Client.java

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Client;

/\*\*

\*

\* @author zhuqihao

\*/

public class Client {

/\*\*

\* 起始函数，调用登录界面

\*/

public static void main(String[] args) {

new LoginClient();

}

}

LoginClient.java

/\*\*

\* 登录界面的主函数，实现账号和密码的填写

\*/

package Client;

import java.awt.Color;

import java.awt.Cursor;

import java.awt.Font;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.ItemEvent;

import java.awt.event.ItemListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.io.BufferedReader;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.FileReader;

import java.io.FileWriter;

import java.io.IOException;

import java.util.ArrayList;

import javax.swing.AbstractListModel;

import javax.swing.BorderFactory;

import javax.swing.ComboBoxModel;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JCheckBox;

import javax.swing.JComboBox;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import Info.Usr\_Info;

import MainPage.MainFace;

/\*\*

\*

\* @author zhuqihao Allrights reservered;

\*/

public class LoginClient extends JFrame implements ActionListener {

/\*\*

\* 界面部件的声明

\*/

private JPanel background;

private JPanel fun;

private JLabel labelaccount;

private JLabel newaccount;

private JButton login;

private JLabel labelpassword;

private JComboBox account;

private JPasswordField password;

private JCheckBox remember;

private JCheckBox autologin;

private JLabel imagetitle;

private String font = new String("Xingkai SC");

private int fontsize = 16;

/\*\*

\* 用来存储从本地读取的需要记住密码的用户名和密码

\*/

ArrayList<String> oldaccount;

/\*\*

\* 界面部件的初始化和摆放位置以及监视器的添加

\*/

public LoginClient() {

oldaccount = new ArrayList<String>();

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setBounds(300, 160, 800, 500); // 确定界面的大小为800\*500

this.setLayout(null); // 确定布局方式为null

initjlabel();

initground();

setVisible(true);

try {

File file = new File("src/usr\_info/usr.txt"); // 读取记住密码的用户信息

if (!file.exists()) {

file.createNewFile();

}

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

String account = new String();

String account\_password = new String();

account = cin.readLine();

while (account != null) {

oldaccount.add(new String(account));

account = cin.readLine();

}

cin.close();

} catch (IOException e) {

e.printStackTrace();

}

if (oldaccount.size() != 0) {

JTextField tf = (JTextField) (account.getEditor().getEditorComponent());

tf.setText(oldaccount.get(0));

password.setText("unknown"); // 密码框内默认显示的是unknown，这样就不知道原来的密码是几位数的

imagetitle.setIcon(new ImageIcon("src/usr\_image/" + oldaccount.get(0))); // 设置记住密码用户的头像

}

}

// 设置登录界面的主背景函数

public void setbackground() {

ImageIcon img = new ImageIcon("./src/image/loginbackground.jpg");

JLabel background = new JLabel(img);

this.background.add(background);

background.setBounds(this.getX(), this.getY(), img.getIconWidth(), img.getIconHeight());

return;

}

// 初始化整个大Panel的函数，包括它添加部件等

public void initground() {

background = new JPanel();

background.setOpaque(true);

background.setBounds(0, 0, this.getWidth(), this.getHeight());

setbackground();

this.add(background);

}

public void initjlabel() {

// 初始化所有的label里的文字内容

labelaccount = new JLabel("账号:");

labelpassword = new JLabel("密码:");

remember = new JCheckBox("记住密码");

password = new JPasswordField();

account = new JComboBox(new comboboxmodel());

autologin = new JCheckBox("自动登录");

login = new JButton("进入悠品");

newaccount = new JLabel("注册账户");

imagetitle = new JLabel();

// 初始化圆框Panel,即登录界面的白框

fun = new RoundRectPanel();// new JPanel();

this.add(fun);

fun.setBounds(20, 50, 370, 250);

fun.setBackground(Color.WHITE);

fun.setOpaque(true);

// 设置label字体颜色

labelaccount.setForeground(Color.gray);

labelaccount.setBounds(120, 50, 50, 40);

labelaccount.setFont(new Font(font, 1, fontsize));

