功能选择模块 66](#_Toc77452771)

[（3）邮件发送模块 66](#_Toc77452772)

[（4）邮件查看模块 67](#_Toc77452773)

[11.2 关键问题设计 67](#_Toc77452774)

[(1) 发送邮件: 67](#_Toc77452775)

[(2) 查看邮件: 68](#_Toc77452776)

[(3) 删除邮件： 68](#_Toc77452777)

[第十二章 邮件客户端程序源代码清单 69](#_Toc77452778)

[(1) Login\_view.java: 69](#_Toc77452779)

[(2) WebPage.java: 71](#_Toc77452780)

[(3) SendPage.java: 73](#_Toc77452781)

[(4) CheckPage.java: 80](#_Toc77452782)

[第十三章 邮件客户端程序运行结果与测试分析 89](#_Toc77452783)

[第十四章 结论与心得 97](#_Toc77452784)

# 第一章 开发工具及环境说明

* 开发工具：IntelliJ IDEA
* 环境说明：Jdk1.8.0,jsoup-1.13.1,javax.mail,javax.activation-1.2.0

# 第二章 网络聊天程序业务分析

* 用Java图形用户界面编写聊天室服务器端和客户端， 支持多个客户端连接到一个服务器。每个客户端能够输入账号。
* 可以实现群聊（聊天记录显示在所有客户端界面）。
* 完成好友列表在各个客户端上显示。
* 可以实现私人聊天，用户可以选择某个其他用户，单独发送信息。
* 服务器能够群发系统消息，能够强行让某些用户下线。
* 客户端的上线下线要求能够在其他客户端上面实时刷新。

# 第三章 网络聊天程序系统设计

## 3.1系统功能定义

根据系统需要实现的功能，将程序分为四个部分：

### （1）界面模块

* 登录界面
* 用户输入账号密码进行登录，设置初始密码为123456，密码错误将会进行提示，输入密码验证正确进入好友界面
* 当用户登陆成功后，向服务器发送一个请求包，要求返回在线好友的包，服务器就回送一个在线好友的信息报，客户端得到包后对好友列表进行处理
* 好友&群聊界面
* 界面上半部分为好友列表，中间为群聊及系统公告显示部分，当好友上线时，群聊界面将会进行提示，好友头像将会变亮表示在线，同样当好友下线时也会进行提示，好友头像变灰
* 私聊窗口界面
* 用户可选择好友进行私聊，私聊消息仅在双方的私聊窗口显示，不会在群聊界面显示
* 服务器界面
* 服务器包括开启、关闭按钮，中间部分包括系统公告和用户上线、下线显示及在线用户情况显示，服务器还包括编辑发送系统公告、强制让用户下线的部分
* 对好友列表的管理类
* 每一位用户对应自己的好友列表，所以单独创建一个类对用户的好友列表进行管理
* 对私聊窗口的管理类
* 每一个用户和私聊对象对应一个私聊窗口，单独创建一个类标识用户编号及私聊对象编号，对私聊窗口进行管理
* 对服务器界面的管理类
* 在后续对服务器界面的文本框进行处理时，需要标识一开始启动的服务器，所以在这里创建一个类对服务器界面进行管理

### （2）消息包模块

* 定义用户信息类
* 用户在登录时需要输入账号、密码，所以在通用包里面创建一个类定义用户信息，存储用户账号、密码
* 定义消息包信息类
* 客户端和服务器在进行通信时传送消息包，创建一个类定义消息包，包括消息包的类型、发送端、接收端、消息本身内容
* 定义消息包类型的接口
* 客户端和服务器在进行通讯时有多种消息包类型，包括普通信息包、客户端要求得到在线好友的信息包、服务器返回在线好友的信息包、系统消息信息包、群消息信息包、下线信息包、强制下线信息包

### （3）服务器模块

* 服务器类
* 服务器在本地监听，设定端口号，等待连接，连接之后就可以接收客户端发来的消息，接收客户端发来的消息，返回一个信息报判断登录是否成功
* 这里采用对象流的方式进行信息传输，在网路间传递对象流，ois.readObject()得到一个对象，通过User.java定义对象类型，里面有用户名和密码，客户端通过ObjectOutputStream传送对象，注意对象要在网络上传输，要序列化，在类那里实现一个接口就可以，public class User implements java.io.Serializable
* 服务器和某个客户端的通信线程类
* 服务器为了可以完成信息转发，我们可以把服务器得到的每个socket和客户端。保存在HashMap，用客户的id标识该socket
* 当服务器和某个客户端形成一个连接时，服务器端就单开一个线程，让这个线程与该客户端通信
* 对服务器和客户端通信线程管理类
* 管理客户端的线程，有一个hashmap，这样就可以得到客户端的线程

### （4）客户端模块

* 客户端连接服务器类
* 在login的时候，ClientLogin中当用户点击登录时就要进行客户端与服务器的连接
* 验证客户端信息类
* 统一管理用户登录，进行check
* 客户端和服务器保持通讯的线程类
* 发送第一次请求，判断登录是否成功
* 验证用户登录的地方，验证成功，就创建一个该qq号和服务器端保持通讯连接的线程，放到hashmap中去
* 等待登录时申请的好友列表回复包
* 管理客户端和服务器保持通讯线程类
* 对客户端和服务器保持通讯的线程进行管理，放到HashMap中去

## 3.2 关键问题设计

### （1）多个客户端连接服务器

* 在服务器端加一个while循环，这样可以实现多个客户端连接

### （2）用户之间私聊

* 点击好友的时候要跳出一个聊天窗口
* 用户点击发送按钮后，通过管理线程的类得到对应的socket，然后得到输出流，客户端将先把消息包发给服务器
* 服务器端不停地接收来自客户端的消息，取出message中的信息，对消息进行类型判断，收到客户端发来的普通信息后，将会对消息进行转发，取得接收人的通讯线程后，将消息内容转发给它
* 客户端1发给服务器，服务器再完成转发任务发给客户端2，再将发送的信息显示在私聊窗口，这样就实现了两个客户端之间的私聊

### （3）多对多聊天

* 一个用户可能同时和多个好友进行私聊，所以这里服务器为了可以完成信息转发，我们可以把服务器得到的每个socket和客户端，保存在HashMap，用客户的id标识该socket
* 当服务器和某个客户端形成一个连接时，服务器端就单开一个线程，让这个线程与该客户端通信
* 把服务器与该客户端的连接赋给s，然后就不停等待客户端发消息过来，得到Sender和Getter和内容Con
* ManageClientThread.java，管理客户端的线程，有一个hashmap，这样就可以得到客户端的线程，这样就可以实现服务器对多个客户端进行转发
* 而同时客户端当一个新的账号登录，就给它创建一个新的线程，让这个线程和服务器保持通讯
* 再创建一个类，ManageClientConServerThread.java对客户端和服务器保持通讯的线程进行管理，放到HashMap中去
* 在ClientConServer.java，验证用户登录的地方，验证成功，就创建一个该qq号和服务器端保持通讯连接的线程，放到hashmap中去
* 在服务器到客户端、客户端到服务器都采用hashmap管理的方式对每个通讯线程进行管理，这样可以实现多个用户同时与多个好友聊天互不干扰，同时在实现后续功能时也提供了方便

### （4）登录时同步好友在线情况

* 在ClientLogin.java中，当用户登陆成功后，向服务器发送一个请求包，指明要的好友情况，要求返回在线好友的包，
* 在SerConClientThread.java中服务器接收到请求包后，就回送一个在线好友的信息报
* ClientConServerThread.java中客户端得到包后对调用friendList.java中的更新方法对好友列表进行处理
* 同时当用户登录成功后，在服务器系统公告栏和在线用户情况同步更新

### （5）系统公告、群发消息

* 系统公告：在myServer\_view.java中，输入内容后点击“系统公告”按钮，首先在服务器界面更新系统公告栏内容，接着服务器端得到在线的用户的线程，遍历hashmap，将系统公告转发给各用户，各客户端接收到服务器发来的消息包类型为系统公告后，调用FriendList.java中的showQgg()方法显示系统公告
* 群发消息：在客户端界面输入内容点击“发送群消息”按钮后，该客户端向服务器发送一个信息包指明消息发送者和消息内容，客户端接收到客户端发来的信息报类型为群消息后，得到在线的用户的线程，遍历hashmap，将群消息转发给各用户，各客户端接收到服务器发来的信息包类型为群消息后，调用FriendList.java中的showQxx()方法显示群消息

### （6）下线时在其他客户端上实时更新

* 在客户端添加监听系统关闭事件，当用户退出时给服务器端发出下线消息，消息包指明发送端和消息类型
* 服务器端接收到客户端发来的下线消息包后，得到在线的用户的线程，遍历hashmap，取出在线的用户的id进行消息包的转发，同时修改服务器端界面，在系统公告区显示该用户下线消息和更新在线用户情况
* 其他各客户端接收到服务器发送来的某用户下线的消息包后，修改相应的好友列表，更新在线好友情况，同时在群消息内容区显示该用户下线信息

### （7）服务器使用户强制下线

* 在服务器界面输入用户编号，点击“强制下线”按钮后，服务器首先向该客户端发送类型为被强制下线的消息包，接着在客户端和服务器两端管理线程的hashmap中删去该客户端与服务器连接的线程，接着服务器取出剩余在线的用户的id，向所有用户发送该用户下线的信息包
* 在服务器界面系统公告区显示该用户下线信息，同时更新服务器界面在线用户情况
* 被强制下线的客户得到服务器发来的被强制下线的信息包后，调用FriendList.java中的out()方法关闭当前界面
* 其他各客户端得到服务器发来的某用户下线的消息包后，修改相应的好友列表，更新在线好友情况，同时在群消息内容区显示该用户下线信息

# 第四章 聊天程序源代码清单

## 4.1 package qqclient\_view

### (1) ClientLogin.java:

/\*

qq客户端登录界面

\*/

package qqclient\_view;

import common.Message;

import common.MessageType;

import common.User;

import qqClient\_model.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.IOException;

import java.io.ObjectOutputStream;

public class ClientLogin extends JFrame implements ActionListener{

//定义北部需要的图片

JLabel jbl1;

//定义中部

//中部有三个JPanel，有一个选项卡窗口管理

JTabbedPane jtp;

JPanel jp2,jp3,jp4;

JLabel jp2\_jbl1,jp2\_jbl2,jp2\_jbl3,jp2\_jbl4;

JButton jp2\_jb1;

JTextField jp2\_jtf;

JPasswordField jp2\_jpf;

JCheckBox jp2\_jcb1,jp2\_jcb2;

//定义南部

JPanel jp1;

JButton jp1\_jb1,jp1\_jb2,jp1\_jb3;

public static void main(String[] args){

ClientLogin qqclientLogin = new ClientLogin();

}

public ClientLogin(){

//处理北边

jbl1 = new JLabel(new ImageIcon("src//image//tou.gif"));

//处理中部

jp2 = new JPanel(new GridLayout(3,3));

jp2\_jbl1 = new JLabel("QQ号码",JLabel.CENTER);

jp2\_jbl2 = new JLabel("QQ密码",JLabel.CENTER);

jp2\_jbl3 = new JLabel("忘记密码",JLabel.CENTER);

jp2\_jbl3.setForeground(Color.BLUE);

jp2\_jbl4 = new JLabel("申请密码保护",JLabel.CENTER);

jp2\_jb1 = new JButton("清除号码");

jp2\_jtf = new JTextField();

jp2\_jpf = new JPasswordField();

jp2\_jcb1 = new JCheckBox("隐身登录");

jp2\_jcb2 = new JCheckBox("记住密码");

//把控件按照顺序加入到jp2

jp2.add(jp2\_jbl1);

jp2.add(jp2\_jtf);

jp2.add(jp2\_jb1);

jp2.add(jp2\_jbl2);

jp2.add(jp2\_jpf);

jp2.add(jp2\_jbl3);

jp2.add(jp2\_jcb1);

jp2.add(jp2\_jcb2);

jp2.add(jp2\_jbl4);

jtp = new JTabbedPane();

jtp.add("QQ号码",jp2);

jp3 = new JPanel();

jp4 = new JPanel();

jtp.add("手机号码",jp3);

jtp.add("电子邮件",jp4);

//处理南部

jp1 = new JPanel(); //默认居中

jp1\_jb1 = new JButton("登录");

//响应用户点击登录

jp1\_jb1.addActionListener(this);

jp1\_jb2 = new JButton("取消");

jp1\_jb3 = new JButton("注册向导");

//三个按钮放入JPanel

jp1.add(jp1\_jb1);

jp1.add(jp1\_jb2);

jp1.add(jp1\_jb3);

this.add(jbl1,"North");

this.add(jp1,"South");

this.add(jtp,"Center");

jbl1.setSize(550,300);

this.setSize(550,500);

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

this.setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e) {

//当用户点击登录按钮

if(e.getSource() == jp1\_jb1){

ClientUser qqClientUser = new ClientUser();

User u = new User();

u.setUserId(jp2\_jtf.getText().trim()); //从输入框得到号码

u.setPasswd(new String(jp2\_jpf.getPassword()));

if(ClientUser.checkUser(u)){

try {

FriendList qqFriendList = new FriendList(u.getUserId());

ManageQqFriendList.addFriendList(u.getUserId(),qqFriendList);

//发送一个要求返回在线好友的请求包

ObjectOutputStream oos = new ObjectOutputStream(ManageClientConServerThread.getClientConServerThread(u.getUserId()).getS().getOutputStream());

//做一个message包

Message m = new Message();

m.setMesType(MessageType.message\_get\_onLineFriend);

//指明我要的是这个qq号的好友情况

m.setSender(u.getUserId());

oos.writeObject(m);

} catch (IOException ioException) {

ioException.printStackTrace();

}

//同时关闭这个登录界面

this.dispose();

}else{

//提示未成功

JOptionPane.showMessageDialog(this,"用户名密码错误");

}

}

}

}

### (2) FriendList.java:

/\*

好友列表，陌生人，黑名单

\*/

package qqclient\_view;

import common.Message;

import common.MessageType;

import qqClient\_model.ManageClientConServerThread;

import qqserver.model.ManageClientThread;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.util.HashMap;

import java.util.Iterator;

public class FriendList extends JFrame implements ActionListener,MouseListener{

//处理第一张卡片

JPanel jphy1,jphy2,jphy3,jphy4,jphy5,jphy6;

JButton jphy\_jb1,jphy\_jb2,jphy\_jb3;

JScrollPane jsp1,jsp3;

JTextArea qxx;

JTextField qxxfs;

JButton jphy4\_jbt;

String owner;

//处理第二张卡片陌生人

JPanel jpmsr1,jpmsr2,jpmsr3;

JButton jpmsr\_jb1,jpmsr\_jb2,jpmsr\_jb3;

JScrollPane jsp2;

JLabel[] jbls;

//把整个JFrame设置成CardLayout布局

CardLayout cl;

public static void main(String[] args){

//FriendList friendList = new FriendList();

}

public FriendList(String ownerId){

this.owner = ownerId;

//处理第一张卡片，显示好友列表

jphy1 = new JPanel(new BorderLayout());

//假定有50个好友

jphy2 = new JPanel(new GridLayout(50,1,4,4));;

//初始化50个好友

jbls = new JLabel[50];

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

jbls[i] = new JLabel(i+1+"",new ImageIcon("src//image//mm.jpg"),JLabel.LEFT);

jbls[i].setEnabled(false);

if(jbls[i].getText().equals(ownerId)){

jbls[i].setEnabled(true);

}

jbls[i].addMouseListener(this);

jphy2.add(jbls[i]);

}

jphy\_jb1 = new JButton("我的好友");

jphy\_jb2 = new JButton("陌生人");

jphy\_jb2.addActionListener(this);

jphy\_jb3 = new JButton("黑名单");

jphy3 = new JPanel(new GridLayout(2,1));

//把两个按钮加到jphy3

jphy3.add(jphy\_jb2);

jphy3.add(jphy\_jb3);

jphy4 = new JPanel(new BorderLayout());

jphy5 = new JPanel();

jphy5.setLayout(new BorderLayout());

jphy6 = new JPanel();

jphy6.setLayout(new GridLayout(2,1,10,10));

qxx = new JTextArea();

jsp3 = new JScrollPane(qxx);

qxxfs = new JTextField();

jphy4\_jbt = new JButton("发送群消息");

jphy4\_jbt.addActionListener(this);

jphy5.add(qxxfs,BorderLayout.CENTER);

jphy5.add(jphy4\_jbt,BorderLayout.EAST);

jphy4.add(jsp3,"Center");

jphy4.add(jphy5,"South");

jsp1 = new JScrollPane(jphy2);

jphy6.add(jsp1);

jphy6.add(jphy4);

//对jphy1初始化

jphy1.add(jphy\_jb1,"North");

jphy1.add(jphy6,"Center");

jphy1.add(jphy3,"South");

//对第二张卡片处理

jpmsr1 = new JPanel(new BorderLayout());

//假定有50个好友

jpmsr2 = new JPanel(new GridLayout(20,1,4,4));;

//初始20个陌生人

JLabel[] jbls2 = new JLabel[20];

for(int i=0; i < jbls2.length; i++){

jbls2[i] = new JLabel(i+1+"",new ImageIcon("src//image//gg.png"),JLabel.LEFT);

jpmsr2.add(jbls2[i]);

}

jpmsr\_jb1 = new JButton("好友");

jpmsr\_jb1.addActionListener(this);

jpmsr\_jb2 = new JButton("陌生人");

jpmsr\_jb3 = new JButton("黑名单");

jpmsr3 = new JPanel(new GridLayout(2,1));

