网络程序设计

（邮件发送程序）

班级：电气 16

姓名：樊凯

学号：01041163

email：fk8472441@163.com

一． 实验任务

Mail Sender (User agent) (Chapter 2)

This Mail Sender provides a graphical interface for the sender ,with fields for sender's e-mail address, recipient's e-mail address, subject of the message, and the message itself. It can set the SMTP server's IP address. GUI you can see at Textbook page 164 .

二.环境

cpu：p4 2.0G，内存 256M，操作系统：windowsXP

开发语言：Java，开发平台：Jbuilder9

三．设计思路

java 语言提供了专门用来编写 email 程序 javamail 包，在 javamail API 里有丰富的类，利用 javamail 包里的类，可以很方便的编写出像 Outlook，Foxmail 等优秀的 email 代理程序。

JavaMail 的基础知识

使用 JavaMail 是发送电子邮件所需要的组件 。

JavaMail 的机构使处理电子邮件非常容易。下面列出了一些我们需要的类：

1．Properties

JavaMail 需要 Properties 来创建一个 session 对象。它将寻找字符串"mail.smtp.host"，属性值就是发送邮件的主机，如：

```
Properties props = new Properties ();
props.put("mail.smtp.host", "smtp.abcd.com");//可以换上你的 smtp 主机名。
```

2．Session

这个 Session 类代表 JavaMail 中的一个邮件 session. 每一个基于 JavaMail 的应用程序至少有一个 session 但是可以有任意多的 session。 在这个例子中, Session 对象需要知道用来处理邮件的 SMTP 服务器。为了做到这一点，你可以参照下面的例子用 Properties 来创建一个 Session 对象

```
Session sendMailSession;
sendMailSession = Session.getInstance(props, null);
```

3．Transport

邮件是既可以被发送也可以被受到。JavaMail 使用了两个不同的类来完成这两个功能：Transport 和 Store。 Transport 是用来发送信息的，而 Store 用来收信。对于这的教程我们只需要用到 Transport 对象。Store 的用法请参看 Sun 的 JavaMail 文档。

用法：Transport transport;

```
transport = sendMailSession.getTransport("smtp");
```

用 JavaMail Session 对象的 getTransport 方法来初始化 Transport。传过去的字符串申明了对象所要使用的协议，如"smtp"。这将为我们省了很多时间。因为 JavaMail 以境内置了很

多协议的实现方法。

注意: JavaMail 并不是绝对支持每一个协议，目前支持 IMAP、 SMTP 和 POP3。

4．Message

Message 对象将存储我们实际发送的电子邮件信息，Message 对象被作为一个 MimeMessage 对象来创建并且需要知道应当选择哪一个 JavaMail session。

使用方法是：Message newMessage = new MimeMessage(sendMailSession);

我的程序实现的功能主要有，除(1)可以正常发送文字消息外，(2)还可发送附件，(3)现在大多数邮件服务器都要求登陆者行身份验证，因此本程序已需要此项功能。(4)另外，为了方便用户使用，要求 smtp 服务器只需要第一次进行配置，以后使用从配置文件直接读取。

下面就是实现以上功能的代码

以下代码用于邮件发送

代码 1

```
//MIME 邮件对象
 MimeMessage mimeMsg = null;
//邮件会话对象
 Session session = null;
Properties props = System.getProperties();    //获得系统属性
props.put("mail.smtp.host", mailhost);          //设置 SMTP 主机

//获得邮件会话对象
session = Session.getDefaultInstance(props,null);
//创建 MIME 邮件对象
mimeMsg = new MimeMessage( session );
//设置发信人
mimeMsg.setFrom(new InternetAddress( from ) );

//设置收信人
if(to!=null){
    mimeMsg.setRecipients( Message.RecipientType.TO, InternetAddress.parse( to ) );
}

//设置抄送人
if(cc!=null){
    mimeMsg.setRecipients( Message.RecipientType.CC, InternetAddress.parse( cc ) );
}

//设置暗送人
if(bcc!=null){
```

```
                    mimeMsg.setRecipients(Message.RecipientType.BCC,
        InternetAddress.parse( bcc ) );
        }

        //设置邮件主题
        mimeMsg.setSubject(subject,"GBK");

        //设置邮件内容
        mimeMsg.setText( content ,"GBK" );
        //发送日期
        mimeMsg.setSentDate(new Date());
        //发送邮件
        Transport.send( mimeMsg );
        System.out.println( "email send！ ");

      } catch (Exception e) {
        e.printStackTrace();
      }
```