// 填写账户信息的列表框，需要设置大小位置以及添加事件监听器，点击切换用户时实时更新头像

account.setBounds(170, 58, 150, 24);

account.setOpaque(true);

account.setEditable(true);

account.addItemListener(new ItemListener() {

public void itemStateChanged(ItemEvent e) {

System.out.println(e.getItem().toString());

imagetitle.setIcon(new ImageIcon("./src/usr\_image/" + e.getItem().toString()));

}

});

// 密码label初始化位置

labelpassword.setForeground(Color.gray);

labelpassword.setBounds(120, 100, 50, 40);

labelpassword.setFont(new Font(font, 1, fontsize));

// 密码框设置填充字符

password.setBounds(170, 108, 150, 27);

password.setEchoChar('\*');

// 两个复选框的设置

remember.setBounds(150, 150, 100, 40);

remember.setFont(new Font(font, 1, fontsize));

remember.setOpaque(false);

remember.setForeground(Color.gray);

autologin.setForeground(Color.GRAY);

autologin.setBounds(250, 150, 100, 40);

autologin.setFont(new Font(font, 1, fontsize));

autologin.setOpaque(false);

// 登录按钮的设置

login.setBackground(new Color(57, 111, 57));

login.setBorderPainted(false);

login.setForeground(Color.white);

login.setBounds(120, 200, 150, 40);

login.setFont(new Font(font, 1, fontsize));

login.addActionListener(this);

login.setOpaque(true);

// 注册新用户了label，添加鼠标点击，移入，移出事件，改变颜色和鼠标的形状

newaccount.setBounds(295, 205, 150, 40);

newaccount.setForeground(Color.gray);

newaccount.setFont(new Font(font, 1, fontsize - 2));

newaccount.addMouseListener(new MouseAdapter() {

public void mouseEntered(MouseEvent e) {

// 将鼠标设置为手型

newaccount.setCursor(Cursor.getPredefinedCursor(Cursor.HAND\_CURSOR));

}

public void mouseExited(MouseEvent e) {

// 将鼠标设置为默认形状

newaccount.setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT\_CURSOR));

}

public void mouseClicked(MouseEvent e) {

dispose();

new RegisterClient();

}

});

// 显示头像的label

imagetitle.setBounds(12, 45, 100, 100);

imagetitle.setBorder(BorderFactory.createRaisedBevelBorder());

// 添加部件

fun.add(imagetitle);

fun.add(newaccount);

fun.add(login);

fun.add(autologin);

fun.add(remember);

fun.add(password);

fun.setLayout(null);

fun.add(labelaccount);

fun.add(labelpassword);

fun.add(account);

}

// 整个类继承了ActionListener接口实现鼠标点击事件的处理

public void actionPerformed(ActionEvent e) {

// 点击注册账号，进入注册界面

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

dispose();

new RegisterClient();

}

// 点击登录按钮，与服务器进行通信，验证密码正确

else if (e.getSource() == login) {

String usrid = account.getSelectedItem().toString();

connection con = new connection(usrid);

// 未连接上服务器

if (!con.active) {

JOptionPane.showMessageDialog(null, "与服务器连接异常，请稍后再试!", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

// 读取记住密码的用户密码

try {

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

if (oldaccount.get(i).equals(usrid)) {

String pass = new String();

DataInputStream out = new DataInputStream(

new FileInputStream("src/usr\_info/" + usrid + ".txt"));

password.setText(out.readUTF());

break;

}

}

} catch (IOException ev) {

ev.printStackTrace();

}

// 验证密码是否正确

String usrpassword = password.getText();

if (!con.sentusr\_password(usrid, usrpassword).equals("matched")) {

JOptionPane.showMessageDialog(null, "不存在该用户或密码错误，请重试", "警告", JOptionPane.ERROR\_MESSAGE);

con.close();

password.setText("unknown");

return;

}

// 关闭与服务器的连接

con.close();

// 是否记住密码，记录在src/usr\_info/文件夹下

if (remember.isSelected()) {

try {

boolean exist = false;

File f = new File("src/usr\_info/" + usrid + ".txt");

if (f.exists()) {

DataOutputStream out = new DataOutputStream(

new FileOutputStream("src/usr\_info/" + usrid + ".txt"));

out.writeUTF(usrpassword);

} else {

f.createNewFile();

DataOutputStream out = new DataOutputStream(

new FileOutputStream("src/usr\_info/" + usrid + ".txt"));

out.writeUTF(usrpassword);