//把两个按钮加到jphy3

jpmsr3.add(jpmsr\_jb1);

jpmsr3.add(jpmsr\_jb2);

jsp2 = new JScrollPane(jpmsr2);

//对jphy1初始化

jpmsr1.add(jpmsr3,"North");

jpmsr1.add(jsp2,"Center");

jpmsr1.add(jpmsr\_jb3,"South");

cl = new CardLayout();

this.setLayout(cl);

this.add(jphy1,"1");

this.add(jpmsr1,"2");

//在窗口显示自己的编号

this.setTitle(ownerId);

this.setSize(300,500);

this.setVisible(true);

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

//监听系统关闭事件，退出时给服务器端发出指定消息

this.addWindowListener(new WindowAdapter() {

@Override

public void windowClosing(WindowEvent e) {

//发送一个下线消息

ObjectOutputStream oos = null;

try {

oos = new ObjectOutputStream(ManageClientConServerThread.getClientConServerThread(ownerId).getS().getOutputStream());

} catch (IOException ioException) {

ioException.printStackTrace();

}

//做一个message包

Message m3 = new Message();

m3.setMesType(MessageType.message\_out);

//指明是谁发的

m3.setSender(ownerId);

m3.setCon(ownerId);

try {

oos.writeObject(m3);

} catch (IOException ioException) {

ioException.printStackTrace();

}

ManageQqFriendList.removeFriendList(ownerId);

ManageClientConServerThread.removeClientConServerThread(ownerId);

System.exit(0);

}

});

}

public void out(){

System.exit(0);

}

@Override

public void actionPerformed(ActionEvent e) {

//如果点击陌生人，显示第二张卡片

if(e.getSource()==jphy\_jb2){

cl.show(this.getContentPane(),"2");

}else if(e.getSource()==jpmsr\_jb1){

cl.show(this.getContentPane(),"1");

}else if(e.getSource() == jphy4\_jbt){

//发送一个群消息

ObjectOutputStream oos = null;

try {

oos = new ObjectOutputStream(ManageClientConServerThread.getClientConServerThread(owner).getS().getOutputStream());

} catch (IOException ioException) {

ioException.printStackTrace();

}

//做一个message包

Message m = new Message();

m.setMesType(MessageType.message\_qxx);

//指明是谁发的

m.setSender(this.owner);

m.setCon(qxxfs.getText());

try {

oos.writeObject(m);

} catch (IOException ioException) {

ioException.printStackTrace();

}

qxxfs.setText("");

}

}

@Override

public void mouseClicked(MouseEvent e) {

//响应用户双击的事件并得到好友的编号

if(e.getClickCount()==2){

//得到该好友的编号

String friendNo = ((JLabel)e.getSource()).getText();

//System.out.println("你希望和" + friendNo + "聊天");

qqchat qqc = new qqchat(this.owner,friendNo);

//把聊天界面加入到管理类

ManageQqChat.addQqChat(this.owner+" "+friendNo,qqc);

}

}

//更新在线的好友情况

public void updateFriendList(Message m){

String onLineFriend [] = m.getCon().split(" ");

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

jbls[Integer.parseInt(onLineFriend[i]) - 1].setEnabled(true);

}

}

//更新下线的好友

public void outFriend(Message m){

String outfriend = m.getCon();

jbls[Integer.parseInt(outfriend)-1].setEnabled(false);

}

//修改群公告

public void showQgg(Message m){

String info = "系统公告： " + m.getCon() + "\r\n";

this.qxx.append(info);

}

public void showQxx(Message m){

String info = m.getSender() + " 说 " + m.getCon() + "\r\n";

this.qxx.append(info);

}

public void showSx(String mes){

String info = mes + "上线啦！" + "\r\n";

this.qxx.append(info);

}

public void showXx(Message m){

String info = m.getCon() + "下线啦！" + "\r\n";

this.qxx.append(info);

}

@Override

public void mousePressed(MouseEvent e) {

}

@Override

public void mouseReleased(MouseEvent e) {

}

@Override

public void mouseEntered(MouseEvent e) {

JLabel jl = (JLabel)e.getSource();

jl.setForeground(Color.ORANGE);

}

@Override

public void mouseExited(MouseEvent e) {

JLabel jl = (JLabel)e.getSource();

jl.setForeground(Color.BLACK);

}

}

### (3) ManageQqChat.java:

/\*

这是一个用于管理用户聊天界面的类

\*/

package qqclient\_view;

import java.util.\*;

public class ManageQqChat {

private static HashMap hm = new HashMap<String,qqchat>();

//加入

public static void addQqChat(String loginIdAnFriendId,qqchat qq)

{

hm.put(loginIdAnFriendId,qq);

}

//取出

public static qqchat getQqChat(String loginIdAnFriendId)

{

return (qqchat)hm.get(loginIdAnFriendId);

}

}

### (4) ManageQqFriendList.java:

/\*

管理好友、黑名单、陌生人、界面类

\*/

package qqclient\_view;

import java.util.\*;

public class ManageQqFriendList {

private static HashMap hm = new HashMap<String, FriendList>();

public static void addFriendList(String qqId,FriendList qqFriendList){

hm.put(qqId,qqFriendList);

}

public static FriendList getQqFriendList(String qqId){

return (FriendList)hm.get(qqId);

}

public static void removeFriendList(String qqId){

hm.remove(qqId);

}

}

### (5) qqchat.java:

/\*

这是与好友聊天的界面,

因为客户端要处于读取的状态，因此我们把它做成一个线程

\*/

package qqclient\_view;

import common.Message;

import common.MessageType;

import qqClient\_model.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.IOException;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.util.\*;

public class qqchat extends JFrame implements ActionListener {

JTextArea jta;

JTextField jtf;

JButton jb;

JPanel jp;

String ownerId;

String friendId;

public static void main(String[] args){

//qqchat qq = new qqchat("1");

}

public qqchat(String ownerId,String friend){

this.ownerId = ownerId;

this.friendId = friend;

jta = new JTextArea();

jtf = new JTextField(15);

jb = new JButton("发送");

jb.addActionListener(this);

jp = new JPanel();

jp.add(jtf);

jp.add(jb);

this.add(jta,"Center");

this.add(jp,"South");

this.setTitle(ownerId + " 正在和 " + friend + " 聊天");

this.setIconImage((new ImageIcon("src//image//qe.jpg")).getImage());

this.setSize(300,200);

this.setVisible(true);

}

//写一个方法显示消息

public void showMessage(Message m){

String info = m.getSender() + " 对 " + m.getGetter() + " 说 " + m.getCon() + "\r\n";

this.jta.append(info);

}

@Override

public void actionPerformed(ActionEvent e) {

if(e.getSource() == jb)

{

//如果用户点击发送按钮

Message m = new Message();

m.setMesType(MessageType.message\_comm\_mes);

m.setSender(this.ownerId);

m.setGetter(this.friendId);

m.setCon(jtf.getText());

m.setSendTime(new Date().toString());

//发送给服务器

try {

//通过管理这个线程的类得到这个线程，然后通过getSocket得到对应的socket，然后得到输出流

ObjectOutputStream oos = new ObjectOutputStream(ManageClientConServerThread.getClientConServerThread(ownerId).getS().getOutputStream());

oos.writeObject(m);

} catch (IOException ioException) {

ioException.printStackTrace();

}

//显示

String info = m.getSender() + " 对 " + m.getGetter() + " 说 " + m.getCon() + "\r\n";

this.jta.append(info);

jtf.setText("");

}

}

}

## 4.2 package qqserver.view:

### (1) myServer\_view.java:

/\*

这是服务器的控制界面，可以完成启动服务器

\*/

package qqserver.view;

import common.Message;

import common.MessageType;

import qqclient\_view.FriendList;

import qqserver.model.\*;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.IOException;

import java.io.ObjectOutputStream;

import java.util.ArrayList;

import java.util.Collections;

import java.util.HashMap;

import java.util.Iterator;

public class myServer\_view extends JFrame implements ActionListener {

JPanel jp1;

JButton jb1,jb2;

JPanel jp2,jp3,jp4;

JButton jp2\_jb1,jp2\_jb2;

JButton jp3\_jb;

JTextField bh,xtxx;

JTextArea jta;

JTextArea jtalist;

JPanel jpc;

String id;

ArrayList<String> zxList = new ArrayList<String>();

public static void main(String[] args){

myServer\_view myserver = new myServer\_view("1");

ManageView.addView("1",myserver);

new myQqServer();

}

public myServer\_view(String id){

this.id = id;

jp1 = new JPanel();

jb1 = new JButton("启动服务器");

jb1.addActionListener(this);

jb2 = new JButton("关闭服务器");

jb2.addActionListener(this);

jp2 = new JPanel(new BorderLayout());

jp3 = new JPanel(new BorderLayout());

jp4 = new JPanel(new GridLayout(2,1,10,10));

jp2\_jb1 = new JButton("用户编号：");

bh = new JTextField();

jp2\_jb2 = new JButton("强制下线");

jp2\_jb2.addActionListener(this);

jp3\_jb = new JButton("系统公告");

jp3\_jb.addActionListener(this);

xtxx = new JTextField();

jp2.add(jp2\_jb1,BorderLayout.WEST);

jp2.add(bh,BorderLayout.CENTER);

jp2.add(jp2\_jb2,BorderLayout.EAST);

jp3.add(xtxx,BorderLayout.CENTER);

jp3.add(jp3\_jb,BorderLayout.EAST);

jp4.add(jp3);

jp4.add(jp2);

jp1.add(jb1);

jp1.add(jb2);

jta = new JTextArea();

jtalist = new JTextArea();

jpc = new JPanel(new GridLayout(1,2,10,10));

jpc.add(jta);

jpc.add(jtalist);

this.add(jp1,"North");

this.add(jpc,"Center");

this.add(jp4,"South");

this.setSize(500,400);

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

this.setVisible(true);

}

@Override

public void actionPerformed(ActionEvent e) {

if(e.getSource() == jb1){

System.out.println("启动成功");

}else if(e.getSource() == jp3\_jb){

try {

SendNotice();

} catch (IOException ioException) {

ioException.printStackTrace();

}

}else if(e.getSource() == jb2){

System.exit(0);

}else if(e.getSource() == jp2\_jb2){

String outId = bh.getText();

Message m = new Message();

m.setCon(outId);

m.setMesType(MessageType.message\_selfout);

ObjectOutputStream oos = null;

try {

oos = new ObjectOutputStream(ManageClientThread.getClientThread(outId).getS().getOutputStream());

} catch (IOException ioException) {

ioException.printStackTrace();

}

m.setGetter(outId);

try {

oos.writeObject(m);

} catch (IOException ioException) {

ioException.printStackTrace();

}

ManageClientThread.removeClientThread(outId);

HashMap hm = ManageClientThread.hm;

Iterator it = hm.keySet().iterator();

Message m4 = new Message();

m4.setCon(outId);

m4.setMesType(MessageType.message\_out);

while(it.hasNext()){

//取出在线的人的id

String onLineUserId = it.next().toString();

ObjectOutputStream ooos = null;

try {

ooos = new ObjectOutputStream(ManageClientThread.getClientThread(onLineUserId).getS().getOutputStream());

} catch (IOException ioException) {

ioException.printStackTrace();

}

m4.setGetter(onLineUserId);

try {

ooos.writeObject(m4);

} catch (IOException ioException) {

ioException.printStackTrace();

}

}

jta.append(outId + "下线啦!" + "\n");

zxList.remove(outId);

jtalist.setText("");

Collections.sort(zxList);

jtalist.append("在线用户： \n");

for (Iterator iter = zxList.iterator(); iter.hasNext();) {

jtalist.append(iter.next().toString()+"\n");

}

bh.setText("");

}

}

//发送系统公告

public void SendNotice() throws IOException {

//得到在线的人的线程

HashMap hm = ManageClientThread.hm;

Iterator it = hm.keySet().iterator();

while(it.hasNext()){

Message m = new Message();

m.setCon(xtxx.getText());

m.setMesType(MessageType.message\_xtxx);

//取出在线的人的id

String onLineUserId = it.next().toString();

ObjectOutputStream oos = new ObjectOutputStream(ManageClientThread.getClientThread(onLineUserId).getS().getOutputStream());

m.setGetter(onLineUserId);

oos.writeObject(m);

}

jta.append("系统公告： " + xtxx.getText() + "\n");

xtxx.setText("");

}

public void showXt(String mes){

String info = mes + "上线啦！" + "\n";

jta.append(info);

zxList.add(mes);

jtalist.setText("");

Collections.sort(zxList);

jtalist.append("在线用户： \n");

for (Iterator iter = zxList.iterator(); iter.hasNext();) {

jtalist.append(iter.next().toString()+"\n");

}

}

public void showXx(String mes){

String info = mes + "下线啦！" + "\n";

zxList.remove(mes);

jta.append(info);

jtalist.setText("");

Collections.sort(zxList);

jtalist.append("在线用户： \n");

for (Iterator iter = zxList.iterator(); iter.hasNext();) {

jtalist.append(iter.next().toString()+"\n");

}

}

}

### (2) ManageView.java:

package qqserver.view;

import java.util.HashMap;

public class ManageView {

private static HashMap hm = new HashMap<String, myServer\_view>();

//加入

public static void addView(String id,myServer\_view view)

{

hm.put(id,view);

}

//取出

public static myServer\_view getView(String id)

{

return (myServer\_view)hm.get(id);

}

}

## 4.3 package qqserver.model:

### (1) myQqServer.java:

/\*

这是qq服务器，监听等待某个qq客户端连接

\*/

package qqserver.model;

import common.\*;

import java.net.\*;

import java.io.\*;

import java.util.\*;

public class myQqServer {

public static void main(String[] args){

}

public myQqServer(){

try{

//在9999监听

System.out.println("我是服务器，在9999监听..");

ServerSocket ss = new ServerSocket(9999);

//阻塞，等待连接

while(true)

{

Socket s = ss.accept();

//接收客户端发来的信息

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

User u = (User) ois.readObject();

System.out.println("服务器接收到用户id:" + u.getUserId() + " 密码:" + u.getPasswd());

Message m = new Message();

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

if (u.getPasswd().equals("123456"))

{

//返回一个成功登录的信息报

m.setMesType("1");

oos.writeObject(m);

//这里就单开一个线程，让该线程与该客户端保持通信

SerConClientThread scct = new SerConClientThread(s);

ManageClientThread.addClientThread(u.getUserId(),scct); //放到那份hashmap里面

//启动与该客户端通讯的线程

scct.start();

//并通知其它在线用户

scct.notifyOther(u.getUserId());

} else

{

m.setMesType("2");

oos.writeObject(m);

//关闭连接,进行下一次连接

s.close();

}

}

}catch (Exception e)

{

e.printStackTrace();

}finally {

}

}

}

### (2) SerConClientThread.java:

/\*

功能：服务器和某个客户端的通讯线程

\*/

package qqserver.model;

import common.Message;

import common.MessageType;

import qqClient\_model.ManageClientConServerThread;

import qqclient\_view.ManageQqFriendList;

import qqserver.view.ManageView;

import qqserver.view.myServer\_view;

import java.io.\*;

import java.net.\*;

import java.util.HashMap;

import java.util.Iterator;

public class SerConClientThread extends Thread{

myServer\_view mview = null;

Socket s;

public SerConClientThread(Socket s)

{

//把服务器与该客户端的连接赋给s

this.s = s;

}

//让该线程去通知其它用户

public void notifyOther(String iam) throws IOException {

//得到在线的人的线程

HashMap hm = ManageClientThread.hm;

Iterator it = hm.keySet().iterator();

while(it.hasNext()){

Message m = new Message();

m.setSender(iam);

m.setCon(iam);

m.setMesType(MessageType.message\_ret\_onLineFriend);

//取出在线的人的id

String onLineUserId = it.next().toString();

ObjectOutputStream oos = new ObjectOutputStream(ManageClientThread.getClientThread(onLineUserId).s.getOutputStream());

m.setGetter(onLineUserId);

oos.writeObject(m);

}

}

public Socket getS() {

return s;

}

public void run()

{

while(true)

{

//这里该线程就可以接收客户端的信息

try {

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

Message m = (Message) ois.readObject();

//取出message中信息,确认服务器拿到了信息

System.out.println(m.getSender() + " 给 " + m.getGetter() + " 说 " + m.getCon());

//对从客户端取得的消息进行类型判断，然后做相应的处理

if(m.getMesType().equals(MessageType.message\_comm\_mes)){

//一会完成转发任务

//取得接收人的通讯线程

SerConClientThread sc = ManageClientThread.getClientThread(m.getGetter());

ObjectOutputStream oos = new ObjectOutputStream(sc.s.getOutputStream());

oos.writeObject(m);

}

else if(m.getMesType().equals(MessageType.message\_get\_onLineFriend)){

System.out.println(m.getSender() + "要他的好友");

//把在服务器的好友给该客户端返回

mview = ManageView.getView("1");

mview.showXt(m.getSender());

String res = ManageClientThread.getAllOnLineUserid();

Message mnew = new Message();

mnew.setCon(res);

mnew.setSender(m.getSender());

mnew.setMesType(MessageType.message\_ret\_onLineFriend);

mnew.setGetter(m.getSender());

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject(mnew);

}else if(m.getMesType().equals(MessageType.message\_qxx)){

System.out.println(m.getSender() + "发送群消息");

//得到在线的人的线程

HashMap hm = ManageClientThread.hm;

Iterator it = hm.keySet().iterator();

while(it.hasNext()){

//取出在线的人的id

String onLineUserId = it.next().toString();