以下代码用于发送附件：

为了发送附件需要定义文件名 String filename

把附件作为信件内容的第二部分一起发送。因此对上面的代码稍作改动。

代码 2

```
        // 第一部分信息
        MimeBodyPart mbp1 = new MimeBodyPart();
        mbp1.setText( content, "GBK");

        // 第二部分信息
        MimeBodyPart mbp2 = new MimeBodyPart();

        // 在第二部分信息中附加一个文件
        FileDataSource fds = new FileDataSource( fileAttachment );
        mbp2.setDataHandler(new DataHandler(fds));
        mbp2.setFileName(fds.getName());
    }
        // 创建 Multipart 并放入每个 MimeBodyPart
        Multipart mp = new MimeMultipart();
        mp.addBodyPart( mbp1 );
        mp.addBodyPart( mbp2 );

        // 增加 Multipart 到信息体
```

mimeMsg.setContent(mp);

目前绝大多数邮件服务器都需要进行身份验证，而用 javamail 进行身份验证也很简单。

首先进行身份验证需要新建一个类 Email_Autherticatorbean。带身份验证的邮件要用到的这个类，这个类一定要继承 Authenticator 类，并覆盖 getPasswordAuthentication 方法
Email_Autherticatorbean 类的代码如下

代码 3

```
import javax.mail.*;
public class Email_Autherticatorbean extends javax.mail.Authenticator
{
   private String m_username = null;
   private String m_userpass = null;

   public void setUsername(String username)
   {
      m_username = username;
   }

   public void setUserpass(String userpass)
   {
      m_userpass = userpass;
   }

   public Email_Autherticatorbean()
   {
      super();
   }

   public Email_Autherticatorbean(String username, String userpass)
   {
      super();
      setUsername(username);
      setUserpass(userpass);
   }
   //在需要身份验证时自动被调用
```

```
public PasswordAuthentication getPasswordAuthentication()
        {
           return new PasswordAuthentication(m_username,m_userpass);
        }
      }
```

在主函数中调用这个类就可验证身份，但是需要对代码 1 做一些改动。

定义用户名与密码变量 String user=null；String password=null；

需要添加设置身份验证的语句

```
props.put("mail.smtp.host", mailhost);          //设置 SMTP 主机
        props.put("mail.smtp.auth","true");          //设置身份验证为真，若须身份验证则必须设为真

//获得邮件会话对象
```

代码 1 中的 session = Session.getDefaultInstance(props,null); 改为

```
session = Session.getDefaultInstance(props, new Email_Autherticatorbean( user, password ) );
```

以上的改动可以完成身份验证功能。

对配置文件的读写

因此要用 java 的 io 流，需要 java.io 包。当程序启动时从文件中读取信息，没改变一次配置，就把新的设置写入到文件中。
这部分代码如下：

写入数据

```
try {
            BufferedWriter out = new BufferedWriter(
            new OutputStreamWriter(new FileOutputStream("conf.txt")));

           out.write(strsmtp);
           out.write("\r");
           out.write(strusername);
           out.write("\r");
           out.write(strpassword);
```

```
            out.close();
      }
          catch(IOException d)
          {}
```

读取数据：把文件中的数据放入数组中

```
try {
          String[] aa = {
                "", "", ""};
          String s;
          int i = 0;
          BufferedReader in = new BufferedReader(new InputStreamReader(new
                FileInputStream("conf.txt")));
          while ( (s = in.readLine()) != null) {
            aa[i] = s;
            i++;

          }
}
      catch (IOException d) {}
```

//

以上代码，可以实现本程序的基本功能。还需要一个图形界面，来方便用户使用。Java 同样提供了，强大的可视化编程的类库，Swing 类和 awt 类。而且 Jbuilder9 的集成开发环境，有很好的 UI 设计功能，可以很方便的设计出美观的图形界面。