FileWriter fout = new FileWriter("src/usr\_info/usr.txt", true);

fout.write(usrid + '\n');

fout.close();

}

} catch (IOException ev) {

ev.printStackTrace();

}

}

dispose();

// 初始化登录账户，并传给Mainface主界面类

Usr\_Info user = new Usr\_Info("非正常登陆", "");

con = new connection(usrid);

try {

user = con.getUsrFromServer(usrid);

} catch (IOException e2) {

// TODO Auto-generated catch block

e2.printStackTrace();

}

File Root, Picture, User;

Root = new File("./YoPin");

if (!Root.exists() || !Root.isDirectory()) {

Root.mkdir();

}

Picture = new File("./YoPin/Pictures/YoPin1");

if (!Picture.exists() || !Picture.isDirectory()) {

Picture.mkdir();

}

/\*

\* File[] list = Picture.listFiles(); Random r = new Random();

\* Picture = list[r.nextInt(list.length)]; Picture = list[1];

\*/

User = new File("./YoPin/User");

if (!User.exists() || !User.isDirectory()) {

User.mkdir();

}

try {

new MainFace(Root, Picture, User, user, con);

} catch (IOException e1) {

// TODO Auto-generated catch block

e1.printStackTrace();

}

}

return;

}

// 实现内部类使得JCombobox可以获得选择的用户号码

class comboboxmodel extends AbstractListModel implements ComboBoxModel {

String item;

public Object getElementAt(int index) {

return oldaccount.get(index++);

}

public int getSize() {

return oldaccount.size();

}

public void setSelectedItem(Object anItem) {

item = (String) anItem;

}

public Object getSelectedItem() {

return item;

}

}

}

registerClient.java

/\*

\* 注册账号界面，作为JPanel添加到上级容器里

\*/

package Client;

/\*\*

\*

\* @author zhuqihao

\*/

import java.awt.BasicStroke;

import java.awt.Color;

import java.awt.Cursor;

import java.awt.Font;

import java.awt.Graphics;

import java.awt.Graphics2D;

import java.awt.Image;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.MouseAdapter;

import java.awt.event.MouseEvent;

import java.awt.image.BufferedImage;

import java.io.File;

import java.io.IOException;

import java.util.Random;

import javax.imageio.ImageIO;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFileChooser;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JOptionPane;

import javax.swing.JPanel;

import javax.swing.JPasswordField;

import javax.swing.JTextField;

import Info.Usr\_Info;

import MainPage.MainFace;

public class RegisterClient extends JFrame implements ActionListener {

// 部件的声明包括 邮箱地址 用户名 密码 确认密码 头像 验证码

private JPanel background;

private JPanel fun;

private JLabel labelemail;

private JTextField email;

private JLabel labelnickname;

private JTextField nickname;

private JLabel labelpassword;

private JPasswordField password;

private JLabel labelconfirmpassword;

private JPasswordField confirmpassword;

private JLabel labelpass;

private JTextField pass;

private JLabel passimage;

private JLabel newpassimage;

private JButton register;

private JLabel back;

private JLabel nickpicture;

private JButton selectnickpicture;

private JButton eyes;

private String passans;

int randomnum = 0;

// 声明字体

private String font = new String("Xingkai SC");

private int fontsize = 18;

// 头像图片，如果没有设置就是默认

private BufferedImage usr\_image;

private String usr\_image\_name = new String();

// 初始化整个panel并在上级容器里面添加这个panel

public RegisterClient() {

setBounds(300, 160, 800, 500);

this.setLayout(null);

initlabel();

initground();

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

setVisible(true);

}

// 设置界面背景图片

public void setbackground() {

ImageIcon img = new ImageIcon("src/image/registerbackground.jpg");

JLabel background = new JLabel(img);

this.background.add(background);

background.setBounds(this.getX(), this.getY(), img.getIconWidth(), img.getIconHeight());

return;

}

public void initground() {

background = new JPanel();

background.setOpaque(true);

background.setBounds(0, 0, this.getWidth(), this.getHeight());

setbackground();

this.add(background);

// this.getContentPane().add(background);

}

public void initlabel() {

fun = new RoundRectPanel();

fun.setBounds(150, 55, 440, 310);

fun.setOpaque(true);

fun.setBackground(Color.WHITE);

fun.setFocusable(true);

fun.setLayout(null);

this.add(fun);