ObjectOutputStream oos = new ObjectOutputStream(ManageClientThread.getClientThread(onLineUserId).getS().getOutputStream());

m.setGetter(onLineUserId);

oos.writeObject(m);

}

}else if(m.getMesType().equals(MessageType.message\_out)){

//得到在线的人的线程

ManageClientThread.removeClientThread(m.getCon());

HashMap hm = ManageClientThread.hm;

Iterator it = hm.keySet().iterator();

while(it.hasNext()){

//取出在线的人的id

String onLineUserId = it.next().toString();

ObjectOutputStream oos = new ObjectOutputStream(ManageClientThread.getClientThread(onLineUserId).getS().getOutputStream());

m.setGetter(onLineUserId);

oos.writeObject(m);

}

mview = ManageView.getView("1");

mview.showXx(m.getSender());

}

} catch (IOException | ClassNotFoundException e) {

e.printStackTrace();

}

}

}

}

### (3) ManageClientThread.java:

package qqserver.model;

import java.util.\*;

public class ManageClientThread {

public static HashMap hm = new HashMap<String,SerConClientThread>();

//向hm中添加一个客户端通讯线程

public static void addClientThread(String uid,SerConClientThread ct)

{

hm.put(uid,ct);

}

public static SerConClientThread getClientThread(String uid)

{

return (SerConClientThread)hm.get(uid);

}

public static void removeClientThread(String uid)

{

hm.remove(uid);

}

//返回当前在线的人的情况

public static String getAllOnLineUserid(){

//用迭代器返回HashMap的k值

Iterator it = hm.keySet().iterator();

String res = "";

while(it.hasNext())

{

res += it.next().toString()+" ";

}

return res;

}

}

## 4.4 package qqClient\_model:

### (1) ClientConServer.java:

/\*

这是客户端连接服务器的后台

\*/

package qqClient\_model;

import common.Message;

import common.User;

import qqclient\_view.qqchat;

import java.net.\*;

import java.io.\*;

import java.util.\*;

public class ClientConServer {

public Socket s;

//发送第一次请求

public boolean SendLoginInfoToServer(Object o){

boolean b = false;

try{

System.out.println("kk");

s = new Socket("127.0.0.1",9999);

ObjectOutputStream oos = new ObjectOutputStream(s.getOutputStream());

oos.writeObject(o);

//发出一个信息后将收到一个信息

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

Message ms = (Message)ois.readObject(); //转换成Message

//这里就是验证用户登录的地方

if(ms.getMesType().equals("1")){

//创建一个该qq号和服务器端保持通讯连接的线程

ClientConServerThread ccst = new ClientConServerThread(s);

//启动该通讯线程

ccst.start();

//把object对象先转为User型，再拿到它的qq号

ManageClientConServerThread.addClientConServerThread(((User)o).getUserId(),ccst);

b = true;

}else{

//关闭socket

s.close();

}

}catch(Exception e){

e.printStackTrace();

}finally{

}

return b;

}

public void SendInfoToServer(Object o){

try{

Socket s = new Socket("127.0.0.1",9999);

}catch(Exception e){

e.printStackTrace();

}finally{

}

}

}

### (2) ClientUser.java:

/\*

验证该用户是否合格

\*/

package qqClient\_model;

import common.\*;

import java.io.ObjectInputStream;

import java.io.ObjectOutputStream;

import java.net.Socket;

public class ClientUser {

public static boolean checkUser(User u){

return new ClientConServer().SendLoginInfoToServer(u);

}

}

### (3) ClientConServerThread.java:

/\*

客户端和服务器端保持通讯的线程

\*/

package qqClient\_model;

import common.Message;

import common.MessageType;

import qqclient\_view.FriendList;

import qqclient\_view.ManageQqChat;

import qqclient\_view.ManageQqFriendList;

import qqclient\_view.qqchat;

import java.net.Socket;

import java.io.\*;

public class ClientConServerThread extends Thread {

private Socket s;

public Socket getS() {

return s;

}

//构造函数

public ClientConServerThread(Socket s)

{

this.s = s;

}

public void run()

{

while (true)

{

//不停的读取从服务器端发来的消息

try{

ObjectInputStream ois = new ObjectInputStream(s.getInputStream());

Message m = (Message)ois.readObject();

//System.out.println("读取到从服务器发来的消息" + m.getSender() + " 给 " + m.getGetter() + " 内容 " + m.getCon());

if(m.getMesType().equals(MessageType.message\_comm\_mes)){

//把从服务器获得的消息显示到该显示的聊天界面

qqchat qqc = ManageQqChat.getQqChat(m.getGetter() + " " +m.getSender());

//显示

qqc.showMessage(m);

}else if(m.getMesType().equals(MessageType.message\_ret\_onLineFriend))

{

System.out.println("客户端接收到" + m.getCon());

String con = m.getCon();

String friends[] = con.split(" ");

String getter = m.getGetter();

System.out.println("getter = " + getter);

//修改相应的好友列表

FriendList qqFriendList = ManageQqFriendList.getQqFriendList(getter);

if(qqFriendList != null){

//更新在线好友

qqFriendList.updateFriendList(m);

}

qqFriendList.showSx(m.getSender());

}else if(m.getMesType().equals(MessageType.message\_xtxx))

{

System.out.println("客户端接收到系统公告" + m.getCon());

String con = m.getCon();

String geter = m.getGetter();

System.out.println("geter = " + geter);

FriendList frilist = ManageQqFriendList.getQqFriendList(geter);

frilist.showQgg(m);

}else if(m.getMesType().equals(MessageType.message\_qxx))

{

System.out.println("客户端接收到群消息" + m.getCon());

String con = m.getCon();

String geter = m.getGetter();

System.out.println("geter = " + geter);

FriendList frilist = ManageQqFriendList.getQqFriendList(geter);

frilist.showQxx(m);

}else if(m.getMesType().equals(MessageType.message\_out)){

String outer = m.getCon();

System.out.println("outer = " + outer);

String gettter = m.getGetter();

//修改相应的好友列表

FriendList qqFriendList = ManageQqFriendList.getQqFriendList(gettter);

if(qqFriendList != null){

//更新在线好友

qqFriendList.outFriend(m);

}

qqFriendList.showXx(m);

}else if(m.getMesType().equals(MessageType.message\_selfout)){

System.out.println(m.getCon() + "被强制下线");

String com = m.getCon();

String geter = m.getGetter();

System.out.println("geter = " + geter);

FriendList frilist = ManageQqFriendList.getQqFriendList(geter);

frilist.out();

}

}catch (Exception e){

e.printStackTrace();

}

}

}

}

### (4) ManageClientConServerThread.java:

/\*

这是一个管理客户端和服务器保持通讯的线程类

\*/

package qqClient\_model;

import java.util.\*;

public class ManageClientConServerThread {

private static HashMap hm = new HashMap<String,ClientConServerThread>();

//把创建好的通讯线程ClientConServerThread放入到hm中

public static void addClientConServerThread(String qqId,ClientConServerThread ccst)

{

hm.put(qqId,ccst);

}

//可与通过qqId取得该线程

public static ClientConServerThread getClientConServerThread(String qqId)

{

return (ClientConServerThread)hm.get(qqId);

}

public static void removeClientConServerThread(String qqId)

{

hm.remove(qqId);

}

}

## 4.5 package common:

### (1) Message.java:

/\*

mesType1 →表示登录成功

mesType2 →表示登录失败

mesType3 →表示是普通的消息包

\*/

package common;

import java.util.Date;

public class Message implements java.io.Serializable{

private String mesType;

private String sender;

private String getter;

private String con; //消息本身内容

private String sendTime;

public String getMesType() {

return mesType;

}

public void setMesType(String mesType) {

this.mesType = mesType;

}

public String getCon() {

return con;

}

public String getGetter() {

return getter;

}

public String getSender() {

return sender;

}

public String getSendTime() {

return sendTime;

}

public void setCon(String con) {

this.con = con;

}

public void setGetter(String getter) {

this.getter = getter;

}

public void setSender(String sender) {

this.sender = sender;

}

public void setSendTime(String sendTime) {

this.sendTime = sendTime;

}

}

### (2) MessageType.java:

/\*

定义包的种类

\*/

package common;

public interface MessageType {

String message\_succeed = "1"; //登陆成功

String message\_login\_fail = "2";//登陆失败

String message\_comm\_mes = "3"; //普通信息包

String message\_get\_onLineFriend = "4";//要求在线好友的包

String message\_ret\_onLineFriend = "5";//返回在线好友的包

String message\_xtxx = "6"; //系统消息

String message\_qxx = "7"; //群消息

String message\_out = "8"; //下线

String message\_selfout = "9";//被强制下线

}

### (3) User.java:

/\*

这是用户信息类

\*/

package common;

import qqclient\_view.qqchat;

public class User implements java.io.Serializable{//序列化，让一个对象可以在网络中传输

private String userId;

private String passwd;

public String getPasswd() {

return passwd;

}

public String getUserId() {

return userId;

}

public void setPasswd(String passwd) {

this.passwd = passwd;

}

public void setUserId(String userId) {

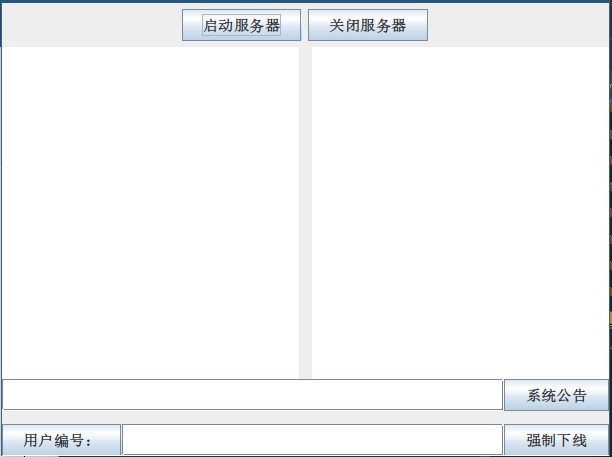
this.userId = userId;

}

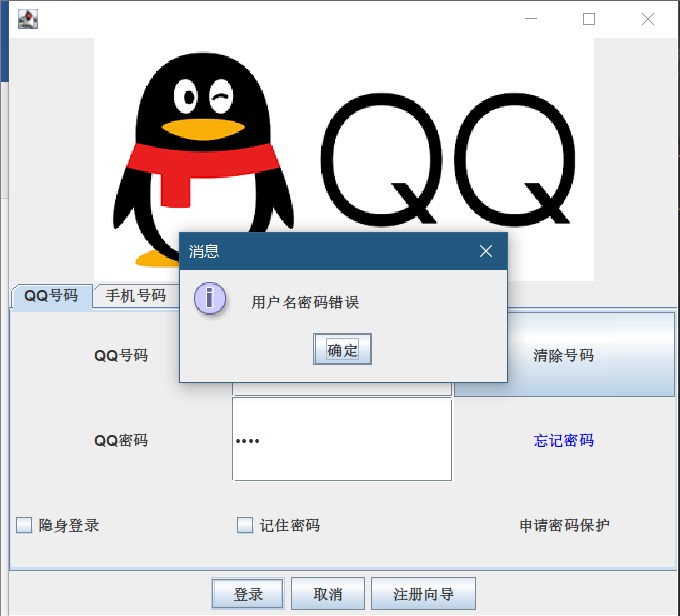
}

# 第五章 聊天程序运行结果与测试分析

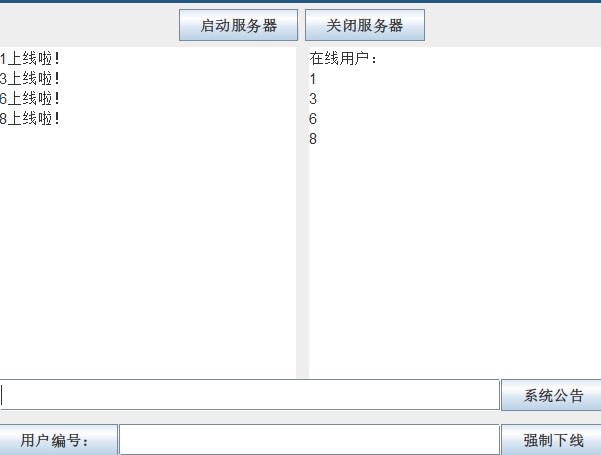
* 启动服务器：

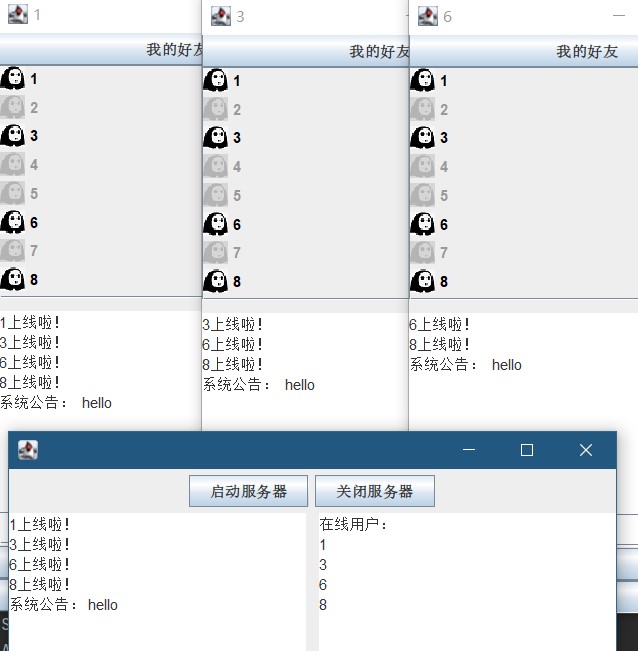


* 用户登录界面，验证密码：



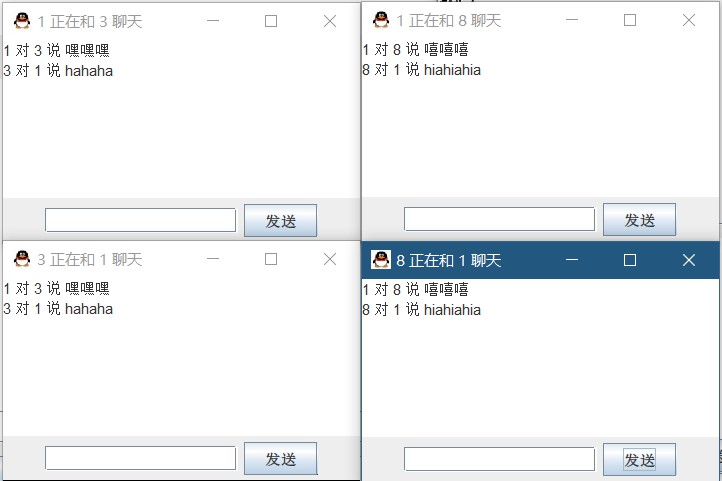
* 登录成功，更新在线好友列表：



* 发送系统公告，发送群聊消息：
* 

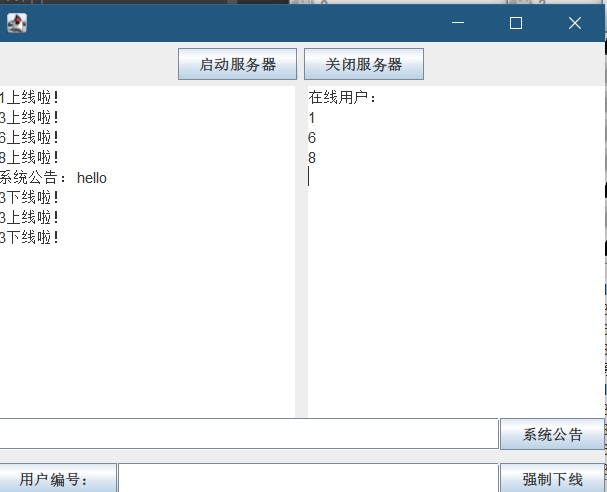


* 好友私聊：

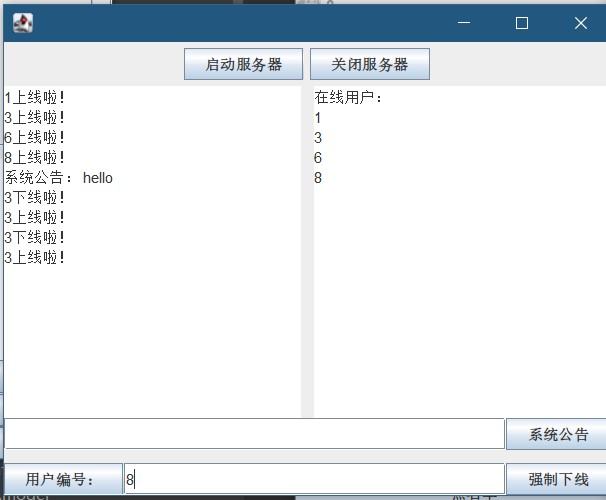


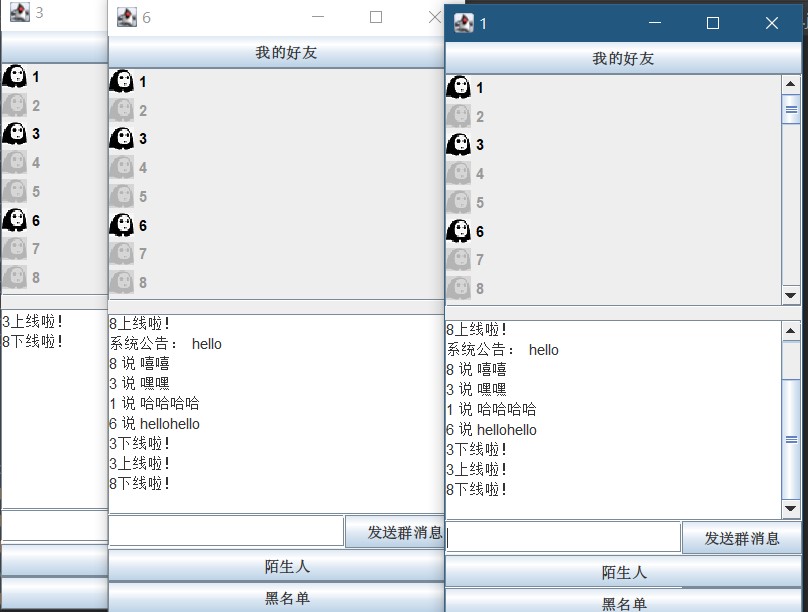
* 3下线，更新其他用户好友情况：





* 让3上线，服务器让8强制下线：





# 第六章 网页爬虫程序业务分析

* 编写界面，输入一个网址，能够爬取该网址上所有的HTML源代码。
* 对网址中的文本进行提取。
* 建立敏感词库，用文本文件保存。
* 将该网址所对应的文本中的敏感词提取并高亮显示。
* 编写文本文件，可以存入多个网址；程序可爬取这些网址中的文本内容，将敏感词记录存入另一个文件，格式自定。
* 编写一个主界面,整合上述功能