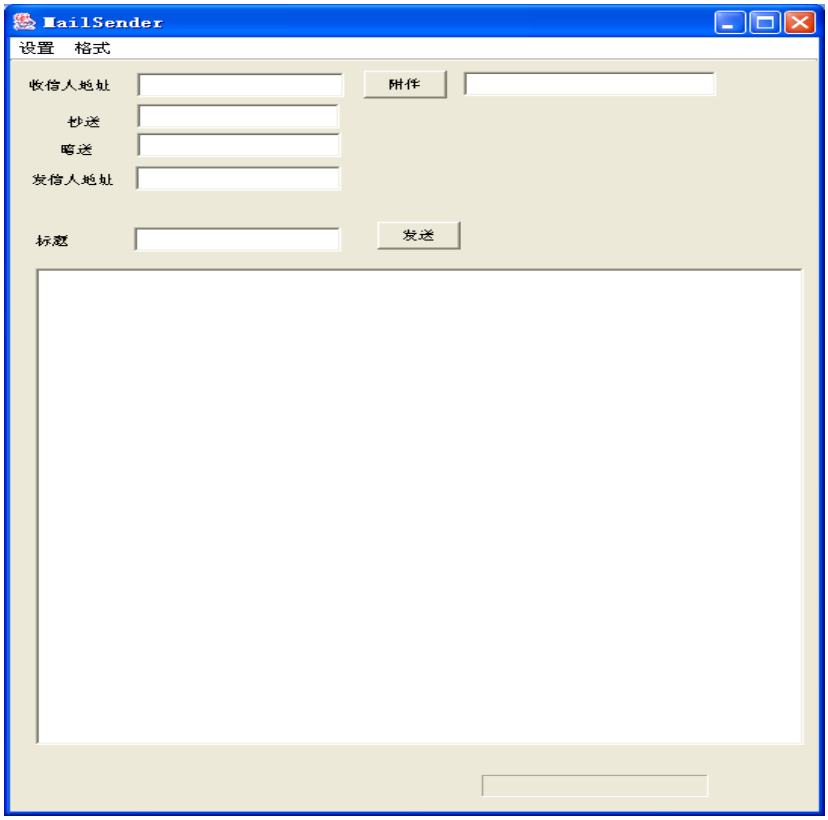
因此以后要做的就是，把功能模块的代码加在其中就可以了。如把代码 1，加到发送按钮的事件处理函数中，每当按按钮就把信发送出去。