// 邮箱的label

labelemail = new JLabel("邮箱地址:");

labelemail.setBounds(20, 0, 100, 40);

labelemail.setFont(new Font(font, 1, fontsize));

// 邮箱地址的textfield部件

email = new JTextField();

email.setBounds(100, 5, 160, 30);

// 昵称的label

labelnickname = new JLabel("昵称:");

labelnickname.setBounds(40, 40, 100, 40);

labelnickname.setFont(new Font(font, 1, fontsize));

// 填写昵称的textfield

nickname = new JTextField();

nickname.setBounds(100, 45, 160, 30);

// 密码的label

labelpassword = new JLabel("密码:");

labelpassword.setBounds(40, 80, 100, 40);

labelpassword.setFont(new Font(font, 1, fontsize));

// 填写密码的JpasswordField，设置掩盖字符

password = new JPasswordField();

password.setText("请输入6-15的英文、数字密码");

password.setEchoChar((char) 0);

password.addMouseListener(new MouseAdapter() {

public void mouseClicked(MouseEvent e) {

password.setText("");

password.setEchoChar('\*');

}

});

password.setBounds(100, 85, 160, 30);

// 确认密码的label

labelconfirmpassword = new JLabel("确认密码:");

labelconfirmpassword.setBounds(20, 120, 100, 40);

labelconfirmpassword.setFont(new Font(font, 1, fontsize));

// 确认密码的密码框JPasswordField

confirmpassword = new JPasswordField();

confirmpassword.setBounds(100, 125, 160, 30);

confirmpassword.setEchoChar('\*');

// 验证码的label

labelpass = new JLabel("验证码:");

labelpass.setBounds(30, 160, 100, 40);

labelpass.setFont(new Font(font, 1, fontsize));

// 验证码的填写区域

pass = new JTextField();

pass.setBounds(100, 165, 60, 30);

// 验证码的图片

passimage = new JLabel(new ImageIcon(newpass()));

passimage.setBounds(170, 165, 80, 30);

// 更换验证码的label，点击重新生成一张验证码图片

newpassimage = new JLabel("<html><u>看不清？换一张</u></html>");

newpassimage.setForeground(Color.red);

newpassimage.setBounds(255, 180, 100, 15);

newpassimage.addMouseListener(new MouseAdapter() {

public void mouseEntered(MouseEvent e) {

// 设置鼠标移入的时候设置鼠标形状为手型

newpassimage.setCursor(Cursor.getPredefinedCursor(Cursor.HAND\_CURSOR));

}

public void mouseExited(MouseEvent e) {

newpassimage.setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT\_CURSOR));

}

});

newpassimage.addMouseListener(new MouseAdapter() {

public void mouseClicked(MouseEvent e) {

// 生成新的验证码图片

passimage.setIcon(new ImageIcon(newpass()));

}

});

// 返回登录界面的label

back = new JLabel("<html><u>已有账号？点击登录</u></html>");

back.setBounds(230, 280, 180, 20);

back.setForeground(Color.red);

back.setFont(new Font("华文仿宋", 0, fontsize - 2));

back.addMouseListener(new MouseAdapter() {

public void mouseEntered(MouseEvent e) {

back.setCursor(Cursor.getPredefinedCursor(Cursor.HAND\_CURSOR));

}

public void mouseExited(MouseEvent e) {

back.setCursor(Cursor.getPredefinedCursor(Cursor.DEFAULT\_CURSOR));

}

public void mouseClicked(MouseEvent e) {

dispose();

new LoginClient();

}

});

// 注册按钮

register = new JButton("立即注册");

register.setBounds(100, 250, 120, 50);

register.addActionListener(this);

register.setFont(new Font(font, 1, fontsize + 2));

// 设置头像的文件选择框

nickpicture = new JLabel();

nickpicture.setBounds(320, 10, 100, 100);

nickpicture.setOpaque(true);

nickpicture.setIcon(new ImageIcon("src/usr\_image/default.jpg"));

fun.add(nickpicture);

/// 选择头像的label

selectnickpicture = new JButton("本地选择头像");

selectnickpicture.setFont(new Font("宋体", 0, 10));

selectnickpicture.setBounds(320, 120, 100, 30);

selectnickpicture.addActionListener(this);

fun.add(selectnickpicture);