# 第七章 网页爬虫程序系统设计

## 7.1 系统功能定义

### （1）界面模块

* 用户输入需要爬取的网址，点击爬虫按钮
* 文本区显示爬取的网页源代码、纯文本
* 用户选择敏感词库
* 匹配后高亮显示文本中的敏感词，显示所有出现的敏感词
* 用户选择网址文件进行爬取，与敏感词库进行匹配
* 统计每个网址文本中敏感词出现的次数，保存统计文件

### （2）单个网址爬虫模块

* 对用户输入的网址进行源代码爬取，保存源代码
* 对用户输入的网址爬取后仅保存文本部分
* 将保存的文本部分与选取的敏感词库进行匹配，高亮显示文本中的敏感词

### （3）敏感词匹配模块

* 选取敏感词库文件，读取文件中敏感词
* 采用DFA算法对文本和敏感词库进行匹配，这里采用最大匹配规则

### （4）文件爬虫模块

* 选取网址文件，读取文件内容
* 遍历文件中所有网址与敏感词库进行匹配，统计敏感词出现的次数
* 将统计结果保存在与程序同文件夹下的data.txt中

## 7.2 关键问题设计

### （1）爬取网页源代码

* 这里在idea中导入jsoup-1.13.1.jar,利用jsoup自带的爬虫功能爬取网页源代码：Document doc = Jsoup.connect(weburl).get();

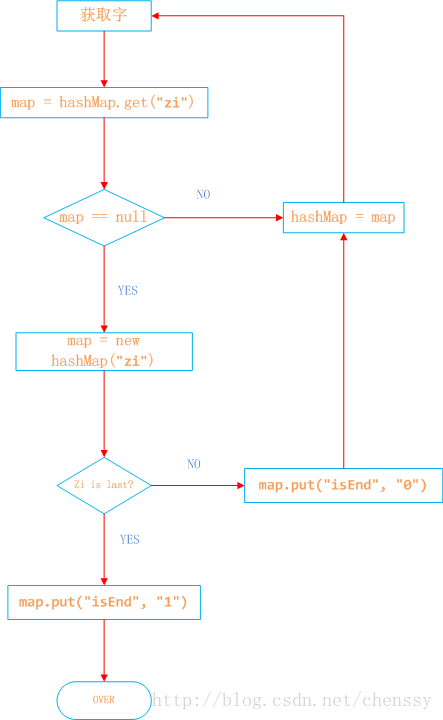
### （2）爬取网页文本

* 利用jsoup自带的爬虫功能获取网页源代码后保存标题和正文部分：Document doc = Jsoup.connect(weburl).get();

String title = doc.title();

String text = title + doc.text();

### （3）敏感词匹配

* 这里采用DFA算法，首先初始化DFA树，接着检查敏感词，1. 是否包含敏感词。 2. 敏感词的长度3. 正常走过字段的长度（便于后期替换优化，避免不必要的循环重复）
* 

### （4）爬取文件中网址文本与敏感词库匹配并保存统计结果

* 打开网址文件，读取文件内容，遍历所有网址，将每个网址爬取到的文本结果与敏感词库进行匹配，保存每个敏感词出现次数，并将每一个网址的统计结果保存在data.txt中

# 第八章 爬虫程序源代码清单

package com.sxf;

import org.jsoup.Jsoup;

import org.jsoup.nodes.Document;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.io.\*;

import java.util.\*;

import javax.swing.\*;

import javax.swing.text.BadLocationException;

import javax.swing.text.DefaultHighlighter;

import javax.swing.text.Highlighter;

import java.util.ArrayList;

public class gui extends JFrame implements ActionListener {

private JPanel jpanel=new JPanel();

private JPanel jptop = new JPanel();

private JPanel jptopright = new JPanel();

private JPanel jphtml = new JPanel();

private JPanel jptext = new JPanel();

private JPanel jpsen = new JPanel();

private JPanel jpword1 = new JPanel();

private JPanel jpword2 = new JPanel();

private JPanel jpdown = new JPanel();

private JTabbedPane alterPane=new JTabbedPane();

private JPanel jpweb = new JPanel();

private JLabel lburl = new JLabel("输入网址");

private JTextField tfurl = new JTextField(10);

private JScrollPane spurl=new JScrollPane(tfurl);

private JButton goSpider=new JButton("开始爬取");

private JButton urls = new JButton("打开网址库");

private JTextArea htmlArea=new JTextArea(15, 25);

private JScrollPane htmlSPane=new JScrollPane(htmlArea);

private JTextArea textArea=new JTextArea(15,25);

private JScrollPane textSPane=new JScrollPane(textArea);

private JTextArea senArea=new JTextArea(15,25);

private JScrollPane senSPane=new JScrollPane(senArea);

private JButton words = new JButton("选择敏感词库");

private JButton match = new JButton("提取敏感词");

private JTextArea senword = new JTextArea();

private JScrollPane wordspane = new JScrollPane(senword);

Set<String> set = null;

HashMap senWordMap;

ArrayList<String> sensList = new ArrayList<String>();

ArrayList<String> senWordList = new ArrayList<String>();

ArrayList<Integer> wordNum = new ArrayList<Integer>();

public gui(){

this.setTitle("Spider");

this.setLocation(400, 200);

this.setSize(600, 500);

this.setDefaultCloseOperation(EXIT\_ON\_CLOSE);

jpanel.setLayout(new BorderLayout());

//顶部,输入网址，开始爬取

jptop.setLayout(new BorderLayout());

lburl.setPreferredSize(new Dimension(70,30));

spurl.setPreferredSize(new Dimension(300,30));

goSpider.setPreferredSize(new Dimension(90,30));

urls.setPreferredSize(new Dimension(90,30));

jptopright.setLayout(new GridLayout(1,2,10,10));

jptopright.add(goSpider);

jptopright.add(urls);

jptop.add(lburl,BorderLayout.WEST);

jptop.add(spurl,BorderLayout.CENTER);

jptop.add(jptopright,BorderLayout.EAST);

//html源代码及抽取的文本

htmlArea.setEditable(false);

htmlArea.setLineWrap(true); //文本区域的换行策略

htmlArea.setFont(new Font("楷体",Font.PLAIN,14));

jphtml.setLayout(new BorderLayout());

jphtml.add(htmlSPane,BorderLayout.CENTER);

textArea.setFont(new Font("楷体",Font.PLAIN,14));

textArea.setEditable(false);

textArea.setLineWrap(true);

jptext.setLayout(new BorderLayout());

jptext.add(textSPane,BorderLayout.CENTER);

senArea.setFont(new Font("楷体",Font.PLAIN,14));

senArea.setEditable(false);

senArea.setLineWrap(true);

jpsen.setLayout(new BorderLayout());

jpsen.add(senSPane,BorderLayout.CENTER);

alterPane.add("HTML源代码",jphtml);

alterPane.add("文本",jptext);

alterPane.add("敏感词汇",jpsen);

jpweb.setLayout(new BorderLayout());

jpweb.add(alterPane,BorderLayout.CENTER);

//敏感词库和提取敏感词汇

senword.setLineWrap(true);

senword.setEditable(false);

wordspane.setPreferredSize(new Dimension(6,400));

jpword1.setLayout(new GridLayout(2,1,10,10));

jpword1.add(words);

jpword1.add(match);

jpword2.setLayout(new BorderLayout());

jpword2.add(jpword1,BorderLayout.NORTH);

jpword2.add(wordspane,BorderLayout.CENTER);

jpdown.setLayout(new BorderLayout());

jpdown.add(jpweb,BorderLayout.CENTER);

jpdown.add(jpword2,BorderLayout.EAST);

jpanel.add(jptop,BorderLayout.NORTH);

jpanel.add(jpdown,BorderLayout.CENTER);

this.add(jpanel);

this.setVisible(true);

//事件

goSpider.addActionListener((ActionListener) this);

urls.addActionListener((ActionListener) this);

words.addActionListener((ActionListener) this);

match.addActionListener((ActionListener) this);

}

public static void main(String[] args){

new gui();

}

@Override

public void actionPerformed(ActionEvent e) {

JButton j = (JButton)e.getSource();

if(j == goSpider){

String weburll = tfurl.getText();

new urlpage(weburll).start();

}else if(j == words){

try {

readSenWordFile();

} catch (Exception exception) {

exception.printStackTrace();

}

}else if (j == match){

hashinit(set);

String weburl = tfurl.getText();

try {

getSenWord((gettext(weburl)),2);

} catch (IOException ioException) {

ioException.printStackTrace();

}

try {

showSenWord();

} catch (BadLocationException badLocationException) {

badLocationException.printStackTrace();

}

String br = senWordList.toString();

senArea.append(br);

}else if(j == urls){

spiderurls();

}

}

public String getHtml(String weburl) throws IOException {

System.out.println("开始爬取");

Document doc = Jsoup.connect(weburl).get();

String html = doc.html();

System.out.println("爬取结束");

return html;

}

public String gettext(String weburl) throws IOException {

Document doc = Jsoup.connect(weburl).get();

String title = doc.title();

String text = title + doc.text();

return text;

}

private void hashinit(Set<String> keyWordSet){

senWordMap = new HashMap(keyWordSet.size());

String key = null;

Map nowMap = null;

Map<String,String> newWordMap = null;

Iterator<String> iterator = keyWordSet.iterator();

while(iterator.hasNext()){

key = iterator.next();

nowMap = senWordMap;

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

char keyChar = key.charAt(i);

Object wordMap = nowMap.get(keyChar);

if(wordMap != null){

nowMap = (Map) wordMap;//存在该key，直接赋值

}else{

newWordMap = new HashMap<String,String>();

newWordMap.put("isEnd", "0");

nowMap.put(keyChar,newWordMap);

nowMap = newWordMap;

}

if(i == key.length()-1){

nowMap.put("isEnd","1");

}

}

}

}

private void readSenWordFile() throws Exception {

String ENCODING = "UTF-8";

JFileChooser fchooser = new JFileChooser(); //文件选择

int ok = fchooser.showOpenDialog(this);

if(ok != JFileChooser.APPROVE\_OPTION) return ; //判断是否正常选择

File file = fchooser.getSelectedFile();//获取选择的文件

InputStreamReader read = new InputStreamReader(new FileInputStream(file),ENCODING);

try {

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

set = new HashSet<String>();

BufferedReader bufferedReader = new BufferedReader(read);

String txt = null;

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

set.add(txt);

sensList.add(txt);

wordNum.add(0);

senword.append(txt + "\n");

}

} else {

throw new Exception("敏感词库文件不存在");

}

}catch(Exception e){

throw e;

}finally{

read.close();

}

}

public void getSenWord(String txt, int matchType){

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

int length = CheckSenWord(txt, i, matchType);

if(length > 0){

senWordList.add(txt.substring(i,i+length));

i = i + length - 1;

}

}

}

public int CheckSenWord(String txt, int beginIndex, int matchType){

boolean flag = false;

int matchFlag = 0;

char word = 0;

int minMatchTYpe = 1; //最小匹配规则

int maxMatchType = 2;

Map nowMap = senWordMap;

for(int i = beginIndex; i < txt.length(); i++){

word = txt.charAt(i);

nowMap = (Map)nowMap.get(word); //获取key

if(nowMap != null){

matchFlag++;

if("1".equals(nowMap.get("isEnd"))){

flag = true;

if(minMatchTYpe == matchType){

break;

}

}

}else{

break;

}

}

if(matchFlag < 2 || !flag){

matchFlag = 0;

}

return matchFlag;

}

public void showSenWord() throws BadLocationException {

Highlighter hg = textArea.getHighlighter();

hg.removeAllHighlights();

String text = textArea.getText();

DefaultHighlighter.DefaultHighlightPainter painter = new DefaultHighlighter.DefaultHighlightPainter(Color.YELLOW);

for(String str:sensList){

int index = 0;

while((index = text.indexOf(str,index)) >= 0){

hg.addHighlight(index,index+str.length(),painter);

index += str.length();

}

}

}

public void spiderurls(){

if(wordNum.size() <= 0){

JOptionPane.showMessageDialog(null,"请先选择敏感词库");

return;

}

JFileChooser fchooser = new JFileChooser();

int ok = fchooser.showOpenDialog(this);

if(ok != JFileChooser.APPROVE\_OPTION) return;

File file = fchooser.getSelectedFile();

new spiderurls(this, file).start();

}

class spiderurls extends Thread{

private File file = null;

public spiderurls(gui gui, File f){

file = f;

}

public void run(){

try{

BufferedReader brr = new BufferedReader(new FileReader(file));

PrintStream ps = new PrintStream(new File("data.txt"));

ps.println("敏感词记录如下：");

int size = sensList.size();

int j = 10;

while(j>0){

j--;

String web = brr.readLine();

if(web == null) break;

ps.println(web + "数据如下：");

String html = getHtml(web);

String text = gettext(web);

for(int i=0;i<size;i++) { //在网页文本中进行匹配

String word=sensList.get(i);

int index=0,account=0,len=word.length();

while((index=text.indexOf(word,index))>=0) {

account++;

int temp=wordNum.get(i); //更新数据

wordNum.set(i,++temp);

index+=len; //更新匹配条件

}

ps.println(word+" 出现 "+account+"次"); //写入当前数据

}

ps.println();

}

brr.close();

System.out.println("爬取完毕");

ps.close();

JOptionPane.showMessageDialog(null,"爬取完毕！请打开文件查看!");

} catch (FileNotFoundException e) {

e.printStackTrace();

} catch (IOException e) {

e.printStackTrace();

}

}

}

class urlpage extends Thread{

private String weburl = null;

public urlpage(String s){

weburl = s;

}

public void run(){

if(weburl.length() <= 0){

JOptionPane.showMessageDialog(null,"网址不能为空");

return;

}

htmlArea.setText("");//清除

textArea.setText("");

String html = null;

try {

html = getHtml(weburl);

} catch (IOException e) {

e.printStackTrace();

}

if(html.length() > 0){

JOptionPane.showMessageDialog(null,"爬取完毕");

htmlArea.append(html);

String text = null;

try {

text = gettext(weburl);

} catch (IOException e) {

e.printStackTrace();

}

textArea.append(text);

/\* getSenWord(text,2);

String br = senWordList.toString();

senArea.append(br);