四. 实验过程

编程阶段

首先就是制作用户界面，如下图

主界面



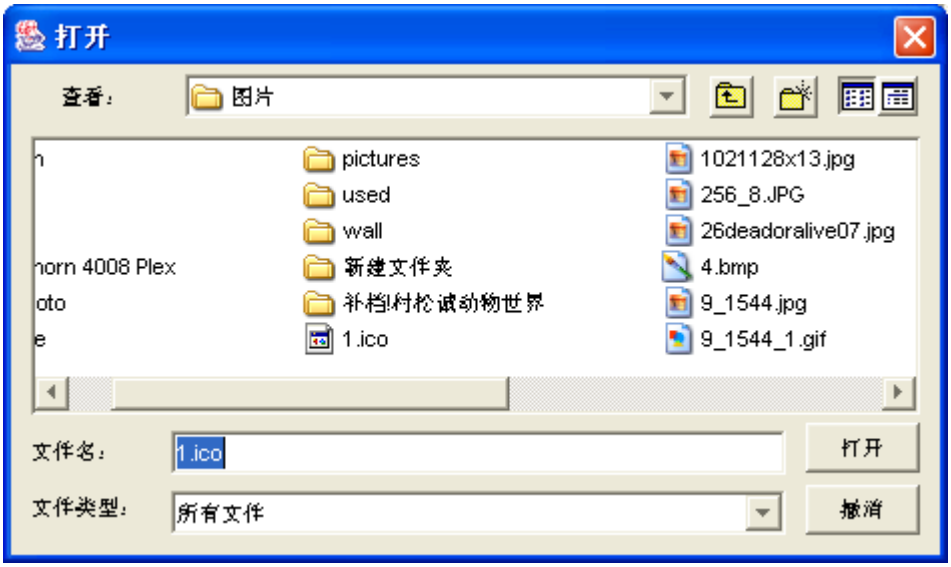
设置面板

然后添加相应的事件处理函数。 在发送按钮的事件处理函数中添加代码 1，并稍作改动使之满足发送附件与身份验证的要求。具体程序见原文件。

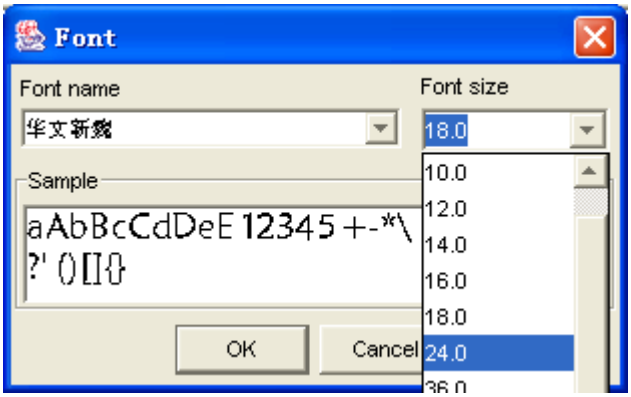