// 查看密码的按钮

ImageIcon img = new ImageIcon(("src/image/eyes.png"));

eyes = new JButton(img);

eyes.setBounds(260, 90, 30, 20);

eyes.addActionListener(this);

fun.add(eyes);

fun.add(back);

fun.add(register);

fun.add(newpassimage);

fun.add(passimage);

fun.add(pass);

fun.add(labelpass);

fun.add(confirmpassword);

fun.add(labelconfirmpassword);

fun.add(password);

fun.add(labelpassword);

fun.add(nickname);

fun.add(labelnickname);

fun.add(email);

fun.add(labelemail);

}

// 获得随机的颜色，来画验证码

Color getRandColor() {

Random random = new Random();

int r = random.nextInt(2 \* randomnum + 100) % 256;

int g = random.nextInt(10 \* randomnum + 122) % 256;

int b = random.nextInt(30 \* randomnum + 299) % 256;

randomnum += 10;

return new Color(r, g, b);

}

// 生成验证码的函数

public BufferedImage newpass() {

int width = 80, height = 30;

BufferedImage image = new BufferedImage(width, height, BufferedImage.TYPE\_INT\_RGB);

// 声明一个画图对象

Graphics g = image.getGraphics();

g.setColor(Color.white);

g.fillRect(0, 0, width, height);

// 再图上先画出想要的四位验证码

g.setFont(new Font("Times New Roman", Font.BOLD, 18));

g.setColor(new Color(0, 0, 0));

g.drawRect(0, 0, width - 1, height - 1);

((Graphics2D) g).setStroke(new BasicStroke(1.5f));

Random random = new Random();

// 在图上随机找两个点，然后在两个点之间随机画线，以此达到干扰效果

for (int i = 0; i < 50; i++) {

g.setColor(getRandColor());

int x = random.nextInt(width);

int y = random.nextInt(height);

int xl = random.nextInt(12);

int yl = random.nextInt(12);

g.drawLine(x, y, x + xl, y + yl);

}

String sRand = "";

// 随机选取数字和字母进行填充验证码

for (int i = 0; i < 4; i++) {

String rand = null;

if (random.nextInt(10) > 5) {

rand = String.valueOf((char) (random.nextInt(10) + 48));

} else {

rand = String.valueOf((char) (random.nextInt(26) + 65));

}

sRand += rand;

g.setColor(getRandColor());

g.drawString(rand, 15 \* i + 10, 16);

}

g.dispose();

passans = new String(sRand);

return image;

}

// 判断填写的字符串是否是一个合法的邮箱地址

static boolean isemailaddress(String s) {

int atloc = -1;

// 判断是否有@符号

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

if (s.charAt(i) == '@') {

if (atloc == -1) {

atloc = i;

} else {

return false;

}

}

}

if (atloc == 0 || atloc == s.length() - 1) {

return false;

}

// 判断是否有合理域名

if (s.charAt(0) == '.' || s.charAt(s.length() - 1) == '.') {

return false;

}

for (int i = atloc + 1; i < s.length(); ++i) {

if (s.charAt(i) == '.') {

if (i == atloc + 1) {

return false;

} else {

return true;

}

}

}

return false;

}

// 取得图片的后缀名，以此来保存图片

public String suffix(String image\_name) {

StringBuilder suffix = new StringBuilder();

for (int i = image\_name.length() - 1; i >= 0; --i) {

if (image\_name.charAt(i) == '.')

break;

suffix.append(image\_name.charAt(i));

}

suffix.reverse();

return suffix.toString();

}

// 处理panle中部件中的各种点击事件和移入事件

public void actionPerformed(ActionEvent e) {

// 点击选择头像，弹出文件选择框，选择合适的图片

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

JFileChooser select = new JFileChooser();

select.showDialog(select, "选择头像");

File file = select.getSelectedFile();

// 将图片进行resize，label的大小固定为100\*100，因此需要将图片进行改变大小

if (file.exists()) {

try {

Image img = ImageIO.read(file);

BufferedImage image = new BufferedImage(100, 100, BufferedImage.TYPE\_INT\_RGB);