\*/

}

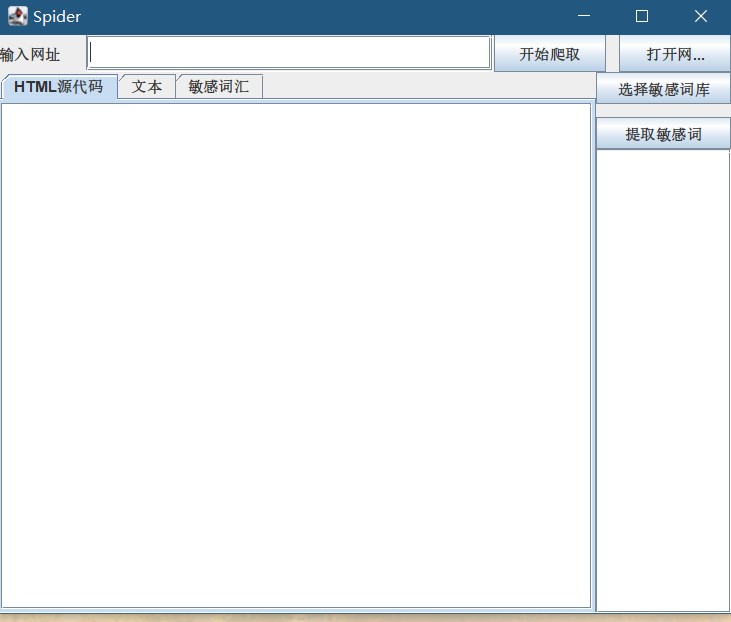
}

}

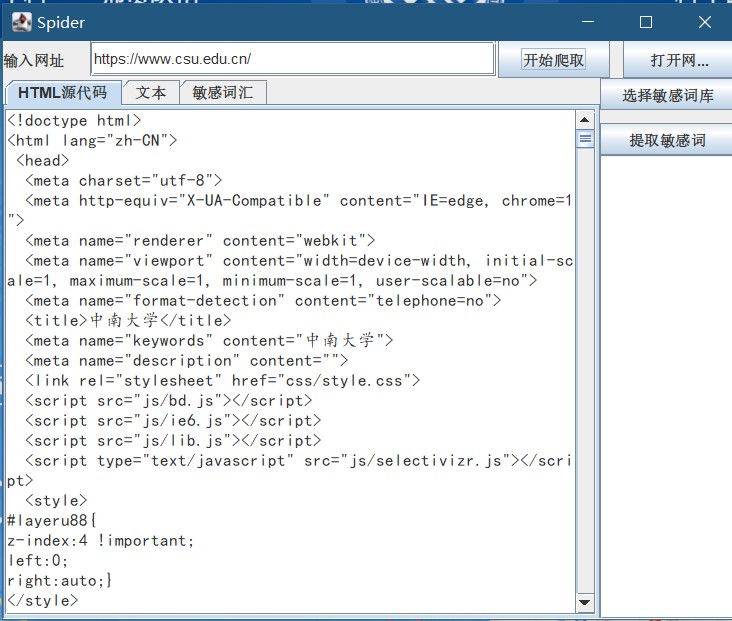
}

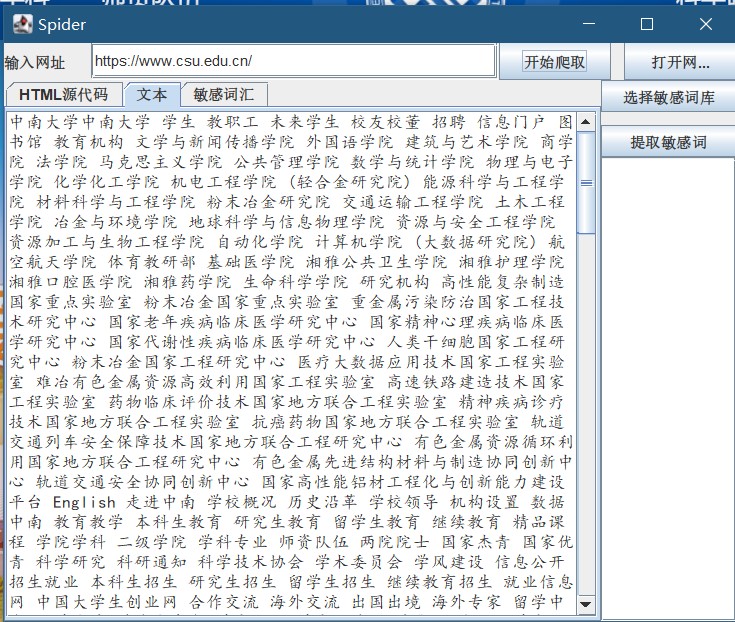
# 第九章 爬虫程序运行结果与测试分析

* 运行界面



* 输入网址进行爬取

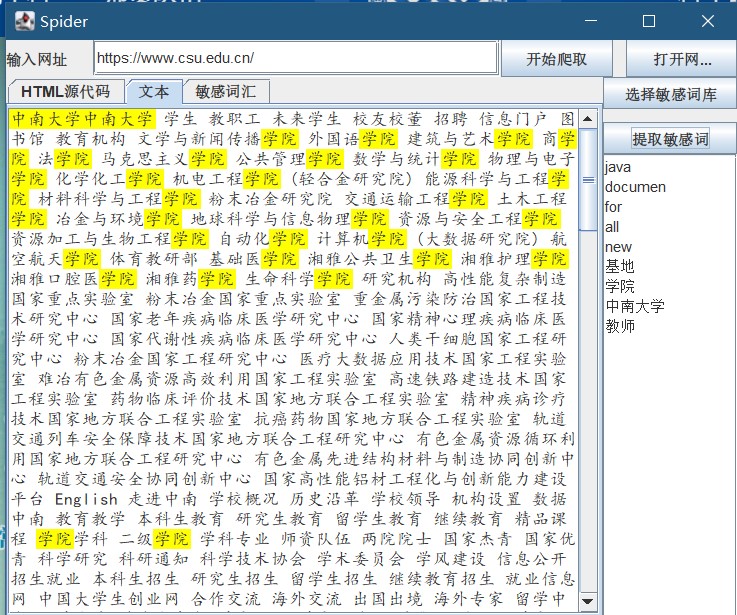


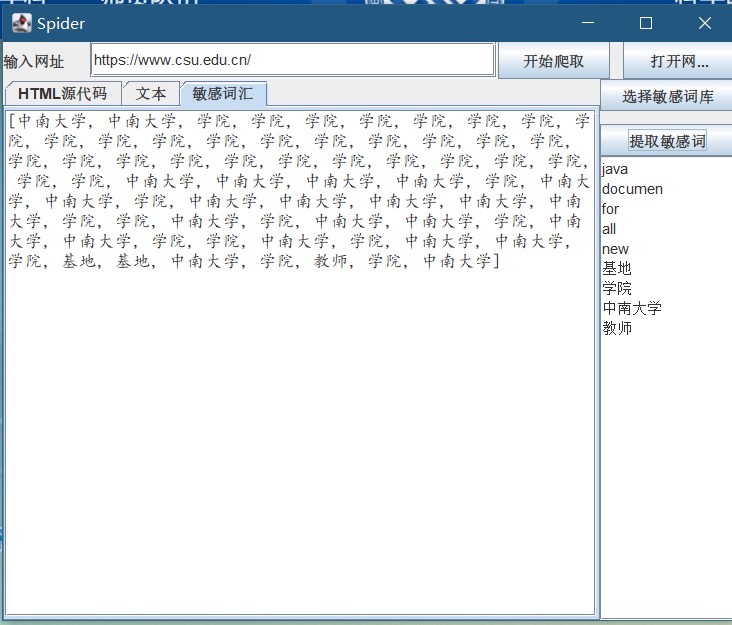


* 选择敏感词库

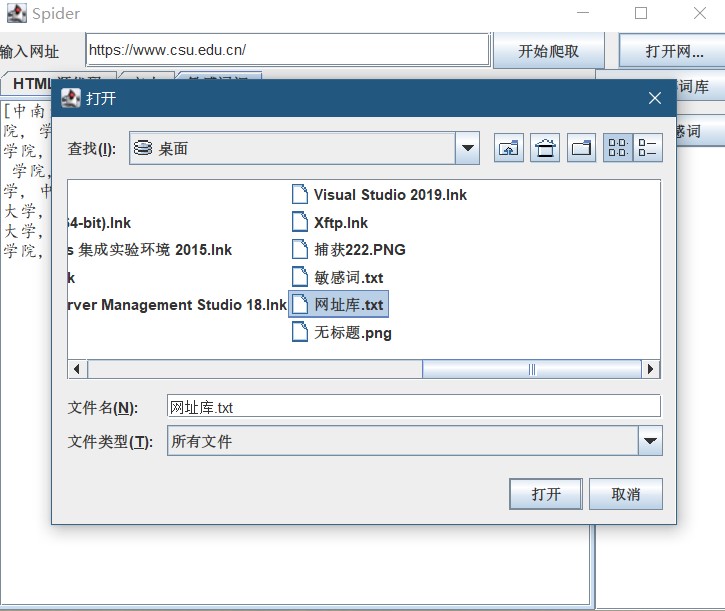


* 提取敏感词，高亮显示

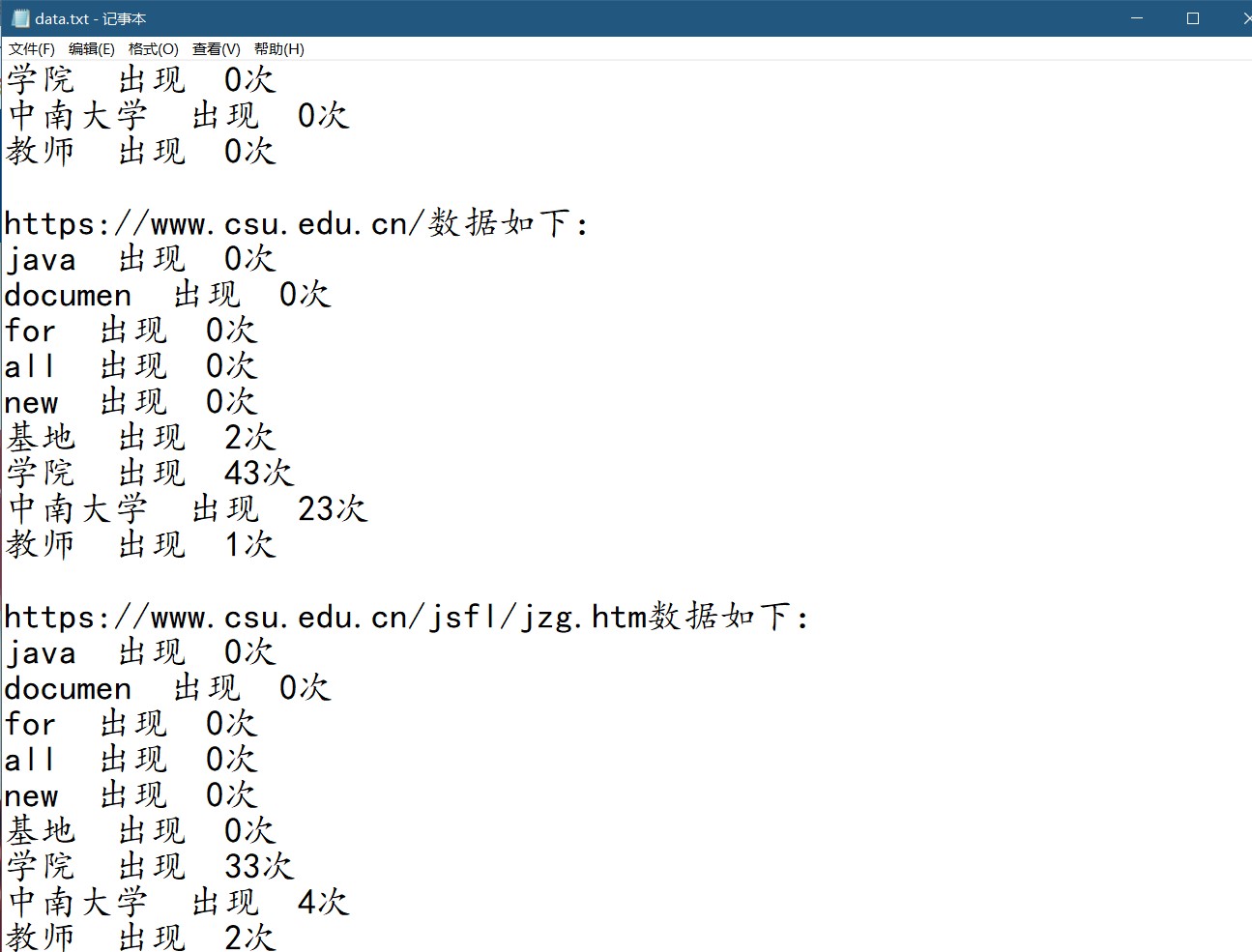
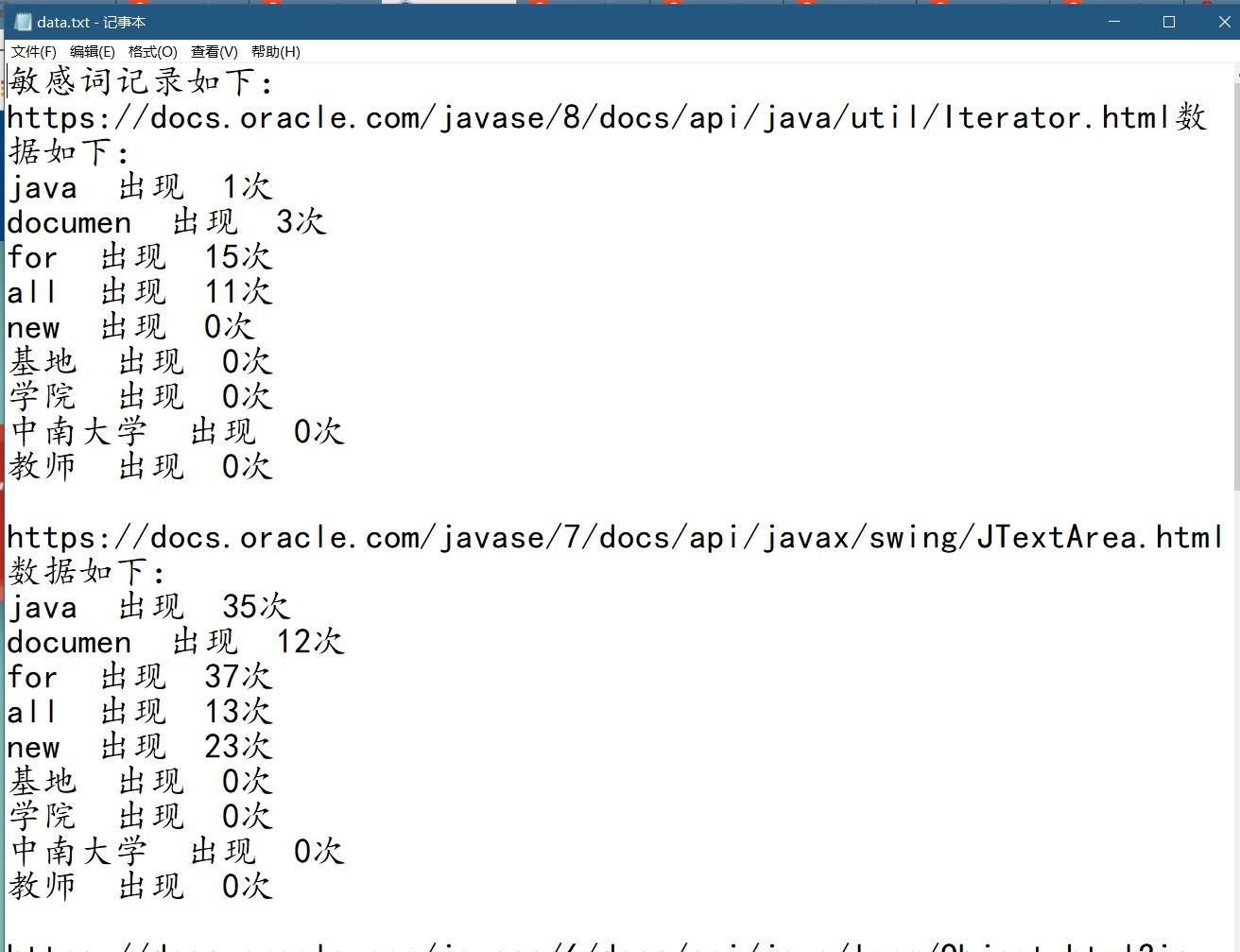




* 打开网址库



* 爬取网址库，记录统计结果



# 第十章 邮件客户端程序业务分析

* 编写图形用户界面，实现邮件客户端软件的设计与开发。用实际的邮箱账号密码进行登录。
* 客户端要求可以完成邮件编辑。
* 客户端要求可以配置与标准邮件服务器的连接。
* 客户端可以发送邮件，并可以粘贴附件。
* 客户端可以接受并且下载邮件、删除邮件。
* 选择一特定的邮件服务器建立邮件账户，编程实现与服务器的通信建立、利用SMTP协议完成邮件发送，利用POP3完成邮件接收。

# 第十一章 邮件客户端程序系统设计

## 11.1 系统功能定义

### （1）登录界面模块

* 用户输入qq邮箱账号和授权码登录
* 注意这里的密码是授权码，不是qq密码

### （2）功能选择模块

* 程序提供两类服务：发送邮件和查看邮件
* 用户点击相应的按钮进入相应的功能模块

### （3）邮件发送模块

* 输入收件人、主题、正文内容后发送邮件，也可以选择添加附件，注意收件人的格式

### （4）邮件查看模块

* 此界面会显示当前邮箱有的邮件数，用户可选择邮件进行查看
* 先选择邮件序号，接着点击refresh按钮，刷新发件人、标题、正文内容，若有附件，则在文本域中提示
* 用户点击download按钮选择下载本邮件中所带附件，设置的下载路径为E盘下的mailFile文件夹，可在本地查看附件是否下载成功
* 用户也可选择删除邮件，选择对应的邮件编号，点击delete按钮即可
* 用户操作完成之后可选择点击reset，使界面回到默认状态

## 11.2 关键问题设计

### (1) 发送邮件:

* 采用Session定义一个基本的邮件会话，利用java.util.Properties对象获取邮件服务器、用户名、密码等信息
* 接着创建要发送的消息，采用Message的一个子类，在大多数情况下，该子类是javax.mail.internet.MimeMessage。通过将Session对象传递给MimeMessage构造器的方法来创建消息：MimeMessage message = new MimeMessage(session);
* 接着创建电子邮件的地址：message.setFrom(new InternetAddress(Account));

message.setRecipients(Message.RecipientType.TO,InternetAddress.parse(jt1.getText()));

* 最后使用使用Transport类，从用于smtp协议的会话中获取一个特定的实例，然后传递用户名和密码并发送消息，最后关闭连接：transport = session.getTransport();transport.connect();

transport.sendMessage(message, message.getAllRecipients());

transport.close();

### (2) 查看邮件:

* 使用session类来获取消息，在获取会话之后，使用用户名和密码来连接Store类：session = Session.getInstance(pro,null);

try

{

store = session.getStore("pop3");

store.connect(Pop3Server, Account, Password);

}

catch(Exception e)

{

e.printStackTrace();

}

* 在连接Store类后，就可以获取一个Folder类，在读取其中的消息前必须先打开该类
* 一旦读取消息，就可以使用getContent()获取其内容
* 阅读完邮件后，就可以关闭文件夹和存储的连接：