程序调试运行结果

设置服务器

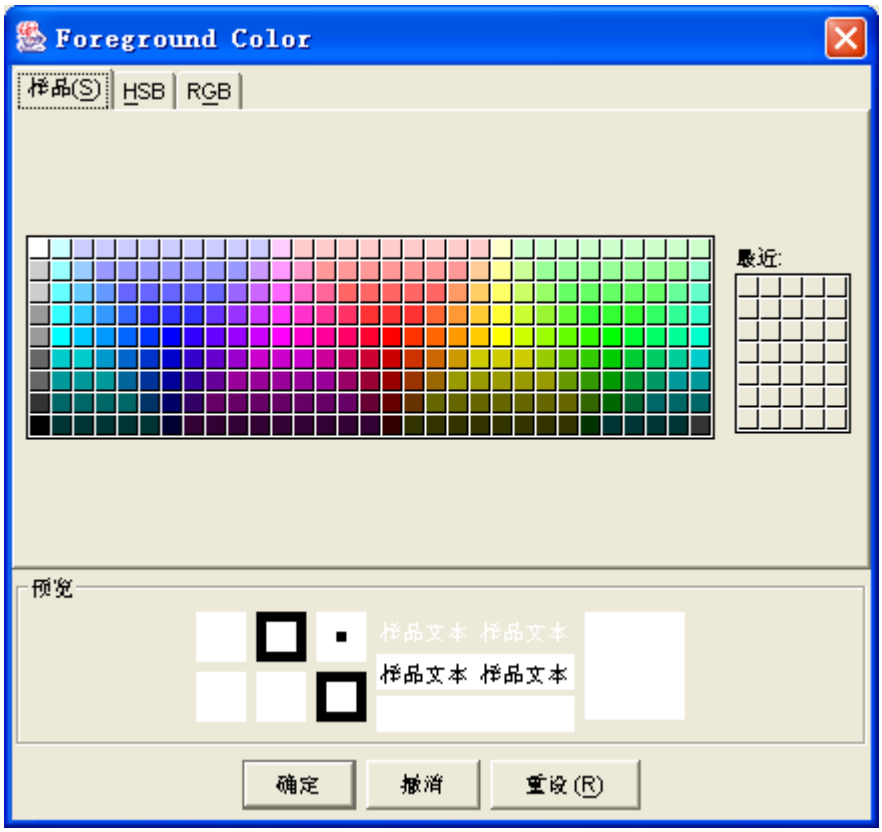
添加附件：



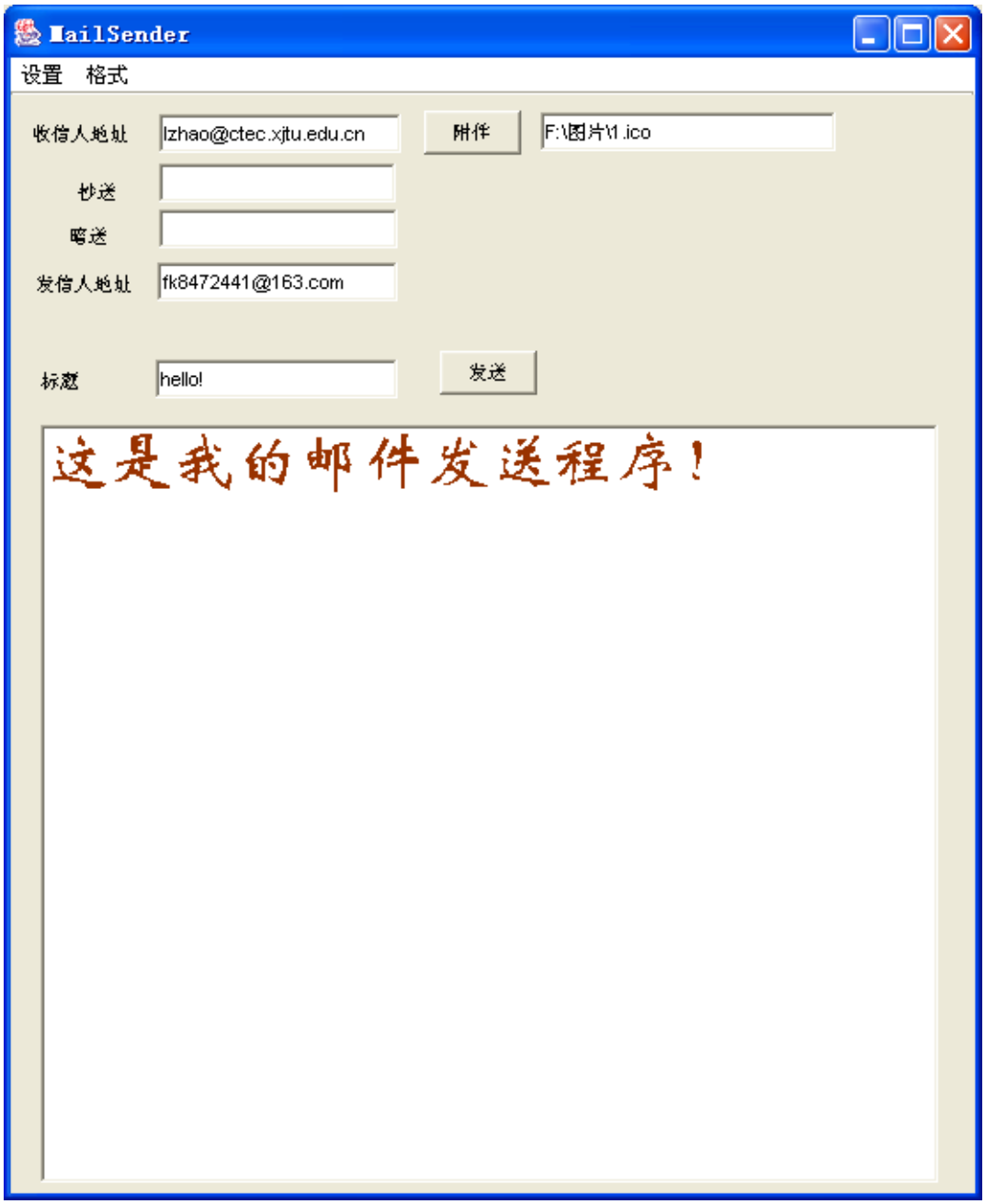
设置字体



设置背景


Foreground Color
样品(S)
HSB
RGB
最近:
预览
样品文本 样品文本
样品文本 样品文本
确定
撤消
重设(R)