Graphics g = image.getGraphics();

g.drawImage(img, 0, 0, 100, 100, Color.LIGHT\_GRAY, null);

g.dispose();

usr\_image = image;

nickpicture.setIcon(new ImageIcon(image));

usr\_image\_name = file.getName();

} catch (IOException event) {

JOptionPane.showMessageDialog(null, "图片读取错误，请重新选择", "警告", JOptionPane.ERROR\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(null, "不存在该文件，请重新选择", "警告", JOptionPane.ERROR\_MESSAGE);

}

}

// 点击注册按钮，和服务器进行连接，并且检查各项的注册输入是否符合规则

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

if (!isemailaddress(email.getText())) {

email.setText("");

JOptionPane.showMessageDialog(null, "非法邮箱地址,请重新输入", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

if (!password.getText().equals(confirmpassword.getText())) {

confirmpassword.setText("");

JOptionPane.showMessageDialog(null, "两次输入的密码不同,请重新输入", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

if (password.getText().length() < 6 || password.getText().length() > 15) {

JOptionPane.showMessageDialog(null, "输入的密码不符合要求，请重新输入", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

if (!passans.equals(pass.getText())) {

pass.setText("");

JOptionPane.showMessageDialog(null, "验证码错误,请重新输入", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

connection con = new connection(nickname.getText());

if (!con.active) {

JOptionPane.showMessageDialog(null, "与服务器连接异常，请稍后再试!", "警告", JOptionPane.ERROR\_MESSAGE);

return;

}

// 向用户的邮箱地址发送说明邮件

new sendemail("1500012943@pku.edu.cn", email.getText(), "欢迎使用悠品软件，开始享受您的优质生活。");

// 服务器返回注册的生成的用户号码

String usr\_id = con.new\_usr(nickname.getText(), password.getText(), email.getText());

JOptionPane.showMessageDialog(null, "注册成功，自动登录中...您的博客号码为" + usr\_id);

try {

// 向本地缓存添加头像，如果没有选择。那么使用默认头像

if (usr\_image == null) {

BufferedImage image = new BufferedImage(100, 100, BufferedImage.TYPE\_INT\_RGB);

File f = new File("src/usr\_image/default.jpg");

Graphics g = image.getGraphics();

g.drawImage(ImageIO.read(f), 0, 0, 100, 100, Color.LIGHT\_GRAY, null);

g.dispose();

usr\_image = image;

usr\_image\_name = "default.jpg";

}

ImageIO.write(usr\_image, suffix(usr\_image\_name), new File("src/usr\_image/" + usr\_id));

} catch (Exception ev) {

ev.printStackTrace();

}

dispose();

con.close();

con = new connection(usr\_id);

Usr\_Info user = new Usr\_Info("非正常进入账号", "");

// 传递新建的、和从服务器上读取的用户信息

try {

user = con.getUsrFromServer(usr\_id);

} catch (IOException e2) {

e2.printStackTrace();

}

File Root, Picture, User;

Root = new File("./YoPin");

if (!Root.exists() || !Root.isDirectory()) {

Root.mkdir();

}

Picture = new File("./YoPin/Pictures/YoPin1");

if (!Picture.exists() || !Picture.isDirectory()) {

Picture.mkdir();

}

User = new File("./YoPin/User");

if (!User.exists() || !User.isDirectory()) {

User.mkdir();

}

try {

new MainFace(Root, Picture, User, user, con);

} catch (IOException e1) {

e1.printStackTrace();

}

}

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

if (password.getEchoChar() != 0) {

password.setEchoChar((char) 0);

} else {

password.setEchoChar('\*');

}

}

}

}

RoundPanel.java

//实现圆角panel

package Client;

import java.awt.Graphics;

import java.awt.geom.RoundRectangle2D;

/\*\*

\*

\* @author zhuqihao

\*/

import javax.swing.JPanel;

public class RoundRectPanel extends JPanel {

RoundRectPanel() {

setBounds(0, 0, 400, 600);

}

// 重写panel的paint函数

public void paint(Graphics g) {

RoundRectangle2D.Double rect = new RoundRectangle2D.Double(0, 0, getWidth(), getHeight(), 20, 20);

g.setClip(rect);

super.paint(g);