Folder folder = store.getFolder(folderName);

folder.open(Folder.READ\_WRITE);

number = folder.getMessageCount();

int Mail\_Index = MailList.getSelectedIndex();

MimeMessage ThisMessage = (MimeMessage)((folder.getMessages())[Mail\_Index]);

Multipart multipart = (Multipart)part.getContent(); //part是传进去folder类里面的内容

int partCount = multipart.getCount();

folder.close(true);

### (3) 删除邮件：

Message DeleteMessage = (folder.getMessages())[index];

# 第十二章 邮件客户端程序源代码清单

## Login\_view.java:

package javaMail;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

public class Login\_view extends JFrame {

JTextField account;

JPasswordField password;

JLabel jbl1;

JButton Account,Password;

JPanel login,jp1,jp2,jp3;

JButton enter;

public static void main(String[] args){

Login\_view login = new Login\_view();

}

public Login\_view(){

jbl1 = new JLabel(new ImageIcon("src//image//tou.png"));

login = new JPanel(new GridLayout(3,1,20,10));

Account = new JButton("账号: ");

Password = new JButton("密码: ");

enter = new JButton("登录");

account = new JTextField();

account.requestFocusInWindow();

jp1 = new JPanel(new BorderLayout());

jp1.add(Account,BorderLayout.WEST);

jp1.add(account,BorderLayout.CENTER);

password = new JPasswordField();

jp2 = new JPanel(new BorderLayout());

jp2.add(Password,BorderLayout.WEST);

jp2.add(password,BorderLayout.CENTER);

jp3 = new JPanel();

jp3.add(enter);

login.add(jp1);

login.add(jp2);

login.add(jp3);

this.add(jbl1,BorderLayout.NORTH);

this.add(login,BorderLayout.CENTER);

this.setTitle("JavaMail");

this.setSize(400,600);

this.setResizable(false);

this.setVisible(true);

this.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);

enter.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

login();

}

});

this.addWindowListener(new WindowAdapter() {

@Override

public void windowClosing(WindowEvent e) {

exit();

}

});

}

public void login(){

if(account.getText().length() == 0 || password.getPassword().length == 0){

JOptionPane.showMessageDialog(Login\_view.this,"账号或密码不能为空！","ERROR",JOptionPane.ERROR\_MESSAGE);

if(account.getText().length() != 0){

password.requestFocusInWindow();

}else{

account.requestFocusInWindow();

}

return;

}

String format = "^(\\w\*|\\d\*)@\\w\*.(com|cn)$";

if(!(account.getText().matches(format))){

JOptionPane.showMessageDialog(Login\_view.this,"账号格式错误","ERROR",JOptionPane.ERROR\_MESSAGE);

account.requestFocusInWindow();

return;

}

Login\_view.this.dispose();

new WebPage(account.getText(),new String(password.getPassword()));

}

private void exit(){

int inquire = JOptionPane.showConfirmDialog(Login\_view.this,"确定退出吗？","Leave",JOptionPane.YES\_NO\_OPTION);

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

System.exit(0);

}else{

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

}

}

## WebPage.java:

package javaMail;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

public class WebPage extends JFrame{

private String Account;

private String Password;

JPanel jp1,jp2,jp3,jp4,jp5;

JButton outbox,inbox;

JLabel send,check;

JLabel jbl1,jbl2;

public WebPage(String account,String password) {

this.Account = account;

this.Password = password;

jbl1 = new JLabel("Welcome！");

jbl1.setPreferredSize(new Dimension(90,30));

jbl2 = new JLabel("");

jbl2.setPreferredSize(new Dimension(170,50));

jp5 = new JPanel(new BorderLayout());

jp5.add(jbl2,BorderLayout.WEST);

jp5.add(jbl1,BorderLayout.CENTER);

jp2 = new JPanel(new GridLayout(2,1,40,40));

send = new JLabel("send a mail please click:");

send.setPreferredSize(new Dimension(70,40));

outbox = new JButton("sendMail");

outbox.setPreferredSize(new Dimension(70,40));

jp2.add(send);

jp2.add(outbox);

jp3 = new JPanel(new GridLayout(2,1,40,40));

check = new JLabel("check a mail please click:");

check.setPreferredSize(new Dimension(70,40));

inbox = new JButton("checkMail");

inbox.setPreferredSize(new Dimension(70,40));

jp3.add(check);

jp3.add(inbox);

jp1 = new JPanel(new GridLayout(1,2,70,70));

jp1.add(jp2);

jp1.add(jp3);

jp4 = new JPanel(new GridLayout(2,1,20,20));

jp4.add(jp5);

jp4.add(jp1);

this.add(jp4);

this.setTitle("JavaMail");

this.setSize(400,250);

this.setResizable(false);

this.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);

this.setVisible(true);

outbox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

new SendPage(Account,Password);

}

});

inbox.addActionListener(new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

try {

new CheckPage(Account,Password);

} catch (Exception exception) {

exception.printStackTrace();

}

}

});

this.addWindowListener(new WindowAdapter() {

@Override

public void windowClosing(WindowEvent e) {

exit();

}

});

}

private void exit(){

int inquire = JOptionPane.showConfirmDialog(WebPage.this,"确定退出吗？","Leave",JOptionPane.YES\_NO\_OPTION);

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

System.exit(0);

}else{

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

}

public static void main(String[] args){

new WebPage("1","1");

}

}

## SendPage.java:

package javaMail;

import javax.activation.DataHandler;

import javax.activation.FileDataSource;

import javax.mail.\*;

import javax.mail.internet.\*;

import javax.sound.midi.Receiver;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import java.io.IOException;

import java.util.ArrayList;

import java.util.Date;

import java.util.Properties;

public class SendPage extends JFrame implements ActionListener{

private String Account;

private String Password;

private Session session;

private Transport transport;

JPanel jpnorth,jpcenter,jp1,jp2,jpsouth,jp3,jp4,jp5,jpall,jp6,jp7;

JButton jb1,jb2,jb3,jb4,jb5,jb6;

JTextField jt1,jt2;

JTextArea jta;

JLabel jbl2,jbl3;

JButton jbl1;

private boolean hasAttach = false;

private ArrayList<String> fileName = new ArrayList<String>();

public SendPage(String account,String password){

this.Account = account;

this.Password = password;

jbl1 = new JButton("Send a email");

jbl2 = new JLabel("");

jpnorth = new JPanel(new BorderLayout());

jpnorth.add(jbl2,BorderLayout.WEST);

jpnorth.add(jbl1,BorderLayout.CENTER);

jb1 = new JButton("收件人:");

jt1 = new JTextField();

jb2 = new JButton("主题:");

jt2 = new JTextField();

jp6 = new JPanel(new GridLayout(2,1,10,10));

jb3 = new JButton("正文:");

jb4 = new JButton("添加附件");

jb4.addActionListener((ActionListener) this);

jb5 = new JButton("发送");

jb5.addActionListener((ActionListener) this);

jp1 = new JPanel(new BorderLayout());

jp2 = new JPanel(new BorderLayout());

jp1.add(jb1,BorderLayout.WEST);

jp1.add(jt1,BorderLayout.CENTER);

jp2.add(jb2,BorderLayout.WEST);

jp2.add(jt2,BorderLayout.CENTER);

jp6.add(jp1);

jp6.add(jp2);

jp3 = new JPanel(new GridLayout(8,1,10,10));

jp3.add(jb3);

jp3.add(jb4);

jp3.add(jb5);

jta = new JTextArea();

jta.setPreferredSize(new Dimension(660,500));

jp4 = new JPanel();

jp4.add(jta);

jp7 = new JPanel(new BorderLayout());

jp7.add(jp3,BorderLayout.WEST);

jp7.add(jp4,BorderLayout.CENTER);

jpcenter = new JPanel(new BorderLayout());

jpcenter.add(jp6,BorderLayout.NORTH);

jpcenter.add(jp7,BorderLayout.CENTER);

jb6 = new JButton("退出");

jb6.addActionListener((ActionListener) this);

jbl3 = new JLabel("");

jp5 = new JPanel(new BorderLayout());

jp5.add(jb6,BorderLayout.CENTER);

jp5.add(jbl3,BorderLayout.WEST);

jpsouth = new JPanel();

jpsouth.add(jp5);

jpall = new JPanel(new BorderLayout());

this.setTitle("JavaMail");

this.setSize(800,600);

jpall.add(jpnorth,BorderLayout.NORTH);

jpall.add(jpcenter,BorderLayout.CENTER);

jpall.add(jpsouth,BorderLayout.SOUTH);

this.add(jpall);

this.setResizable(false);

this.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);

this.setVisible(true);

this.addWindowListener(new WindowAdapter()

{

@Override

public void windowClosing(WindowEvent e)

{

exit();

}

});

}

public static void main(String[] args){

new SendPage("1","1");

}

@Override

public void actionPerformed(ActionEvent e) {

JButton j = (JButton) e.getSource();

if (j == jb6) {

exit();

}else if(j == jb5){

if(hasAttach){

try {

SendMailPro();

} catch (MessagingException messagingException) {

messagingException.printStackTrace();

}

}else{

try {

SendMail();

} catch (MessagingException messagingException) {

messagingException.printStackTrace();

}

}

}else if(j == jb4){

try {

AppendAttach();

} catch (IOException ioException) {

ioException.printStackTrace();

}

}

}

public void exit(){

int inquire = JOptionPane.showConfirmDialog(SendPage.this,

"确定退出？","Leave",

JOptionPane.YES\_NO\_OPTION);

if(inquire==JOptionPane.YES\_OPTION)

{

this.dispose();

}

else

{

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

}

private void SendMail() throws MessagingException {

Properties pro = new Properties();

pro.put("mail.transport.protocol","smtp");

pro.put("mail.smtp.class","com.sun.mail.smtp.SMTPTransport");

pro.put("mail.smtp.host","smtp.qq.com");

/\*\*SMTP port.\*/

pro.put("mail.smtp.port","25");

/\*\*Verify account.\*/

pro.put("mail.smtp.auth","true");

session = Session.getInstance(pro, new Authenticator()

{

public PasswordAuthentication getPasswordAuthentication()

{

return new PasswordAuthentication(Account, Password);

}

});

transport = session.getTransport();

Session NewSession = Session.getDefaultInstance(new Properties());

MimeMessage message = new MimeMessage(NewSession);

message.setFrom(new InternetAddress(Account));

message.setRecipients(Message.RecipientType.TO,InternetAddress.parse(jt1.getText()));

message.setSentDate(new Date());

message.setSubject(jt2.getText());

message.setText(jta.getText());

message.saveChanges();

transport.connect();

transport.sendMessage(message, message.getAllRecipients());

JOptionPane.showMessageDialog(SendPage.this, "邮件发送成功！");

jt1.setText("");

jt2.setText("");

jta.setText("");

transport.close();

}

public void AppendAttach() throws IOException {

JFileChooser fileChooser = new JFileChooser();

if(fileChooser.showOpenDialog(SendPage.this) == JFileChooser.APPROVE\_OPTION){

String fileAddr = fileChooser.getSelectedFile().getCanonicalPath();

if(fileAddr != null && fileAddr.length()!=0){

hasAttach = true;

fileName.add(fileAddr);

}

}

}

private void SendMailPro() throws MessagingException {

Properties pro = new Properties();

pro.put("mail.transport.protocol","smtp");

pro.put("mail.smtp.class","com.sun.mail.smtp.SMTPTransport");

pro.put("mail.smtp.host","smtp.qq.com");

/\*\*SMTP port.\*/

pro.put("mail.smtp.port","25");

/\*\*Verify account.\*/

pro.put("mail.smtp.auth","true");

session = Session.getInstance(pro, new Authenticator()

{

public PasswordAuthentication getPasswordAuthentication()

{

return new PasswordAuthentication(Account, Password);

}

});

transport = session.getTransport();

Session NewSession = Session.getDefaultInstance(new Properties());

MimeMessage message = new MimeMessage(NewSession);

message.setFrom(new InternetAddress(Account));

message.setRecipients(Message.RecipientType.TO,InternetAddress.parse(jt1.getText()));

message.setSentDate(new Date());

message.setSubject(jt2.getText());

MimeBodyPart ContentPart = CreateCon(jta.getText());

ArrayList<MimeBodyPart> FileList = new ArrayList<MimeBodyPart>();

for(int i=0;i<fileName.size();++i)

{

FileList.add(CreateAttach(fileName.get(i)));

}

MimeMultipart AllMultiPart =new MimeMultipart("mixed");

AllMultiPart.addBodyPart(ContentPart);

for(int i=0;i<FileList.size();++i)

{

AllMultiPart.addBodyPart(FileList.get(i));

}

message.setContent(AllMultiPart);

message.saveChanges();

transport.connect();

transport.sendMessage(message, message.getAllRecipients());

JOptionPane.showMessageDialog(SendPage.this, "邮件发送成功!");

transport.close();

}

public static MimeBodyPart CreateCon(String body) throws MessagingException {

MimeBodyPart HTMLBodyPart = new MimeBodyPart();

HTMLBodyPart.setContent(body, "text/html;charset=UTF-8");

return HTMLBodyPart;

}

public static MimeBodyPart CreateAttach(String fileName) throws MessagingException {

MimeBodyPart Attachment = new MimeBodyPart();

FileDataSource file = new FileDataSource(fileName);

Attachment.setDataHandler(new DataHandler(file));

Attachment.setFileName(file.getName());

return Attachment;

}

}

## CheckPage.java:

package javaMail;

import javax.mail.\*;

import javax.mail.internet.MimeMessage;

import javax.mail.internet.MimeMultipart;

import javax.mail.internet.MimeUtility;

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.WindowAdapter;

import java.awt.event.WindowEvent;

import java.io.\*;

import java.util.Properties;

public class CheckPage extends JFrame implements ActionListener{

private final String folderName = "inbox";

private final String savePosition = "E:\\mailFile\\";

private String Account;

private String Password;

JButton jbdelete,jbreset,jbupdate,jbdownload,jbsender,jbtopic,jbbody;

JTextArea jtabody;

JLabel jtanote;

JLabel jlbmail,jlbtopic;

JPanel jpnorth,jp11,jp12,jp1,jp2,jpsouth,jpCenter,jpWest,jpEast,jpall;

JComboBox MailList;

private Session session;

private Store store;

private String Pop3Server = "pop.qq.com";

private int number;

public CheckPage(String account,String password) throws Exception {

this.Account = account;

this.Password = password;

Properties pro = new Properties();

pro.put("mail.store.protocol", "pop3");

pro.put("mail.imap.class", "com.sun.mail.imap.IMAPStore");

pro.put("mail.pop3.class", "com.sun.mail.pop3.POP3Store");

session = Session.getInstance(pro,null);

try

{

store = session.getStore("pop3");

store.connect(Pop3Server, Account, Password);

}

catch(Exception e)

{

e.printStackTrace();

}

jtanote = new JLabel("",SwingConstants.RIGHT);

ReceiveMes();

jpnorth = new JPanel();

jpnorth.add(jtanote);

jbsender = new JButton("from:");

jlbmail = new JLabel();

jp11 = new JPanel(new BorderLayout());

jp11.add(jbsender,BorderLayout.WEST);

jp11.add(jlbmail,BorderLayout.CENTER);

jbtopic = new JButton("topic:");

jlbtopic = new JLabel();

jp12 = new JPanel(new BorderLayout());

jp12.add(jbtopic,BorderLayout.WEST);

jp12.add(jlbtopic,BorderLayout.CENTER);

jp1 = new JPanel(new GridLayout(2,1,10,10));

jp1.add(jp11);

jp1.add(jp12);

jbbody = new JButton("body:");

jtabody = new JTextArea();

jp2 = new JPanel(new BorderLayout());

jp2.add(jbbody,BorderLayout.WEST);

jp2.add(jtabody,BorderLayout.CENTER);

jpsouth = new JPanel(new BorderLayout());

jpsouth.add(jp1,BorderLayout.NORTH);

jpsouth.add(jp2,BorderLayout.CENTER);

jpCenter = new JPanel(new BorderLayout());

jpCenter.add(jpnorth,BorderLayout.NORTH);

jpCenter.add(jpsouth,BorderLayout.CENTER);

MailList = new JComboBox();

MailList.setMaximumRowCount(5);

for(int i = 0; i < number; i++){

MailList.addItem("Mail-No." + (i+1));

}

MailList.setSelectedIndex(0);

jbdelete = new JButton("delete");

jbdelete.addActionListener(this);

jpEast = new JPanel(new GridLayout(2,1,10,10));

jpEast.add(MailList);

jpEast.add(jbdelete);

jbreset = new JButton("reset");

jbreset.addActionListener(this);

jbupdate = new JButton("refresh");

jbupdate.addActionListener(this);

jbdownload = new JButton("download");

jbdownload.addActionListener(this);

jpWest = new JPanel(new GridLayout(5,1,10,10));

jpWest.add(jbreset);

jpWest.add(jbupdate);

jpWest.add(jbdownload);

jpall = new JPanel(new BorderLayout());

jpall.add(jpWest,BorderLayout.WEST);

jpall.add(jpCenter,BorderLayout.CENTER);

jpall.add(jpEast,BorderLayout.EAST);

this.setTitle("JavaMail");

this.setSize(800,600);

this.add(jpall);

this.setResizable(false);

this.setDefaultCloseOperation(DISPOSE\_ON\_CLOSE);

this.setVisible(true);

this.addWindowListener(new WindowAdapter()

{

@Override

public void windowClosing(WindowEvent e)

{

exit();

}

});

}

@Override

public void actionPerformed(ActionEvent e) {

JButton j = (JButton) e.getSource();

if(j == jbreset){

try {

ReceiveMes();