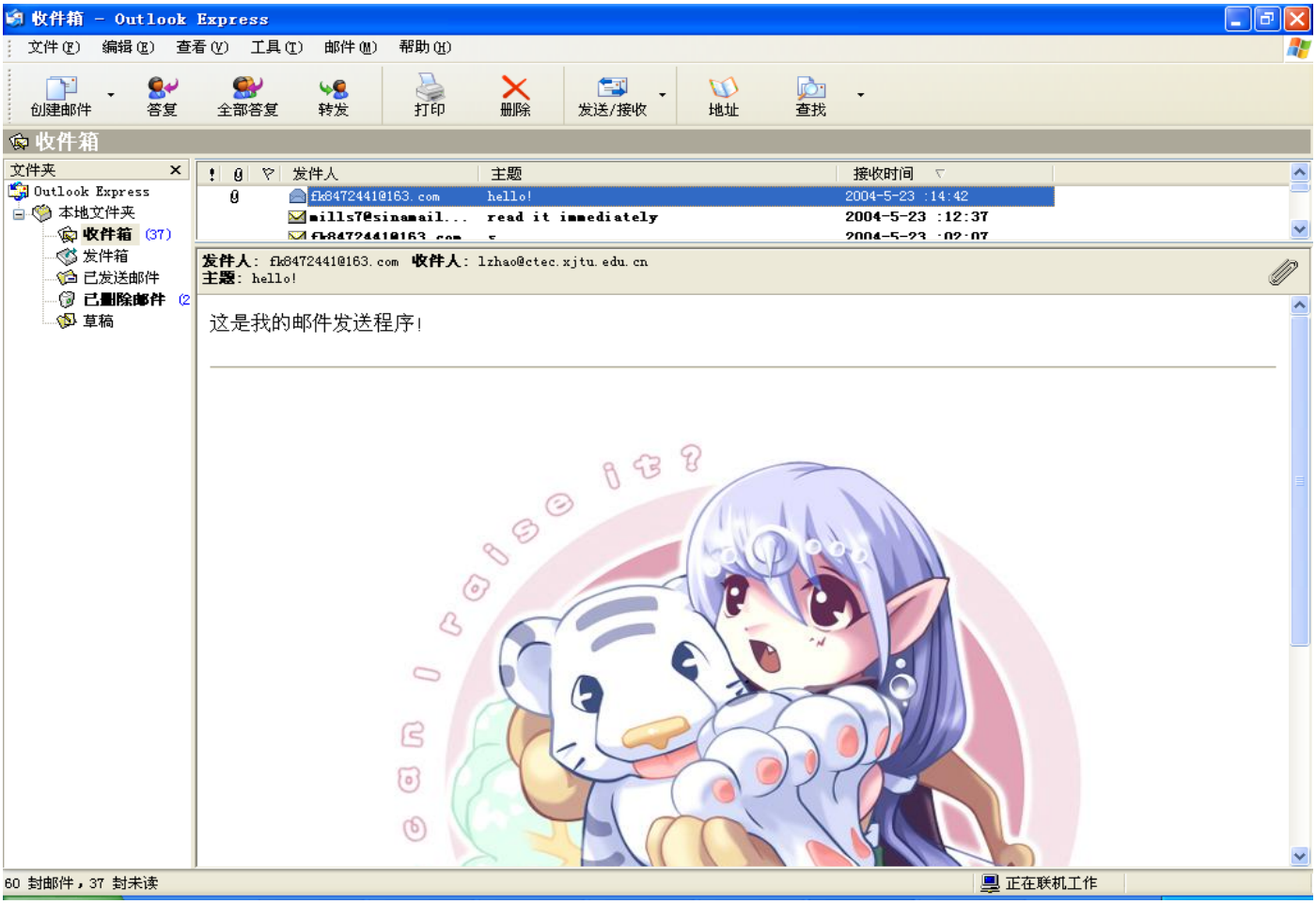
邮件撰写：


MailSender
设置
格式
收信人地址
lzhao@ctec.xjtu.edu.cn
附件
F:\图片\1.ico
抄送
暗送
发信人地址
fk8472441@163.com
标题
hello!
发送
这是我的邮件发送程序！

发送成功：


About
Send Success!
Ok

用 OutLook 接受测试：



五.分析结论

本程序运行结果完全满足题目要求，而且完成题目要求以外的功能，如可以发送附件，进行身份验证，使用配置文件等。

通过这次编程实验，使我对网络编程有了更进一步的认识，网络编程其实没有想象的那么神秘。通过实验，我对 java 的学习又深入一步，学会了 javamail 的使用，还学会了 java 可视化编程。这些都是以前不太会用的。在编程的过程中我看了很多 java 的书，分析了很多例子，感觉收获不少，并且也觉得 java 的内容真的很多。

比较可惜的是本想在发送过程中显示发送进度，由于对 Timer 类不太了解，所以没能成功。还有一些功能没时间去做，比如接收邮件，地址簿。这些只能以后再作了。