}

}

sendmail.java

/\*

\* 给指定邮箱地址发送邮件

\*/

package Client;

import java.util.Date;

import java.util.Properties;

import javax.mail.Message;

import javax.mail.Session;

import javax.mail.Transport;

import javax.mail.internet.InternetAddress;

import javax.mail.internet.MimeMessage;

public class sendemail {

private static final String account = "";

private static final String password = "";

public sendemail(String fromemail, String tomail, String body) {

try {

Properties props = new Properties();

props.put("mail.trasport.protocol", "smtp");

props.put("mail.smtp.host", "smtp.pku.edu.cn");

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

props.put("mail.smto.port", "25"); // 邮件发送的端口为25

Session session = Session.getInstance(props);

MimeMessage message = new MimeMessage(session);

message.setFrom(new InternetAddress(fromemail));

message.setRecipient(Message.RecipientType.TO, new InternetAddress(tomail));

message.setText(body);

message.setSentDate(new Date());

// message.saveChanges();

Transport transport = session.getTransport("smtp");

transport.connect(account, password); // 发送邮件的账号和密码

transport.sendMessage(message, message.getAllRecipients());// 发送邮件,其中第二个参数是所有已设好的收件人地址

transport.close();

} catch (Exception e) {

System.out.print(e);

}

}

}

connection.java

/\*

\*和服务器进行沟通的connection类，实现各种交互功能

\*/

package Client;

import java.io.DataInputStream;

import java.io.DataOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

/\*\*

\*

\* @author zqh

\*/

import java.net.Socket;

import Community.blogAbstract;

import Community.blogComment;

import Info.Usr\_Info;

public class connection {

private String serverip;

private int port;

private Socket client\_socket;

private DataOutputStream cout;

private DataInputStream cin;

public boolean active;

private String usrId;

public connection(String usrid) {

usrId = usrid;

// 向port8888和服务器建立连接

port = 8888;

// 本地测试

serverip = new String("127.0.0.1");

// 启用服务器

// serverip=new String("120.77.203.54");

active = true;

// 向套接字发出连接请求

try {

client\_socket = new Socket(serverip, port);

cout = new DataOutputStream(client\_socket.getOutputStream());

cin = new DataInputStream(client\_socket.getInputStream());

cout.writeUTF(usrid);

} catch (Exception e) {

active = false;

e.printStackTrace();

}

}

// 用户登录请求

public String sentusr\_password(String usr\_id, String usr\_password) {

String result = "";

try {

cout.writeUTF("usr\_password");

cout.writeUTF(usr\_id);

cout.writeUTF(usr\_password);

result = cin.readUTF();

} catch (Exception e) {

System.out.print(e);

}

return result;

}

// 申请新用户

public String new\_usr(String usr\_id, String usr\_password, String usr\_email) {

String result = "";

try {

cout.writeUTF("usr\_newusr");

cout.writeUTF(usr\_id);

cout.writeUTF(usr\_password);

cout.writeUTF(usr\_email);

result = cin.readUTF();

} catch (IOException e) {

} catch (Exception e) {

e.printStackTrace();

}

return result;

}

// 修改用户信息

public void usr\_info\_change(Usr\_Info new\_usr) {

try {

cout.writeUTF("change\_usr\_info");

cout.writeUTF(new\_usr.getName());

cout.writeUTF(new\_usr.getID());

cout.writeUTF(new\_usr.getKeyword());

cout.writeUTF(new\_usr.getStar());

cout.writeUTF(new\_usr.getAge());

cout.writeUTF(new\_usr.getJob());

cout.writeUTF(new\_usr.getHobby());

cout.writeUTF(new\_usr.getMail());

} catch (Exception e) {

System.out.println(e);

}

}

// 获取帖子标题内容

public blogAbstract getBlogText(int kind, int id) {

blogAbstract result = new blogAbstract();

try {

cout.writeUTF("getblogtext");

cout.writeInt(kind);

cout.writeInt(id);

String title = cin.readUTF();

String authorname = cin.readUTF();

String time = cin.readUTF();

String commentid = cin.readUTF();

byte buf[] = new byte[8000];

String name = cin.readUTF();

long flength = cin.readLong();

String filename = "src/blog\_icon/" + id + name;

DataOutputStream fout = new DataOutputStream(new FileOutputStream(filename));

long rlen = 0;

while (true) {

int read = 0;

read = cin.read(buf);

rlen += read;

fout.write(buf, 0, read);

if (rlen == flength)

break;