} catch (Exception exception) {

exception.printStackTrace();

}

jlbmail.setText("the sender.");

jlbtopic.setText("the Topic.");

jtabody.setText("the body");

}else if(j == jbupdate){

try {

viewCon();

} catch (Exception exception) {

exception.printStackTrace();

}

}else if(j == jbdownload){

try {

DownloadAttach(MailList.getSelectedIndex());

} catch (Exception exception) {

exception.printStackTrace();

}

}else if(j == jbdelete){

try {

DeleteMail(MailList.getSelectedIndex());

} catch (Exception exception) {

exception.printStackTrace();

}

}

}

private void ReceiveMes() throws Exception {

Folder folder = store.getFolder(folderName);

if(folder==null)

{

throw new Exception(folderName+" does not exist.");

}

folder.open(Folder.READ\_WRITE);

number = folder.getMessageCount();

jtanote.setText("Totally, mail number is "+number+".");

}

private void viewCon() throws Exception {

Folder folder = store.getFolder(folderName);

if(folder==null)

{

throw new Exception(folderName+" does not exist.");

}

folder.open(Folder.READ\_WRITE);

int Mail\_Index = MailList.getSelectedIndex();

MimeMessage ThisMessage = (MimeMessage)((folder.getMessages())[Mail\_Index]);

String sender = String.valueOf((ThisMessage.getFrom())[0]);

jlbmail.setText(sender);

jlbtopic.setText(ThisMessage.getSubject());

if(hasAttachment(ThisMessage))

{

StringBuffer textbody = new StringBuffer();

GetTextBody(ThisMessage,textbody);

jtabody.setText(textbody.toString()+"\n\nNOTE:此邮件带有附件.");

}

else

{

String textBody = String.valueOf(ThisMessage.getContent());

jtabody.setText(textBody);

}

ThisMessage.setFlag(Flags.Flag.SEEN, true);

folder.close(true);

}

private boolean hasAttachment(Part part) throws MessagingException, IOException {

boolean has = false;

if(part.isMimeType("multipart/\*"))

{

MimeMultipart multipart = (MimeMultipart)part.getContent();

int partCount = multipart.getCount();

for(int i=0;i<partCount;++i)

{

BodyPart bodyPart = multipart.getBodyPart(i);

String disp = bodyPart.getDisposition();

if(disp!=null&&(disp.equalsIgnoreCase(Part.ATTACHMENT)||disp.equalsIgnoreCase(Part.INLINE)))

{

has = true;

}

else if(bodyPart.isMimeType("multipart/\*"))

{

has = hasAttachment(bodyPart);

}

else

{

String contentType = bodyPart.getContentType();

if(contentType.indexOf("application")!=-1)

{

has = true;

}

if(contentType.indexOf("name")!=-1)

{

has = true;

}

}

if(has)

{

break;

}

}

}

else if(part.isMimeType("message/rfc822"))

{

has = hasAttachment((Part)part.getContent());

}

return has;

}

private StringBuffer GetTextBody(Part part, StringBuffer textbody) throws MessagingException, IOException {

boolean hasTextAttach = part.getContentType().indexOf("name")>0;

if(part.isMimeType("text/\*")&&!hasTextAttach)

{

textbody.append(part.getContent().toString());

}

else if(part.isMimeType("message/rfc822"))

{

GetTextBody((Part)part.getContent(),textbody);

}

else if(part.isMimeType("multipart/\*"))

{

Multipart multipart = (Multipart)part.getContent();

int partCount = multipart.getCount();

for(int i=0;i<partCount;++i)

{

BodyPart bodypart = multipart.getBodyPart(i);

GetTextBody(bodypart, textbody);

}

}

return textbody;

}

private void DownloadAttach(int index) throws Exception {

Folder folder = store.getFolder(folderName);

if(folder==null)

{

throw new Exception(folderName+" does not exist.");

}

folder.open(Folder.READ\_WRITE);

int Mail\_Index = MailList.getSelectedIndex();

MimeMessage ThisMessage = (MimeMessage)((folder.getMessages())[Mail\_Index]);

if(hasAttachment(ThisMessage))

{

SaveAttach(ThisMessage,savePosition+ThisMessage.getSubject()+"\_");

JOptionPane.showMessageDialog(CheckPage.this, "附件下载成功!");

}

else

{

JOptionPane.showMessageDialog(CheckPage.this,"This mail has no attachment.","ERROR",JOptionPane.ERROR\_MESSAGE);

}

}

private void SaveAttach(Part part, String s) throws Exception {

if(part.isMimeType("multipart/\*"))

{

Multipart multipart = (Multipart)part.getContent();

int partCount = multipart.getCount();

for (int i=0;i<partCount;++i)

{

BodyPart bodypart = multipart.getBodyPart(i);

String disp = bodypart.getDisposition();

if(disp!=null&&(disp.equalsIgnoreCase(Part.ATTACHMENT)||disp.equalsIgnoreCase(Part.INLINE)))

{

InputStream is = bodypart.getInputStream();

SaveFile(is,savePosition,Decode(bodypart.getFileName()));

}

else if(bodypart.isMimeType("multipart/\*"))

{

SaveAttach(bodypart, savePosition);

}

else

{

String contentType = bodypart.getContentType();

if(contentType.indexOf("name")!=-1||contentType.indexOf("application")!=-1)

{

SaveFile(bodypart.getInputStream(), savePosition, Decode(bodypart.getFileName()));

}

}

}

}

}

private void SaveFile(InputStream is,String savePosition,String fileName) throws Exception

{

BufferedInputStream bis = new BufferedInputStream(is);

BufferedOutputStream bos = new BufferedOutputStream(new FileOutputStream(new File(savePosition+fileName)));

int index=-1;

while((index=bis.read())!=-1)

{

bos.write(index);

bos.flush();

}

bos.close();

bis.close();

}

private String Decode(String encode) throws Exception

{

if(encode==null||encode.equals(""))

{

return "";

}

else

{

return MimeUtility.decodeText(encode);

}

}

private void DeleteMail(int index) throws Exception

{

Folder folder = store.getFolder(folderName);

if(folder==null)

{

throw new Exception(folderName+" does not exist.");

}

folder.open(Folder.READ\_WRITE);

int inquire = JOptionPane.showConfirmDialog(CheckPage.this,

"确定删除Mail-No."+(index+1)+" ?","Delete",

JOptionPane.YES\_NO\_OPTION);

if(inquire==JOptionPane.YES\_OPTION)

{

Message DeleteMessage = (folder.getMessages())[index];

DeleteMessage.setFlag(Flags.Flag.DELETED, true);

JOptionPane.showMessageDialog(CheckPage.this, "Mail-No."+(index+1)+" 已删除");

}

else

{

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

folder.close(true);

}

private void exit()

{

int inquire = JOptionPane.showConfirmDialog(CheckPage.this,

"确定退出吗?","Leave",

JOptionPane.YES\_NO\_OPTION);

if(inquire==JOptionPane.YES\_OPTION)

{

this.dispose();

}

else

{

this.setDefaultCloseOperation(DO\_NOTHING\_ON\_CLOSE);

}

}

public static void main(String[] args) throws Exception {

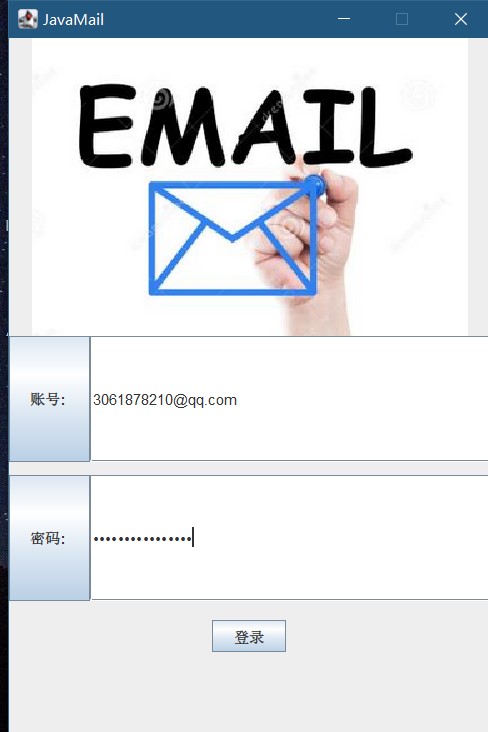
new CheckPage("1","1");