}

fout.flush();

result.authorName = new String(authorname);

result.commentID = Integer.parseInt(commentid);

result.time = new String(time);

result.title = new String(title);

result.filename = new String(filename);

} catch (Exception e) {

System.out.println(e);

}

return result;

}

// 获取贴子主要内容

public void getblogmaintext(int kind, int blogID, String filepath) {

try {

cout.writeUTF("getblogmaintext");

cout.writeInt(kind);

cout.writeInt(blogID);

int filenum = cin.readInt();

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

String filename = cin.readUTF();

long flength = cin.readLong();

File file = new File(filepath + "/" + filename);

file.createNewFile();

DataOutputStream fout = new DataOutputStream(new FileOutputStream(file));

byte buf[] = new byte[8000];

long rlen = 0;

while (true) {

int read = 0;

read = cin.read(buf);

rlen += read;

fout.write(buf, 0, read);

if (rlen == flength)

break;

}

fout.flush();

fout.close();

}

} catch (Exception e) {

e.printStackTrace();

}

return;

}

// 获得帖子的已有评论

public blogComment getblogcomment(int kind, int blogID, int commentID) {

blogComment buf = new blogComment();

try {

cout.writeUTF("getblogcomment");

cout.writeInt(kind);

cout.writeInt(blogID);

cout.writeInt(commentID);

buf.authorName = cin.readUTF();

buf.time = cin.readUTF();

buf.commentText = cin.readUTF();

} catch (Exception e) {

e.printStackTrace();

}

return buf;

}

// 新建帖子

public void createblog(blogAbstract abstractblog, String filename, File file) {

try {

cout.writeUTF("createblog");

cout.writeInt(abstractblog.kind);

cout.writeUTF(abstractblog.title);

cout.writeUTF(abstractblog.authorName);

cout.writeUTF(abstractblog.time);

cout.writeUTF(String.valueOf(abstractblog.commentID));

DataInputStream fin = new DataInputStream(new FileInputStream(file));

cout.writeUTF(file.getName());

cout.writeLong(file.length());

byte buf[] = new byte[8000];

while (true) {

int read = 0;

read = fin.read(buf);

if (read == -1) {

break;

}

cout.write(buf, 0, read);

}

Thread.sleep(10);

File Allfile = new File(filename);

File[] files = Allfile.listFiles();

cout.writeInt(files.length);

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

// 重要，需要休眠10ms

Thread.sleep(10);

fin = new DataInputStream(new FileInputStream(files[i]));

cout.writeUTF(files[i].getName());

cout.writeLong(files[i].length());

buf = new byte[8000];

while (true) {

int read = 0;

read = fin.read(buf);

System.out.println(read);

if (read == -1) {

break;

}

cout.write(buf, 0, read);

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

// 添加评论

public void addcomment(int kind, int blogid, blogComment comment) {

try {

cout.writeUTF("addcomment");

cout.writeInt(kind);

cout.writeInt(blogid);

cout.writeUTF(comment.authorName);

cout.writeUTF(comment.time);

cout.writeUTF(comment.commentText);

} catch (Exception e) {

e.printStackTrace();

}

}

// 获得当前已有帖子数目

public int blognumber(int kind) {

int ans = 0;

try {

cout.writeUTF("getblognumber");

cout.writeInt(kind);

ans = cin.readInt();

} catch (Exception e) {

e.printStackTrace();

}

return ans;

}

// 从服务器获取用户信息

public Usr\_Info getUsrFromServer(String usrName) throws IOException {

cout.writeUTF("getUsr");

cout.writeUTF(usrName);

Usr\_Info tmp;

String name, id, star, age, job, hobby, mail;

name = cin.readUTF();

id = cin.readUTF();

star = cin.readUTF();

age = cin.readUTF();

job = cin.readUTF();

hobby = cin.readUTF();

mail = cin.readUTF();

tmp = new Usr\_Info(id, "test");

tmp.setName(name);

tmp.setAge(age);

tmp.setStar(star);

tmp.setJob(job);

tmp.setHobby(hobby);

tmp.setMail(mail);

return tmp;

}

// 关闭套接字连接

public void close() {

try {

cout.writeUTF("bye");

client\_socket.close();

} catch (IOException e) {

e.printStackTrace();

}

}

}