}

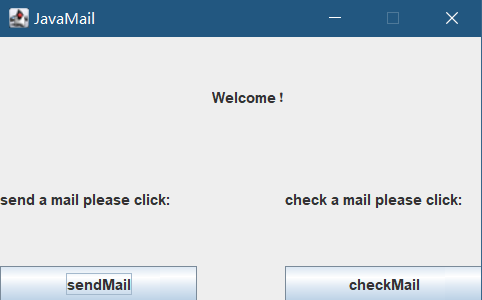
}

# 第十三章 邮件客户端程序运行结果与测试分析

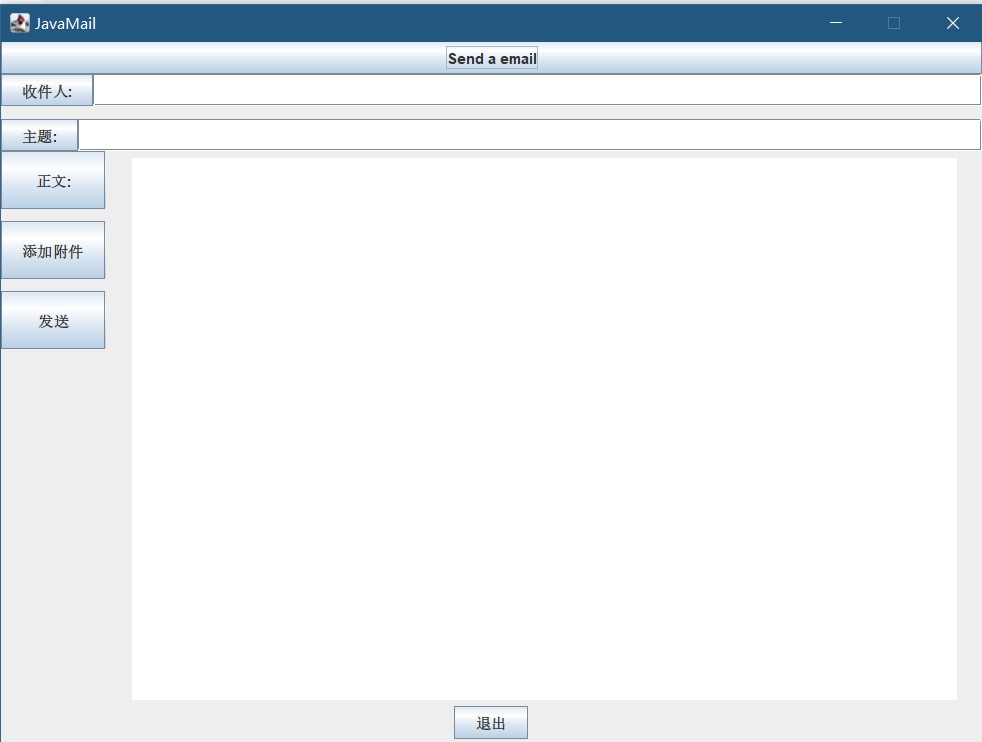
* 登录界面

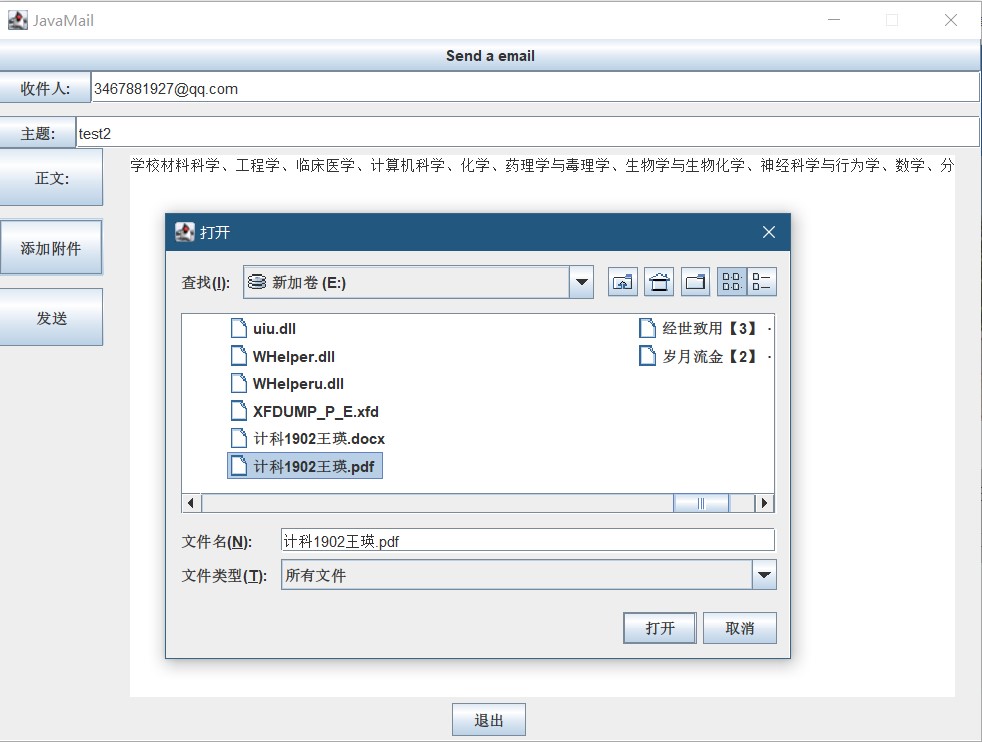


* 登陆成功，选择功能界面

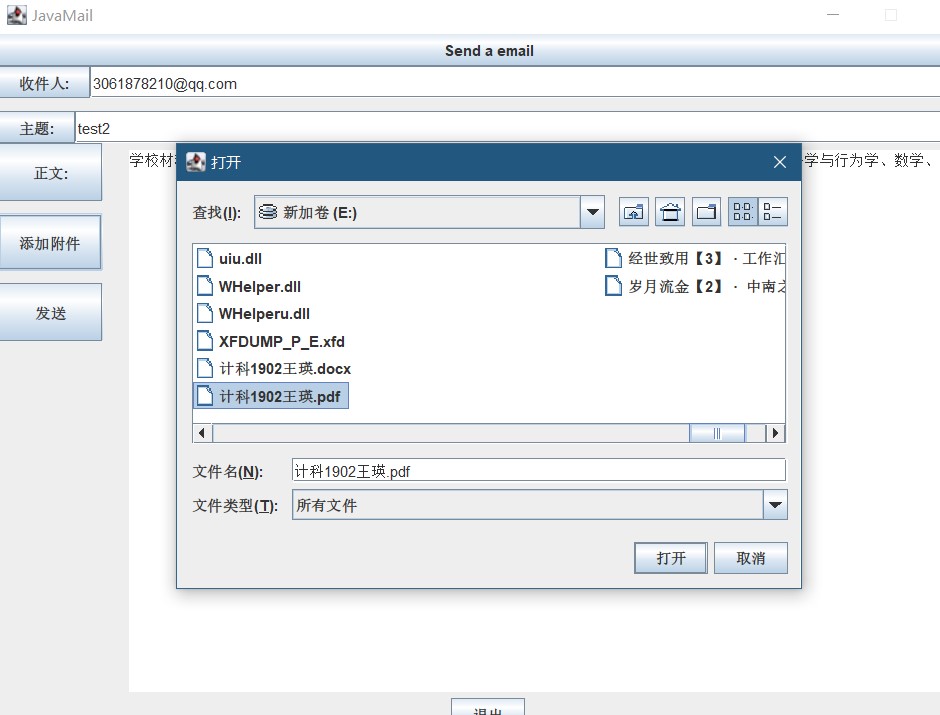


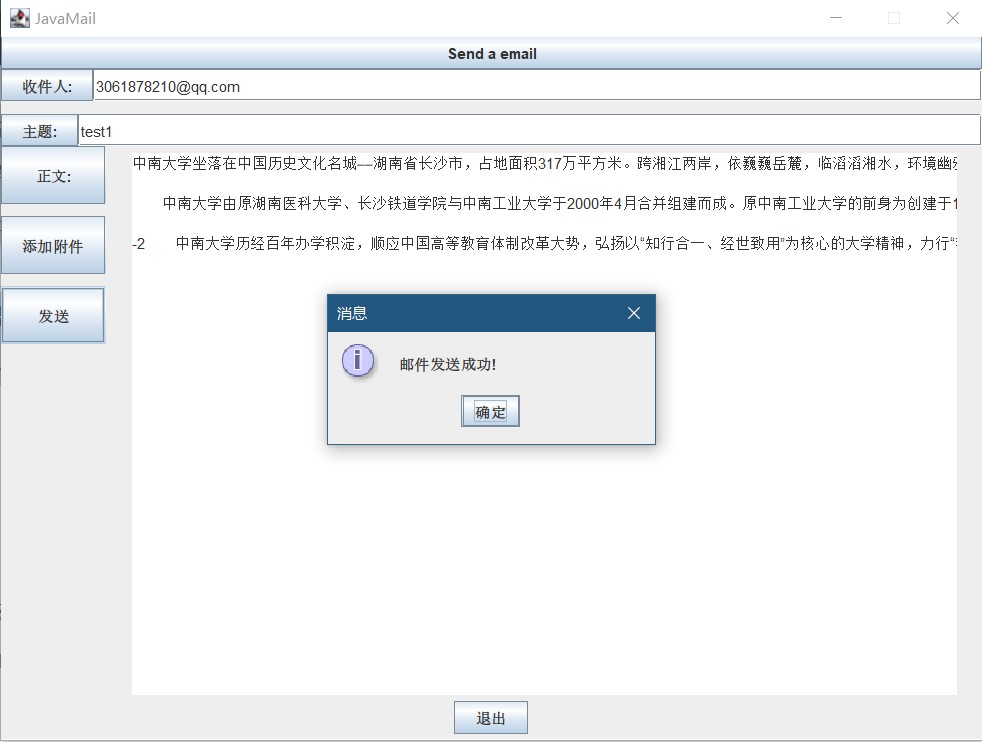
* 发送邮件界面



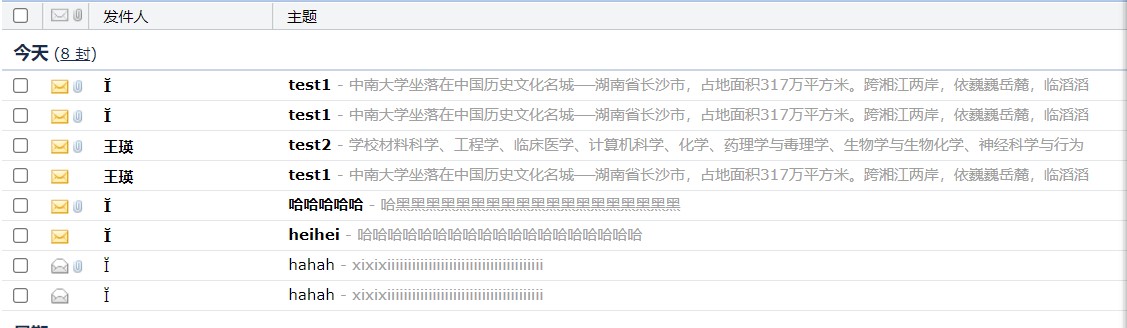




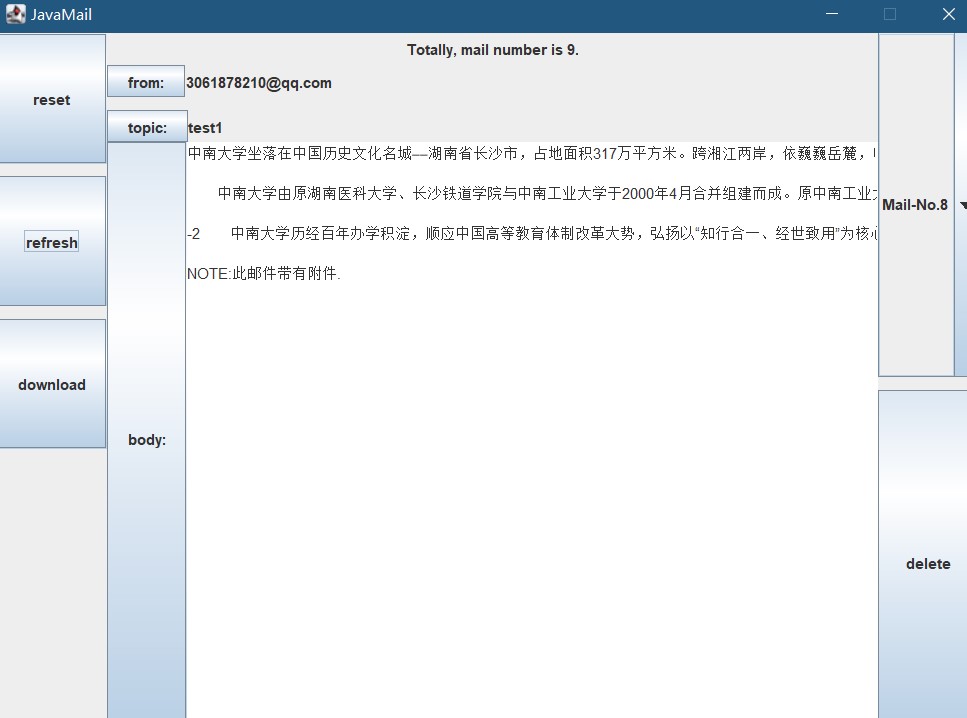




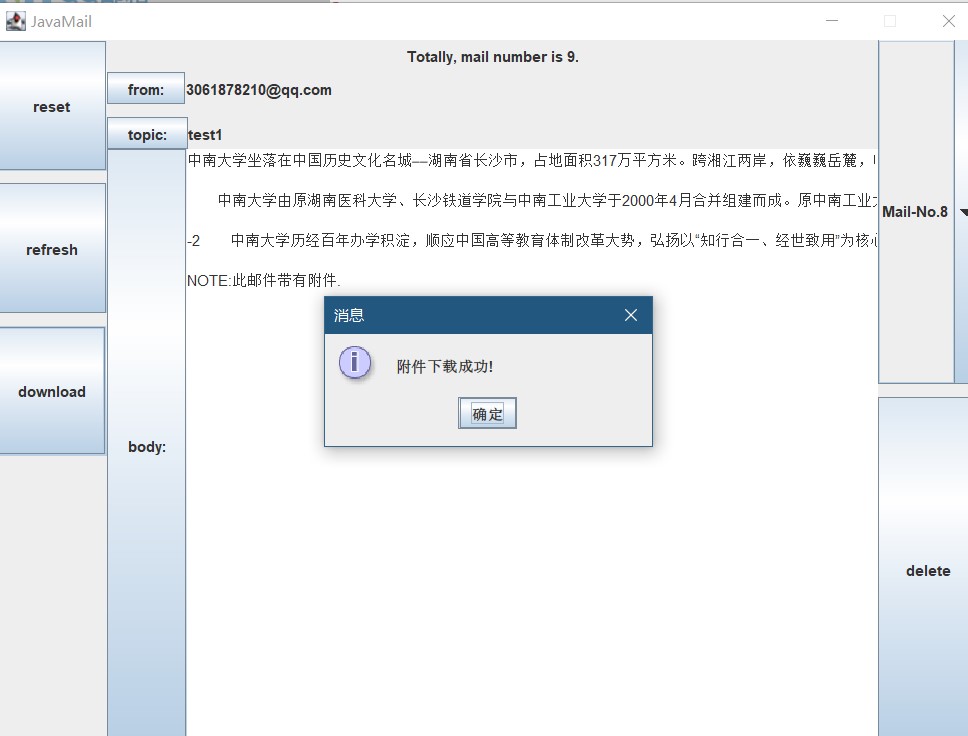
* 在浏览器中登陆检查邮件是否发送成功



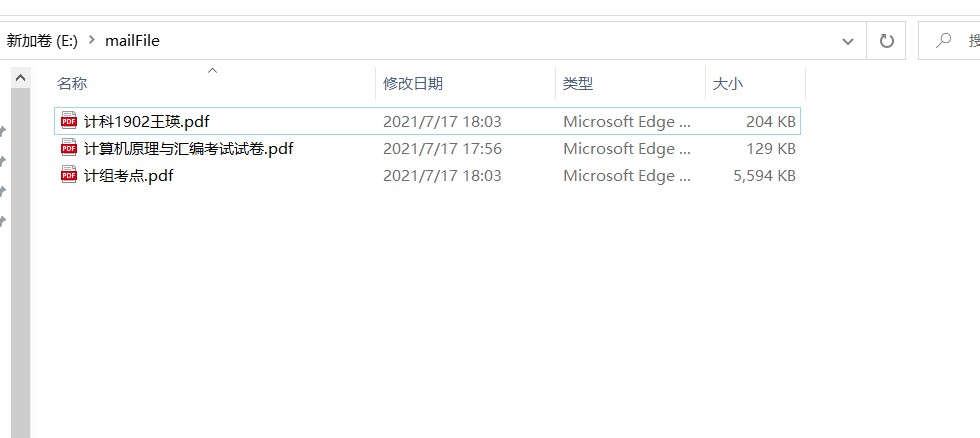
* 查阅邮件界面



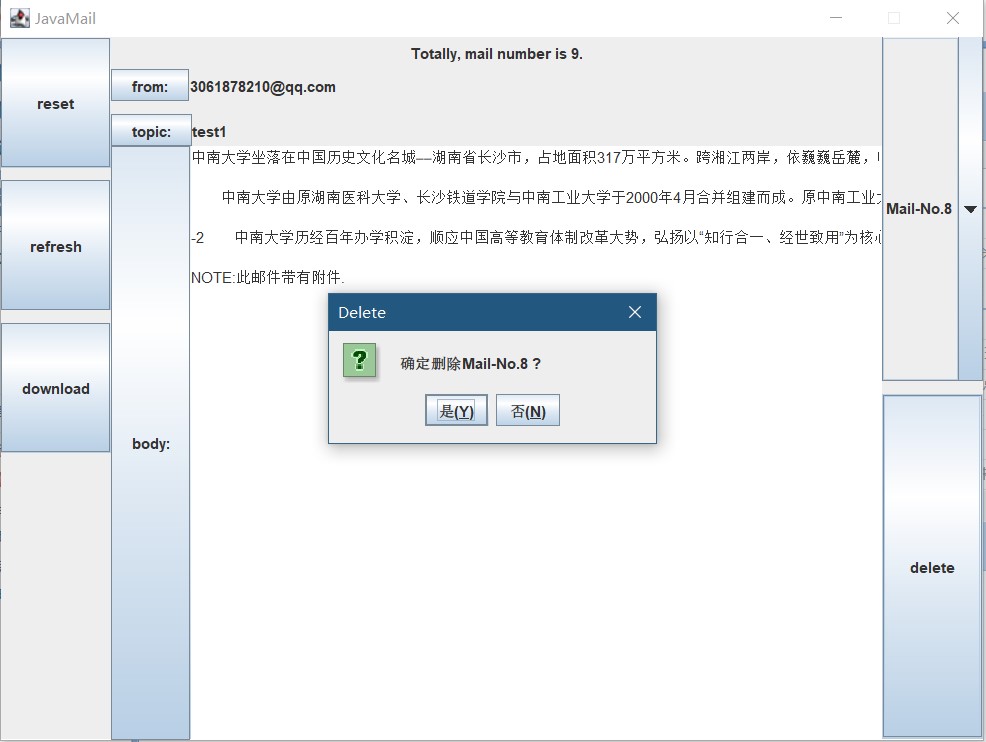
* 下载邮件所带附件

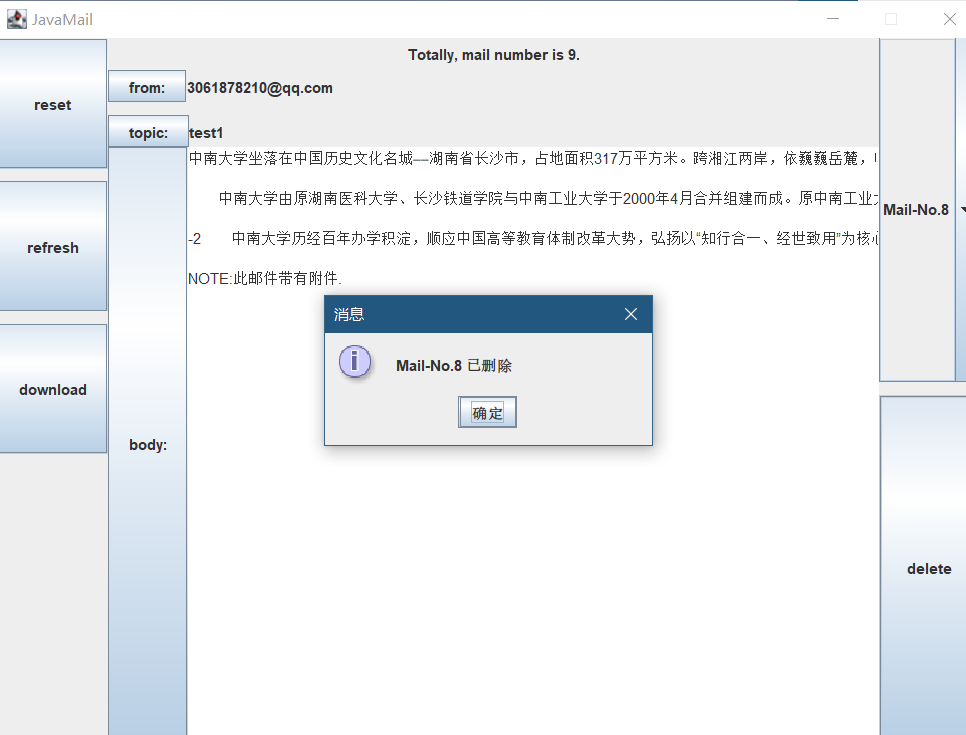


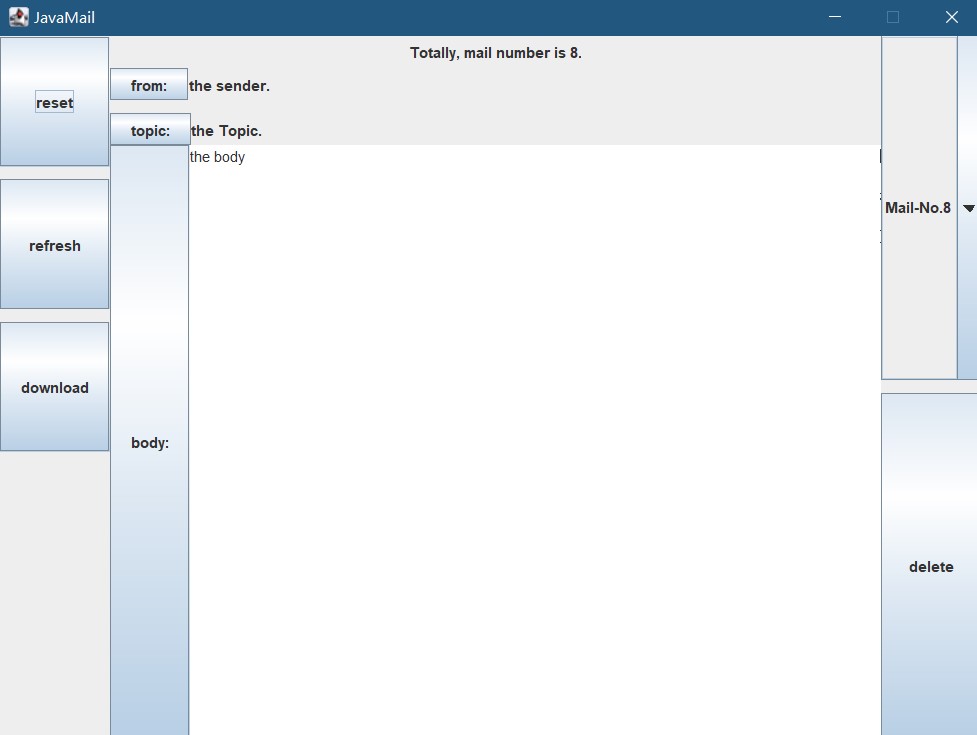
* 附件设置的保存位置是E盘下的mailFile文件夹：



* 删除邮件







* 在浏览器中查看，可以看到与之前相比最上面的test1已经被删除



# 第十四章 结论与心得

* 虽然以前学过java，但是并不熟练，尤其图形界面就一点儿不会，课设刚开始的时候感觉非常吃力，完全不知道从何下手，在谷歌上查了很多关于Java爬虫的资料，了解到idea上用jsoup就可以直接实现爬虫，用DFA算法可以实现敏感词匹配，所以一开始的时候选择先做爬虫那个程序，关于爬虫的界面构思了很长时间没有想清楚，后来在CSDN上找到类似项目，模仿了别人的界面开始整这个程序，写完爬虫这个程序对java图形界面掌握了一些，在后面写聊天室的时候提供了很多帮助
* 写聊天室的时候逻辑有一些乱，看了几个关于模仿qq项目的学习视频后思路清晰了一些，emm虽然写得还是有些乱，但是聊天室写完之后，对图形界面、线程的使用有了更进一步的理解掌握，由于时间紧凑，一开始计划实现的更多扩展功能还没有全部实现，计划后面暑假的时候慢慢补全，让程序更加完善
* 做前两个的时候稍微有些头绪，但是JavaMail是完全陌生，相关的资料也不是很完善，后面时间更少，做这个的时候对界面的处理就没有前两个好了，勉强实现相关功能，但是对JavaMail的掌握尚且不够，可能后续还是需要花些时间再研究研究
* 在做课设的整个过程中有的时候感觉非常吃力，前期感觉自己就是在专注地写bug，一个接一个，因为自己的不熟练导致bug很多，效率很低，比如一开始写爬虫的时候重写actionPerformed一直报错，界面的按钮一直没有反应，后来找到问题是在一开始少了implements ActionListener；后面也遇到很多问题，有的查来查去换了很多解决办法，耗费了很多时间，比如在做JavaMail的时候，一开始以为只需要一个mail.jar，后来缺了一个activation.jar,疯狂报错，找了很长时间才找到这个问题，找到下载的途径导进去，在写聊天室的时候因为定义的类很多，很容易搞混，很多次因为搞混一些定义的对象比如消息包m和新的消息包mnew的sender和getter就会一直报错，感觉还是因为自己对java熟练程度不够，虽然bug很多，但是在找解决办法的时候也感觉是进步的过程，课设虽然时间短任务大些，但是是学东西的高效途径
* Java写图形界面确实很方便，java很多包直接拿来用也非常方便，以后多尝试用java解决问题。在以后的学习过程中，还是应当多动手，这样能知道会遇到什么问题，解决问题之后就能学到更多东西，更加熟练，一是提高动手能力，二是要提高独立思考解决问题的能力，多写代码，少写bug。